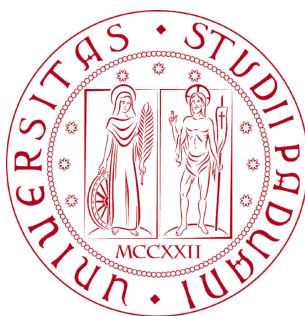




UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI - DTG

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA
MECCATRONICA

APPLICAZIONE DI RETI NEURALI
CONVOLUZIONALI A SISTEMI
RGB-D PER IL RICONOSCIMENTO E
IDENTIFICAZIONE DI
COMPONENTI MECCANICI

RELATORE: ING. RICCARDO MINTO

LAUREANDO: REBUSTI THOMAS

MATRICOLA: 2106686

ANNO ACCADEMICO 2025-2026

Indice

Sommario	IX
Introduzione	XI
1 Stato dell'Arte e obiettivo della tesi	1
1.1 Sensori RGB-D e acquisizione di informazioni	1
1.1.1 Panorama sulle tecnologie per la misura della profondità .	1
1.2 Principio fisico del Time of Flight	4
1.3 Riconoscimento del pezzo: da 2D a RGB-D	5
1.4 Stima della posa	7
1.5 Reti neurali artificiali e deep learning	9
1.5.1 Neurone artificiale e modello di rete	9
1.5.2 Paradigmi di apprendimento	10
1.5.3 Reti neurali profonde	10
1.5.4 Reti neurali convoluzionali	10
1.5.5 Object detection: riconoscimento e localizzazione	11
1.5.6 Approccio one-stage e famiglie YOLO	11
1.5.7 Verso YOLOX	12
1.6 Rete neurale utilizzata: YOLOX in ambiente MATLAB	12
1.7 Addestramento della rete neurale	15
1.8 Obiettivo del progetto	17
2 Telecamera Orbbec Femto Bolt	19
2.1 Presentazione generale della camera	19
2.2 Architettura e funzionamento	20

2.3	Mappa di profondità e rappresentazione tridimensionale	21
2.4	Scelte progettuali e filosofia dell'approccio adottato	24
2.5	Organizzazione dati e file Output	25
2.5.1	Acquisizione e configurazione degli stream	25
2.6	Avvio della pipeline e interazione con il dispositivo	26
2.7	Stabilizzazione iniziale del sensore e motivazioni	27
2.8	Allineamento tra immagine RGB e mappa di profondità	28
2.9	Salvataggio delle immagini e conservazione dell'informazione . . .	30
2.10	Considerazioni finali sul programma	31
3	Reti Neurali Artificiali	33
3.1	Introduzione alle reti neurali Artificiali	33
3.1.1	Origini storiche delle reti neurali	34
3.2	Affermazione del deep learning	36
3.3	Struttura generale	37
3.3.1	Caratteristiche fondamentali	38
3.4	Apprendimento e ottimizzazione	38
3.5	Continuità tra deep learning e MATLAB	40
3.5.1	Motivazioni di MATLAB nel contesto del progetto	41
3.6	Valutazione delle architetture di object detection	43
3.6.1	Introduzione alla fase di valutazione delle reti neurali . . .	43
3.6.2	Motivazioni della scelta delle versioni "Tiny"	43
3.7	Confronto concettuale tra YOLOv4 e YOLOX	44
3.7.1	YOLOX Tiny: evoluzione architetturale e approccio anchor-free	45
3.8	Costruzione del dataset per riconoscimento dei pezzi	46
3.8.1	Composizione del dataset	46
3.8.2	Script MATLAB per l'acquisizione automatizzata del dataset	47
3.9	Struttura generale dello script di training	49
3.9.1	Caricamento del dataset e salto del training	50
3.9.2	Normalizzazione delle bounding box	51
3.10	Creazione dei datastore MATLAB	51

3.10.1	Verifica visiva del dataset prima del training (preview) . . .	52
3.11	Suddivisione Train / Validation	52
3.11.1	Strategie di suddivisione del dataset	52
3.11.2	Creazione del detector Tiny pre-addestrato	53
3.12	Impostazione delle metriche e delle opzioni di training	53
3.12.1	mAP@0.5 come metrica	53
3.12.2	Scelte di training	55
3.13	Training del detector e valutazione delle prestazioni	57
3.14	Risultati finali	62
3.14.1	Valutazione YOLO-V4	62
3.14.2	Valutazione YOLOX	63
4	Riconoscimento posizione e distanza	65
4.1	Ruolo del riconoscimento nella pipeline	65
4.1.1	Gestione dell'ambiente di esecuzione	65
4.1.2	Raffinamento delle bounding box	66
4.1.3	Limiti intrinseci della localizzazione	67
4.1.4	Segmentazione e adattamento alla geometria	68
4.2	Il raffinamento delle bounding box utilizzato	68
4.2.1	Principio di base	69
4.2.2	Pulizia morfologica	69
4.2.3	Margine di sicurezza e controllo vincoli	70
4.2.4	Significato per la pipeline RGB-D	70
4.3	Calibrazione del piano	71
4.3.1	Principi teorici	72
4.4	Dalla mappa depth alla nuvola di punti	72
4.5	Inclinazione dell'asse Z - TILT	75
4.5.1	Definizione geometrica dell'angolo di tilt	75
4.5.2	Effetto del tilt sulle coordinate senza compensazione	76
4.6	Dal riferimento di piano alla calibrazione RGB-Depth	77
4.7	Calibrazione manuale RGB-Depth	78
4.7.1	Trasformazione affine RGB→Depth	79

4.8	Limiti del modello affine singolo	80
4.9	Modello dual-affine con blending	80
4.9.1	Difetti e limiti della calibrazione manuale	81
4.10	Calibrazione automatica RGB–Depth	82
4.10.1	Architettura del flusso software	82
4.10.2	Acquisizione automatica e allineamento	83
4.10.3	Conversione della profondità	84
4.10.4	Integrazione con la rete neurale	84
4.10.5	Aspetti di robustezza e limiti	85
4.11	Confronto tra: manuale e automatica	86
5	Interfaccia Grafica	87
5.1	Struttura generale dell’interfaccia	88
5.1.1	Area di visualizzazione delle immagini	89
5.1.2	Area riconoscimento pezzi	90
5.2	Sezione “Dati Immagine”	91
5.2.1	Selettore di visualizzazione RGB/DEPTH	93
5.2.2	Selezione del modello di rete neurale	94
5.2.3	Pulsante “Calibra RGB–DEPTH”	94
5.3	Sezione: accesso, stato e supervisione	95
5.3.1	Sezione “Impostazioni”	96
5.3.2	Sezione “Stato”	96
5.4	Sezione “Controllo altezze” e logica dei LED	98
5.5	Gestione dell’errore e robustezza d’uso	99
5.6	Dal riconoscimento alla misura	99
5.7	Tolleranze e decisione binaria	100
5.8	Considerazioni finali	100
	Conclusioni	101
A	Paradigma <i>anchor-free</i> nella object detection	105
A.1	Introduzione	105
A.2	Approccio <i>anchor-based</i> : richiamo concettuale	106

A.3	Definizione del paradigma <i>anchor-free</i>	106
A.4	Significato operativo del paradigma	107
A.5	Collegamento con YOLOX	107
A.6	Vantaggi e limiti	108
A.7	Conclusione	108
B	Funzione di perdita composita in YOLOX	111
B.1	Introduzione	111
B.2	Struttura generale della loss	111
B.3	Termine di classificazione	112
B.4	Termine di regressione del bounding box	113
B.5	Termine di objectness (confidenza)	113
B.6	Interpretazione multi-task della loss	114
B.7	Ruolo dei coefficienti di pesatura	114
B.8	Osservazione sulle varianti implementative	114
B.9	Conclusione	115
C	Trasformazione di <i>shear</i> nel modello affine	117
C.1	Definizione e significato geometrico	117
C.2	Shear orizzontale e verticale in forma esplicita	117
C.3	Forma matriciale (coordinate omogenee)	118
C.4	Shear all'interno di una trasformazione affine generale	118
C.5	Interpretazione nel mapping RGB→Depth	118
D	Funzione <i>smoothstep</i> per il blending nel modello dual-affine	121
D.1	Contesto: blending morbido tra due trasformazioni affini	121
D.2	Normalizzazione della coordinata e funzione di clipping	122
D.3	Definizione della funzione <i>smoothstep</i> usata nel programma	122
D.4	Blending finale tra i due modelli affini	123
D.5	Parametri del programma: x_{split} e w_b	123
D.6	Osservazioni pratiche e possibili limiti	124
E	Funzione di <i>clipping</i> e variabile normalizzata t nel blending dual-affine	125

E.1	Contesto: perché servono t e <code>clip</code> nel blending	125
E.2	Definizione della fascia di blending	126
E.3	Costruzione della variabile normalizzata t	126
E.4	Funzione <code>clip</code> : definizione e interpretazione	127
E.5	Dal parametro t al peso di blending w : collegamento con <i>smoothstep</i>	127
E.6	Blending finale tra le due trasformazioni affini	128
E.7	Nota conclusiva	128

Bibliografia	131
---------------------	------------

Sommario

Il progetto mira allo sviluppo di una pipeline integrata per l'acquisizione 3D della scena, l'individuazione automatica del pezzo e la stima affidabile delle sue dimensioni geometriche principali.

Il sistema sviluppato combina l'utilizzo di una rete neurale profonda per il riconoscimento del pezzo con algoritmi di elaborazione geometrica basati sulle informazioni di profondità fornite dal sensore. Particolare attenzione è stata dedicata all'integrazione delle diverse componenti software e hardware, al fine di ottenere una soluzione coerente e facilmente estendibile.

A completamento del sistema, è stata realizzata un'interfaccia grafica che consente all'operatore di supervisionare il processo, visualizzare i risultati dell'elaborazione e interagire in modo intuitivo con il sistema. L'approccio adottato mira ad avvicinare il prototipo sviluppato a un contesto applicativo di tipo industriale, garantendo al contempo semplicità d'uso e controllo del flusso operativo.

Nel complesso, il lavoro dimostra come l'integrazione di sensori RGB-D, reti neurali e strumenti di analisi tridimensionale possa costituire una soluzione efficace per applicazioni di visione artificiale orientate al riconoscimento e alla misura senza contatto.

Introduzione

Negli ultimi anni, la visione artificiale ha assunto un ruolo sempre più centrale nei contesti industriali, in particolare nei processi di ispezione, controllo qualità e misura dimensionale senza contatto. L'evoluzione dei sensori tridimensionali e delle tecniche di intelligenza artificiale ha reso possibile lo sviluppo di sistemi in grado non solo di acquisire informazioni geometriche dettagliate, ma anche di interpretarle automaticamente, riducendo l'intervento umano e aumentando ripetibilità ed efficienza dei processi produttivi. In questo contesto si colloca il presente lavoro di tesi, che affronta il problema del riconoscimento e della misura di componenti meccanici mediante l'integrazione di una telecamera RGB-D e di una rete neurale profonda.

L'obiettivo del progetto è la realizzazione di un sistema completo di visione artificiale capace di acquisire immagini e informazioni di profondità, riconoscere automaticamente la presenza di un pezzo all'interno della scena e ricavarne le principali dimensioni geometriche in modo affidabile e ripetibile. Il sistema è stato progettato con particolare attenzione alla sua aderenza a uno scenario industriale reale, includendo non solo lo sviluppo degli algoritmi di elaborazione, ma anche la realizzazione di un pannello operatore che consenta la supervisione del processo e la visualizzazione dei risultati.

Il lavoro è strutturato in sei capitoli, ciascuno dei quali affronta un aspetto specifico del sistema sviluppato.

Nel Capitolo 1, viene presentato lo stato dell'arte relativo ai sensori RGB-D, alle tecniche di riconoscimento basate su deep learning e ai metodi di misura tridimensionale. Il capitolo fornisce il contesto teorico e tecnologico di riferimento, evidenziando i principali approcci presenti in letteratura e motivando le scelte

progettuali adottate nel corso della tesi. Nello stesso capitolo viene inoltre definito in modo formale l'obiettivo del progetto, chiarendo i requisiti funzionali del sistema e le finalità applicative del lavoro svolto.

Il Capitolo 2 è dedicato alla descrizione della telecamera RGB-D utilizzata, la Orbbec Femto Bolt. In questo capitolo vengono illustrati i principi di funzionamento del sensore, le caratteristiche tecniche principali e le modalità di acquisizione delle immagini a colori e delle mappe di profondità. Particolare attenzione è rivolta alla corretta interpretazione del dato di profondità e ai vincoli geometrici imposti dal sistema di acquisizione, elementi fondamentali per le successive fasi di elaborazione tridimensionale.

Nel Capitolo 3 viene introdotta la rete neurale utilizzata per il riconoscimento automatico del pezzo. Viene descritta l'architettura scelta, le motivazioni alla base della sua adozione e le modalità di integrazione all'interno del sistema complessivo. Il capitolo affronta inoltre il processo di addestramento della rete e le principali considerazioni legate alla scelta del dataset e delle metriche di valutazione. Alla fine del capitolo, saranno presenti dei commenti sui primi risultati della rete.

Il Capitolo 4 descrive le tecniche adottate per il riconoscimento della posizione del pezzo e per la stima delle distanze e delle dimensioni geometriche. A partire dall'output della rete neurale e dalle informazioni di profondità fornite dalla telecamera, vengono illustrati i metodi di ricostruzione tridimensionale, di stima del piano di riferimento e di calcolo delle misure dimensionali. Questo capitolo rappresenta il collegamento tra il riconoscimento basato su deep learning e l'analisi metrologica tridimensionale.

Il Capitolo 5 è dedicato all'interfaccia grafica, ovvero al pannello operatore sviluppato. Viene descritto il ruolo dell'interfaccia come strumento di controllo e supervisione del sistema, illustrandone le funzionalità principali, le modalità di interazione con l'utente e la gestione del flusso operativo. L'obiettivo di questo capitolo è mostrare come il sistema sviluppato possa essere utilizzato in modo pratico e intuitivo, avvicinandosi alle esigenze tipiche di un contesto industriale.

Infine, vengono tratte le conclusioni del lavoro svolto. Nella sezione delle conclusioni vengono riassunti i risultati ottenuti, analizzati i punti di forza e i limiti

del sistema e discusse le possibili evoluzioni future del progetto, sia dal punto di vista algoritmico sia applicativo.

Nel complesso, la struttura della tesi riflette l'approccio adottato nello sviluppo del progetto, che combina aspetti teorici, implementativi e applicativi con l'obiettivo di realizzare un sistema di visione artificiale avanzato, robusto e orientato a scenari reali di utilizzo.

Capitolo 1

Stato dell'Arte e obiettivo della tesi

1.1 Sensori RGB-D e acquisizione di informazioni

Negli ultimi anni, l'impiego di sensori RGB-D si è diffuso in numerosi ambiti applicativi, tra cui visione artificiale industriale, robotica, e controllo qualità. A differenza delle telecamere 2D tradizionali, i sensori RGB-D forniscono, oltre all'informazione cromatica, una stima della profondità per ciascun pixel dell'immagine, consentendo una ricostruzione tridimensionale della scena osservata. Tale caratteristica risulta fondamentale quando è richiesto il passaggio da misure puramente bidimensionali (in pixel) a misure metriche espresse in unità fisiche, come millimetri o metri.

1.1.1 Panorama sulle tecnologie per la misura della profondità

La misura della profondità (depth) in una scena può essere ottenuta con approcci molto diversi, che si distinguono principalmente in:

- **metodi passivi:** non richiedono l'emissione di un segnale verso l'ambiente

- **metodi attivi:** illuminano o “sondano” la scena con un segnale e ne misurano la risposta

La scelta della tecnologia dipende dal compromesso tra accuratezza, risoluzione spaziale, robustezza alle condizioni ambientali, range operativo, costo e complessità di integrazione. Tra i metodi **passivi**, la famiglia più comune è la **visione stereo**. Due (o più) camere osservano la scena da punti di vista differenti; la profondità si ricava stimando la disparità tra punti corrispondenti nelle immagini. Questo approccio è concettualmente semplice e non richiede sorgenti di illuminazione dedicate, ma la qualità della stima dipende fortemente dalla presenza di texture (pattern visivi) e da condizioni favorevoli di illuminazione [1]. Superfici uniformi, riflettenti o trasparenti possono rendere difficoltosa la corrispondenza dei punti e introdurre errori. Inoltre, la precisione decresce all'aumentare della distanza e richiede una buona calibrazione del sistema e una geometria stereo adeguata (baseline).

Nei metodi **attivi** rientra la **luce strutturata** (Structured Light), in cui un proiettore emette un pattern (tipicamente infrarosso) sulla scena; una camera osserva la deformazione del pattern e ricostruisce la profondità tramite triangolazione. Questa tecnologia può produrre mappe dense e accurate a breve raggio, ed è stata storicamente popolare per applicazioni indoor. Tuttavia, può soffrire in presenza di luce solare intensa (che “satura” l'infrarosso), su materiali con scarsa diffusività (speculari) e in scenari con più sensori vicini (interferenze tra pattern).

Un'altra classe molto diffusa è il **Time-of-Flight** (ToF) ottico, in cui la profondità si ottiene misurando il tempo di percorrenza della luce (o, più spesso, lo sfasamento di un segnale luminoso modulato). Esistono dispositivi direct ToF (impulsi e misura del tempo di volo) e indirect ToF (misura di fase). I sensori ToF offrono tipicamente una stima diretta della distanza lungo la linea di vista (asse ottico) e una mappa di profondità densa allineata al piano immagine, risultando efficaci per tracking e ricostruzione 3D in tempo reale. Le principali criticità includono errori dovuti a multi-path interference (riflessioni multiple), sensibilità alla riflettanza dei materiali, rumore crescente con la distanza e possibili interferenze

tra più dispositivi ToF operanti nello stesso ambiente.

Accanto ai ToF “a matrice” (ad esempio la Femto Bolt), si colloca il LiDAR (Light Detection and Ranging), che usa impulsi laser e misura il tempo di ritorno per ottenere distanze. Il LiDAR può essere a scansione (meccanica o MEMS) o solid-state e produce tipicamente una nuvola di punti più che una mappa densa per pixel [2].

È molto usato in robotica e automotive per la robustezza sul range medio-lungo, ma può avere densità spaziale inferiore rispetto a una RGB-D e costi più elevati. Anche qui, superfici altamente riflettenti o trasparenti possono generare misure degradate, e l’illuminazione ambientale può incidere in alcune implementazioni. Infine, per applicazioni a corto raggio e a basso costo, si usano sensori ultrasuoni, che stimano la distanza dal tempo di volo di onde acustiche. Sono semplici e robusti, ma con bassa risoluzione angolare, sensibilità alla geometria delle superfici e difficoltà nel ricostruire dettagli fini; per questo sono spesso impiegati come sensori di prossimità più che per percezione 3D completa.

In sintesi, nessuna tecnologia è universalmente “migliore”: le soluzioni ottiche (stereo, luce strutturata, ToF, LiDAR) privilegiano la ricchezza geometrica e l’allineamento con l’immagine, mentre le soluzioni radio/acustiche privilegiano robustezza e affidabilità in ambienti ostili, a scapito del dettaglio. In applicazioni industriali e di metrologia leggera, i sensori RGB-D (in particolare ToF e alcune varianti attive) rappresentano spesso un compromesso efficace tra densità del dato, semplicità di integrazione e prestazioni in tempo reale.

1.2 Principio fisico del Time of Flight

Il funzionamento della Femto Bolt si basa sul principio del Time of Flight [3], una tecnica di misura che sfrutta la velocità finita della luce per determinare la distanza tra il sensore e gli oggetti presenti nella scena. Il concetto alla base è relativamente semplice: la telecamera emette un segnale luminoso, tipicamente nella banda dell'infrarosso, che si propaga nello spazio, colpisce gli oggetti e viene parzialmente riflesso verso il sensore. Misurando il tempo impiegato dal segnale per compiere questo percorso di andata e ritorno, è possibile calcolare la distanza dell'oggetto dal sensore.

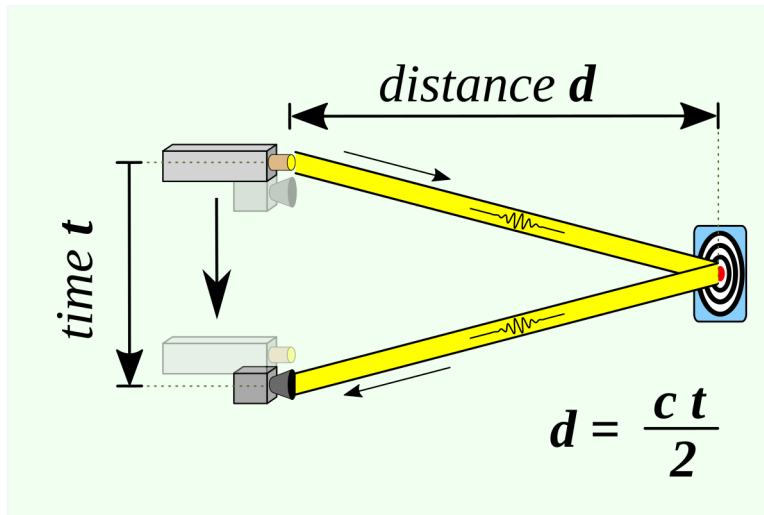


Figura 1.1: Funzionamento TOF (fonte: [4])

Dal punto di vista matematico, la relazione che lega il tempo di volo alla distanza è espressa dalla formula:

$$d = \frac{c \cdot \Delta t}{2} \quad (1.1)$$

dove d rappresenta la distanza tra il sensore e il punto riflettente, c è la velocità della luce nel vuoto e Δt è il tempo totale impiegato dal segnale per raggiungere l'oggetto e tornare al sensore. Il fattore di divisione per due è necessario poiché il tempo misurato include sia il percorso di andata sia quello di ritorno.

Nel caso della Femto Bolt, questa misura non viene effettuata su un singolo punto, ma simultaneamente su tutti i pixel del sensore. Ciò significa che ogni elemento fotosensibile calcola in modo indipendente la distanza del punto della scena che proietta su di esso, permettendo di ottenere in un'unica acquisizione una mappa completa delle profondità.

1.3 Riconoscimento del pezzo: da 2D a RGB-D

Il riconoscimento automatico di un pezzo all'interno di una scena rappresenta una fase fondamentale nei sistemi di visione artificiale industriale, poiché costituisce il punto di partenza per tutte le elaborazioni successive, come la misura dimensionale, l'ispezione qualitativa o la manipolazione robotica. Tradizionalmente, questo problema è stato affrontato mediante tecniche di visione bidimensionale, basate esclusivamente su immagini RGB o in scala di grigi. In tali approcci, l'identificazione del pezzo avveniva tramite l'estrazione di caratteristiche manuali, come contorni, soglie di intensità o descrittori locali, seguita da regole decisionali definite a priori. Sebbene efficaci in contesti controllati, questi metodi risultano fortemente sensibili a variazioni di illuminazione, rumore, occlusioni parziali e cambiamenti nell'aspetto superficiale del pezzo.

L'introduzione delle reti neurali profonde, in particolare delle reti convoluzionali, ha segnato un netto miglioramento nello stato dell'arte del riconoscimento degli oggetti. Le moderne architetture di object detection consentono di apprendere automaticamente le caratteristiche rilevanti direttamente dai dati, superando i limiti dei descrittori manuali. Queste reti sono in grado di riconoscere e localizzare uno o più oggetti all'interno di un'immagine, fornendo in output informazioni di alto livello, come bounding box e classi di appartenenza. In ambito industriale, tale capacità si traduce in una maggiore robustezza del sistema, anche in presenza di condizioni operative non ideali.

Nel contesto dei sistemi RGB-D [5], il riconoscimento del pezzo può essere affrontato secondo diverse strategie. Un primo approccio consiste nell'utilizzare esclusivamente l'informazione bidimensionale, tipicamente l'immagine RGB, per il riconoscimento, demandando l'uso della profondità alle fasi successive di ela-

borazione geometrica. Questo approccio è ampiamente diffuso, poiché la maggior parte delle reti neurali di object detection è stata progettata e ottimizzata per immagini 2D. L'immagine RGB fornisce infatti informazioni semantiche ricche, come colore, texture e pattern, che risultano particolarmente discriminanti per il riconoscimento.

Un secondo approccio prevede l'utilizzo diretto della mappa di profondità come input della rete neurale. In questo caso, il riconoscimento si basa su informazioni puramente geometriche, come la forma e la disposizione spaziale del pezzo. Tale strategia può risultare vantaggiosa in scenari caratterizzati da scarsa variabilità cromatica o da illuminazione non controllata. Tuttavia, l'assenza di informazioni semantiche [6] — intese come indizi legati al significato visivo dell'oggetto e al suo contesto, tipicamente ricavati dal canale RGB o da annotazioni/feature apprese su texture e parti distintive — rende spesso più complesso il processo di addestramento e può ridurre la capacità discriminante della rete in presenza di oggetti geometricamente simili. Esempi di informazioni semantiche includono la presenza di loghi o scritte, pattern superficiali (graffi, zigrinature, trame), colori o marcature funzionali, la distinzione di sotto-componenti riconoscibili (fori, etichette, zone lavorate), oppure il contesto di scena (ad esempio l'oggetto posizionato su un certo supporto o vicino a specifici elementi), tutti elementi che possono aiutare a separare classi che in profondità risultano quasi indistinguibili.

Una terza strategia, sempre più diffusa nello stato dell'arte, consiste nella fusione delle informazioni RGB e depth. In questi approcci, la rete neurale elabora congiuntamente informazioni bidimensionali e tridimensionali, beneficiando sia della ricchezza semantica dell'RGB sia dei vincoli geometrici forniti dalla profondità. Sebbene questa soluzione possa migliorare la robustezza complessiva del sistema, essa comporta un aumento della complessità architetturale e computazionale, rendendo talvolta più difficile l'integrazione in contesti industriali real-time.

1.4 Oltre il riconoscimento: stima della posa e misure 3D

Una volta riconosciuto e localizzato il pezzo all'interno della scena, molte applicazioni avanzate richiedono una descrizione più completa della sua configurazione spaziale. In particolare, la stima della posa di un oggetto, ovvero la descrizione della sua posizione e del suo orientamento nello spazio, rappresenta un problema centrale in ambiti quali la manipolazione robotica, l'assemblaggio automatico e la visione artificiale avanzata. La posa consente di descrivere in modo completo la relazione tra l'oggetto e il sistema di riferimento della camera o dell'ambiente.

Lo stato dell'arte propone numerosi metodi per la stima della posa a partire da dati RGB-D, (la determinazione della posa di oggetti rigidi da un singolo input RGB-D è stata formalizzata anche nel benchmark BOP (*Benchmark for 6D Object Pose Estimation*) [7]. Alcuni approcci si basano sull'utilizzo di modelli geometrici o CAD noti dell'oggetto e sulla successiva ricerca della trasformazione rigida che meglio allinea il modello alla nuvola di punti osservata. Altri metodi, più recenti, sfruttano reti neurali profonde in grado di stimare direttamente la posa a partire dall'immagine o dalla combinazione di immagine e profondità. Sebbene queste soluzioni abbiano mostrato risultati promettenti in contesti di ricerca, esse risultano spesso complesse da implementare e da validare in applicazioni industriali, soprattutto quando sono richiesti elevati livelli di affidabilità metrologica.

In alcune applicazioni pratiche, invece, l'obiettivo principale non è la stima completa della posa, bensì la misura accurata di alcune dimensioni geometriche fondamentali del pezzo, come lunghezza, larghezza e altezza. In questi casi, la letteratura evidenzia come sia possibile ottenere misure affidabili senza ricorrere a una posa completa, adottando invece un approccio basato sulla ricostruzione tridimensionale locale e sull'analisi geometrica della nuvola di punti. Questo approccio risulta particolarmente adatto quando il pezzo è posizionato su un piano di riferimento noto, come un piano di appoggio.

Un elemento centrale di questa metodologia è la stima del piano di riferimento, che consente di separare il pezzo dallo sfondo e di definire un sistema di riferimento coerente per la misura. La stima del piano viene generalmente effettuata tramite algoritmi di fitting robusto, capaci di individuare il piano anche in presenza di rumore e outlier (anomalia). Una volta definito il piano, è possibile calcolare l'altezza del pezzo come distanza dei suoi punti tridimensionali dal piano stesso, ottenendo una misura direttamente interpretabile in termini metrici.

Per quanto riguarda le dimensioni in pianta, la distribuzione dei punti appartenenti al pezzo può essere analizzata per stimare bounding box tridimensionali. L'utilizzo di bounding box orientate secondo le direzioni principali dell'oggetto consente di ridurre l'errore di misura in presenza di rotazioni rispetto al sistema di riferimento della camera. Questo tipo di analisi geometrica permette di ottenere una descrizione compatta ma significativa della forma del pezzo, sufficiente per molte applicazioni di controllo qualità e ispezione dimensionale.

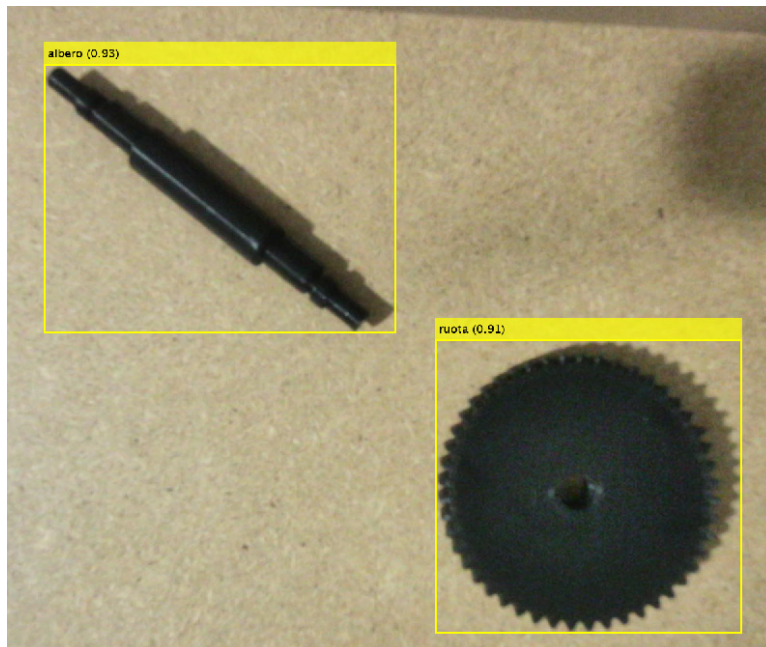


Figura 1.2: Esempio bounding box

Nel progetto sviluppato, si è scelto di adottare un approccio che privilegia la combinazione tra riconoscimento automatico e misura geometrica esplicita, evi-

tando una stima end-to-end della posa tramite rete neurale. La rete neurale viene utilizzata esclusivamente per individuare la regione di interesse, mentre la stima delle dimensioni avviene mediante modelli geometrici deterministici applicati ai dati tridimensionali.

1.5 Reti neurali artificiali e deep learning

Negli ultimi decenni, le reti neurali artificiali hanno assunto un ruolo sempre più centrale nello sviluppo di sistemi intelligenti per l'elaborazione dei dati. In particolare, l'avvento del *deep learning* ha reso possibile affrontare in modo efficace problemi complessi di percezione visiva, superando i limiti delle tecniche tradizionali basate su modelli analitici e sull'estrazione manuale di caratteristiche.

Nel contesto della visione artificiale industriale, le reti neurali rappresentano oggi uno strumento fondamentale per il riconoscimento automatico di oggetti, la classificazione di difetti e la localizzazione di componenti all'interno di una scena, grazie alla loro capacità di apprendere rappresentazioni robuste direttamente dai dati [8].

1.5.1 Neurone artificiale e modello di rete

Il concetto di neurone artificiale costituisce l'unità fondamentale delle reti neurali. Ciascun neurone realizza una combinazione lineare degli ingressi, seguita dall'applicazione di una funzione di attivazione non lineare. Questa struttura consente alla rete di approssimare funzioni complesse e di modellare relazioni non lineari tra le variabili di input e output.

Una rete neurale è ottenuta collegando un elevato numero di neuroni in una struttura a strati. L'informazione fluisce tipicamente dallo strato di ingresso verso lo strato di uscita secondo un'architettura *feed-forward*. Durante la fase di addestramento, i pesi delle connessioni vengono aggiornati tramite algoritmi di ottimizzazione basati sul gradiente, come la retropropagazione dell'errore [9].

1.5.2 Paradigmi di apprendimento

L'addestramento delle reti neurali può avvenire secondo diversi paradigmi. Nell'apprendimento supervisionato, largamente utilizzato nelle applicazioni industriali, la rete apprende a partire da esempi etichettati, minimizzando una funzione di perdita che misura l'errore tra l'output previsto e quello desiderato. Questo approccio consente di ottenere modelli controllabili e facilmente validabili.

Altri paradigmi includono l'apprendimento non supervisionato, finalizzato all'individuazione di strutture latenti nei dati, e l'apprendimento per rinforzo, in cui la rete apprende interagendo con un ambiente. Tuttavia, nel contesto del riconoscimento di oggetti e della visione artificiale, l'apprendimento supervisionato rimane l'approccio di riferimento [9].

1.5.3 Reti neurali profonde

Con il termine *reti neurali profonde* si indicano modelli caratterizzati da un numero elevato di strati nascosti. L'aumento della profondità consente alla rete di apprendere rappresentazioni gerarchiche dei dati: gli strati più superficiali estraggono caratteristiche semplici, mentre quelli più profondi combinano tali informazioni in descrittori sempre più astratti.

Questo meccanismo risulta particolarmente efficace nel trattamento di immagini, in cui la struttura spaziale e la composizione gerarchica delle forme giocano un ruolo fondamentale. Le reti profonde hanno dimostrato prestazioni superiori rispetto ai metodi tradizionali in numerosi benchmark di visione artificiale [8].

1.5.4 Reti neurali convoluzionali

Le reti neurali convoluzionali (CNN) rappresentano una classe di reti profonde specificamente progettata per l'elaborazione di immagini. Esse introducono l'operazione di convoluzione, che consente di estrarre automaticamente caratteristiche locali sfruttando la correlazione spaziale tra i pixel.

Le CNN presentano due vantaggi fondamentali:

1. la riduzione del numero di parametri rispetto alle reti completamente connesse

2. maggiore robustezza alle traslazioni e alle variazioni locali dell'immagine

Grazie a queste proprietà, le reti convoluzionali sono diventate lo standard per applicazioni quali classificazione di immagini, segmentazione e riconoscimento di oggetti [10].

1.5.5 Object detection: riconoscimento e localizzazione

L'object detection rappresenta un'evoluzione rispetto alla semplice classificazione, poiché richiede non solo di riconoscere la presenza di un oggetto, ma anche di determinarne la posizione all'interno dell'immagine. Dal punto di vista concettuale, questo problema combina due compiti distinti: la classificazione e la regressione delle coordinate spaziali.

Le prime soluzioni di object detection si basavano su pipeline complesse, con una separazione netta tra estrazione delle regioni di interesse e classificazione. Con l'avvento del deep learning, sono state sviluppate architetture end-to-end in grado di risolvere il problema in modo integrato, migliorando significativamente accuratezza ed efficienza [11].

1.5.6 Approccio one-stage e famiglie YOLO

Tra le architetture di object detection, un ruolo di rilievo è ricoperto dai modelli *one-stage*, nei quali la detection viene formulata come un problema di regressione diretta dall'immagine di input alle bounding box e alle classi degli oggetti. Questo approccio è stato reso popolare dalla famiglia di modelli YOLO (*You Only Look Once*) [12].

Le architetture YOLO si distinguono per l'elevata velocità di inferenza e per la capacità di analizzare l'immagine in modo globale, evitando la generazione esplicita di region proposal. Le successive evoluzioni della famiglia YOLO hanno introdotto miglioramenti significativi in termini di accuratezza, stabilità del training e capacità di generalizzazione [13].

1.5.7 Verso YOLOX

YOLOX rappresenta una delle evoluzioni più recenti delle architetture YOLO e introduce scelte progettuali mirate a migliorare ulteriormente le prestazioni dei modelli one-stage. In particolare, YOLOX adotta un approccio *anchor-free* e una separazione dei rami di classificazione e regressione, riducendo l'interferenza tra i due compiti durante l'addestramento [14].

Dal punto di vista teorico, queste scelte rendono YOLOX particolarmente adatto a contesti applicativi industriali, in cui è richiesta una localizzazione accurata degli oggetti anche in presenza di dataset di dimensioni contenute. L'architettura consente inoltre una più semplice integrazione con pipeline di elaborazione geometrica, come quelle necessarie per la misura tridimensionale basata su sensori RGB-D.

Nel capitolo successivo verrà descritta nel dettaglio l'architettura YOLOX adottata, le motivazioni della scelta della versione *Tiny* e la sua integrazione in ambiente MATLAB per il riconoscimento automatico dei pezzi meccanici.

1.6 Rete neurale utilizzata: YOLOX in ambiente MATLAB

Prima di introdurre l'architettura adottata nel presente lavoro, è opportuno fornire una panoramica generale delle principali famiglie di reti neurali utilizzabili in ambiente MATLAB per l'elaborazione di dati visivi. MATLAB, attraverso il Deep Learning Toolbox, mette infatti a disposizione un insieme articolato di strumenti per la progettazione, l'addestramento e l'inferenza di modelli neurali, supportando diversi paradigmi architetturali consolidati in letteratura.

Le reti neurali convoluzionali (Convolutional Neural Networks, CNN) rappresentano il riferimento principale per l'analisi di immagini. Esse si basano sull'applicazione di filtri convoluzionali che consentono di estrarre automaticamente caratteristiche gerarchiche a partire dai dati in ingresso, passando da rappresentazioni locali a descrizioni sempre più astratte. Questa famiglia include modelli destinati alla classificazione, alla regressione e, soprattutto, alla comprensione

della scena visiva. In MATLAB, tali architetture costituiscono il nucleo delle applicazioni di visione artificiale e sono supportate sia tramite modelli pre-addestrati sia attraverso strumenti per la progettazione personalizzata.

Un'estensione naturale delle CNN è rappresentata dalle reti per la segmentazione semantica e istanza, nelle quali l'obiettivo non è soltanto classificare un'intera immagine, ma assegnare una classe a ciascun pixel oppure distinguere le singole occorrenze degli oggetti. Queste architetture risultano particolarmente rilevanti quando è richiesta una comprensione dettagliata della scena, ad esempio per applicazioni di ispezione o analisi metrologica avanzata. MATLAB offre supporto a tali approcci mediante framework dedicati che permettono di gestire dati annotati a livello pixel-wise.

Un'altra famiglia di rilievo è quella delle reti per object detection, che si collocano a metà strada tra classificazione e segmentazione. In questo caso, il modello è chiamato a individuare la presenza di oggetti e a determinarne la posizione nello spazio dell'immagine, tipicamente mediante bounding box. All'interno di questa categoria si distinguono due principali approcci: i metodi a due stadi, basati su una fase preliminare di generazione di regioni candidate seguita da una classificazione, e i metodi a singolo stadio, nei quali la localizzazione e il riconoscimento avvengono simultaneamente. MATLAB supporta entrambe le strategie, consentendo di scegliere la soluzione più adeguata in funzione dei requisiti di accuratezza e prestazioni computazionali.

Accanto alle architetture strettamente dedicate alla visione, MATLAB include anche strumenti per l'impiego di reti neurali ricorrenti (Recurrent Neural Networks, RNN) e varianti più evolute, utilizzate principalmente per l'elaborazione di dati sequenziali. Sebbene queste non rappresentino la scelta primaria per l'analisi di immagini statiche, esse trovano applicazione in contesti in cui l'informazione temporale risulta significativa, come nel caso di sequenze video o segnali acquisiti nel tempo.

Infine, è opportuno menzionare le architetture ibride e i modelli basati su trasferimento di apprendimento (transfer learning), che consentono di riutilizzare conoscenze apprese su grandi dataset per adattarle a problemi specifici con quantità limitate di dati. Questo approccio è ampiamente supportato in MATLAB e

risulta particolarmente vantaggioso in ambito industriale, dove la disponibilità di dati annotati può essere ridotta.

Alla luce di questa panoramica, risulta evidente come MATLAB offra un ecosistema completo per l'implementazione di diverse famiglie di reti neurali, permettendo di selezionare di volta in volta l'architettura più idonea in funzione del problema applicativo considerato. In questo contesto si inserisce la scelta effettuata nel presente lavoro, descritta nel seguito, relativa all'impiego di una rete appartenente alla famiglia degli object detector a singolo stadio.

Nel presente lavoro di tesi, il riconoscimento automatico del pezzo all'interno delle immagini acquisite è stato affrontato mediante l'impiego di una rete neurale per object detection. In particolare, è stata adottata l'architettura YOLOX (You Only Look Once – eXtended), implementata e utilizzata in ambiente MATLAB. La scelta di questa architettura è il risultato di una valutazione attenta tra accuratezza, velocità di inferenza, semplicità di integrazione e robustezza complessiva del sistema, aspetti fondamentali in un contesto applicativo di tipo industriale. (Nel Capitolo 3 viene descritta in modo approfondito l'architettura del sistema.)

YOLOX appartiene alla famiglia delle reti di object detection a singolo stadio (single-stage detector), nelle quali il riconoscimento e la localizzazione degli oggetti avvengono in un'unica fase di elaborazione. Questo approccio si distingue dai metodi a due stadi, come quelli basati su region proposal, per una maggiore efficienza computazionale e una latenza ridotta, caratteristiche particolarmente rilevanti quando si desidera un sistema reattivo e potenzialmente utilizzabile in tempo reale. Nel caso in esame, sebbene il tempo reale non costituisca un vincolo stringente, l'efficienza dell'algoritmo rappresenta comunque un requisito importante per garantire scalabilità e possibilità di estensione futura del sistema.

Nel sistema sviluppato, la rete YOLOX svolge il ruolo di primo stadio della pipeline di elaborazione. A partire dall'immagine acquisita dalla telecamera Femto Bolt, la rete ha il compito di determinare se il pezzo di interesse è presente nella scena e, in caso affermativo, di fornirne una localizzazione approssimata tramite una bounding box. Questa informazione viene successivamente utilizzata per delimitare la regione di interesse sulla mappa di profondità, consentendo di concentrare l'elaborazione tridimensionale esclusivamente sulla porzione di scena

rilevante. Tale scelta progettuale consente di ridurre l'influenza dello sfondo, migliorare la robustezza delle misure e contenere il carico computazionale delle fasi successive.

È importante sottolineare che la rete neurale non viene utilizzata per stimare direttamente dimensioni metriche ma al contrario, essa è impiegata esclusivamente come strumento di riconoscimento e localizzazione semantica. Questa separazione tra riconoscimento basato su deep learning e misura basata su modelli geometrici deterministici è finalizzata a garantire una maggiore interpretabilità e affidabilità del sistema. In ambito metrologico e industriale, infatti, è spesso preferibile affidare la stima delle grandezze fisiche a modelli espliciti, piuttosto che a reti neurali end-to-end difficilmente validabili.

1.7 Addestramento della rete neurale

L'addestramento della rete neurale è stato affrontato secondo il paradigma del supervised learning, che rappresenta l'approccio di riferimento per problemi di riconoscimento e localizzazione in ambito di visione artificiale. In tale contesto, il modello apprende a partire da un insieme di dati etichettati, nei quali a ciascuna immagine è associata un'informazione esplicita relativa alla presenza e alla posizione dell'oggetto di interesse. Questo approccio è stato adottato anche nel caso in esame, come descritto nel seguito, e costituisce un elemento centrale per il raggiungimento delle prestazioni richieste dal sistema.

L'addestramento supervisionato della rete neurale rappresenta una fase cruciale del progetto, in quanto determina direttamente la capacità del modello di riconoscere il pezzo in modo robusto e ripetibile. Un modello adeguatamente addestrato è infatti in grado di generalizzare rispetto a variazioni nelle condizioni operative, quali cambiamenti nella posizione, nell'orientazione e nella distanza del pezzo rispetto alla telecamera. Per tale motivo, particolare attenzione è stata dedicata alla costruzione del dataset e alla definizione della procedura di training, aspetti che risultano determinanti nell'ambito del supervised learning.

Il dataset utilizzato è stato costruito a partire da immagini acquisite mediante la stessa telecamera impiegata nel sistema finale, garantendo così coerenza tra la

fase di addestramento e quella di utilizzo operativo. Le immagini sono state raccolte in condizioni diversificate, includendo variazioni realistiche della scena, al fine di evitare fenomeni di overfitting e favorire la capacità di generalizzazione del modello. In un contesto di apprendimento supervisionato, la qualità e la rappresentatività dei dati rivestono infatti un ruolo fondamentale, spesso più rilevante della complessità dell'architettura adottata.

Elemento caratterizzante del supervised learning è la presenza di annotazioni esplicite: nel presente lavoro, ogni immagine è stata etichettata manualmente mediante bounding box, indicanti la posizione del pezzo all'interno della scena. Questa operazione è stata svolta con particolare cura, in quanto errori sistematici nelle etichette possono compromettere il processo di apprendimento, inducendo il modello a convergere verso soluzioni subottimali. Il dataset è stato quindi suddiviso in insiemi distinti di addestramento, validazione e test, secondo una prassi consolidata che consente di monitorare l'andamento del training e valutare la capacità del modello di generalizzare su dati non visti.

Una volta completata la fase di addestramento supervisionato, la rete neurale è stata integrata nella pipeline di elaborazione. In fase di inferenza, il modello riceve in ingresso un'immagine e restituisce una bounding box che identifica la regione di interesse, replicando il comportamento appreso durante il training. Tale informazione viene successivamente utilizzata per estrarre la porzione rilevante della mappa di profondità, sulla quale vengono eseguite le elaborazioni tridimensionali. In questo modo, il supervised learning consente di demandare alla rete il compito di riconoscimento e localizzazione semantica, mantenendo separata la stima delle grandezze metriche, affidata a metodi geometrici deterministici.

L'addestramento della rete YOLOX, condotto secondo questo paradigma, ha permesso di ottenere un modello affidabile e coerente con gli obiettivi applicativi del progetto. Più in generale, l'esperienza evidenzia come, nell'ambito del supervised learning, la corretta costruzione del dataset e la qualità delle annotazioni rappresentino fattori determinanti per le prestazioni complessive del sistema. L'integrazione tra tecniche di deep learning supervisionato e metodi geometrici espliciti si configura quindi come una soluzione efficace, capace di coniugare flessibilità, robustezza e interpretabilità, in linea con le esigenze dei sistemi di visione

industriale e delle applicazioni metrologiche senza contatto.

1.8 Obiettivo del progetto

L'obiettivo del presente progetto è lo sviluppo di un sistema completo di visione artificiale in grado di riconoscere automaticamente componenti meccanici e di ricavarne le principali dimensioni geometriche mediante l'impiego di una telecamera RGB-D e di tecniche di deep learning. Il lavoro si propone di integrare in un'unica pipeline coerente le fasi di acquisizione dei dati, riconoscimento del pezzo, ricostruzione tridimensionale e analisi metrologica, ponendo particolare attenzione alla robustezza del sistema e alla sua aderenza a scenari applicativi di tipo industriale.

Il sistema utilizza una telecamera RGB-D Orbbec Femto Bolt per l'acquisizione sincrona di immagini a colori e mappe di profondità. La gestione diretta dell'hardware di acquisizione è demandata a un eseguibile esterno sviluppato in linguaggio C++, che si occupa del controllo della telecamera e della produzione dei dati necessari alle successive fasi di elaborazione.

Un aspetto centrale del progetto è la progettazione di un sistema che non si limiti alla sola sperimentazione algoritmica, ma che risulti utilizzabile e comprensibile anche dal punto di vista operativo. Per questo motivo, accanto allo sviluppo degli algoritmi di riconoscimento e misura, è stato realizzato un pannello operatore che funge da interfaccia di controllo e supervisione dell'intero sistema. Tale interfaccia rappresenta un elemento chiave per avvicinare il prototipo sviluppato a una reale applicazione industriale, consentendo all'utente di interagire in modo diretto con il sistema e di interpretarne facilmente i risultati. Il pannello consente inoltre all'utente di personalizzare dinamicamente la visualizzazione dei risultati, attivando o disattivando la visualizzazione delle bounding box, dei centri dei pezzi e delle informazioni metrologiche senza la necessità di ripetere l'elaborazione. È infine prevista la possibilità di esportare i dati ottenuti in formato CSV, rendendo il sistema compatibile con flussi di lavoro industriali, archiviazione dei risultati e sistemi di tracciabilità.

Nel complesso, l'obiettivo del progetto è dimostrare la fattibilità e l'efficacia di un sistema integrato che combina sensori RGB-D, reti neurali profonde e strumenti di analisi metrologica tridimensionale, supportato da un'interfaccia operatore progettata per garantire facilità d'uso, controllo del processo e robustezza operativa. Il sistema sviluppato si presta sia a contesti di ricerca applicata sia a possibili estensioni verso applicazioni industriali reali, costituendo una base solida per ulteriori sviluppi futuri.

Capitolo 2

Telecamera Orbbec Femto Bolt

2.1 Presentazione generale della camera

Negli ultimi anni, l'evoluzione dei sistemi di visione artificiale ha portato a una crescente diffusione di sensori in grado di acquisire informazioni tridimensionali dell'ambiente. In questo contesto si inserisce la telecamera Femto Bolt, sviluppata dall'azienda Orbbec, che rappresenta una soluzione avanzata per la percezione spaziale basata sulla tecnologia Time of Flight (ToF). Questo tipo di dispositivo è progettato per superare i limiti della visione bidimensionale tradizionale, fornendo non solo informazioni cromatiche o di intensità luminosa, ma anche dati metrici relativi alla distanza degli oggetti presenti nella scena.



Figura 2.1: Femto Bolt Orbbec (fonte: Orbbec [15])

La Femto Bolt rientra nella categoria delle telecamere di profondità attive, ovvero sistemi che emettono un segnale luminoso controllato e ne analizzano il comportamento dopo l'interazione con l'ambiente circostante. L'obiettivo principale di tali dispositivi è la generazione di una rappresentazione tridimensionale della scena osservata, indispensabile in applicazioni come la robotica, l'automazione industriale, la navigazione autonoma e l'interazione uomo-macchina.

Dal punto di vista concettuale, la necessità di acquisire informazioni di profondità nasce dal fatto che una normale immagine 2D descrive il mondo reale esclusivamente nel piano dell'immagine, perdendo completamente la dimensione spaziale associata alla distanza. Al contrario, un sistema di visione 3D permette di associare a ogni punto dell'immagine una coordinata di profondità, consentendo una ricostruzione geometrica molto più fedele della realtà fisica. La Femto Bolt risponde a questa esigenza utilizzando un principio di misura diretto e fisicamente fondato, ovvero il tempo di volo della luce.

2.2 Architettura e funzionamento

Dal punto di vista hardware, la Femto Bolt integra diversi sottosistemi che cooperano per realizzare il processo di acquisizione tridimensionale. Un ruolo centrale è svolto dall'emettitore di luce infrarossa, che proietta un segnale luminoso modulato verso la scena. L'utilizzo della banda infrarossa consente di evitare interferenze con la visione umana e di ridurre la dipendenza dalle condizioni di illuminazione ambientale.

Il sistema ottico della telecamera, composto da lenti calibrate, definisce il campo visivo e influisce direttamente sulla risoluzione spaziale e sulla precisione delle misure. L'intero processo di acquisizione è supportato da un'unità di elaborazione integrata, che si occupa di trasformare i segnali grezzi in dati strutturati, come mappe di profondità e nuvole di punti tridimensionali. I dati così ottenuti vengono infine trasmessi a un sistema esterno, tipicamente un computer o un'unità di controllo, attraverso interfacce di comunicazione ad alta velocità.

2.3 Mappa di profondità e rappresentazione tridimensionale

La principale informazione fornita dalla telecamera Femto Bolt è la mappa di profondità, che costituisce una rappresentazione bidimensionale in cui a ciascun pixel dell'immagine viene associata una misura di distanza reale tra il punto osservato e il sensore. A differenza di una comune immagine bidimensionale, nella quale il valore del pixel rappresenta un'intensità luminosa, nella mappa di profondità tale valore è direttamente correlato alla coordinata spaziale lungo la direzione di osservazione della camera. La mappa di profondità può quindi essere formalmente descritta come una funzione che associa a ogni coppia di coordinate del piano immagine $(\mathbf{X}, \mathbf{Y})$ una distanza $\mathbf{Z}$, ovvero $\mathbf{D}(\mathbf{X}, \mathbf{Y}) = \mathbf{Z}$.

Questa rappresentazione costituisce il primo livello di descrizione tridimensionale della scena osservata, ma risulta ancora intrinsecamente legata al sistema di coordinate dell'immagine. Per ottenere una descrizione spaziale completa è necessario introdurre il sistema di riferimento della telecamera e il modello geometrico che governa il processo di proiezione prospettica. Nel caso della Femto Bolt, il sistema di riferimento della camera ha origine nel centro ottico del sensore; gli assi $\mathbf{X}$ e $\mathbf{Y}$ giacciono sul piano immagine, rispettivamente lungo le direzioni orizzontale e verticale, mentre l'asse $\mathbf{Z}$ è orientato lungo la direzione di osservazione della telecamera, uscente al piano del sensore, come si vede in Figura 2.2. La coordinata $\mathbf{Z}$ rappresenta pertanto la distanza misurata dal sensore verso la scena.

Un aspetto rilevante nel caso specifico della Femto Bolt è che l'asse $\mathbf{Z}$ della telecamera non risulta perfettamente allineato con la verticale del sistema di riferimento globale, ma presenta un'inclinazione pari a 6° . Tale inclinazione ha un impatto diretto sull'interpretazione delle misure di profondità, in quanto le distanze fornite dal sensore sono riferite all'asse ottico della camera e non alla verticale reale dell'ambiente.

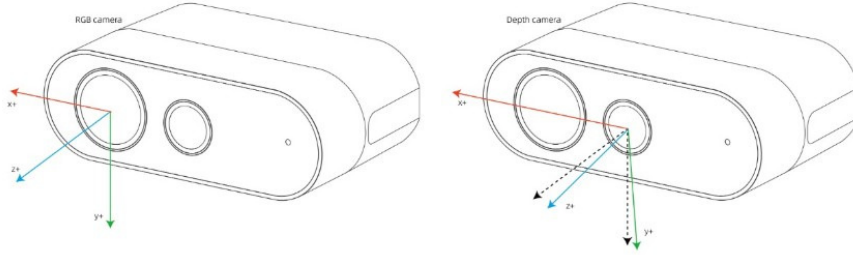


Figura 2.2: Sistema di riferimento Femto Bolt (fonte: Orbbec [15]).

Di conseguenza, per ottenere una rappresentazione tridimensionale coerente con la geometria del mondo reale, è necessario tenere conto di questa inclinazione nella fase di elaborazione dei dati.

Dal punto di vista matematico, ciò implica che le coordinate tridimensionali ottenute dalla trasformazione della mappa di profondità devono essere opportunamente corrette mediante una rotazione per essere espresse in un sistema di riferimento globale allineato con la verticale. Indicando con $\theta = 6^\circ$ l'angolo di inclinazione dell'asse $\mathbf{Z}$ rispetto alla verticale, è possibile descrivere la relazione tra il sistema di riferimento della camera e quello del mondo attraverso una matrice di rotazione. Considerando, ad esempio, una rotazione attorno all'asse $\mathbf{X}$ della camera, la trasformazione può essere espressa come:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \quad (2.1)$$

dove (X_c, Y_c, Z_c) rappresentano le coordinate di un punto nel sistema di riferimento della camera e (X_w, Y_w, Z_w) le coordinate dello stesso punto nel sistema di riferimento del mondo. Questa trasformazione consente di compensare l'inclinazione della telecamera e di ottenere una rappresentazione coerente con la geometria reale dell'ambiente osservato.

Una volta definito il sistema di riferimento della camera, la ricostruzione tridimensionale della scena avviene attraverso la trasformazione delle coordinate del piano immagine in coordinate spaziali reali. Tale trasformazione si basa sul modello di camera pinhole e sull'utilizzo dei parametri intrinseci del sistema ottico, in particolare delle lunghezze focali f_x e f_y [16] espresse in pixel lungo le direzioni orizzontale e verticale, e delle coordinate del centro ottico (c_x, c_y) . Indicando (x, y) le coordinate di un pixel e con $Z = D(x, y)$ a profondità associata, le coordinate tridimensionali del punto nel sistema di riferimento della camera possono essere ricostruite mediante le seguenti relazioni:

$$X = \frac{(x - c_x) \cdot Z}{f_x}, \quad Y = \frac{(y - c_y) \cdot Z}{f_y}, \quad Z = D(x, y) \quad (2.2)$$

Queste equazioni descrivono il processo di retro-proiezione, attraverso il quale ogni pixel della mappa di profondità viene trasformato in un punto dello spazio tridimensionale. Le coordinate X e Y rappresentano rispettivamente lo spostamento orizzontale e verticale del punto rispetto all'asse ottico della camera, mentre la coordinata Z descrive la distanza lungo la direzione di osservazione. Le lunghezze focali f_x e f_y svolgono un ruolo fondamentale nel determinare il fattore di scala che lega le coordinate dell'immagine alle dimensioni reali nello spazio, rendendo la calibrazione intrinseca un passaggio essenziale per garantire l'accuratezza della ricostruzione.

L'insieme delle coordinate tridimensionali così ottenute costituisce una nuvola di punti, ovvero una discretizzazione spaziale della superficie degli oggetti presenti nella scena. Tale rappresentazione risulta particolarmente adatta a successive fasi di elaborazione, come la segmentazione, la stima delle forme geometriche e l'analisi delle distanze. Nel caso della Femto Bolt, la conoscenza dell'inclinazione dell'asse Z rispetto alla verticale e la corretta applicazione delle trasformazioni geometriche risultano fondamentali per garantire misure accurate di altezza e posizione, soprattutto in contesti applicativi quali la robotica e l'automazione industriale.

2.4 Scelte progettuali e filosofia dell'approccio adottato

L'approccio seguito nello sviluppo del programma può essere descritto come modulare, robusto e orientato alla misura, piuttosto che come un semplice esempio di utilizzo dell'SDK Orbbec. Le scelte progettuali principali possono essere sintetizzate in tre principi guida:

1. Separazione delle responsabilità

L'acquisizione dei dati è concettualmente separata dalle fasi di riconoscimento e di misura. Il programma non esegue alcuna elaborazione di alto livello (come object detection o stima geometrica), ma si limita a fornire dati affidabili e ben documentati alle fasi successive.

2. Robustezza operativa

L'algoritmo è progettato per gestire condizioni non ideali: inizializzazione del sensore, auto-esposizione, fluttuazioni iniziali del depth, assenza temporanea di frame validi. L'obiettivo è ridurre al minimo la probabilità di acquisire dati instabili o non rappresentativi.

3. Riproducibilità sperimentale

Ogni acquisizione produce non solo immagini, ma anche metadati e parametri di calibrazione, in modo che l'intero processo possa essere ripetuto e verificato a posteriori, anche a distanza di tempo.

Questi principi sono particolarmente rilevanti in applicazioni industriali, dove la trasparenza del processo e la possibilità di validare i risultati sono requisiti fondamentali.

2.5 Organizzazione dati e file Output

Una delle prime scelte progettuali riguarda l'organizzazione dei dati prodotti dal programma. Invece di salvare semplicemente le immagini in una cartella generica, l'algoritmo crea una struttura di directory dedicata all'acquisizione corrente. Questa scelta risponde a due esigenze principali.

La prima è la chiarezza: separare i dati di acquisizione da altri file di progetto riduce il rischio di confusione, soprattutto quando si lavora con numerosi esperimenti o configurazioni differenti. La seconda è la scalabilità: una struttura ben definita consente di estendere facilmente il programma per acquisizioni multiple, costruzione di dataset o salvataggio di versioni successive delle immagini. L'organizzazione strutturata dei dati di acquisizione, comprensiva di immagini, metadati e parametri di calibrazione, è fondamentale per garantire la riproducibilità degli esperimenti e la verificabilità dei risultati [17].

All'interno della cartella di output, il programma salva:

- la mappa di profondità allineata
- un file di testo con informazioni riepilogative
- un file contenente i parametri di calibrazione e la scala della profondità

2.5.1 Acquisizione e configurazione degli stream

L'acquisizione dei dati avviene tramite l'SDK Orbbec, che fornisce una struttura ad alto livello basata su una pipeline di elaborazione. La pipeline rappresenta l'insieme dei componenti che gestiscono la comunicazione con il dispositivo, la sincronizzazione degli stream e la consegna dei frame all'applicazione.

Il programma non assume a priori che la telecamera fornisca determinati formati o risoluzioni, ma interroga il dispositivo per ottenere la lista dei profili disponibili per ciascun sensore. Questo approccio aumenta la portabilità del algoritmo e riduce il rischio di errori dovuti a configurazioni non supportate dall'hardware o dall'SDK.

La selezione dei profili RGB e depth è un passaggio cruciale, poiché influisce direttamente:

- sulla qualità dell'immagine utilizzata per il riconoscimento
- sulla risoluzione e precisione della mappa di profondità
- sulla compatibilità tra i due stream in fase di allineamento

In un contesto industriale, questa flessibilità consente di adattare il sistema a differenti requisiti di precisione o velocità, semplicemente modificando la scelta del profilo, senza intervenire sulla logica complessiva del programma.

Dopo la fase di stabilizzazione e allineamento, il programma seleziona un frameset valido e ne verifica la completezza. Questo passaggio riflette una filosofia di programmazione difensiva: prima di procedere con il salvataggio dei dati, il software controlla esplicitamente che tutte le informazioni necessarie siano presenti.

Questa scelta riduce il rischio di produrre file incompleti o incoerenti, che potrebbero essere erroneamente utilizzati nelle fasi successive di elaborazione. In un contesto di tesi o di applicazione industriale, è preferibile interrompere l'acquisizione con un errore chiaro piuttosto che produrre dati di dubbia qualità.

2.6 Avvio della pipeline e interazione con il dispositivo

Una volta definiti i profili di acquisizione, la pipeline viene avviata e il dispositivo viene messo in funzione. Da questo momento, la telecamera inizia a produrre flussi continui di frame RGB e depth.

Il programma recupera anche un riferimento diretto al dispositivo fisico, che viene successivamente utilizzato per ottenere informazioni di calibrazione e parametri specifici del sensore. Questa scelta evidenzia un aspetto importante: la calibrazione non è trattata come un'informazione esterna o “nota a priori”, ma come un dato intrinseco al dispositivo, da acquisire e salvare contestualmente alle immagini. L'avvio della pipeline di acquisizione e la gestione dello streaming dei dati da sensori RGB-D richiedono una corretta sincronizzazione e integrazione dei flussi di colore e profondità [18].

Questo approccio riduce il rischio di errori dovuti all'uso di parametri di calibrazione non coerenti con il sensore effettivamente utilizzato.

2.7 Stabilizzazione iniziale del sensore e motivazioni

Uno degli aspetti più rilevanti del programma è la gestione della fase iniziale di acquisizione, durante la quale il sensore non ha ancora raggiunto condizioni di funzionamento stabili. In particolare, le prime decine di frame prodotti da una telecamera RGB-D possono essere affette da:

- variazioni di esposizione e guadagno automatico
- rumore elevato nel segnale di profondità
- disallineamenti temporanei tra stream RGB e depth

Le misure fornite dai sensori di profondità possono presentare instabilità e rumore significativo nelle fasi iniziali di acquisizione, rendendo necessaria una fase di stabilizzazione prima dell'utilizzo dei dati a fini metrici [19].

In un sistema di misura, l'acquisizione di dati instabili può compromettere l'intera catena di elaborazione, portando a risultati non ripetibili o a errori sistematici difficili da diagnosticare. Per questo motivo, il programma adotta una strategia di stabilizzazione basata sullo scarto dei primi frame acquisiti.

Dal punto di vista concettuale, questa fase può essere vista come una fase di "stabilizzazione" del sensore, analogo a quanto avviene in molti strumenti di misura industriali. Il software attende che il flusso di dati diventi stabile prima di selezionare il frame da salvare, riducendo la probabilità di acquisire immagini non rappresentative delle reali condizioni operative.

Questa scelta, sebbene semplice, ha un impatto significativo sulla qualità delle misure successive, soprattutto quando si richiede una precisione millimetrica.

2.8 Allineamento tra immagine RGB e mappa di profondità

Un aspetto fondamentale nell'elaborazione dei dati provenienti da sensori RGB-D è rappresentato dall'allineamento tra l'immagine a colori (RGB) e la corrispondente mappa di profondità. Tale operazione risulta necessaria per garantire una coerenza geometrica tra le informazioni radiometriche e quelle spaziali, consentendo un'integrazione efficace delle due modalità sensoriali all'interno della pipeline di elaborazione.

Un passaggio chiave della pipeline è l'allineamento della mappa di profondità al frame RGB. In un sensore RGB-D, i due stream sono generalmente acquisiti da sensori fisicamente distinti, caratterizzati da ottiche differenti e da punti di vista non coincidenti. Di conseguenza, i pixel dell'immagine RGB e quelli della mappa depth non corrispondono automaticamente agli stessi punti nello spazio [20]. Questa discrepanza è dovuta sia alla diversa posizione relativa tra i sensori (parametri estrinseci), sia alle differenti caratteristiche interne delle camere (parametri intrinseci), quali lunghezza focale, centro ottico e distorsioni.

L'allineamento software si pone quindi come soluzione a tale problematica, trasformando la mappa di profondità in modo che essa sia espressa nel sistema di riferimento dell'immagine colore. Questo processo si basa su un modello geometrico che descrive la relazione tra i due sistemi di acquisizione e richiede una fase preliminare di calibrazione, finalizzata alla stima accurata dei parametri intrinseci ed estrinseci delle due camere. Una calibrazione precisa è essenziale per evitare errori sistematici che comprometterebbero la corretta corrispondenza tra le due modalità.

Dal punto di vista metodologico, la tecnica più diffusa per l'allineamento RGB-depth è quella basata sulla proiezione geometrica. In questo approccio, ogni pixel della mappa di profondità viene inizialmente retro-proiettato nello spazio tridimensionale, ottenendo così una rappresentazione 3D della scena nel sistema di riferimento del sensore depth. Successivamente, tali punti vengono trasformati nel sistema di riferimento della camera RGB mediante una trasformazione rigida definita dai parametri estrinseci. Infine, i punti trasformati vengono proiettati sul

piano immagine della camera colore utilizzando i parametri intrinseci, ottenendo così una mappa di profondità allineata pixel-per-pixel con l'immagine RGB.

Questo approccio, sebbene ampiamente utilizzato per la sua solidità teorica e semplicità implementativa, presenta alcune criticità legate alla natura dei sensori. In particolare, la differenza di punto di vista può generare occlusioni, mentre la discretizzazione della profondità può causare la presenza di regioni non definite (holes) o sovrapposizioni multiple nella fase di proiezione. Tali effetti vengono generalmente mitigati mediante tecniche di interpolazione o strategie di filtraggio post-processo.

Accanto ai metodi basati su proiezione, esistono approcci alternativi basati sulla registrazione tra immagini, che sfruttano l'estrazione di caratteristiche visive per stimare una trasformazione tra le due viste. Tuttavia, tali tecniche risultano meno adatte nel contesto RGB-D, in quanto le mappe di profondità non presentano informazioni radiometriche comparabili a quelle dell'immagine RGB, limitandone l'efficacia e la robustezza.

Nel contesto applicativo della presente tesi, la scelta di allineare la mappa di profondità al frame RGB, piuttosto che viceversa, è coerente con la struttura della pipeline proposta. In particolare, il riconoscimento del pezzo viene effettuato sull'immagine RGB, mentre l'informazione di profondità viene impiegata in una fase successiva per la ricostruzione tridimensionale. L'allineamento consente pertanto di utilizzare direttamente le coordinate pixel dell'immagine colore — ad esempio quelle associate a una bounding box — per individuare in modo immediato la regione corrispondente nella mappa di profondità.

Questa scelta progettuale garantisce una transizione semplice, coerente e deterministica tra la fase di analisi visiva e quella di elaborazione geometrica, riducendo la complessità computazionale e semplificando l'integrazione tra i diversi moduli della pipeline, come verrà approfondito nel capitolo 4. In conclusione, l'allineamento RGB-depth rappresenta un elemento imprescindibile nei sistemi RGB-D, fungendo da collegamento tra percezione visiva e modellazione tridimensionale della scena.

2.9 Salvataggio delle immagini e conservazione dell'informazione

Il salvataggio delle immagini avviene utilizzando formati non compressi o lossless, in modo da preservare l'informazione originale fornita dal sensore. In particolare, la scelta di salvare la mappa di profondità in formato a 16 bit è fondamentale per applicazioni di misura, poiché consente di rappresentare con sufficiente precisione l'intervallo di distanze misurabili dal sensore.

Dal punto di vista metodologico, questa scelta riflette l'obiettivo del progetto: non si tratta di produrre immagini “esteticamente gradevoli”, ma di preservare la massima quantità di informazione utile per la ricostruzione tridimensionale e la stima dimensionale.

Uno degli aspetti più significativi del programma è il salvataggio esplicito dei parametri di calibrazione e della scala della profondità. Questo passaggio rende il dataset autosufficiente: ogni immagine è accompagnata dalle informazioni necessarie per interpretarne correttamente i valori.

Dal punto di vista della ricostruzione 3D, la presenza dei parametri intrinseci e della scala depth consente di convertire i pixel della mappa di profondità in coordinate tridimensionali reali, secondo il modello di camera pinhole. Questo rende possibile l'uso diretto dei dati in ambienti di calcolo come MATLAB, senza dover ricorrere a ipotesi implicite o parametri esterni.

Nel contesto complessivo del progetto, il programma di acquisizione rappresenta il ponte tra il mondo fisico e il dominio digitale. Le immagini RGB alimentano la rete neurale per il riconoscimento del pezzo, mentre la mappa di profondità, correttamente allineata e calibrata, consente la ricostruzione tridimensionale della regione di interesse.

La separazione concettuale tra riconoscimento e misura, resa possibile dalla struttura dei dati acquisiti, costituisce uno dei punti di forza dell'approccio adottato. Essa consente di combinare la flessibilità del deep learning con la precisione e l'interpretabilità dei metodi geometrici deterministici.

2.10 Considerazioni finali sul programma

Il programma sviluppato non è semplicemente un esempio di utilizzo dell'SDK Orbbec, ma un componente fondamentale di una pipeline di visione industriale orientata alla misura. Le scelte progettuali adottate — dalla stabilizzazione del sensore al salvataggio dei parametri di calibrazione — riflettono una particolare attenzione alla qualità del dato e alla riproducibilità del processo.

In conclusione, il software di acquisizione rappresenta un elemento chiave per garantire che le prestazioni complessive del sistema non siano limitate da errori o incoerenze introdotte nelle fasi iniziali, ponendo basi solide per le successive elaborazioni di riconoscimento e misura dimensionale.

Capitolo 3

Reti Neurali Artificiali

3.1 Introduzione alle reti neurali Artificiali

Negli ultimi decenni, l'evoluzione delle tecnologie informatiche ha portato a una crescita esponenziale della quantità di dati disponibili e della potenza di calcolo accessibile. In questo scenario, si è resa sempre più evidente la necessità di sviluppare modelli computazionali in grado non solo di elaborare grandi moli di informazioni, ma anche di apprendere automaticamente strutture, regolarità e relazioni latenti nei dati. Le reti neurali artificiali rappresentano una delle risposte più significative a questa esigenza, configurandosi come strumenti fondamentali nell'ambito dell'intelligenza artificiale e, in particolare, del machine learning.

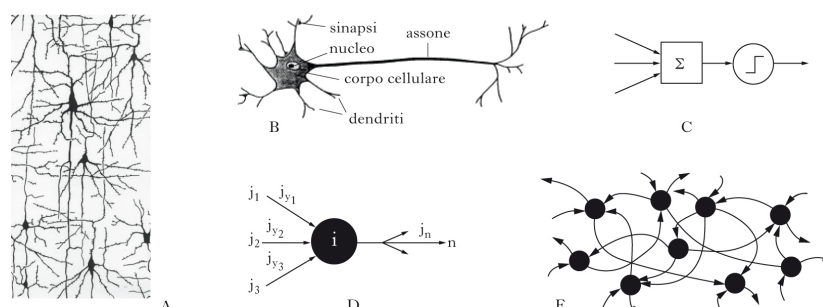


Figura 3.1: Astrazione reti neurali (fonte: [21].)

L'idea alla base delle reti neurali è quella di imitare, in modo astratto e semplificato, il funzionamento del cervello umano [22], come illustrato in figura 3.1. Il

cervello è infatti composto da miliardi di neuroni interconnessi, capaci di elaborare informazioni in modo parallelo, adattivo e robusto. Sebbene i modelli artificiali siano molto lontani dalla complessità biologica reale, essi riescono comunque a catturare alcuni principi chiave dell'elaborazione neurale, come l'apprendimento per esperienza e la capacità di generalizzare a partire da esempi.

Nel contesto accademico e industriale, le reti neurali sono oggi impiegate in un numero crescente di applicazioni: dal riconoscimento vocale e visivo, alla traduzione automatica, dall'analisi di dati biomedici alla finanza quantitativa, fino alla guida autonoma e ai sistemi di raccomandazione. Tale diffusione rende imprescindibile una comprensione approfondita dei fondamenti teorici, storici e architetturali di questi modelli, obiettivo principale della presente introduzione.

3.1.1 Origini storiche delle reti neurali

Le prime idee riconducibili alle reti neurali artificiali risalgono alla metà del XX secolo. Nel 1943, Warren McCulloch e Walter Pitts proposero un modello matematico semplificato del neurone biologico, noto come neurone di McCulloch-Pitts.

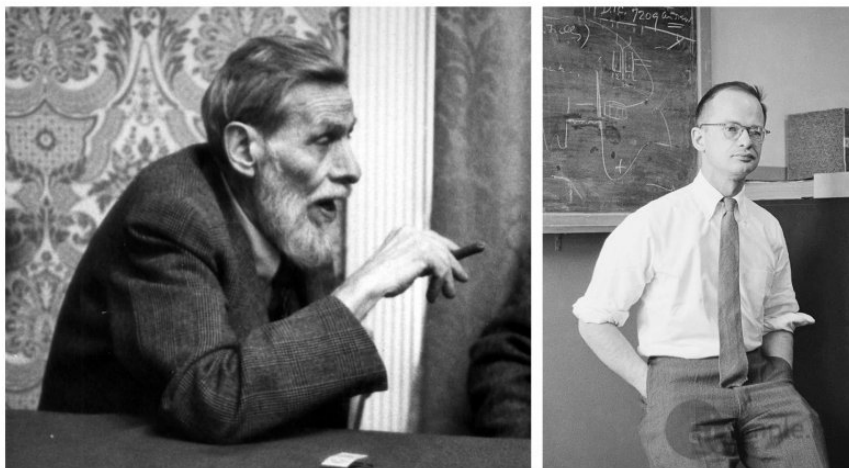


Figura 3.2: A sinistra: Warren McCulloch a destra: Walter Pitts (fonte: [23])

Questo modello descriveva il neurone come un'unità logica capace di produrre un'uscita binaria in funzione di una combinazione lineare degli ingressi. Sebbene estremamente semplice, tale approccio pose le basi concettuali per lo sviluppo successivo delle reti neurali. Negli anni Cinquanta, Frank Rosenblatt introdusse il

perceptrone, uno dei primi modelli di apprendimento automatico basati su neuroni artificiali. Il perceptrone era in grado di apprendere una funzione di classificazione lineare a partire da esempi etichettati, adattando i propri parametri attraverso una regola di apprendimento iterativa.

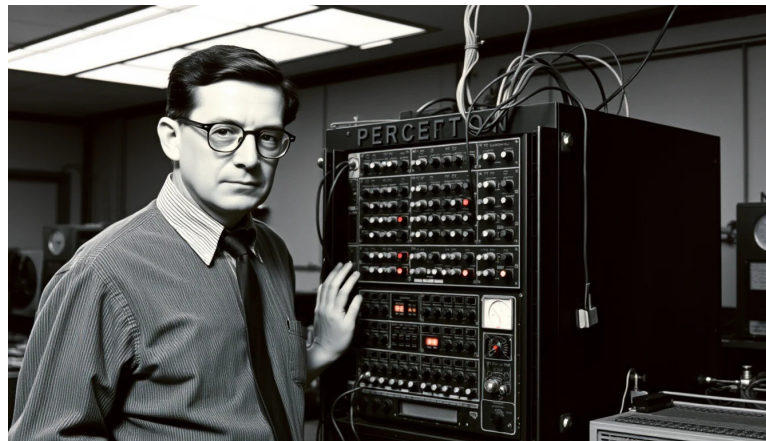


Figura 3.3: Frank Rosenblatt e il suo perceptrone (fonte: [24].)

Nonostante l'entusiasmo iniziale, le limitazioni teoriche del perceptrone, in particolare l'impossibilità di risolvere problemi non linearmente separabili, portarono a un periodo di ridotto interesse verso le reti neurali, noto come "inverno dell'intelligenza artificiale".

Una nuova fase di sviluppo ebbe inizio negli anni Ottanta, grazie all'introduzione dell'algoritmo di retropropagazione dell'errore (backpropagation) [25]. Questo metodo consentì di addestrare reti neurali multistrato, superando i limiti del perceptrone semplice e permettendo l'apprendimento di relazioni non lineari complesse. Da quel momento in poi, le reti neurali iniziarono a dimostrare un potenziale sempre maggiore, sebbene la loro diffusione pratica fosse ancora limitata dalle risorse computazionali disponibili.

3.2 Affermazione del deep learning

Il vero punto di svolta si è verificato a partire dagli anni Duemila, con la cosiddetta rinascita delle reti neurali profonde, comunemente indicata come deep learning [26]. Questo cambiamento è stato reso possibile dalla convergenza di tre fattori principali: la disponibilità di grandi quantità di dati (big data), l'aumento della potenza di calcolo, in particolare grazie alle unità di elaborazione grafica (GPU), e lo sviluppo di nuove architetture e tecniche di addestramento.

Le reti neurali profonde si caratterizzano per la presenza di numerosi strati intermedi, detti strati nascosti, che consentono di apprendere rappresentazioni gerarchiche dei dati. In ambito visivo, ad esempio, i primi strati possono apprendere caratteristiche semplici come bordi e angoli, mentre gli strati più profondi catturano strutture sempre più complesse, fino a concetti di alto livello come oggetti o scene. Questa capacità di apprendere automaticamente le caratteristiche rilevanti distingue il deep learning dagli approcci tradizionali, che richiedevano una progettazione manuale delle feature.

L'affermazione del deep learning ha avuto un impatto profondo sia sulla ricerca scientifica sia sulle applicazioni industriali. Risultati di livello umano, e in alcuni casi superiore, sono stati raggiunti in compiti come il riconoscimento di immagini, il gioco strategico e l'elaborazione del linguaggio naturale, consolidando il ruolo centrale delle reti neurali nel panorama dell'intelligenza artificiale moderna.

3.3 Struttura generale

Dal punto di vista architetturale, una rete neurale artificiale è tipicamente organizzata in strati. Il primo strato, detto strato di input (pallini gialli in figura 3.4) che riceve i dati in ingresso sotto forma di vettori numerici. Seguono uno o più strati nascosti (pallini verdi in figura 3.4) responsabili dell'elaborazione interna delle informazioni, e infine uno strato di output (pallini rossi in figura 3.4) che produce il risultato finale del modello.

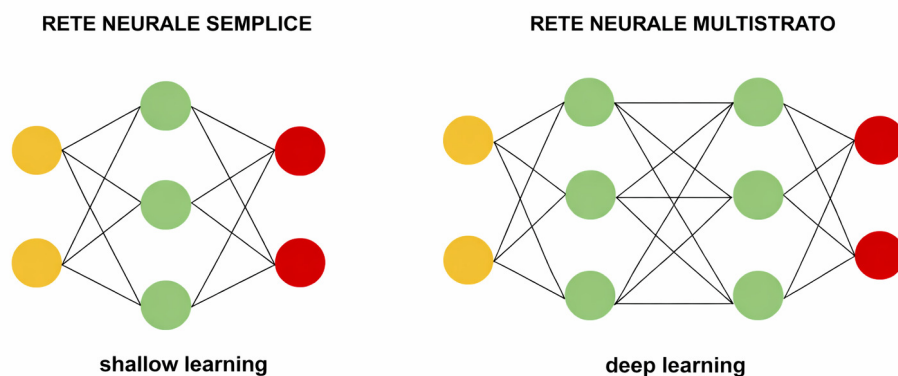


Figura 3.4: Esempi di struttura generale delle reti neurali (fonte: [27].)

Ogni strato è composto da un certo numero di unità computazionali, chiamate neuroni artificiali. Ciascun neurone riceve in ingresso i segnali provenienti dai neuroni dello strato precedente, li combina linearmente attraverso pesi sinaptici e aggiunge un termine di bias. Il risultato di questa combinazione viene poi trasformato mediante una funzione di attivazione non lineare, che introduce la capacità di modellare relazioni complesse.

Le funzioni di attivazione rivestono un ruolo fondamentale nel comportamento delle reti neurali. Tra le più comuni si annoverano la funzione sigmoide, la tangente

iperbolica e la funzione ReLU (Rectified Linear Unit). La scelta della funzione di attivazione influisce significativamente sulla capacità di apprendimento del modello e sulla stabilità del processo di addestramento.

3.3.1 Caratteristiche fondamentali

Le reti neurali artificiali presentano alcune caratteristiche distintive che ne spiegano il successo. In primo luogo, esse sono modelli universali di approssimazione: è stato dimostrato teoricamente che una rete neurale con almeno uno strato nascosto può approssimare qualsiasi funzione continua, a patto di disporre di un numero sufficiente di neuroni.

In secondo luogo, le reti neurali sono intrinsecamente adattive. Grazie al processo di apprendimento, esse modificano il proprio comportamento in risposta ai dati, rendendole particolarmente adatte a contesti dinamici e complessi. Inoltre, la loro natura distribuita conferisce una certa robustezza al rumore e agli errori locali.

Infine, le reti neurali consentono un'elaborazione altamente parallela, sia a livello concettuale sia implementativo. Questa caratteristica le rende particolarmente compatibili con le moderne architetture hardware e spiega in parte le prestazioni elevate ottenute nelle applicazioni su larga scala.

3.4 Apprendimento e ottimizzazione

La fase di apprendimento rappresenta un elemento centrale nei sistemi basati su reti neurali, in quanto determina la capacità del modello di apprendere relazioni significative dai dati e, soprattutto, di generalizzare correttamente a esempi non osservati durante l'addestramento [9]. In questo contesto, l'apprendimento può essere classificato in tre principali paradigmi: apprendimento supervisionato, non supervisionato e per rinforzo, ciascuno caratterizzato da differenti modalità di utilizzo dell'informazione disponibile.

Nel caso dell'apprendimento supervisionato, il modello viene addestrato a partire da un insieme di dati etichettati, in cui a ogni input è associato un output

desiderato. L'obiettivo della rete è apprendere una funzione di mapping che minimizzi l'errore tra le predizioni e i valori reali. A tal fine, viene definita una funzione di perdita, il cui gradiente rispetto ai parametri della rete viene calcolato mediante l'algoritmo di retropropagazione dell'errore. Questo algoritmo consente di determinare in modo efficiente come ciascun peso e bias contribuisca all'errore complessivo, propagando l'informazione dagli strati di uscita verso quelli iniziali.

I gradienti così ottenuti vengono quindi utilizzati da algoritmi di ottimizzazione, quali la discesa del gradiente e le sue varianti, per aggiornare iterativamente i parametri del modello. La convergenza del processo di apprendimento dipende da diversi fattori, tra cui il tasso di apprendimento, la scelta dell'inizializzazione dei parametri e la complessità architetturale della rete. Questo paradigma trova ampia applicazione in problemi di classificazione e regressione, dove è disponibile un ground truth esplicito.

Diversamente, nell'apprendimento non supervisionato il modello opera su dati privi di etichette, con l'obiettivo di individuare strutture latenti, regolarità o rappresentazioni compatte dell'informazione. In assenza di un errore definito rispetto a un output noto, il processo di apprendimento si basa su criteri alternativi, come la minimizzazione della ricostruzione (nel caso degli autoencoder) o l'organizzazione topologica dei dati (come nelle mappe auto-organizzanti). Questo approccio risulta particolarmente utile in contesti esplorativi o quando l'annotazione dei dati non è disponibile, sebbene la valutazione delle prestazioni risulti meno immediata.

Il terzo paradigma è rappresentato dall'apprendimento per rinforzo, nel quale un agente apprende attraverso l'interazione con un ambiente dinamico. In questo scenario, il modello non riceve etichette esplicite, ma segnali di ricompensa o penalità che guidano l'apprendimento di una politica ottimale. L'obiettivo è massimizzare la ricompensa cumulativa nel tempo, affrontando il compromesso tra esplorazione di nuove strategie e sfruttamento delle conoscenze acquisite. Tale approccio è ampiamente utilizzato in ambiti quali il controllo automatico, la robotica e i sistemi decisionali complessi.

Indipendentemente dal paradigma adottato, un aspetto critico dell'addestramento delle reti neurali è rappresentato dal rischio di overfitting, ossia la tendenza

del modello a memorizzare i dati di addestramento anziché apprendere regolarità generalizzabili. Questo fenomeno compromette la capacità del sistema di operare correttamente su dati non visti. Per contrastarlo, sono state sviluppate diverse tecniche di regolarizzazione, tra cui il dropout, la penalizzazione dei pesi e l'early stopping, che contribuiscono a migliorare la robustezza e la capacità di generalizzazione del modello.

In conclusione, i tre paradigmi di apprendimento costituiscono le fondamenta teoriche delle reti neurali moderne e, sebbene distinti, possono essere combinati per sfruttarne i vantaggi complementari. La progettazione efficace di un sistema di apprendimento richiede pertanto una scelta consapevole del paradigma più adatto, nonché un'attenta gestione del processo di ottimizzazione e delle problematiche legate alla generalizzazione.

3.5 Continuità tra deep learning e MATLAB

Nei capitoli precedenti sono state analizzate le principali caratteristiche delle reti neurali profonde impiegate per il riconoscimento automatico degli oggetti, con particolare riferimento alle architetture di object detection basate su reti convoluzionali. Tali modelli rappresentano attualmente lo stato dell'arte per l'estrazione automatica di informazioni semantiche da immagini, grazie alla loro capacità di apprendere direttamente dai dati caratteristiche robuste e discriminanti.

Tuttavia, l'adozione di una rete neurale in un contesto applicativo reale, e in particolare in un sistema di visione industriale orientato alla misura, richiede un ulteriore livello di riflessione che va oltre la sola architettura del modello. È infatti necessario considerare l'ambiente di sviluppo, gli strumenti disponibili per l'addestramento e la validazione, nonché le modalità di integrazione della rete neurale all'interno della pipeline complessiva del sistema.

In questo contesto, il passaggio da framework di deep learning orientati principalmente alla ricerca e allo sviluppo di modelli generici a un ambiente come MATLAB rappresenta una scelta progettuale significativa, che merita di essere analizzata e motivata.

È importante sottolineare che l'utilizzo di MATLAB non implica un cambiamento nei principi fondamentali del deep learning. I concetti chiave introdotti nei capitoli precedenti — apprendimento supervisionato, ottimizzazione tramite minimizzazione di una funzione di perdita, generalizzazione del modello — rimangono invariati.

Ciò che cambia è il livello di astrazione e il modo in cui questi concetti vengono declinati operativamente. In MATLAB, l'attenzione si sposta maggiormente su:

- la definizione esplicita dei dati di input e delle etichette;
- il controllo delle fasi di addestramento e validazione;
- l'analisi quantitativa delle prestazioni del modello;
- l'integrazione del risultato della rete neurale con le successive fasi di elaborazione.

In questo senso, MATLAB non sostituisce i framework di deep learning, ma rappresenta un ambiente complementare, particolarmente efficace quando l'obiettivo non è soltanto il riconoscimento, ma l'interazione della rete con un sistema di misura più ampio.

3.5.1 Motivazioni di MATLAB nel contesto del progetto

Nel progetto sviluppato, l'obiettivo principale non è la realizzazione di un sistema di object detection general-purpose, bensì l'integrazione del riconoscimento automatico del pezzo all'interno di una pipeline di visione industriale orientata alla misura dimensionale.

Questa specificità ha guidato la scelta di MATLAB come ambiente di sviluppo per la rete neurale finale, per una serie di motivazioni principali:

- **Integrazione con la fase geometrica** MATLAB consente di combinare in modo naturale l'output della rete neurale (bounding box, maschere o regioni di interesse) con le successive elaborazioni geometriche basate sulla mappa di profondità e sulla nuvola di punti.

- **Controllo e interpretabilità** In un contesto metrologico, è spesso preferibile utilizzare modelli e strumenti che consentano un controllo esplicito sui parametri e una facile interpretazione dei risultati. MATLAB offre strumenti di analisi che facilitano la verifica e la validazione delle prestazioni della rete.
- **Prototipazione rapida e validazione sperimentale** L'ambiente MATLAB permette di testare rapidamente diverse configurazioni di rete, parametri di training e strategie di validazione, mantenendo una stretta connessione con i dati reali acquisiti dal sistema.
- **Uniformità dell'ambiente di sviluppo** Dal momento che le fasi successive di elaborazione (ricostruzione 3D, misura dimensionale, visualizzazione dei risultati) sono già implementate in MATLAB, l'utilizzo dello stesso ambiente anche per la rete neurale semplifica l'integrazione complessiva del sistema.

Alla luce delle considerazioni precedenti, il capitolo successivo si concentra sulla descrizione dettagliata della rete neurale implementata e addestrata in ambiente MATLAB. Verranno analizzati:

- la struttura della rete;
- il dataset utilizzato per l'addestramento;
- le scelte relative alla procedura di training e validazione;
- le prestazioni ottenute in termini di accuratezza e robustezza;
- l'integrazione della rete all'interno della pipeline di misura.

Questo passaggio rappresenta il proseguimento del percorso iniziato con l'analisi delle architetture di deep learning generiche e che conduce alla presentazione di una soluzione concreta, sviluppata e validata nel contesto specifico del progetto.

3.6 Valutazione delle architetture di object detection

3.6.1 Introduzione alla fase di valutazione delle reti neurali

Prima di procedere allo sviluppo e all'addestramento della rete neurale definitiva in ambiente MATLAB, è stata condotta una fase preliminare di valutazione delle architetture di object detection più idonee al contesto applicativo del progetto. L'obiettivo di questa analisi non era quello di individuare la rete con le migliori prestazioni assolute su benchmark generici, ma di selezionare una soluzione che offrisse un compromesso adeguato tra accuratezza, robustezza e complessità computazionale, in linea con le esigenze di un sistema di visione industriale orientato alla misura.

In particolare, l'attenzione si è concentrata su architetture appartenenti alla famiglia YOLO (*You Only Look Once*), ampiamente utilizzate per problemi di object detection in tempo reale. Tra le diverse varianti disponibili, sono state prese in considerazione le versioni **YOLOv4 Tiny** e **YOLOX Tiny**, entrambe progettate per applicazioni in cui le risorse computazionali sono limitate e la latenza deve essere contenuta.

3.6.2 Motivazioni della scelta delle versioni “Tiny”

Le versioni Tiny delle architetture YOLO rappresentano una semplificazione delle reti originali, ottenuta riducendo il numero di parametri, la profondità della rete e la complessità delle operazioni convoluzionali. Tale riduzione consente di mantenere tempi di inferenza contenuti e un utilizzo limitato delle risorse computazionali, aspetti di particolare rilevanza nei sistemi di visione industriale e nei contesti applicativi real-time.

Le versioni complete delle architetture YOLO, sebbene caratterizzate da prestazioni superiori in termini di accuratezza su benchmark generici, richiedono tipicamente risorse hardware di elevata capacità, quali GPU dedicate o CPU ad

alte prestazioni, per garantire tempi di elaborazione compatibili con l'uso operativo. Nel contesto del progetto, l'impiego di tali risorse non risulta giustificato, poiché il sistema è progettato per operare su piattaforme hardware comuni e per integrarsi con ulteriori moduli di elaborazione, come la ricostruzione tridimensionale e la misura geometrica.

L'adozione delle versioni Tiny consente quindi di ottenere un compromesso efficace tra prestazioni e complessità computazionale, rendendo possibile l'esecuzione della rete neurale anche in assenza di hardware altamente performante. Questa scelta risulta coerente con l'obiettivo di sviluppare un sistema modulare, scalabile e potenzialmente trasferibile in un contesto industriale, dove i vincoli di costo, consumo energetico e integrazione hardware rivestono un ruolo fondamentale.

3.7 Confronto concettuale tra YOLOv4 e YOLOX

YOLOv4 Tiny rappresenta una versione alleggerita dell'architettura YOLOv4, progettata per mantenere buone prestazioni di detection riducendo drasticamente il numero di parametri. Dal punto di vista concettuale, YOLOv4 Tiny si basa su un'architettura anchor-based, in cui la rete predice bounding box come offset rispetto a un insieme predefinito di ancore. Questa impostazione presenta alcuni vantaggi:

- semplicità concettuale;
- buona stabilità durante l'addestramento;
- ampia diffusione e disponibilità di esempi applicativi.

Tuttavia, l'uso di ancore richiede una fase preliminare di definizione o adattamento delle stesse al dataset specifico. In contesti fortemente specializzati, come il riconoscimento di pochi pezzi meccanici con geometria nota, questa dipendenza dalle ancore può introdurre una rigidità non sempre desiderabile.

3.7.1 YOLOX Tiny: evoluzione architetturale e approccio anchor-free

YOLOX Tiny rappresenta un'evoluzione più recente della famiglia YOLO e introduce un cambiamento concettuale significativo rispetto alle versioni precedenti: l'adozione di un approccio anchor-free. In questo paradigma, la rete non utilizza bounding box predefinite, ma apprende direttamente la posizione e le dimensioni degli oggetti a partire dalle caratteristiche estratte. Dal punto di vista metodologico, questo comporta diversi vantaggi:

- semplificazione della fase di progettazione del modello;
- maggiore flessibilità nell'adattamento a oggetti di forma diversa;
- riduzione della dipendenza da iperparametri legati alle ancore.

Inoltre, YOLOX introduce una separazione esplicita tra il ramo di classificazione e il ramo di regressione delle bounding box, migliorando la capacità della rete di apprendere rappresentazioni più specializzate per ciascun compito.

Nel contesto del progetto, caratterizzato da:

- un numero limitato di classi (ruota e albero),
- un ambiente relativamente controllato,
- l'esigenza di un riconoscimento robusto piuttosto che estremamente generico,

YOLOX Tiny risulta particolarmente interessante per la sua maggiore flessibilità e per la riduzione della complessità legata alla gestione delle ancore. YOLOv4 Tiny, pur rimanendo una soluzione valida e consolidata, risulta invece più vincolata a scelte architettureali che non sempre si adattano in modo ottimale a dataset di dimensioni ridotte e fortemente specializzati.

3.8 Costruzione del dataset per riconoscimento dei pezzi

Una fase cruciale nello sviluppo di qualsiasi rete neurale supervisionata è la costruzione di un dataset rappresentativo del problema reale. Nel progetto in esame, si è scelto di realizzare un dataset dedicato, acquisendo immagini direttamente dalla telecamera utilizzata nel sistema finale.

Questa scelta è motivata dalla volontà di:

- ridurre il domain shift tra fase di addestramento e fase operativa;
- garantire che le condizioni di acquisizione (illuminazione, punto di vista, risoluzione) siano coerenti;
- evitare l'uso di dataset generici non rappresentativi del contesto industriale specifico.

3.8.1 Composizione del dataset

Il dataset è stato costruito includendo 100 immagini per ciascun pezzo meccanico, ovvero:

- 100 immagini della ruota;
- 100 immagini dell'albero;

Le immagini sono state acquisite variando:

- la posizione del pezzo all'interno della scena;
- l'orientazione sul piano di appoggio;
- la distanza dalla camera.

Questa variabilità controllata consente alla rete di apprendere una rappresentazione più robusta del pezzo, evitando un'eccessiva specializzazione su configurazioni particolari.

3.8.2 Script MATLAB per l'acquisizione automatizzata del dataset

La creazione del dataset è stata supportata da uno script MATLAB dedicato, sviluppato con l'obiettivo di automatizzare il processo di acquisizione delle immagini e garantire uniformità e ripetibilità nei dati raccolti. L'utilizzo di uno script separato consente di mantenere distinta la fase di raccolta dei dati da quella di addestramento della rete neurale, migliorando la chiarezza del flusso di lavoro e la manutenibilità del progetto.

Lo script è stato progettato in modo flessibile, permettendo di adattare rapidamente la procedura di acquisizione a diverse esigenze sperimentali senza modificare la struttura del codice. In particolare, è sufficiente intervenire su un numero limitato di parametri per definire:

- il numero di immagini da acquisire per ciascuna classe;
- il tempo di attesa tra uno scatto e il successivo, utile per riposizionare manualmente il pezzo o variare le condizioni di acquisizione;
- le classi di immagini da creare, corrispondenti ai diversi pezzi meccanici oggetto di studio.

Una volta impostati questi parametri, l'operatore può semplicemente avviare il programma, che procede automaticamente all'acquisizione e al salvataggio delle immagini secondo le specifiche definite. Questo approccio riduce la possibilità di errori manuali, assicura una distribuzione controllata delle immagini nel dataset e facilita la creazione di nuovi insiemi di dati in caso di estensione del progetto o introduzione di ulteriori classi.

Dal punto di vista metodologico, l'automazione della fase di acquisizione contribuisce a migliorare la coerenza del dataset e rende il processo facilmente replicabile, aspetti fondamentali per l'addestramento e la validazione di reti neurali in contesti applicativi reali. L'annotazione delle immagini è stata effettuata tramite lo strumento Image Labeler di MATLAB, che consente di:

- definire classi di oggetti;
- disegnare bounding box in modo interattivo;
- gestire in modo strutturato le etichette associate alle immagini.

L'utilizzo di Image Labeler offre il vantaggio di una stretta integrazione con l'ambiente MATLAB, facilitando il passaggio diretto dai dati annotati alla fase di addestramento della rete neurale.

Durante l'annotazione, particolare attenzione è stata posta alla precisione della delimitazione delle bounding box, in modo da includere l'intero pezzo meccanico senza introdurre porzioni significative di sfondo. Questa scelta è particolarmente importante nel contesto del progetto, poiché la bounding box definisce la regione di interesse che verrà successivamente utilizzata per l'estrazione della mappa di profondità e per la ricostruzione tridimensionale.

Una bounding box accurata non solo migliora le prestazioni della rete neurale, ma contribuisce anche alla qualità delle misure geometriche effettuate nelle fasi successive.

Una sintesi delle principali differenze tra le due architetture è riportata in Tabella 3.1, dalla quale emerge come YOLOX Tiny offra una maggiore flessibilità rispetto a YOLOv4 Tiny, in particolare in contesti applicativi con dataset ridotti e altamente specializzati.

Caratteristica	YOLOv4 Tiny	YOLOX Tiny
Approccio architetturale	Anchor-based (uso di bounding box predefinite)	Anchor-free (predizione diretta delle bounding box)
Gestione delle bounding box	Offset rispetto ad ancore predefinite	Predizione diretta di posizione e dimensioni
Complessità progettuale	Richiede definizione e tuning delle ancore	Nessuna definizione di ancore necessaria
Flessibilità	Limitata, dipendente dalle ancore	Elevata, adattabile a diverse forme di oggetti
Stabilità di training	Buona stabilità	Buona, con miglioramenti nella separazione dei compiti
Struttura della rete	Architettura semplificata derivata da YOLOv4	Separazione esplicita tra classificazione e regressione
Dipendenza da iperparametri	Elevata (ancore da configurare)	Ridotta (assenza di ancore)
Adattabilità a dataset piccoli	Limitata, rischio di rigidità	Maggiore adattabilità e flessibilità
Adeguatezza al contesto applicativo	Soluzione consolidata ma meno flessibile	Più adatta a scenari controllati e con poche classi

Tabella 3.1: Confronto concettuale tra YOLOv4 Tiny e YOLOX Tiny

3.9 Struttura generale dello script di training

Lo scopo dello script è costruire un flusso completo e ripetibile per l'addestramento di un detector di object detection in MATLAB, a partire da un dataset annotato (ruota e albero), fino alla valutazione quantitativa delle prestazioni e al salvataggio del modello finale.

Nel progetto, questo passaggio è cruciale perché l'output della rete (bounding box del pezzo nell'immagine RGB) abilita la fase successiva della pipeline: estrazione della ROI nella mappa di profondità allineata e ricostruzione 3D per la misura dimensionale. Di conseguenza, lo script non è pensato solo per "allenare

una rete”, ma per produrre un modello utilizzabile operativamente e verificabile.

3.9.1 Caricamento del dataset e salto del training

Il primo blocco dello script implementa una logica molto utile in sperimentazioni iterative: se sul disco esiste già un file *.mat* con un detector addestrato, lo script lo carica e salta l'intera fase di training.

Questo approccio è stato scelto per motivi pratici:

- l'addestramento di una rete, anche “Tiny”, richiede tempo e risorse;
- durante lo sviluppo si eseguono molte prove (modifica dataset, augmentations, opzioni di training);
- è importante poter riprendere rapidamente l'analisi (test, inferenza, metriche) senza dover ri-addestrare ogni volta.

In altre parole, lo script distingue due modalità operative:

- modalità training (quando il modello non esiste ancora)
- modalità riuso/test (quando il modello è già disponibile)

Questa soluzione aumenta la produttività e riduce la variabilità accidentale (ad esempio piccoli cambiamenti dovuti a run differenti). Lo script non legge direttamente le etichette dall'*Image Labeler* in forma grezza, ma parte da un file *.mat* generato da uno “script di conversione” esterno. Questo significa che è stata introdotta una fase intermedia che normalizza e rende uniforme la struttura del dataset.

Il motivo è tecnico e molto comune in MATLAB: le funzioni di training per object detection si aspettano formati di input precisi (ad esempio tabella con *imageFilename* e colonne di *bounding box* per classe). L'*Image Labeler* salva le annotazioni in un formato che spesso necessita di essere trasformato per diventare “training-ready”.

Subito dopo il caricamento, lo script:

- controlla che esista la variabile *trainingData*;
- verifica che le colonne attese siano presenti (*imageFilename*, *albero*, *ruota*);
- controlla che i file immagine esistano realmente sul disco.

3.9.2 Normalizzazione delle bounding box

Una parte consistente dello script serve a garantire che le bounding box siano nel formato più robusto per MATLAB: una cella per immagine, contenente una matrice $N \times 4$ (o 0×4 se non ci sono oggetti) con le box nel formato standard $[x,y,w,h]$.

Questa normalizzazione è stata necessaria perché nella pratica i dataset possono assumere forme diverse:

- alcune pipeline producono bounding box come matrici numeriche;
- altre producono celle;
- a volte un'immagine può contenere zero, uno o più oggetti.

Forzare tutto in *cell* con $N \times 4$ evita incompatibilità con *boxLabelDatastore* e con le funzioni di training/aumentazione, e rende il dataset gestibile anche in presenza di casi limite (immagini senza box, box vuote, ecc.). È una scelta che aiuta a prevenire errori a runtime e rende la pipeline più stabile.

3.10 Creazione dei datastore MATLAB

Una volta sistemata la tabella, lo script costruisce:

- un *imageDatastore* per le immagini;
- un *boxLabelDatastore* per le bounding box;
- e li combina con *combine()*.

Questa è la struttura standard di MATLAB per training con dati “paired” (immagine e label assieme). Il vantaggio è duplice:

1. **efficienza**: i datastore gestiscono caricamento lazy, batch e accesso incrementale;
2. **compatibilità**: le funzioni di training di MATLAB lavorano nativamente con questa interfaccia.

3.10.1 Verifica visiva del dataset prima del training (preview)

Lo script include una fase di *preview* che visualizza alcune immagini con bounding box sovrapposte.

Questa parte è più importante di quanto sembri: in un dataset di object detection, piccoli errori di labeling (box fuori posto, scambi di classi, box mancanti) possono distruggere la qualità del training, specialmente con dataset piccoli come quello del progetto.

La preview ha quindi lo scopo di:

- verificare rapidamente che le classi (“albero”, “ruota”) siano corrette;
- controllare che le box siano nel formato giusto;
- individuare errori grossolani prima di sprecare tempo computazionale in training.

3.11 Suddivisione Train / Validation

3.11.1 Strategie di suddivisione del dataset

Lo script divide il dataset in:

- **training set (80%)** : usato per ottimizzare i pesi della rete;
- **validation set (20%)**: usato per misurare generalizzazione e scegliere impostazioni.

Questa scelta è “standard”, ma in questo progetto ha una motivazione specifica: la rete deve funzionare in condizioni operative realistiche, quindi serve un controllo sul rischio di overfitting. Con un dataset relativamente piccolo (100 immagini per classe), la separazione train/validation è essenziale per interpretare correttamente le prestazioni.

3.11.2 Creazione del detector Tiny pre-addestrato

Il detector viene costruito a partire da un modello pre-addestrato su COCO (tiny-coco). Questa scelta è tipica del transfer learning: invece di allenare da zero, si riadattano i pesi di una rete che ha già imparato feature generiche utili (bordi, texture, pattern).

Con dataset piccoli, il transfer learning è praticamente obbligatorio se si vuole ottenere un modello stabile. Lo script definisce `classNames = [“albero”, “ruota”]` e crea un detector finale che sostituisce lo strato di classificazione del modello pre-addestrato per farlo lavorare sulle due classi specifiche.

Questo passaggio è il ponte tra una rete general-purpose e una rete “mirata” per un contesto industriale ristretto.

3.12 Impostazione delle metriche e delle opzioni di training

3.12.1 mAP@0.5 come metrica

Lo script usa come metrica mAP@50 (*mean Average Precision con IoU=0.5*). È una scelta che combina:

- Capacità di classificare correttamente;
- capacità di localizzare correttamente

Nel progetto la bounding box serve come ROI (*Region Of Interest*) per estrarre profondità e ricostruzione: quindi una buona mAP@0.5 indica che la rete sta localizzando l’oggetto in modo sufficientemente accurato per abilitare la fase 3D.

Entrando nel dettaglio, tale metrica rappresenta uno standard consolidato nella letteratura di computer vision e viene adottata anche negli ambienti di sviluppo come MATLAB per la valutazione di modelli basati su YOLO.

Formalmente, la mAP@50 è definita come la media delle Average Precision calcolate per ciascuna classe del problema:

$$\text{mAP@50} = \frac{1}{C} \sum_{c=1}^C AP_c^{(\text{IoU} \geq 0.5)} \quad (3.1)$$

dove (C) rappresenta il numero totale di classi e (AP_c) è la Average Precision associata alla classe (c), calcolata considerando esclusivamente le predizioni che soddisfano la condizione ($\text{IoU} \geq 0.5$).

L'Average Precision per ciascuna classe è definita come l'area sottesa alla curva Precision-Recall:

$$AP_c = \int_0^1 p_c(r), dr \quad (3.2)$$

dove ($p_c(r)$) rappresenta la precisione in funzione del recall per la classe (c).

Dal punto di vista operativo, il calcolo della mAP@50 avviene attraverso i seguenti passaggi. Per ciascuna classe, le predizioni del modello vengono ordinate in base al livello di confidenza. Successivamente, ogni predizione viene classificata come True Positive (TP) o False Positive (FP) in funzione della soglia di sovrapposizione con il ground truth, definita tramite l'Intersection over Union:

$$\text{IoU} = \frac{\text{Area di intersezione}}{\text{Area di unione}} \quad (3.3)$$

Una predizione è considerata corretta se soddisfa la condizione ($\text{IoU} \geq 0.5$). A partire da tali classificazioni, vengono calcolate le metriche di precisione e recall:

$$P = \frac{TP}{TP + FP} \quad (3.4)$$

$$R = \frac{TP}{TP + FN} \quad (3.5)$$

La curva Precision-Recall viene quindi costruita variando la soglia di confidenza del modello, e l'area sottesa a tale curva fornisce il valore di (AP_c). Infine,

la media delle Average Precision su tutte le classi produce la metrica complessiva mAP@50.

Nel contesto di MATLAB, funzioni come `evaluateDetectionPrecision` o `evaluateObjectDetection` implementano automaticamente tale procedura, restituendo sia le Average Precision per ciascuna classe sia il valore medio complessivo. Nel caso specifico considerato in questa tesi, la metrica mAP@50 risulta particolarmente adeguata, in quanto consente di valutare simultaneamente la correttezza della classificazione e l'accuratezza della localizzazione degli oggetti rilevati, garantendo una misura sintetica e interpretabile delle prestazioni del modello.

3.12.2 Scelte di training

La definizione delle opzioni di training rappresenta una fase fondamentale nello sviluppo di una rete neurale affidabile, poiché influisce in modo diretto sulla stabilità del processo di apprendimento, sulla velocità di convergenza e sulla capacità del modello di generalizzare correttamente a dati non visti. Nel progetto in esame, particolare attenzione è stata dedicata alla scelta dei parametri di addestramento, con l'obiettivo di privilegiare un comportamento robusto e controllabile della rete, piuttosto che una convergenza rapida ma potenzialmente instabile.

Il metodo di ottimizzazione adottato è basato su Stochastic Gradient Descent with Momentum (SGDM), una tecnica ampiamente utilizzata nel training di reti convoluzionali per problemi di visione artificiale. L'uso del momento consente di smorzare le oscillazioni del gradiente e di favorire una discesa più regolare lungo la superficie della funzione di perdita, riducendo il rischio di rimanere intrappolati in minimi locali poco significativi. Questa caratteristica risulta particolarmente importante in presenza di dataset di dimensioni contenute, come nel caso in esame, dove il rumore statistico può influenzare maggiormente l'andamento dell'ottimizzazione.

Il valore del learning rate iniziale, impostato a 10^{-4} , è stato scelto in modo conservativo, al fine di evitare aggiornamenti eccessivamente aggressivi dei pesi della

rete. In un contesto applicativo industriale, un learning rate troppo elevato può portare a instabilità del training o a una convergenza verso soluzioni subottimali. La scelta di un valore relativamente basso consente invece di ottenere un apprendimento più graduale e controllato, favorendo una migliore generalizzazione del modello.

Per migliorare ulteriormente la stabilità del processo di training, è stata adottata una strategia di riduzione del learning rate a gradini (*piecewise schedule*). In questo approccio, il learning rate viene diminuito dopo un certo numero di epoche, consentendo alla rete di effettuare aggiustamenti più fini dei pesi nelle fasi finali dell'addestramento. Questa strategia riflette una pratica consolidata nel deep learning, secondo cui le prime epoche servono a individuare una regione promettente dello spazio dei parametri, mentre le epoche successive permettono di raffinare la soluzione. Un altro aspetto rilevante riguarda la scelta della dimensione del mini-batch, fissata a un valore relativamente ridotto. Un mini-batch di dimensioni contenute introduce una certa variabilità nel gradiente stimato, che può agire come una forma implicita di regolarizzazione, riducendo il rischio di overfitting. Allo stesso tempo, questa scelta è coerente con i vincoli hardware del sistema, evitando un utilizzo eccessivo della memoria e rendendo il training eseguibile anche in assenza di GPU ad alte prestazioni.

La robustezza del training è ulteriormente garantita dall'introduzione di un controllo sul valore del gradiente, attraverso il *gradient thresholding*. Limitare la norma del gradiente consente di prevenire fenomeni di instabilità numerica, come l'esplosione del gradiente, che possono compromettere l'intero processo di addestramento. Questo accorgimento risulta particolarmente utile quando si utilizzano reti profonde o quando si lavora con dati reali, potenzialmente affetti da rumore e variazioni non ideali.

Un ruolo centrale nel processo di addestramento è svolto dalla validazione periodica del modello. Durante il training, la rete viene valutata a intervalli regolari su un insieme di dati di validazione, separato dal training set. Questa procedura consente di monitorare l'evoluzione delle prestazioni del modello su dati non utilizzati per l'aggiornamento dei pesi, fornendo un'indicazione diretta della ca-

pacità di generalizzazione. Nel progetto, la metrica principale utilizzata per la validazione è la mAP@0.5, scelta in quanto rappresentativa sia della correttezza della classificazione sia della precisione nella localizzazione delle bounding box. L'uso della mAP@0.5 come metrica obiettivo durante il training consente di allineare il processo di ottimizzazione agli obiettivi applicativi del sistema. Poiché l'output della rete neurale viene utilizzato per definire una regione di interesse destinata a successive elaborazioni tridimensionali, è essenziale che la localizzazione dell'oggetto sia sufficientemente accurata, anche più della sola classificazione. In questo senso, la metrica adottata riflette in modo diretto le esigenze del progetto.

Infine, la configurazione delle opzioni di training prevede una gestione esplicita dell'ambiente di esecuzione, consentendo di scegliere tra CPU e GPU in funzione delle risorse disponibili. Questa flessibilità rende lo script riutilizzabile su piattaforme hardware diverse e coerente con l'obiettivo di sviluppare una soluzione trasferibile in contesti industriali reali, dove l'hardware disponibile può variare significativamente.

Nel complesso, le scelte adottate per le opzioni di training riflettono una filosofia progettuale orientata alla robustezza, alla riproducibilità e al controllo del processo di apprendimento. Piuttosto che massimizzare la velocità di convergenza o ottimizzare parametri in modo aggressivo, l'approccio seguito privilegia un addestramento stabile e interpretabile, in linea con le esigenze di un sistema di visione artificiale destinato a supportare operazioni di misura e controllo qualità.

3.13 Training del detector e valutazione delle prestazioni

La fase di addestramento del detector rappresenta un momento cruciale per comprendere il comportamento delle diverse architetture considerate e per valutare in modo oggettivo la loro idoneità al problema specifico affrontato nel progetto. I grafici riportati in questa sezione mostrano in modo sintetico ma significativo l'andamento del processo di training e le prestazioni ottenute in validazione per

le due reti analizzate, YOLOv4 Tiny e YOLOX Tiny, consentendo un confronto diretto sia dal punto di vista qualitativo sia quantitativo.

Nel caso di YOLOv4 Tiny, il grafico relativo al training (figura 3.5) evidenzia una riduzione progressiva della funzione di perdita nel corso delle epoche. Questo andamento suggerisce che la rete riesce a ottimizzare correttamente la funzione obiettivo sui dati di training, mostrando un comportamento stabile e privo di oscillazioni marcate.

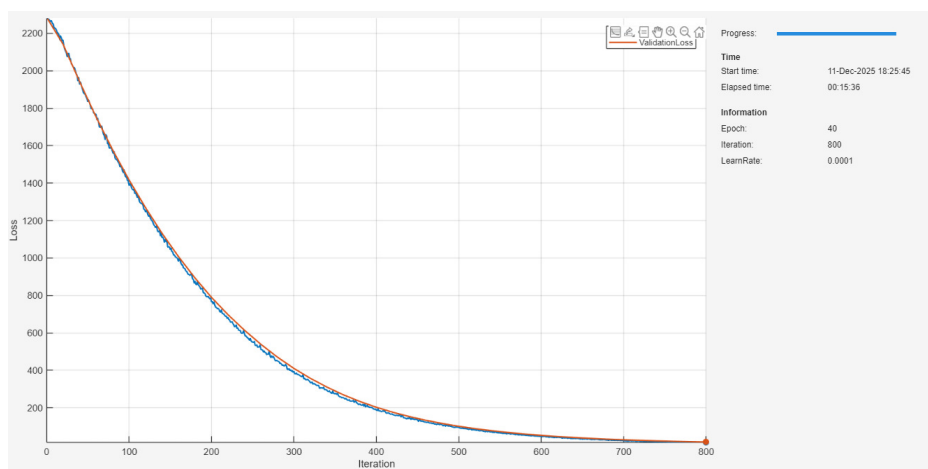


Figura 3.5: Andamento Training YOLO-v4

Tuttavia, l'analisi congiunta delle curve di training e dei risultati in validazione mette in evidenza una discrepanza significativa tra l'ottimizzazione interna del modello e le prestazioni effettive di detection. In particolare, nonostante la diminuzione della loss, la metrica mAP@0.5 calcolata sul validation set rimane su valori relativamente contenuti, con una marcata differenza tra le due classi considerate.

Il grafico per classe della mAP (fig. 3.6) relativo a YOLOv4 Tiny mostra infatti come la rete riesca a riconoscere la classe "ruota" con una precisione maggiore rispetto alla classe "albero", suggerendo una difficoltà nel modellare in modo uniforme entrambe le tipologie di oggetti. Questo comportamento può essere interpretato come un segnale di limitata capacità di generalizzazione, probabilmente influenzata dalla struttura anchor-based dell'architettura e dalla necessità di un tuning più accurato degli anchor in funzione delle dimensioni e delle proporzioni specifiche dei pezzi meccanici osservati. In un dataset relativamente piccolo e

specializzato, come quello utilizzato nel progetto, tali aspetti possono incidere in modo significativo sulle prestazioni finali.

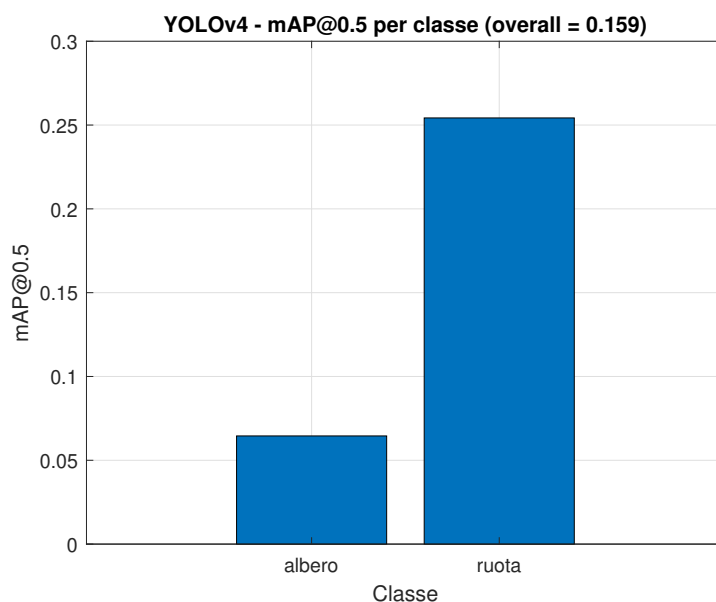


Figura 3.6: Rappresentazione mAP per classi (YOLO-v4)

Un quadro sensibilmente diverso emerge dall'analisi dei grafici relativi a YOLOX Tiny. L'andamento del training mostra una rapida convergenza della loss nelle prime epoche, seguita da una fase di stabilizzazione su valori contenuti. Questo comportamento indica che la rete riesce a individuare rapidamente una soluzione efficace nello spazio dei parametri, senza manifestare instabilità numeriche o difficoltà di convergenza. La regolarità dell'andamento suggerisce inoltre che le scelte adottate per le opzioni di training, come il learning rate e la strategia di validazione, risultano adeguate al problema affrontato.

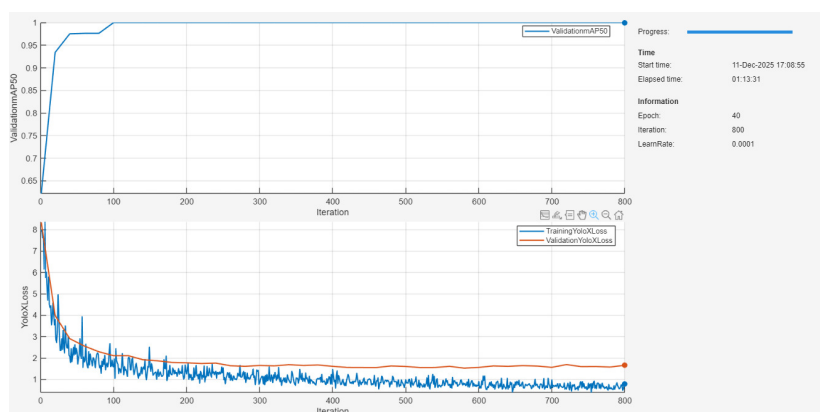


Figura 3.7: Andamento Training YOLOX

La mAP50 di figura 3.7 cresce rapidamente durante il training e raggiunge valori prossimi all'unità, mantenendosi stabile nelle epoche successive. Il diagramma a barre della mAP (fig. 3.8) per classe mostra inoltre una distribuzione uniforme delle prestazioni tra “albero” e “ruota”, indicando che la rete è in grado di apprendere in modo coerente le caratteristiche distintive di entrambe le classi. Questo risultato evidenzia una migliore capacità di generalizzazione rispetto a YOLOv4 Tiny e suggerisce che l'architettura YOLOX Tiny sia maggiormente adatta a gestire il tipo di dati e di oggetti presenti nel dataset.

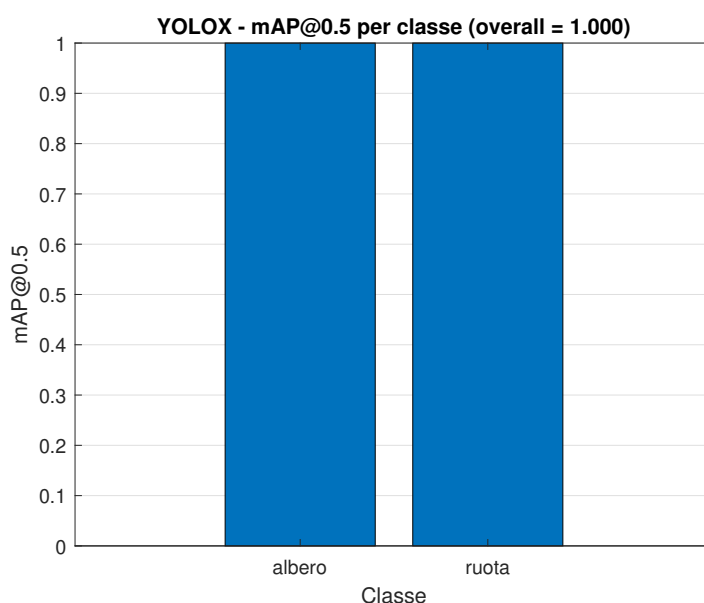


Figura 3.8: Rappresentazione mAP per classi (YOLOX)

Il confronto diretto tra i quattro grafici consente quindi di trarre alcune considerazioni di carattere generale. In primo luogo, emerge chiaramente come una buona convergenza della loss non sia di per sé sufficiente a garantire elevate prestazioni applicative. Il caso di YOLOv4 Tiny dimostra che una rete può apparire ben addestrata dal punto di vista dell'ottimizzazione interna, ma risultare meno efficace quando valutata su dati non visti. In secondo luogo, l'architettura del modello gioca un ruolo determinante nel modo in cui le informazioni vengono apprese e generalizzate. L'approccio anchor-free di YOLOX Tiny, unito alla separazione tra il ramo di classificazione e quello di regressione delle bounding box, sembra offrire una maggiore flessibilità e una migliore adattabilità a dataset piccoli e altamente specializzati.

Nel contesto del progetto, questi risultati assumono un significato particolarmente rilevante. Poiché il detector non costituisce il fine ultimo del sistema, ma rappresenta un passaggio intermedio per la definizione della regione di interesse destinata alle successive elaborazioni tridimensionali, è fondamentale che le bounding box prodotte siano affidabili, stabili e coerenti tra le diverse classi. Un detector che fornisce prestazioni elevate e bilanciate in validazione riduce il rischio di errori nella selezione della ROI e contribuisce direttamente alla qualità delle misure geometriche effettuate nelle fasi successive.

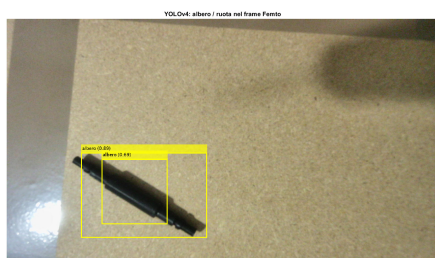
In conclusione, l'analisi combinata del training e della valutazione quantitativa in validazione evidenzia in modo chiaro la superiorità di YOLOX Tiny rispetto a YOLOv4 Tiny nel contesto applicativo considerato. I grafici mostrano come YOLOX Tiny non solo converga più rapidamente, ma produca anche risultati più consistenti e interpretabili, rendendolo la scelta più appropriata per l'integrazione nel sistema di visione industriale sviluppato.

3.14 Risultati finali

I risultati finali ottenuti dall'applicazione dei due detector addestrati, YOLOv4 Tiny e YOLOX Tiny, consentono di effettuare una valutazione conclusiva non solo dal punto di vista quantitativo, ma anche qualitativo, osservando direttamente il comportamento delle reti su immagini reali acquisite dalla telecamera Orbbec Femto Bolt. Le immagini riportate in questa sezione mostrano esempi rappresentativi dell'output dei due modelli, evidenziando le differenze nella capacità di riconoscimento e localizzazione dei pezzi meccanici “albero” e “ruota”.

3.14.1 Valutazione YOLO-V4

Dalla figura 3.9a, relativa ai risultati di YOLOv4 Tiny, è possibile osservare come il detector riesca a individuare correttamente il pezzo presente nella scena, ma presenti alcune criticità. In particolare, la rete genera più bounding box sovrapposte associate alla stessa classe, ciascuna con un diverso valore di confidenza. Questo comportamento suggerisce una minore precisione nella localizzazione dell'oggetto e una certa incertezza nella stima della bounding box ottimale. Sebbene il pezzo venga riconosciuto, la presenza di box multiple ridondanti può complicare le fasi successive della pipeline, in cui è necessario selezionare una singola regione di interesse affidabile per l'estrazione della profondità e la ricostruzione tridimensionale.



(a) Doppia bounding box



(b) Solo una classe trovata

Figura 3.9: Risultati post training con YOLO-v4

Nel caso della figura 3.9b, YOLOv4 Tiny mostra una maggiore stabilità, con una bounding box più coerente e un valore di confidenza elevato. Tuttavia, anche

in questo scenario, il comportamento del detector risulta meno uniforme tra le diverse classi, confermando quanto emerso dall'analisi quantitativa delle metriche di validazione. La differenza di prestazioni tra “albero” e “ruota” riflette una capacità di generalizzazione non omogenea, già evidenziata dai valori di mAP@0.5 relativamente bassi e sbilanciati ottenuti durante la fase di valutazione.

3.14.2 Valutazione YOLOX

Dall'immagine 3.10, si deduce, invece, un comportamento nettamente più robusto e coerente. Entrambi i pezzi meccanici presenti nella scena vengono riconosciuti correttamente e localizzati mediante bounding box ben definite, ciascuna associata a un valore di confidenza elevato. In questo caso, il detector produce una singola bounding box per ciascun oggetto, senza ridondanze o ambiguità, facilitando in modo diretto l'individuazione della regione di interesse da utilizzare nelle fasi successive di elaborazione tridimensionale. Questo risultato qualitativo

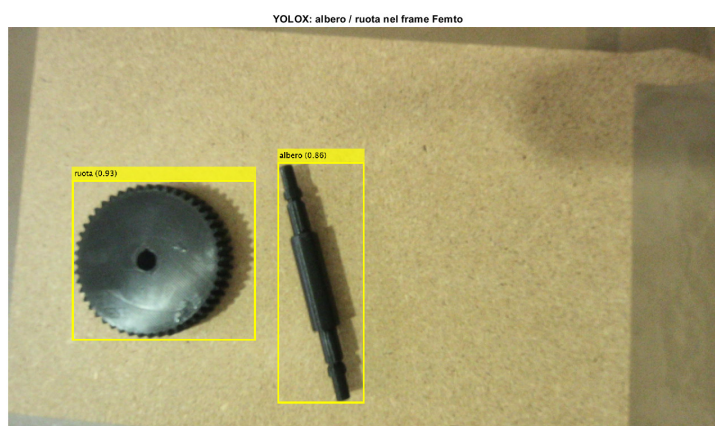


Figura 3.10: Risultato YOLOX

risulta pienamente coerente con quanto osservato nei grafici di training e di validazione. L'elevata mAP@0.5 ottenuta da YOLOX Tiny, pari a 1.0 su entrambe le classi nel validation set, trova infatti riscontro diretto nella stabilità e nella precisione delle predizioni mostrate nelle immagini. La rapidità di convergenza della rete e la stabilizzazione delle metriche durante l'addestramento si traducono

in un comportamento affidabile anche in fase di inferenza su dati reali.

Il confronto visivo tra i risultati di YOLOv4 Tiny e YOLOX Tiny evidenzia quindi come le differenze architetturali tra i due modelli abbiano un impatto concreto sulle prestazioni applicative. L'approccio anchor-free di YOLOX Tiny e la separazione dei rami di classificazione e regressione consentono alla rete di adattarsi meglio al dataset specifico del progetto, riducendo l'incertezza nella localizzazione degli oggetti e migliorando la coerenza delle predizioni tra classi diverse.

In conclusione, l'analisi congiunta dei risultati quantitativi, dei grafici di training e delle immagini di output conferma in modo consistente la superiorità di YOLOX Tiny rispetto a YOLOv4 Tiny per il problema affrontato. I risultati finali mostrano come YOLOX Tiny rappresenti la scelta più efficace e affidabile per il riconoscimento automatico dei pezzi meccanici nel contesto applicativo considerato, risultando pienamente coerente con gli obiettivi di robustezza, precisione e integrazione del sistema di visione industriale sviluppato.

Capitolo 4

Riconoscimento posizione e distanza

4.1 Ruolo del riconoscimento nella pipeline

Nel sistema di visione sviluppato, la fase di riconoscimento del pezzo rappresenta il collegamento tra l'acquisizione dell'immagine RGB e le successive elaborazioni tridimensionali basate sulla mappa di profondità. L'obiettivo di questa fase non è soltanto identificare la presenza di un determinato componente meccanico nella scena, ma anche fornire una localizzazione sufficientemente accurata da consentire l'estrazione di una regione di interesse (ROI) affidabile per la ricostruzione tridimensionale e la misura dimensionale.

Il riconoscimento viene quindi concepito come una fase abilitante dell'intera pipeline: la qualità delle bounding box prodotte dalla rete neurale influisce direttamente sulla correttezza delle misure geometriche effettuate a valle. Per questo motivo, il processo di detection è stato progettato con particolare attenzione alla robustezza, alla stabilità e alla coerenza delle informazioni restituite.

4.1.1 Gestione dell'ambiente di esecuzione

Un primo aspetto rilevante della funzione riguarda la gestione del detector e dell'ambiente di esecuzione. Il modello di rete neurale non viene ricaricato a ogni

chiamata della funzione, ma viene gestito in modalità lazy loading: il detector viene caricato una sola volta e mantenuto in memoria per le elaborazioni successive, mediante una variabile persistente [28].

Questa scelta progettuale risponde a due esigenze principali. Da un lato, riduce il carico computazionale e i tempi di risposta del sistema, evitando operazioni ripetitive e costose. Dall'altro, garantisce coerenza tra le elaborazioni successive, assicurando che tutte le inferenze vengano eseguite utilizzando lo stesso modello e la stessa configurazione.

Il programma sviluppato aggiorna inoltre lo stato del sistema, riportando informazioni sul modello utilizzato e sull'ambiente di calcolo selezionato (CPU o GPU). La disponibilità di una GPU compatibile può essere verificata a runtime, permettendo di selezionare in modo controllato l'ambiente di esecuzione più adatto [29]. Questo aspetto, pur non influenzando direttamente il riconoscimento, contribuisce alla tracciabilità e alla trasparenza del sistema, caratteristiche particolarmente importanti in un contesto applicativo e sperimentale.

4.1.2 Raffinamento delle bounding box

Nel contesto dei sistemi di object detection basati su reti neurali convoluzionali, le bounding box rappresentano una stima approssimata della posizione e dell'estensione spaziale degli oggetti all'interno dell'immagine. Sebbene le moderne architetture di deep learning, come YOLOX, siano in grado di fornire localizzazioni accurate, le bounding box prodotte devono essere interpretate come stime probabilistiche, influenzate sia dalle caratteristiche del dataset sia dalla funzione di perdita utilizzata durante il training.

Dal punto di vista teorico, la bounding box è una rappresentazione rettangolare che approssima l'area occupata dall'oggetto, ma non ne descrive in modo esatto la geometria. In applicazioni di puro riconoscimento o tracking, tale approssimazione è spesso sufficiente. Tuttavia, in un sistema orientato alla misura tridimensionale e alla metrologia senza contatto, anche piccoli errori di localizzazione possono avere un impatto significativo sulle elaborazioni successive, in

particolare quando la bounding box viene utilizzata come regione di interesse per l'estrazione della mappa di profondità.

Per questo motivo, il raffinamento delle bounding box assume un ruolo cruciale all'interno della pipeline sviluppata. L'obiettivo di questa fase non è migliorare il riconoscimento in senso stretto, ma aumentare la coerenza geometrica tra la regione individuata dalla rete neurale e l'effettiva estensione fisica del pezzo osservato.

4.1.3 Limiti intrinseci della localizzazione

Le reti di object detection apprendono a localizzare gli oggetti ottimizzando una funzione di perdita che bilancia diversi obiettivi, tra cui classificazione e regressione delle coordinate della bounding box. Questa formulazione comporta inevitabilmente un compromesso: la rete privilegia una localizzazione sufficientemente accurata in media, ma non garantisce un allineamento preciso con i bordi reali dell'oggetto in ogni singola immagine.

Inoltre, fattori come variazioni di illuminazione, riflessi superficiali, rumore dell'immagine e parziale sovrapposizione con lo sfondo possono influenzare la posizione stimata della bounding box. Dal punto di vista teorico, ciò significa che la bounding box fornita dalla rete può includere porzioni di sfondo oppure escludere parti marginali dell'oggetto, introducendo un errore sistematico nella definizione della regione di interesse.

Questi limiti diventano particolarmente critici quando la bounding box viene utilizzata come base per operazioni geometriche, come il calcolo di dimensioni metriche a partire da una mappa di profondità. In tali casi, l'errore bidimensionale si propaga nello spazio tridimensionale, compromettendo l'accuratezza delle misure.

4.1.4 Segmentazione e adattamento alla geometria

Un approccio concettualmente efficace al raffinamento delle bounding box consiste nell'utilizzare tecniche di segmentazione per separare il pezzo dallo sfondo all'interno della regione inizialmente individuata. Dal punto di vista teorico, la segmentazione consente di passare da una rappresentazione rettangolare grossolana a una descrizione più aderente alla forma reale dell'oggetto.

Anche quando non viene utilizzata una segmentazione completa a livello di pixel, è possibile sfruttare informazioni locali, come variazioni di intensità, contorni o regioni omogenee, per adattare le dimensioni della bounding box. Questo processo riduce l'influenza di elementi estranei presenti nella scena e migliora la corrispondenza tra bounding box e oggetto fisico.

Nel contesto di pezzi meccanici, spesso caratterizzati da geometrie ben definite e superfici relativamente uniformi, tali tecniche risultano particolarmente efficaci. La presenza di bordi netti e di differenze cromatiche rispetto allo sfondo facilita l'individuazione di regioni coerenti, rendendo il raffinamento una strategia affidabile e ripetibile.

4.2 Il raffinamento delle bounding box utilizzato

La funzione di raffinamento adottata nel progetto implementa un approccio di post-processing locale sulle bounding box prodotte dal detector (YOLOX). In una pipeline di object detection, la bounding box restituita dalla rete neurale rappresenta una stima robusta della localizzazione dell'oggetto, ma non garantisce necessariamente un allineamento preciso con i bordi reali del pezzo. In ambito metrologico, questa approssimazione può diventare critica: porzioni di sfondo incluse nella ROI o la perdita di parti dell'oggetto possono introdurre outlier nella nuvola di punti estratta dalla profondità e, di conseguenza, degradare l'accuratezza delle misure dimensionali.

Per mitigare questo problema, la funzione esegue una segmentazione all'interno di una regione ristretta attorno alla bounding box originaria e ricava da essa una nuova bounding box più aderente alla regione effettivamente occupata dall'oggetto. Questo permette di coniugare i vantaggi del deep learning (robustezza e capacità di riconoscere l'oggetto anche in condizioni non ideali) con la precisione di un metodo deterministico locale (adattamento geometrico ai contorni reali).

4.2.1 Principio di base

Il punto chiave dell'approccio è che la segmentazione non viene applicata all'intera immagine, ma solo a una ROI ampliata attorno alla bounding box stimata dal detector. Questo schema può essere visto come una strategia *coarse-to-fine*:

- **Coarse localization:** la rete neurale individua una zona plausibile in cui si trova l'oggetto.
- **Fine refinement:** all'interno di quella zona, una segmentazione a basso costo computazionale delimita la regione dell'oggetto con maggiore precisione.

Questa scelta è teoricamente vantaggiosa perché riduce drasticamente le ambiguità tipiche della segmentazione globale. In una scena industriale, infatti, la segmentazione sull'intera immagine risulta spesso instabile per effetto di ombre, riflessi e texture dello sfondo. Vincolando la segmentazione a una regione già "semantizzata" dal detector, si limita lo spazio delle soluzioni e si aumenta la probabilità di isolare correttamente il pezzo.

4.2.2 Pulizia morfologica

La segmentazione binaria grezza può contenere rumore (pixel isolati), piccoli fori all'interno della regione dell'oggetto o discontinuità lungo i bordi. Per stabilizzare il risultato, la funzione utilizza operazioni di **morfologia matematica** (apertura e chiusura), che hanno una motivazione teorica precisa:

- **Apertura (opening):** elimina piccole strutture e rumore, preservando la forma globale dell'oggetto.

- **Chiusura (closing):** colma piccole lacune e rende la regione più compatta.
- **Rimozione di componenti piccole:** elimina regioni spurie non compatibili con l'area attesa del pezzo.

Questi passaggi consentono di ottenere una maschera più pulita e strutturata, riducendo la sensibilità a disturbi locali. In un contesto industriale, tale stabilizzazione è essenziale per ottenere risultati ripetibili.

4.2.3 Margine di sicurezza e controllo vincoli

Un aspetto importante, in ottica applicativa, è l'introduzione di un margine di sicurezza attorno alla bounding box raffinata. Teoricamente, questa espansione controllata serve a evitare che una segmentazione troppo aggressiva (ad esempio dovuta a riflessi o a porzioni a basso contrasto) “tagli” parti dell'oggetto. In altre parole, l'algoritmo produce una bounding box più precisa ma conserva una tolleranza minima per garantire che l'oggetto rimanga interamente contenuto nella ROI.

Inoltre, vengono applicati vincoli per assicurare che la bounding box finale rimanga all'interno dell'immagine e che la ROI abbia dimensioni sufficienti per essere significativa. Questo rende la procedura robusta anche in presenza di bounding box iniziali piccole o parzialmente fuori campo.

4.2.4 Significato per la pipeline RGB-D

Il raffinamento tramite segmentazione locale produce un beneficio diretto sulle fasi successive della pipeline:

1. **ROI più pulita sulla profondità:** riducendo l'inclusione dello sfondo, diminuisce la quantità di punti non pertinenti nella nuvola 3D estratta dalla mappa depth.
2. **Maggiore ripetibilità metrologica:** la dimensione stimata del pezzo (lunghezza, larghezza, altezza) risulta meno influenzata da variazioni marginali nella localizzazione.

3. **Riduzione di ambiguità:** la selezione della componente principale stabilizza il comportamento del sistema quando la scena presenta dettagli o elementi secondari.

In sintesi, la funzione implementa un raffinamento coarse-to-fine che, partendo da una stima semantica affidabile (YOLOX), migliora la coerenza geometrica della ROI tramite segmentazione e morfologia. Una volta completata la fase di riconoscimento del pezzo e definita una regione di interesse affidabile mediante il raffinamento delle bounding box, il sistema dispone delle informazioni necessarie per passare da una descrizione puramente bidimensionale a una rappresentazione tridimensionale della scena. Tuttavia, affinché le misure geometriche ricavate dalla mappa di profondità risultino fisicamente interpretabili, è necessario introdurre un ulteriore elemento di riferimento: **il piano di appoggio** sul quale il pezzo è posizionato.

4.3 Calibrazione del piano

Nel contesto del sistema sviluppato, la profondità fornita dalla telecamera RGB-D rappresenta una distanza misurata lungo l'asse ottico della camera, e non una quota verticale rispetto a un piano di riferimento assoluto. Di conseguenza, la sola informazione di profondità non è sufficiente per determinare grandezze geometriche come l'altezza reale del pezzo rispetto al piano di appoggio. Per colmare questa lacuna, è necessario stimare in modo accurato la posizione e l'orientamento del piano su cui giace l'oggetto.

La calibrazione del piano costituisce quindi il passaggio che consente di collegare il riconoscimento del pezzo alla sua effettiva interpretazione geometrica nello spazio tridimensionale. Essa permette di separare concettualmente e operativamente la superficie di riferimento dallo spazio occupato dal pezzo, rendendo possibile una misura dimensionale coerente con la geometria reale dell'ambiente.

4.3.1 Principi teorici

La calibrazione del piano ha l'obiettivo di stimare i parametri geometrici di una superficie piana osservata dalla telecamera RGB-D, tipicamente il piano di appoggio su cui vengono collocati i pezzi meccanici. Dal punto di vista matematico, un piano nello spazio tridimensionale può essere descritto mediante un vettore normale $\mathbf{n}$ e una distanza d dall'origine del sistema di riferimento della camera, secondo l'equazione:

$$(\mathbf{n} \cdot \mathbf{X}) + d = 0 \quad (4.1)$$

dove $\mathbf{X} = (X_p, Y_p, Z_p)$ rappresenta un punto appartenente al piano espresso nel sistema di riferimento della camera. La funzione di calibrazione implementata nel progetto stima tali parametri a partire dalla mappa di profondità espressa in millimetri. Il principio alla base dell'algoritmo è quello di selezionare un insieme di punti tridimensionali appartenenti al piano di appoggio e di approssimarli mediante un modello planare, minimizzando l'errore quadratico complessivo.

4.4 Dalla mappa depth alla nuvola di punti

La calibrazione del piano di appoggio richiede innanzitutto la trasformazione della mappa di profondità, espressa in millimetri, in un insieme di punti tridimensionali nel sistema di riferimento della camera. La mappa $D=(u,v)$ associa a ciascuna coordinata pixel (u,v) un valore di profondità Z . Per effettuare la ricostruzione 3D, si utilizza il modello pinhole e i parametri intrinseci della camera (f_x, f_y, c_x, c_y) . Nel caso in cui la mappa di profondità abbia una risoluzione differente rispetto a quella per cui gli intrinseci sono definiti, tali parametri vengono scalati in base ai fattori:

$$s_x = \frac{W_d}{W_{ref}}, \quad s_y = \frac{H_d}{H_{ref}} \quad (4.2)$$

dove (W_d, H_d) rappresentano larghezza e altezza effettive della depth map, mentre (W_{ref}, H_{ref}) sono le dimensioni di riferimento (nel progetto 1920×1080). Di conseguenza:

$$f'_x = f_x \cdot s_x, \quad f'_y = f_y \cdot s_y, \quad c'_x = c_x \cdot s_x, \quad c'_y = c_y \cdot s_y \quad (4.3)$$

Una volta definiti gli intrinseci coerenti con la risoluzione reale, i pixel (u, v) campionati nella regione centrale dell'immagine (per ridurre l'influenza di bordi rumorosi e zone con dati mancanti) vengono convertiti in punti 3D tramite back-projection. Indicando con $Z = D(u, v)$ la profondità (in mm), le coordinate metriche nel riferimento camera (X, Y, Z) sono calcolate come:

$$X = \frac{(u - c'_x) \cdot Z}{f'_x}, \quad Y = \frac{(v - c'_y) \cdot Z}{f'_y} \quad Z = D(u, v) \quad (4.4)$$

L'insieme dei punti così ottenuti viene raccolto in una matrice:

$$P = \begin{bmatrix} X_1 & Y_1 & Z_1 \\ X_2 & Y_2 & Z_2 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ X_n & Y_n & Z_n \end{bmatrix} \in \mathbb{R}^{N \times 3} \quad (4.5)$$

dove $\mathbf{N}$ è il numero di campioni validi (campioni a profondità non nulla). Una volta ottenuta la matrice $\mathbf{P}$, contenente le coordinate tridimensionali dei punti appartenenti al piano di appoggio nel sistema di riferimento della camera, il problema della calibrazione può essere affrontato in termini puramente geometrici. L'obiettivo diventa infatti quello di individuare il piano che meglio approssima, in senso globale, l'insieme dei punti ricostruiti a partire dalla mappa di profondità.

Il primo passo consiste nel calcolare il punto medio della nuvola di punti. Questo punto rappresenta il baricentro geometrico dell'insieme dei campioni e fornisce un riferimento centrale attorno al quale analizzare la distribuzione spaziale dei dati. Il calcolo del baricentro consente di eliminare la dipendenza dalla posizione assoluta dei punti rispetto all'origine del sistema di riferimento della camera, permettendo di concentrarsi esclusivamente sulla loro disposizione relativa.

Successivamente, tutti i punti vengono considerati in forma centrata, ovvero traslati in modo tale che il baricentro coincida con l'origine. Questo passaggio è fondamentale per l'analisi statistica della nuvola di punti, poiché consente di descrivere la geometria del piano attraverso le direzioni principali di variazione dei dati. In altre parole, si studia come i punti si distribuiscono nello spazio indipendentemente dalla loro posizione media.

A questo punto, la nuvola di punti centrata viene analizzata mediante una decomposizione ai valori singolari. Dal punto di vista geometrico, tale operazione permette di individuare le tre direzioni ortogonali principali lungo le quali i punti presentano la maggiore e la minore dispersione. Due di queste direzioni giacciono approssimativamente sul piano di appoggio, mentre la terza, associata alla minima varianza, identifica la direzione ortogonale al piano stesso.

La direzione di minima dispersione viene quindi interpretata come il vettore normale del piano. Questa scelta ha una chiara giustificazione geometrica: se i punti appartengono idealmente a un piano, essi presentano una variabilità significativa lungo le direzioni contenute nel piano, ma una variabilità minima lungo la direzione perpendicolare ad esso. In presenza di rumore e imperfezioni nella misura della profondità, tale direzione non è esattamente nulla, ma rimane comunque identificabile come quella lungo cui la dispersione dei punti è minima.

Una volta determinata la normale del piano, il piano stesso viene completamente definito imponendo che esso passi per il baricentro della nuvola di punti. In questo modo si ottiene una rappresentazione compatta e coerente del piano di appoggio, espressa direttamente nel sistema di riferimento della camera. Questa rappresentazione permette di calcolare, in modo univoco, la distanza ortogonale tra qualsiasi punto tridimensionale e il piano stimato.

Un aspetto rilevante della calibrazione è il controllo dell'orientamento della normale del piano. In particolare, viene verificata la direzione della componente lungo l'asse ottico della camera, in modo da garantire una convenzione di segno coerente. Questo accorgimento risulta importante nelle fasi successive, in cui la distanza punto-piano viene interpretata come altezza del pezzo rispetto al piano di appoggio.

Infine, dall'orientamento della normale del piano è possibile ricavare l'angolo di inclinazione del piano rispetto all'asse Z del sistema di riferimento della camera. Tale informazione fornisce una misura diretta della geometria del setup di acquisizione e consente di verificare la coerenza tra il modello matematico e le caratteristiche fisiche del sistema. Nel caso della telecamera Orbbec Femto Bolt, l'inclinazione stimata risulta compatibile con l'inclinazione nominale dell'asse ot-

tico rispetto alla verticale, confermando la necessità di una correzione geometrica delle misure di profondità.

4.5 Inclinazione dell'asse Z - TILT

L'inclinazione del piano stimato è un indicatore fondamentale per valutare la correttezza della calibrazione e la stabilità meccanica del setup. In un sistema ideale (camera perfettamente orientata e piano perfettamente “fronto-parallelo” al sensore), la normale del piano risulta allineata con l'asse ottico della camera. In un contesto reale, invece, piccoli disallineamenti (tilt del sensore, non planarità, vibrazioni, fissaggi non rigidi) producono una rotazione della normale rispetto all'asse Z.

4.5.1 Definizione geometrica dell'angolo di tilt

Assumendo il sistema di riferimento della camera con asse ottico coincidente con

$$\mathbf{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad (4.6)$$

e indicando con $\mathbf{n}$ la normale del piano stimata, l'angolo di inclinazione θ tra la normale e l'asse ottico è definito dal prodotto scalare:

$$\cos(\theta) = \frac{\mathbf{n}^\top \mathbf{k}}{\|\mathbf{n}\| \|\mathbf{k}\|}. \quad (4.7)$$

Poiché $\|\mathbf{k}\| = 1$, si ottiene:

$$\cos(\theta) = \frac{n_z}{\|\mathbf{n}\|}. \quad (4.8)$$

Se $\mathbf{n}$ è normalizzata ($\|\mathbf{n}\| = 1$), allora:

$$\theta = \arccos(n_z). \quad (4.9)$$

Infine la conversione in gradi:

$$\theta_{\text{deg}} = \theta \cdot \frac{180}{\pi}. \quad (4.10)$$

Se $\theta \approx 0^\circ$ la normale del piano è quasi parallela all'asse Z : il piano risulta quasi perpendicolare alla direzione di vista della camera (configurazione tipica: camera “dall'alto” verso il piano).

Se θ cresce, significa che il piano (o la camera) sono inclinati: questa condizione può introdurre errori sistematici nella misura dell'altezza se il sistema assume un riferimento di quota globale non compensato.

4.5.2 Effetto del tilt sulle coordinate senza compensazione

Nel caso in cui non venga applicata alcuna compensazione del tilt, le coordinate dei punti rimangono espresse nel sistema di riferimento della camera, e la coordinata di profondità z continua a essere interpretata lungo l'asse ottico. Tuttavia, quando $\theta \neq 0$, tale asse non è più ortogonale al piano osservato.

Di conseguenza, la coordinata z non rappresenta più la distanza reale dal piano, ma una sua proiezione lungo la direzione di vista. In particolare, considerando un piano con normale $\mathbf{n} = [n_x; n_y; n_z]^\top$, la distanza ortogonale di un punto $\mathbf{p} = [x; y; z]^\top$ dal piano è:

$$h = \frac{\mathbf{n}^\top \mathbf{p} + d}{|\mathbf{n}|}. \quad (4.11)$$

Se però si utilizza direttamente la coordinata z come stima della quota, si introduce un errore sistematico che dipende dall'inclinazione del piano. Infatti, anche per punti appartenenti allo stesso piano, il valore di z varia in funzione delle coordinate x e y :

$$z = -\frac{n_x}{n_z}x - \frac{n_y}{n_z}y - \frac{d}{n_z}. \quad (4.12)$$

Questa espressione evidenzia che, in presenza di tilt ($n_x \neq 0$ o $n_y \neq 0$), la profondità misurata non è costante sul piano ma presenta un gradiente spaziale. In altre parole, il piano appare inclinato nel sistema della camera e la quota dipende linearmente dalla posizione nel campo visivo.

In termini di interpretazione geometrica, l'errore commesso utilizzando z al posto della distanza ortogonale è legato all'angolo θ tra la normale del piano e l'asse ottico. Per piccoli angoli, tale errore può essere approssimato come proporzionale a $\tan(\theta)$ e alla distanza laterale dal centro dell'immagine.

Pertanto, in assenza di compensazione del tilt, le misure di altezza risultano affette da una distorsione sistematica che cresce con:

- l'angolo di inclinazione θ ;
- la distanza dal centro ottico (coordinate x, y);
- la distanza dal piano (effetti prospettici).

Questo comportamento deve essere tenuto in considerazione quando si interpretano direttamente le mappe di profondità fornite dalla camera, in quanto può portare a errori anche significativi nelle stime di quota.

4.6 Dal riferimento di piano alla calibrazione RGB-Depth

La calibrazione del piano fornisce il primo riferimento metrologico globale del sistema: stabilisce quale superficie debba essere considerata “quota zero” per la misura delle altezze. Tuttavia, tale riferimento risulta utilizzabile solo se le misure di profondità associate alle regioni d'interesse degli oggetti sono coerenti con le coordinate in cui la rete neurale fornisce il rilevamento (bounding boxes o maschere).

Nel programma, la rete neurale opera sull'immagine RGB e restituisce una lista di oggetti con:

- **classe** (nel progetto sono: ruota, albero),
- **bounding box** in coordinate RGB,
- **punteggio di confidenza**.

La metrologia, invece, si basa sulla Depth, che può avere:

- **risoluzione** differente rispetto all'RGB
- **sistema di coordinate** differente (in assenza di allineamento),
- eventuali **distorsioni** e/o **traslazioni** dovute alla baseline tra i sensori.

Ne consegue che, per poter calcolare l'altezza di un oggetto a partire dalla ROI definita in RGB, è necessario un collegamento geometrico tra le coordinate RGB e quelle Depth. Questo collegamento costituisce il nucleo della cosiddetta “calibrazione RGB–Depth”, che si colloca logicamente a valle della calibrazione del piano.

Nel flusso del programma esistono due riferimenti fondamentali:

1. **Riferimento di quota** (piano): consente di interpretare i valori Depth come “distanza camera–punto” e di definire un livello di riferimento corrispondente al piano di appoggio.
2. **Riferimento di corrispondenza** (RGB–Depth): consente di associare correttamente le regioni d'immagine dove la rete individua l'oggetto alle regioni di profondità da cui estrarre la quota dell'oggetto.

La grandezza finale di interesse è l'altezza dell'oggetto sopra il piano. A livello concettuale, l'idea è confrontare la profondità del piano (distanza camera–piano) con la profondità della superficie superiore dell'oggetto (distanza camera–oggetto). In prima approssimazione, l'altezza dell'oggetto sopra il piano si ottiene come:

$$h = Z_{\text{piano}} - Z_{\text{oggetto}}. \quad (4.13)$$

dove: Z_{piano} è la quota del piano (in mm) espressa come distanza camera–piano, e Z_{oggetto} è una stima robusta della quota camera–oggetto ottenuta campionando i pixel Depth nella ROI dell'oggetto.

4.7 Calibrazione manuale RGB–Depth

Nel programma, la calibrazione manuale è stata implementata tramite una procedura a punti corrispondenti: l'utente seleziona manualmente una serie di punti sull'immagine RGB (figura 4.1) e gli stessi punti sulla mappa di profondità (figura 4.2) (visualizzata in modo da rendere identificabili i medesimi riferimenti). Tali corrispondenze consentono di stimare una trasformazione geometrica che mappa le coordinate RGB nel sistema Depth.

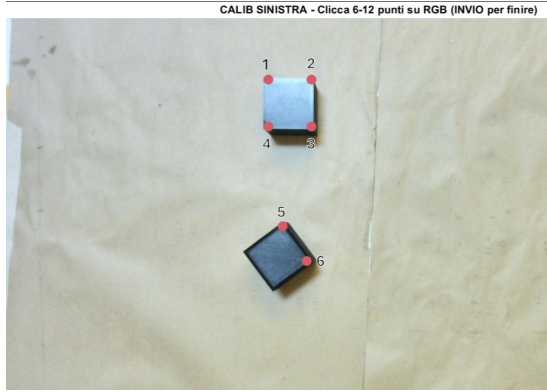


Figura 4.1: Punti su immagine RGB

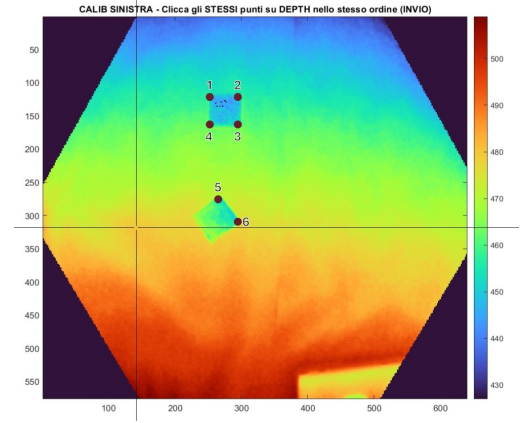


Figura 4.2: Stessi punti su DEPTH

4.7.1 Trasformazione affine RGB→Depth

Si adotta un modello affine 2D (in coordinate pixel), espresso in forma omogenea:

$$\begin{bmatrix} u_D \\ v_D \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}, \quad \mathbf{A} \in \mathbb{R}^{3 \times 3}. \quad (4.14)$$

Dove (u_R, v_R) sono coordinate pixel in RGB, (u_D, v_D) sono coordinate pixel in Depth e $\mathbf{A}$ contiene rotazione/scalatura, shear (cfr. Appendice C) e traslazione (entro i limiti del modello affine).

Operativamente separando le componenti:

$$u_D = a_{11}u_R + a_{12}v_R + a_{13}, \quad v_D = a_{21}u_R + a_{22}v_R + a_{23}. \quad (4.15)$$

Dati N punti, si costruisce:

$$\mathbf{U} = \begin{bmatrix} u_{R,1} & v_{R,1} & 1 \\ \vdots & \vdots & \vdots \\ u_{R,N} & v_{R,N} & 1 \end{bmatrix}, \quad (4.16)$$

e si stimano i parametri con minimi quadrati:

$$\mathbf{a} = \mathbf{U}^\dagger \mathbf{u}_D, \quad \mathbf{b} = \mathbf{U}^\dagger \mathbf{v}_D. \quad (4.17)$$

4.8 Limiti del modello affine singolo

In sistemi reali, un singolo modello affine può risultare insufficiente per almeno tre motivi:

1. la relazione RGB–Depth può contenere effetti prospettici e non linearità (distorsioni residue) non modellabili con affine;
2. la corrispondenza può variare lungo il campo visivo, producendo un errore che cresce ai bordi;
3. piccoli errori di selezione punti possono polarizzare la stima, specialmente se i punti non coprono uniformemente l'immagine.

Per ridurre l'errore “spazialmente variabile”, nel programma è stata adottata una strategia dual-affine: si stimano due trasformazioni affini distinte, una valida prevalentemente nella parte sinistra del FOV e una nella parte destra, e si effettua una fusione morbida (blend) in una zona di transizione.

4.9 Modello dual-affine con blending

Si stimano due matrici: A_{left} e A_{right} e si definisce un punto di split x_{split} e una larghezza di w_b .

Per un punto RGB (u_R, v_R) si calcolano:

$$\begin{bmatrix} u_{D,L} \\ v_{D,L} \\ 1 \end{bmatrix} = \mathbf{A}_{\text{left}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}, \quad \begin{bmatrix} u_{D,R} \\ v_{D,R} \\ 1 \end{bmatrix} = \mathbf{A}_{\text{right}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}. \quad (4.18)$$

Si definisce poi un coefficiente di blending $w(u_R) \in [0, 1]$ mediante una funzione “smoothstep” (cfr. Appendice D) (continua e con derivata continua agli estremi).

Una forma tipica, coerente con l'implementazione, è:

$$t = \text{clip} \left(\frac{u_R - (x_{\text{split}} - \frac{w_b}{2})}{w_b}, 0, 1 \right), \quad w = t^2(3 - 2t), \quad (4.19)$$

dove la variabile t è una coordinata normalizzata della posizione orizzontale u all'interno della fascia di transizione; la funzione $\text{clip}(\cdot, 0, 1)$ satura t nell'intervallo $[0, 1]$ così da imporre che, fuori dalla fascia, il peso di blending assuma

valori estremi (0 o 1), selezionando rispettivamente il modello sinistro o destro (cfr. Appendice E).

Infine si ottiene la proiezione dual-affine:

$$u_D = (1 - w) u_{D,L} + w u_{D,R}, \quad v_D = (1 - w) v_{D,L} + w v_{D,R}. \quad (4.20)$$

Questa strategia mantiene un modello semplice e computazionalmente leggero, ma riduce l'errore sistematico legato alla variazione spaziale della corrispondenza.

4.9.1 Difetti e limiti della calibrazione manuale

La calibrazione manuale dipende dalla capacità dell'operatore di selezionare punti realmente corrispondenti tra RGB e Depth. Anche con strumenti di visualizzazione adeguati, la selezione presenta un'incertezza intrinseca dovuta a:

- discretizzazione in pixel (errore di quantizzazione),
- rumore e buchi di depth,
- ambiguità visiva di alcuni riferimenti.

Questa componente umana riduce la ripetibilità della calibrazione tra sessioni diverse e, in scenari produttivi, impone di limitare la calibrazione manuale a casi di manutenzione o ritaratura. Il modello affine (anche dual-affine) non rappresenta rigorosamente distorsioni ottiche residue e/o deformazioni prospettiche non lineari, di conseguenza, l'errore può variare sul campo e può dipendere anche dalla distanza degli oggetti dal sensore.

Effetti sul calcolo delle altezze

La misura dell'altezza dipende dalla correttezza della ROI depth campionata. Se la mappa RGB→Depth presenta un errore anche di pochi pixel, la ROI può includere porzioni del piano o bordi dell'oggetto, alterando la stima del percentile e introducendo bias sull'altezza stimata. L'effetto è particolarmente critico quando:

1. le dimensioni dell'oggetto sono piccole rispetto alla risoluzione depth,

2. la ROI è vicina a discontinuità (spigoli, bordi),
3. la depth presenta rumore o zone invalid.

4.10 Calibrazione automatica RGB–Depth

La calibrazione automatica RGB–Depth ha l’obiettivo di rendere confrontabili, nello stesso sistema di riferimento immagine, due sorgenti di informazione eterogenee: l’immagine a colori (RGB) e la mappa di profondità (Depth). Nel contesto del presente lavoro, tale allineamento è un prerequisito fondamentale per la metrologia: la rete neurale effettua la segmentazione/detection sul frame RGB, mentre le altezze degli oggetti vengono stimate interrogando la profondità corrispondente ai pixel racchiusi nel bounding box rilevato.

In assenza di una calibrazione adeguata, anche piccoli disallineamenti tra i due stream (differenze di intrinseci, distorsioni, traslazioni tra ottiche, variazioni di FOV e risoluzione) si traducono in errori sistematici nella selezione dei pixel di profondità, con conseguente degradazione della stima dell’altezza. La calibrazione automatica mira quindi a:

- eliminare l’offset tra RGB e Depth e ridurre l’errore di proiezione pixel-a-pixel;
- garantire ripetibilità dell’allineamento senza intervento dell’operatore;
- stabilizzare il processo di misura in presenza di piccole variazioni operative (posizionamento, illuminazione, drift termico, ecc.).

4.10.1 Architettura del flusso software

La procedura automatica si inserisce in una pipeline che combina componenti eterogenei:

1. acquisizione sincrona (o quasi-sincrona) dei frame RGB e Depth tramite un eseguibile C++ basato su Orbbec SDK;

2. lettura e conversione della Depth in millimetri, utilizzando il fattore di scala fornito dai metadati del frame;
3. stima della quota del piano di appoggio (piano di riferimento) e dei parametri di conversione pixel→millimetri su RGB;
4. detection/riconoscimento sul frame RGB mediante rete neurale addestrata per le classi di interesse;
5. associazione bounding box→ROI di profondità e calcolo dell'altezza dell'oggetto rispetto al piano.

La calibrazione automatica RGB-Depth, in questo schema, è il meccanismo che assicura che la ROI di profondità interrogata corrisponda spazialmente alla ROI individuata in RGB.

4.10.2 Acquisizione automatica e allineamento

Il sistema utilizza un eseguibile C++ (realizzato su Orbbec SDK) per acquisire:

- un frame RGB ad alta risoluzione;
- un frame Depth in formato a 16 bit;
- un file di metadati contenente almeno le dimensioni della Depth e lo scale factor per la conversione in millimetri.

Il punto chiave è l'impiego delle funzionalità di *alignment* disponibili nell'SDK (filtro di allineamento Depth→Color), che producono una Depth già riportata nel sistema di riferimento immagine della camera colore. In altre parole, la Depth risultante è una mappa in cui ogni pixel è coerente con un pixel della RGB (stesso sistema di coordinate immagine), riducendo drasticamente la necessità di trasformazioni geometriche manuali.

Questa fase fornisce un allineamento di base che, in molti casi, è sufficiente per ottenere un accoppiamento consistente tra bbox rilevati su RGB e profondità campionata.

4.10.3 Conversione della profondità

La depth viene letta come matrice D a 16 bit e convertita in millimetri tramite un fattore di scala s (mm/unit) fornito dai metadati:

$$D_{\text{mm}} = s \cdot D. \quad (4.21)$$

Nel caso del dispositivo e della configurazione adottata, lo scale factor è tipicamente pari a 1 mm/unità, ma il software non assume tale valore a priori: esso viene letto da file, rendendo il sistema robusto a modifiche di configurazione.

I pixel di profondità a valore nullo o non valido (ad esempio per assenza di misura, riflessioni o saturazione) vengono marcati come `NaN`, così da escluderli dai calcoli statistici successivi e prevenire stime errate dell'altezza.

4.10.4 Integrazione con la rete neurale

Una volta acquisito il frame RGB e stimati i parametri di calibrazione necessari, l'immagine viene processata dalla rete neurale.

È importante notare che la rete lavora esclusivamente sul frame RGB; la Depth viene coinvolta in modo deterministico *dopo* la detection, per calcolare grandezze metriche associate alla regione di interesse.

Per ogni bounding box, viene estratta una ROI dalla mappa D_{mm} allineata. La stima della profondità dell'oggetto deve essere robusta rispetto a:

- rumore locale e outlier;
- pixel invalidi (NaN);
- porzioni di bounding box che includono anche piano o sfondo.

Nel programma, la profondità dell'oggetto viene stimata tramite un percentile basso (ad esempio il 10-esimo percentile) dei valori validi nella ROI. Questa scelta è coerente con la geometria della scena: l'oggetto, essendo più vicino alla camera rispetto al piano, presenta profondità minori. Un percentile basso tende quindi a privilegiare la superficie più vicina, riducendo l'influenza di pixel appartenenti al piano retrostante.

Infine, l'altezza dell'oggetto rispetto al piano è calcolata come:

$$h = Z_{\text{piano}} - Z_{\text{oggetto}}, \quad (4.22)$$

dove Z_{piano} è la profondità del piano e Z_{oggetto} la profondità stimata dell'oggetto nella bbox. Il risultato viene poi riportato nel pannello operatore e usato per la logica di controllo qualità (range di accettazione).

4.10.5 Aspetti di robustezza e limiti

La calibrazione automatica presenta vantaggi operativi evidenti:

- riduce la dipendenza dall'operatore;
- migliora la ripetibilità delle misure;
- sfrutta le funzionalità di allineamento dell'SDK, evitando modelli geometrici manuali complessi.

Permangono tuttavia alcuni limiti tipici dei sistemi RGB-Depth:

- in presenza di superfici riflettenti o con geometrie complesse, la Depth può presentare buchi o outlier;
- l'allineamento SDK è ottimale in condizioni nominali, ma può degradare ai bordi del campo visivo o con forti discontinuità;
- una bbox troppo ampia può includere porzioni di piano e alterare la stima dell'altezza, rendendo necessarie statistiche robuste o maschere più selettive.

Per questi motivi, il progetto mantiene anche la modalità di calibrazione manuale, utilizzabile in casi particolari (set-up non nominali, necessità di correzioni locali o verifiche di laboratorio). La relazione tra calibrazione manuale e automatica è approfondita nel capitolo successivo, evidenziando quando ciascuna modalità risulta preferibile.

4.11 Confronto tra: manuale e automatica

La calibrazione automatica tende a garantire una maggiore ripetibilità perché basata su parametri di fabbrica e su una pipeline di allineamento coerente con la geometria del dispositivo. La calibrazione manuale, pur potendo raggiungere buoni risultati localmente, presenta una variabilità legata alla selezione dei punti e all'aderenza del modello affine al fenomeno reale.

In entrambe le modalità, la stima dell'altezza dipende dalla qualità della depth nella ROI. La scelta del percentile riduce la sensibilità a outlier, ma resta la necessità di:

- garantire un numero sufficiente di pixel validi,
- evitare ROI che includano troppo piano o bordi.

In modalità manuale, un errore di pochi pixel nella mappa RGB→Depth può influenzare notevolmente la ROI; in modalità automatica, tale rischio è ridotto perché l'allineamento produce direttamente una depth nello stesso sistema della RGB.

Capitolo 5

Interfaccia Grafica

Il pannello operatore costituisce il livello di interazione principale tra l'utente e il sistema di acquisizione, riconoscimento e misura sviluppato nel presente lavoro. La sua funzione non si limita alla semplice visualizzazione dei risultati, ma integra in un'unica interfaccia la gestione dell'acquisizione immagini, l'esecuzione del riconoscimento tramite rete neurale, l'estrazione delle misure geometriche dai dati RGB-Depth, il controllo di qualità rispetto alle tolleranze impostate e le operazioni di configurazione e manutenzione del sistema.

L'interfaccia è stata realizzata mediante *App Designer* di MATLAB, scelta che ha permesso di sviluppare un ambiente grafico integrato con il codice di elaborazione numerica, mantenendo continuità tra acquisizione, visualizzazione e logica di controllo. Dal punto di vista progettuale, il pannello è stato concepito come una *Human-Machine Interface* (HMI) orientata all'impiego pratico: l'operatore deve essere in grado di avviare la scansione, leggere i risultati, salvare i frame acquisiti e ricevere un riscontro immediato sullo stato di conformità dei pezzi analizzati. Parallelamente, l'amministratore deve poter accedere a funzionalità avanzate, quali la calibrazione manuale e la modifica di alcune impostazioni operative, senza compromettere la semplicità d'uso per l'utente standard.

L'interfaccia è strutturata in tre aree verticali principali: una colonna sinistra dedicata alla visualizzazione delle immagini, una colonna centrale dedicata ai dati di acquisizione e ai comandi principali, e una colonna destra che contiene la gestione dell'accesso, lo stato del sistema e il feedback finale sulle misure.

Tale organizzazione non risponde unicamente a criteri estetici, ma riflette la logica funzionale del processo: nella sezione laterale sinistra si colloca la fase di osservazione, nella parte centrale si svolgono le operazioni di elaborazione e interpretazione, mentre nella sezione laterale destra sono concentrate le attività di controllo e validazione del sistema.

5.1 Struttura generale dell'interfaccia

Come indicato, la disposizione dei componenti nel pannello segue una struttura a tre colonne con specializzazione funzionale. Tale organizzazione facilita la lettura visiva e separa in modo chiaro:

- la componente *visiva* (immagini),
- la componente *operativa e informativa* (comandi e misure),
- la componente di *controllo e supervisione* (accesso, stato, conformità).



Figura 5.1: Pannello operatore sviluppato in App Designer di MATLAB.

Nella Figura 5.1 è riportata la schermata del pannello nella sua configurazione completa. L'impianto visivo adotta una gerarchia semplice e leggibile: le aree funzionali sono delimitate da pannelli, i comandi principali sono centrali, mentre le

informazioni di stato sono collocate a destra, in posizione costante, così da rimanere sempre visibili durante l'uso. L'uso di titoli di sezione chiari (*Lato Immagini*, *Lato Dati*, *Info*) favorisce un'immediata comprensione della funzione di ciascuna zona. Inoltre, la distinzione cromatica tra sfondo, pannelli e controlli contribuisce a separare i livelli gerarchici dell'interfaccia, migliorandone la leggibilità.

5.1.1 Area di visualizzazione delle immagini

La colonna sinistra del pannello è dedicata alla visualizzazione dei frame associati al processo di misura. Nella parte superiore è presente l'area *Frame Scattato*, che mostra l'immagine acquisita dalla camera. In condizioni operative normali, si tratta del frame RGB utilizzato come input per la rete neurale. Questa vista rappresenta il riferimento "grezzo" della scena, ossia l'immagine così com'è acquisita prima delle elaborazioni di riconoscimento e sovrapposizione grafica.

Dal punto di vista funzionale, la presenza del frame scattato è importante per almeno tre motivi. In primo luogo, consente all'operatore di verificare immediatamente che la scena sia stata acquisita correttamente: illuminazione, messa a fuoco, centratura del pezzo e assenza di occlusioni, possono essere valutate visivamente. In secondo luogo, permette di confrontare l'immagine originale con quella successivamente elaborata dal detector, distinguendo chiaramente tra dato acquisito e dato interpretato. In terzo luogo, costituisce una forma di tracciabilità: il frame può essere salvato come evidenza documentale dell'acquisizione.

Sotto l'area del frame scattato è presente un pulsante di salvataggio dedicato. Tale pulsante consente di esportare il frame acquisito in un file immagine, permettendo la conservazione del dato per finalità di:

- archiviazione delle prove sperimentali;
- analisi offline;
- verifica a posteriori in caso di risultati anomali;
- documentazione dei test e redazione di report.

L'introduzione di un comando di salvataggio separato per il frame non elaborato risponde a una precisa scelta progettuale: il dato originale deve poter essere

conservato indipendentemente dagli overlay grafici o dai risultati della rete, così da mantenere una traccia del contenuto effettivamente acquisito dal sensore.

5.1.2 Area riconoscimento pezzi

Nella parte inferiore della colonna sinistra è presente l'area *Frame Riconoscimento Pezzi*, che visualizza l'immagine elaborata. In questa vista compaiono i risultati del detector neurale: tipicamente bounding box, etichette di classe, punti notevoli, testi di misura o altri overlay grafici prodotti dal post-processing, come in figura 5.2.

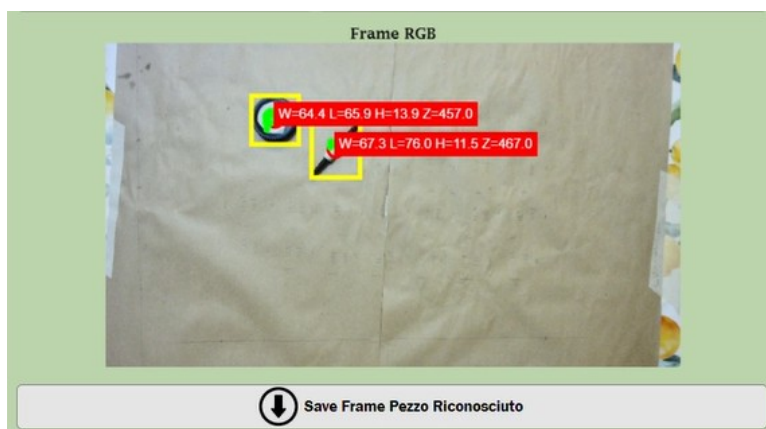


Figura 5.2: Esempio di Immagine elaborata a pannello

La separazione visiva tra frame originale e frame elaborato consente all'operatore di confrontare immediatamente il comportamento del sistema di riconoscimento con la scena reale. Questo aspetto è particolarmente utile in fase di taratura e debug, ma rimane vantaggioso anche in esercizio, perché permette di:

- verificare la correttezza del riconoscimento della classe;
- osservare il posizionamento dei bounding box;
- interpretare i valori numerici mostrati in sovrapposizione;
- individuare eventuali errori di segmentazione o falsi positivi.

Anche il frame elaborato dispone di un pulsante di salvataggio dedicato. Questa funzione è particolarmente rilevante in un sistema di misura, poiché consente

di conservare l'immagine con tutte le annotazioni utili all'interpretazione del risultato: classe riconosciuta, posizione del pezzo, dimensioni, altezze, eventuali etichette di qualità.

La disponibilità di un'immagine annotata semplifica la ricostruzione del processo di misura e favorisce la comunicazione dei risultati sia in ambito sperimentale sia in contesto produttivo. In pratica, il salvataggio del frame elaborato costituisce un *report visivo* sintetico, immediatamente interpretabile anche da un utente che non abbia accesso ai dettagli del codice.

5.2 Sezione “Dati Immagine”

La parte superiore della colonna centrale è dedicata ai *Dati Immagine*. Questa area raccoglie una serie di informazioni sintetiche associate all'acquisizione corrente, tra cui:

- dimensione dell'immagine colore in pixel;
- dimensione dell'immagine colore in millimetri;
- quota del piano di riferimento;
- tempo complessivo di acquisizione ed elaborazione.

Questi dati hanno una doppia funzione. Da un lato, forniscono all'operatore una conferma numerica del fatto che il sistema abbia eseguito correttamente la conversione tra scala immagine e scala fisica. Dall'altro, costituiscono un insieme di metadati tecnici utili per valutare la consistenza del processo. Ad esempio, un tempo di elaborazione anomalo può indicare un problema di caricamento del detector, di accesso ai file o di disponibilità della GPU; una quota del piano incoerente può suggerire la necessità di ricalibrare il piano o verificare la scena.

Al centro della sezione è collocato il pulsante principale del sistema, denominato *Scatta + Elabora*. Questo comando rappresenta il punto di ingresso del flusso operativo standard. Alla pressione del pulsante, il pannello avvia una sequenza articolata che include:

- lancio dell'eseguibile di acquisizione;
- attesa della generazione dei file immagine e metadati;
- lettura del frame RGB e della Depth;
- applicazione della calibrazione (automatica o manuale, se richiesta);
- esecuzione del detector neurale;
- estrazione delle misure e aggiornamento dell'interfaccia.

Dal punto di vista dell'utente, l'intera sequenza è incapsulata in un'unica azione. Questa scelta semplifica l'uso e riduce la possibilità di errori procedurali. Dal punto di vista del software, invece, il pulsante attiva una funzione ad alta complessità, responsabile di coordinare componenti esterni, controlli di stato, gestione degli errori, aggiornamento delle immagini e popolamento dei campi di output.



Figura 5.3: Esempio di Immagine elaborata a pannello

Sotto la sezione dedicata ai dati dell'immagine si trova l'area *Dati Pezzi*, destinata alla visualizzazione delle misure dei componenti riconosciuti. Questa sezione comprende:

- un menu a tendina per selezionare il pezzo di interesse;
- una text area contenente le informazioni dettagliate;

- un pulsante di salvataggio in CSV.

Il menu a tendina consente di filtrare la visualizzazione quando sono presenti più oggetti, selezionando un elemento specifico oppure scegliendo la visualizzazione complessiva. La text area è il principale strumento di comunicazione numerica del pannello: in essa vengono riportati i parametri stimati dal sistema, quali dimensioni, posizione, quota, altezza sopra piano e altre eventuali grandezze diagnostiche.

La scelta della text area come contenitore dei risultati risponde all'esigenza di disporre di una visualizzazione flessibile, capace di adattarsi a set di dati non sempre identici. A differenza di una tabella rigida, la *text area* consente di presentare in forma leggibile un insieme eterogeneo di informazioni, ordinato per oggetto e accompagnato da etichette descrittive.

Il pulsante *Save Info (CSV)* consente di esportare i risultati numerici in formato tabellare. Questa funzionalità è importante per collegare il pannello operatore a fasi successive di analisi, statistica o tracciabilità. Il formato CSV permette infatti:

- importazione semplice in MATLAB, Excel o strumenti di analisi dati;
- archiviazione storica delle misure;
- confronto tra lotti, configurazioni o condizioni di test;
- generazione di report e grafici.

Dal punto di vista metodologico, la presenza di una funzione di export separa chiaramente il livello di supervisione in tempo reale dal livello di analisi differita, rendendo il sistema adatto sia all'uso operativo sia alla documentazione sperimentale.

5.2.1 Selettore di visualizzazione RGB/DEPTH

Più in basso nella colonna centrale è presente il controllo *Imposta il frame*, che permette di selezionare se visualizzare la modalità RGB oppure la modalità Depth.

Questo comando non modifica il processo di acquisizione in sé, ma agisce sulla rappresentazione mostrata all'utente.

Tale funzione è particolarmente utile in due contesti. In fase di debug o messa a punto, la visualizzazione Depth consente di verificare la qualità del segnale di profondità, identificare buchi di misura, aree sature o artefatti di allineamento. In fase operativa, la visualizzazione RGB resta generalmente la più intuitiva per l'utente, ma la possibilità di passare a Depth aumenta la trasparenza del sistema e consente controlli puntuali in caso di risultati inattesi.

5.2.2 Selezione del modello di rete neurale

Il pulsante *Seleziona Modello per Training* permette di scegliere quale modello di rete utilizzare. Questa funzione risponde alla necessità di gestire configurazioni diverse del detector, ad esempio:

- modelli addestrati su set di dati differenti;
- versioni aggiornate del detector;
- modelli sperimentali da confrontare;
- configurazioni ottimizzate per precisione o velocità.

La possibilità di selezionare dinamicamente il modello rende il sistema flessibile e adatto alla sperimentazione. Dal punto di vista architetturale, la rete non viene necessariamente ricaricata a ogni elaborazione, ma può essere gestita con logica di *lazy loading*, così da ridurre i tempi di risposta e mantenere coerenza tra inferenze successive. Il pannello, mostrando il percorso del modello e lo stato del detector, rende visibile all'utente quale configurazione stia effettivamente guidando il riconoscimento.

5.2.3 Pulsante “Calibra RGB-DEPTH”

L'ultimo comando della colonna centrale è dedicato alla calibrazione manuale RGB-Depth. Questo pulsante avvia una procedura specifica, distinta dal flusso

di scansione ordinario, finalizzata alla costruzione o all'aggiornamento della mappa di corrispondenza tra coordinate RGB e coordinate Depth. Si tratta di una funzione avanzata, resa disponibile solo in presenza di privilegi adeguati

La sua collocazione in basso nella colonna centrale riflette il fatto che si tratta di una funzione importante ma non di uso continuo. Essa fa parte delle procedure di configurazione e manutenzione, non del ciclo elementare di misura. Tuttavia, la sua presenza nel pannello principale evita la necessità di strumenti esterni e rende il sistema auto-consistente anche sotto il profilo della calibrazione.

5.3 Sezione: accesso, stato e supervisione

La parte superiore della colonna destra contiene la sezione *Accesso*, che implementa il meccanismo di login. Essa include:

- visualizzazione di data e ora;
- indicazione della modalità corrente (operatore o amministratore);
- selettore utente;
- campo password;
- pulsante di login.

La distinzione tra modalità *operatore* e modalità *amministratore* costituisce un elemento importante della progettazione del pannello. In un sistema di misura, non tutte le funzioni devono essere accessibili a qualunque utente. L'operatore deve poter eseguire la misura e leggere il risultato; l'amministratore, invece, deve poter intervenire su funzioni sensibili che modificano la configurazione del sistema. Il login realizza quindi una separazione dei ruoli, introducendo un livello minimo ma efficace di controllo degli accessi. Anche in un prototipo sperimentale, tale scelta è significativa perché riduce il rischio di modifiche involontarie ai parametri di funzionamento e rende il comportamento del sistema più vicino a quello richiesto in ambiente industriale.

5.3.1 Sezione “Impostazioni”

Sotto la sezione di accesso si trova il pannello *Impostazioni*, che raccoglie opzioni di visualizzazione e configurazione. Tra queste compaiono:

- attivazione/disattivazione della visualizzazione dei bounding box;
- attivazione/disattivazione della visualizzazione del centro;
- attivazione/disattivazione della visualizzazione dei testi di misura;
- checkbox per la calibrazione manuale;
- controllo per il salvataggio automatico;
- pulsante di calibrazione del piano.

Questa sezione è normalmente bloccata in modalità operatore e viene abilitata in modalità amministratore. La scelta è coerente con la logica di separazione dei ruoli: le impostazioni modificano la rappresentazione del risultato o il modo in cui il sistema si calibra, e pertanto non devono essere alterabili da chi esegue soltanto le misure correnti.

La presenza della checkbox di calibrazione manuale è particolarmente importante perché seleziona una diversa logica di funzionamento: quando attiva, il sistema deve utilizzare la mappa di calibrazione manuale e richiedere un flusso coerente con tale scelta. Il pannello, quindi, non è un semplice contenitore di opzioni grafiche, ma uno strumento che modifica la modalità operativa interna del software.

5.3.2 Sezione “Stato”

Una delle aree più rilevanti dal punto di vista della trasparenza del sistema è l'area *Stato*. In esso vengono riportate informazioni sintetiche ma cruciali sul funzionamento corrente, come:

- stato della camera;
- esito dell'eseguibile di acquisizione;

- ultimo errore;
- stato del detector;
- ambiente di calcolo selezionato (CPU o GPU);
- percorso del modello di rete in uso.

Questa sezione svolge una funzione diagnostica fondamentale, come si evince dalla figura 5.4.

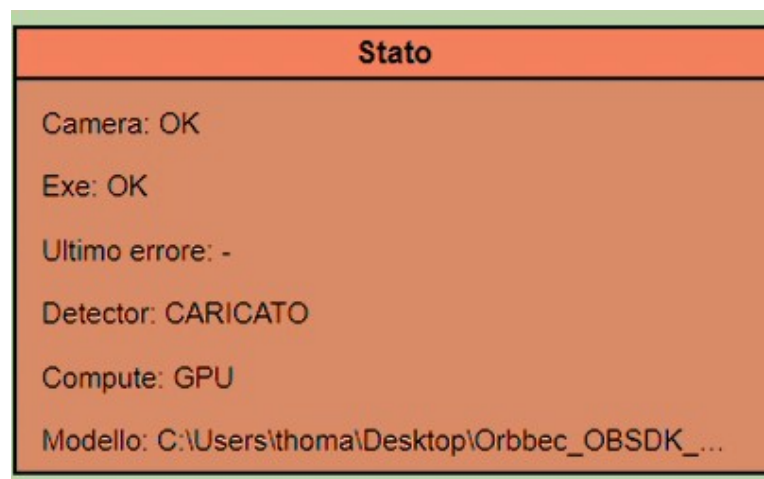


Figura 5.4: Informazioni riguardo i livelli software

In sistemi che integrano più livelli software e componenti esterni, la causa di un malfunzionamento non è sempre immediatamente evidente all'utente. Rendere visibile lo stato delle componenti consente di:

- identificare rapidamente se il problema è legato all'acquisizione o al detector;
- verificare se il modello sia stato correttamente caricato;
- controllare se il sistema stia operando in modalità GPU o CPU;
- ricondurre un errore a una fase specifica del processo.

Dal punto di vista della progettazione HMI, il pannello di stato riduce l'opacità del sistema e rappresenta un elemento essenziale per la manutenzione e il debug.

5.4 Sezione “Controllo altezze” e logica dei LED

Nella parte inferiore della colonna destra è presente il pannello *Controllo altezze*, che contiene due indicatori luminosi:

- un LED verde, associato alla conformità delle misure;
- un LED rosso, associato alla non conformità.

Questa sezione costituisce il livello di sintesi finale del sistema. Dopo che la rete ha riconosciuto gli oggetti e il software ha calcolato le rispettive altezze, il pannello confronta i valori ottenuti con le tolleranze specificate. In particolare, la logica implementata considera:

- *ruota* conforme se la sua altezza è pari a 12 mm con tolleranza ± 2 mm;
- *albero* conforme se la sua altezza è pari a 11 mm con tolleranza ± 2 mm.

Da un punto di vista formale, la condizione di conformità complessiva può essere espressa come:

$$|h_{\text{ruota}} - 12| \leq 2 \quad \text{e} \quad |h_{\text{albero}} - 11| \leq 2 \quad (5.1)$$

Se entrambe le condizioni risultano vere, si accende il LED verde; se almeno una delle due non è soddisfatta, si accende il LED rosso. Questa logica binaria consente di trasformare un insieme di misure numeriche in un esito immediatamente leggibile dall'operatore, riducendo il carico cognitivo e avvicinando il comportamento del sistema a quello di un dispositivo di controllo qualità industriale.

L'uso di indicatori luminosi è una scelta progettuale semplice ma efficace: l'utente non deve necessariamente interpretare tutte le misure numeriche per comprendere se il pezzo rientri o meno nelle tolleranze. Tuttavia, i valori dettagliati restano disponibili nella sezione *Dati Pezzi* garantendo sia sintesi immediata sia accesso al dettaglio.

5.5 Gestione dell'errore e robustezza d'uso

Il pannello non è solo uno strumento di comando, ma anche un sistema di segnalazione degli errori. La presenza della sezione *Ultimo errore* e la gestione di messaggi di stato riducono il rischio che il software fallisca in modo silenzioso. Dal punto di vista dell'ingegneria del software, questa è una scelta importante: un sistema di misura deve essere in grado non solo di produrre risultati, ma anche di dichiarare in modo leggibile quando tali risultati non possono essere garantiti.

La robustezza dell'interazione è rafforzata anche dalla presenza di blocchi e disabilitazioni contestuali: ad esempio, alcuni pulsanti sono disattivati quando non vi sono dati validi o quando non è stato effettuato il login appropriato. Ciò impedisce sequenze di utilizzo incoerenti e guida implicitamente l'utente verso il corretto ordine delle operazioni.

5.6 Dal riconoscimento alla misura

Il pannello non si limita a mostrare il risultato della rete, ma rende esplicito il passaggio dal riconoscimento alla misura. Questa caratteristica è centrale per il valore applicativo del progetto. Il sistema non si ferma alla classificazione dell'oggetto, ma utilizza il bounding box per interrogare la profondità, stimare la quota dell'oggetto e calcolarne l'altezza rispetto al piano. Tale passaggio è reso leggibile nel pannello attraverso:

- la visualizzazione delle bbox sul frame elaborato;
- la text area delle misure;
- l'indicazione della quota del piano;
- il controllo finale di conformità tramite LED.

L'operatore, quindi, non percepisce il sistema come un semplice classificatore, ma come uno strumento di controllo dimensionale assistito da visione artificiale.

5.7 Tolleranze e decisione binaria

La scelta di rappresentare l'esito finale con due LED è coerente con una logica di controllo qualità di tipo industriale. La decisione binaria (conforme / non conforme) viene costruita a partire da misure continue, ma viene sintetizzata in un'informazione immediatamente azionabile. Questo è un aspetto cruciale dal punto di vista ergonomico.

In contesti operativi, l'utente non ha sempre il tempo o la necessità di interpretare valori numerici dettagliati; spesso è sufficiente sapere se il pezzo rientra o meno nelle tolleranze. Al contempo, il pannello conserva l'accesso ai dettagli numerici, permettendo di risalire all'origine della non conformità.

5.8 Considerazioni finali

Il pannello operatore sviluppato in App Designer di MATLAB rappresenta un elemento chiave del sistema proposto, poiché concentra in un'unica interfaccia le funzioni di acquisizione, visualizzazione, controllo, configurazione e validazione metrologica.

Dal punto di vista ingegneristico, il pannello non è soltanto un contenitore grafico, ma un vero e proprio livello di orchestrazione tra telecamera Orbbec, codice MATLAB, eseguibili esterni e rete neurale. La presenza di funzioni di salvataggio, strumenti di calibrazione, controllo dei ruoli e indicatori di qualità rende l'interfaccia adatta non solo alla dimostrazione sperimentale, ma anche a un utilizzo strutturato, ripetibile e potenzialmente estendibile verso scenari applicativi più vicini all'ambiente industriale.

La progettazione del pannello ha quindi avuto un duplice valore. Da un lato, ha fornito all'utente finale uno strumento di utilizzo chiaro e immediato. Dall'altro, ha reso visibile e governabile un processo interno complesso, trasformando una pipeline articolata di visione artificiale e metrologia in un sistema interattivo, leggibile e controllabile.

Conclusioni

Il progetto sviluppato nel presente lavoro ha portato alla realizzazione di un sistema integrato per l’acquisizione, il riconoscimento e la misura metrologica di componenti meccanici mediante camera RGB–Depth (Orbbec Femto Bolt) e pipeline software basata su MATLAB. L’obiettivo principale non era soltanto dimostrare la capacità di riconoscere oggetti tramite una rete neurale, ma costruire una catena completa *vision-to-measure* in cui il risultato dell’inferenza (classificazione e localizzazione mediante bounding box) venisse trasformato in una misura fisica, interpretabile e verificabile, in particolare l’altezza dei pezzi rispetto a un piano di riferimento. In questo senso, il contributo del progetto si colloca nell’ambito della metrologia assistita da visione artificiale, dove la qualità del risultato dipende dalla coerenza tra acquisizione, calibrazione, riconoscimento e post-processing numerico.

Un aspetto centrale del lavoro è stato l’allineamento tra dominio RGB e dominio Depth, indispensabile per poter interrogare la profondità nella stessa regione spaziale in cui la rete rileva l’oggetto. La gestione di stream con risoluzioni differenti e con geometrie di proiezione non coincidenti ha imposto l’adozione di una strategia di calibrazione RGB–Depth articolata in due modalità operative. La prima è la modalità di calibrazione manuale, sviluppata per consentire all’amministratore un controllo diretto del mapping RGB→Depth mediante selezione di punti corrispondenti e stima di trasformazioni affini. L’adozione della variante *dual-affine* con blending morbido basato su *smoothstep* ha rappresentato una scelta pratica per ridurre errori spazialmente variabili sul campo visivo, mantenendo al contempo un modello computazionalmente semplice. Questa modalità si è dimostrata utile come strumento di manutenzione o di recovery, in particolare quan-

do si intende compensare disallineamenti residui o verificare sperimentalmente il comportamento del sistema in condizioni non nominali.

La seconda modalità, progettata come impostazione principale in esercizio, è la calibrazione automatica basata sull'SDK Orbbec e su un eseguibile C++ dedicato alla cattura e all'allineamento Depth→Color. Questa scelta ha permesso di sfruttare la calibrazione interna del dispositivo (intrinseci/estrinseci e correzioni implementate dal produttore) e di ottenere una depth coerente con la griglia del frame RGB, semplificando in modo sostanziale la catena di misura: i bounding box prodotti dal detector diventano direttamente regioni di interesse su cui campionare la profondità. L'integrazione dell'eseguibile nel flusso MATLAB ha garantito un compromesso efficace tra robustezza e semplicità d'uso: da un lato la pipeline di acquisizione e allineamento viene delegata a una componente esterna ottimizzata, dall'altro l'utente interagisce con un unico comando operativo di scansione, senza dover gestire direttamente la complessità della configurazione sensoriale.

Parallelamente alla calibrazione RGB-Depth, il progetto ha affrontato la necessità di definire un riferimento metrologico stabile tramite la calibrazione del piano di appoggio. Tale passaggio è stato determinante per convertire la profondità, interpretata come distanza camera-punto, in una misura di altezza sopra piano coerente e ripetibile. La stima del piano e l'impiego di misure robuste per stimare la quota dell'oggetto nella ROI (ad esempio tramite percentile basso dei valori depth validi) hanno contribuito a migliorare la stabilità della misura in presenza di rumore, outlier e contaminazione del piano all'interno dei bounding box. L'idea di fondo, implementata nel codice, consiste nel ricavare una quota oggetto che rappresenti la superficie più vicina alla camera e nel confrontarla con la quota del piano per ottenere l'altezza; la protezione contro valori negativi quando la ROI intercetta solo piano completa la robustezza del post-processing e riduce i falsi risultati in condizioni limite.

Un contributo pratico di particolare rilievo è rappresentato dal pannello operatore sviluppato in App Designer. Esso non ha avuto un ruolo meramente estetico, ma ha costituito un livello di orchestrazione tra componenti eterogenee: acquisizione tramite eseguibile esterno, lettura dei frame, gestione del detector

in modalità *lazy loading*, selezione dell'ambiente di calcolo CPU/GPU, gestione del login e dei privilegi, calibrazioni, visualizzazione dei risultati e salvataggio di immagini e dati. L'interfaccia a tre colonne ha consentito di mantenere separati ma coordinati i tre piani fondamentali del sistema: osservazione (immagini RGB e frame elaborato), interpretazione (misure numeriche e metadati dell'acquisizione) e supervisione (accesso, stato, diagnostica e controllo qualità). In particolare, l'introduzione dei LED di controllo altezze (verde/rosso), guidati da tolleranze ± 2 mm per i valori nominali di ruota e albero, ha trasformato un risultato numerico in un esito immediatamente azionabile dall'operatore, avvicinando il comportamento complessivo del prototipo a quello richiesto da un contesto industriale di controllo qualità.

Nonostante i risultati raggiunti, il sistema evidenzia anche criticità che rappresentano direzioni naturali di miglioramento. La prima riguarda la qualità intrinseca della profondità: superfici riflettenti, geometrie complesse o condizioni di illuminazione non ideali possono generare buchi e rumore nella mappa depth, riducendo la quantità di pixel validi nella ROI e degradando la stima dell'altezza. La seconda riguarda la dipendenza metrologica dalla qualità della detection: bounding box non aderenti o falsi positivi possono includere porzioni di piano o di sfondo e alterare la statistica con cui si stima la quota dell'oggetto. In tale contesto, l'uso di percentili migliora la robustezza ma non elimina completamente il problema quando la ROI è fortemente contaminata o quando l'oggetto è parzialmente occluso. Infine, la calibrazione manuale, pur essendo uno strumento efficace in casi specifici, rimane legata alla ripetibilità dell'operatore e potrebbe necessitare di procedure di validazione o ricalibrazione periodica per prevenire drift non rilevati.

Alla luce di tali considerazioni, l'architettura ottenuta dimostra la fattibilità di una pipeline end-to-end che collega in modo coerente acquisizione RGB-Depth, calibrazione, object detection e metrologia, offrendo un sistema utilizzabile e interpretabile, con adeguate possibilità di estensione verso scenari applicativi più strutturati.

Appendice A

Paradigma *anchor-free* nella object detection

A.1 Introduzione

Nel contesto della *object detection*, il termine *anchor-free* indica una famiglia di metodi in cui il rilevamento degli oggetti non si basa su un insieme predefinito di *anchor boxes* distribuite sulle feature map, ma avviene mediante una predizione più diretta della posizione e dell'estensione dell'oggetto. Questo paradigma si contrappone agli approcci *anchor-based*, nei quali la rete parte da box iniziali di dimensioni e rapporti d'aspetto fissati a priori e apprende a correggerli.

Nel presente lavoro, il riferimento a tale paradigma è motivato dal fatto che YOLOX adotta esplicitamente una formulazione *anchor-free*, introducendo un'evoluzione concettuale rispetto a molte versioni precedenti della famiglia YOLO. In termini pratici, questa scelta semplifica la pipeline di rilevamento e riduce la dipendenza da iperparametri legati alla definizione delle anchor.

A.2 Approccio *anchor-based*: richiamo concettuale

Nei detector *anchor-based*, su ogni posizione della feature map viene associato un insieme di box di riferimento:

$$\mathcal{A} = \{a_1, a_2, \dots, a_K\}, \quad (\text{A.1})$$

dove ogni anchor a_k è caratterizzata da una scala e da un aspect ratio predefiniti.

La rete, per ciascuna anchor, apprende due tipi di informazione:

$$\text{classe} \quad \text{e} \quad \Delta b, \quad (\text{A.2})$$

dove Δb rappresenta l'offset di regressione necessario a trasformare l'anchor iniziale nel bounding box finale.

Questo schema ha il vantaggio di essere storicamente consolidato, ma presenta alcune criticità:

- richiede la definizione manuale di scale, aspect ratio e numero di anchor;
- introduce una fase di *matching* tra ground truth e anchor, che complica il training;
- può aumentare il costo computazionale e la sensibilità a iperparametri non ottimali.

A.3 Definizione del paradigma *anchor-free*

Nei metodi *anchor-free*, il detector non parte da box predefiniti. La rete apprende invece a predire direttamente, per ogni posizione spaziale della feature map, le informazioni necessarie a ricostruire il bounding box dell'oggetto.

In una formulazione generale, per un punto candidato (x, y) si può esprimere la predizione come:

$$(c, l, t, r, b), \quad (\text{A.3})$$

dove:

- c è il punteggio di classificazione (o objectness / confidence);
- l, t, r, b sono le distanze del punto candidato dai lati sinistro, superiore, destro e inferiore del bounding box.

Il bounding box finale può quindi essere ricostruito come:

$$x_{\min} = x - l, \quad y_{\min} = y - t, \quad x_{\max} = x + r, \quad y_{\max} = y + b. \quad (\text{A.4})$$

Questa formulazione elimina la necessità di definire *anchor boxes* a priori e consente di trattare il rilevamento come una predizione diretta di posizione e classe.

A.4 Significato operativo del paradigma

Dal punto di vista implementativo, il paradigma *anchor-free* introduce una semplificazione significativa. In particolare:

- viene rimossa la dipendenza da un insieme discreto di box iniziali predefiniti;
- il training risulta meno legato alla scelta di iperparametri geometrici (numero, scala e forma delle anchor);
- la pipeline di inferenza tende a risultare più lineare e più semplice da interpretare.

Dal punto di vista concettuale, l'oggetto non viene più descritto come “correzione di una proposta iniziale”, ma come risultato di una predizione geometrica diretta effettuata a partire dalle feature locali.

A.5 Collegamento con YOLOX

YOLOX riprende e integra il paradigma *anchor-free* all'interno della famiglia YOLO. Il passaggio a una formulazione anchor-free è descritto dagli autori come uno degli elementi centrali dell'evoluzione architetturale del modello, insieme ad altre scelte progettuali quali la decoupled head e una strategia avanzata di label assignment.

Nel contesto del presente lavoro, tale aspetto è rilevante perché il detector impiegato eredita proprio questa filosofia: la predizione non dipende da una griglia di anchor prestabilite, ma da una stima diretta dei bounding box. Questo contribuisce a rendere il sistema più semplice da addestrare e più coerente con una pipeline moderna di *object detection end-to-end*.

A.6 Vantaggi e limiti

L'approccio *anchor-free* presenta diversi vantaggi:

- riduzione della complessità del modello rispetto a schemi fortemente dipendenti da anchor;
- minore necessità di tuning manuale degli iperparametri geometrici;
- maggiore chiarezza concettuale nella relazione tra punto di predizione e box finale.

Permangono tuttavia alcuni aspetti critici:

- la qualità della predizione dipende fortemente dalla strategia di assegnazione dei campioni positivi/negativi;
- in scenari con forte sovrapposizione di oggetti, la definizione dei punti responsabili del rilevamento può essere meno banale;
- la semplificazione dell'architettura non elimina la necessità di una progettazione accurata della loss e delle regole di training.

A.7 Conclusione

Il paradigma *anchor-free* rappresenta una transizione concettuale importante nell'evoluzione dei detector one-stage. Invece di partire da un insieme finito di proposte geometriche predefinite, la rete apprende a localizzare direttamente gli oggetti a partire dalle feature spaziali. Tale scelta, adottata in YOLOX, costituisce uno

degli elementi che giustificano la sua maggiore semplicità architetture rispetto a detector precedenti fortemente ancorati a schemi *anchor-based*. Nel quadro del presente progetto, questo aspetto contribuisce a rendere il riconoscimento più coerente con una pipeline rapida, integrata e adatta a un sistema di misura automatico.

Appendice B

Funzione di perdita composita in YOLOX

B.1 Introduzione

Nel paradigma della *object detection* one-stage, il processo di apprendimento non si limita a un solo obiettivo, ma richiede la simultanea ottimizzazione di più compiti distinti. In particolare, la rete deve imparare: la classificazione dell’oggetto, la localizzazione geometrica del bounding box e la stima della confidenza (o *objectness*), ossia la probabilità che una determinata predizione corrisponda effettivamente a un oggetto reale.

Per questo motivo, YOLOX adotta una *funzione di perdita composita*, cioè una funzione obiettivo formata dalla combinazione di più termini, ciascuno dedicato a un aspetto specifico del problema di rilevamento. Tale scelta è coerente con la natura multi-task del detector e consente di addestrare la rete in modo end-to-end, ottimizzando simultaneamente classificazione e localizzazione.

B.2 Struttura generale della loss

In forma generale, la funzione di perdita totale può essere descritta come:

$$\mathcal{L}_{\text{tot}} = \lambda_{\text{cls}} \mathcal{L}_{\text{cls}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}} + \lambda_{\text{obj}} \mathcal{L}_{\text{obj}}, \quad (\text{B.1})$$

dove:

- $\mathcal{L}_{\text{cls}}$ è il termine di perdita associato alla classificazione;
- $\mathcal{L}_{\text{reg}}$ è il termine di perdita associato alla regressione del bounding box;
- $\mathcal{L}_{\text{obj}}$ è il termine di perdita associato alla stima della confidenza (*objectness*);
- $\lambda_{\text{cls}}, \lambda_{\text{reg}}, \lambda_{\text{obj}}$ sono coefficienti di pesatura che regolano il contributo relativo dei diversi termini.

Questa formulazione esprime in modo compatto il principio fondamentale della loss multi-task: ogni componente della rete contribuisce all'aggiornamento dei parametri in funzione del proprio obiettivo, ma il training avviene in maniera congiunta, attraverso un'unica funzione scalare minimizzata durante l'ottimizzazione.

B.3 Termine di classificazione

Il termine di classificazione misura quanto la classe predetta dalla rete si discosti dalla classe reale dell'oggetto. In un detector multi-classe, la rete associa a ciascuna predizione una distribuzione di probabilità sulle classi disponibili e l'apprendimento mira a massimizzare la probabilità assegnata alla classe corretta.

In una formulazione generale, il termine di classificazione può essere scritto come:

$$\mathcal{L}_{\text{cls}} = \ell_{\text{cls}}(\hat{\mathbf{p}}, \mathbf{p}), \quad (\text{B.2})$$

dove $\hat{\mathbf{p}}$ rappresenta il vettore delle probabilità predette e $\mathbf{p}$ il vettore target (tipicamente codificato in forma binaria o one-hot).

Nel contesto dei detector one-stage moderni, questo termine è generalmente implementato tramite una loss di tipo binaria o multi-label, in modo da trattare la classificazione in maniera stabile anche in presenza di classi multiple e squilibrio tra esempi positivi e negativi.

B.4 Termine di regressione del bounding box

Il termine di regressione ha il compito di misurare la qualità geometrica della localizzazione. La rete non deve infatti soltanto riconoscere la presenza di un oggetto, ma anche predire un bounding box il più possibile aderente alla sua estensione reale.

In forma astratta, si può scrivere:

$$\mathcal{L}_{\text{reg}} = \ell_{\text{reg}}(\hat{b}, b), \quad (\text{B.3})$$

dove $\hat{b}$ è il bounding box predetto e b quello di riferimento (ground truth).

Nei detector moderni, compreso YOLOX, questo termine è spesso basato su misure di sovrapposizione geometrica tra box, piuttosto che su una semplice distanza coordinata-per-coordinata. L'idea è che due bounding box debbano essere considerati “vicini” se si sovrappongono bene, e non soltanto se i loro parametri numerici sono simili. Questo rende la loss di regressione più coerente con la metrica finale di valutazione del detector.

B.5 Termine di objectness (confidenza)

Il termine di *objectness* ha una funzione distinta dalla classificazione. Mentre la classificazione stabilisce *quale* oggetto sia presente, la confidenza misura *se* in quella posizione sia realmente presente un oggetto.

Si tratta quindi di un problema binario, che può essere formalizzato come:

$$\mathcal{L}_{\text{obj}} = \ell_{\text{obj}}(\hat{o}, o), \quad (\text{B.4})$$

dove:

- $\hat{o}$ è la confidenza predetta dalla rete;
- $o \in \{0, 1\}$ è il target binario che indica rispettivamente assenza o presenza di un oggetto.

Questo termine è particolarmente importante nei detector one-stage, poiché il numero di posizioni candidate è elevato e la maggior parte di esse non corrisponde a oggetti reali. La loss di objectness permette quindi di separare in modo efficace le predizioni rilevanti dallo sfondo.

B.6 Interpretazione multi-task della loss

Dal punto di vista teorico, la funzione di perdita composita di YOLOX può essere interpretata come un problema di ottimizzazione multi-task. La rete condivide una parte comune di estrazione delle feature, ma ottimizza contemporaneamente sottoproblemi differenti:

- un sottoproblema semantico, relativo alla classificazione;
- un sottoproblema geometrico, relativo alla localizzazione;
- un sottoproblema di affidabilità, relativo alla presenza dell'oggetto.

Questa struttura riflette il comportamento operativo del detector: una predizione viene considerata corretta soltanto se è simultaneamente ben classificata, ben localizzata e associata a una confidenza elevata.

B.7 Ruolo dei coefficienti di pesatura

I coefficienti λ_{cls} , λ_{reg} e λ_{obj} non hanno soltanto un significato numerico, ma controllano il bilanciamento tra compiti diversi. Se uno dei termini dominasse in modo eccessivo sugli altri, il processo di addestramento potrebbe convergere verso una soluzione sbilanciata, ad esempio:

- ottima classificazione ma localizzazione scadente;
- box accurati ma confidenza instabile;
- elevata objectness ma discriminazione insufficiente tra classi.

La loss composta consente quindi di regolare il contributo di ciascun compito, rendendo l'addestramento più stabile e coerente con l'obiettivo finale del detector.

B.8 Osservazione sulle varianti implementative

A livello implementativo, il dettaglio esatto delle singole funzioni ℓ_{cls} , ℓ_{reg} e ℓ_{obj} può variare tra versioni, framework o configurazioni di training. In YOLOX, la

formulazione concreta dipende dalla specifica implementazione della testa di detection e dalla strategia di assegnazione dei campioni positivi e negativi. Tuttavia, il principio concettuale rimane invariato: la loss totale nasce dalla combinazione di termini separati per classificazione, localizzazione e confidenza.

B.9 Conclusione

La funzione di perdita composita è un elemento strutturale essenziale di YOLOX. Essa traduce il problema del rilevamento oggetti in una forma di apprendimento congiunto, in cui la rete viene ottimizzata simultaneamente per riconoscere gli oggetti, localizzarli correttamente e stimare l'affidabilità della predizione. Questo approccio consente di mantenere una pipeline end-to-end e costituisce uno dei motivi per cui YOLOX si inserisce in modo naturale tra i detector moderni ad alte prestazioni.

Appendice C

Trasformazione di *shear* nel modello affine

C.1 Definizione e significato geometrico

Nel contesto delle trasformazioni affini bidimensionali, lo *shear* (in italiano spesso indicato come *taglio* o *scorrimento*) è una trasformazione che modifica la geometria di una figura “inclinandola” lungo una direzione in proporzione alla coordinata nell’altra direzione. A differenza di una rotazione pura, lo *shear* non preserva gli angoli (ad esempio un rettangolo può diventare un parallelogramma), pur preservando la linearità delle rette e il parallelismo.

C.2 Shear orizzontale e verticale in forma esplicita

Uno *shear orizzontale* (x-shear) è descritto da:

$$x' = x + k y, \quad y' = y, \quad (\text{C.1})$$

dove k è il parametro di shear. All’aumentare di y , le coordinate x vengono incrementate proporzionalmente.

Uno *shear verticale* (y-shear) è descritto da:

$$x' = x, \quad y' = y + k x. \quad (\text{C.2})$$

C.3 Forma matriciale (coordinate omogenee)

In coordinate omogenee (utili per includere anche la traslazione), uno shear orizzontale è:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & k & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad (\text{C.3})$$

mentre uno shear verticale è:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ k & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \quad (\text{C.4})$$

C.4 Shear all'interno di una trasformazione affine generale

Una trasformazione affine 2D generale può essere espressa come:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & b \\ c & d \end{bmatrix}}_{\text{parte lineare}} \begin{bmatrix} x \\ y \end{bmatrix} + \underbrace{\begin{bmatrix} t_x \\ t_y \end{bmatrix}}_{\text{traslazione}}. \quad (\text{C.5})$$

Nella matrice lineare, i termini “incrociati” b e c (che mescolano x con y) sono responsabili del comportamento di *shear* (oltre a poter concorrere alla rotazione, a seconda della decomposizione scelta). In particolare:

- un termine $b \neq 0$ indica che x' dipende anche da y (componente di shear in x);
- un termine $c \neq 0$ indica che y' dipende anche da x (componente di shear in y).

C.5 Interpretazione nel mapping RGB→Depth

Nel caso del mapping tra coordinate pixel RGB e coordinate pixel Depth, il modello affine adottato può essere scritto come:

$$u_D = a_1 u_R + a_2 v_R + a_3, \quad v_D = b_1 u_R + b_2 v_R + b_3. \quad (\text{C.6})$$

I coefficienti a_2 e b_1 rappresentano i contributi “incrociati” (rispettivamente v_R in u_D e u_R in v_D), e sono quindi associati al comportamento di *shear*. In un sistema reale RGB–Depth, tali termini possono modellare disallineamenti residuali tra i due sistemi di proiezione, come piccole non-ortogonalità apparenti, deformazioni prospettiche residue o skew non catturabili da un semplice modello di sola traslazione/scala.

Appendice D

Funzione *smoothstep* per il blending nel modello dual-affine

D.1 Contesto: blending morbido tra due trasformazioni affini

Nel modello *dual-affine* adottato nel programma, la corrispondenza tra coordinate RGB e coordinate Depth viene modellata mediante due trasformazioni affini distinte:

$$\begin{bmatrix} u_{D,L} \\ v_{D,L} \\ 1 \end{bmatrix} = \mathbf{A}_{\text{left}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}, \quad \begin{bmatrix} u_{D,R} \\ v_{D,R} \\ 1 \end{bmatrix} = \mathbf{A}_{\text{right}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}. \quad (\text{D.1})$$

Poiché l'errore di mapping RGB→Depth tende a variare sul campo visivo, l'utilizzo di un unico modello affine può essere penalizzante. Si impiega quindi un blending continuo tra i due modelli in una regione di transizione centrata attorno a una coordinata di split x_{split} (in pixel RGB), con ampiezza w_b (in pixel RGB).

Per evitare discontinuità (salti improvvisi) nella mappa risultante, è necessario che il peso di blending sia una funzione continua e regolare: nel programma viene adottata la funzione *smoothstep*.

D.2 Normalizzazione della coordinata e funzione di clipping

Sia u_R la coordinata orizzontale (asse x) di un punto in immagine RGB. Si definisce una variabile normalizzata $t \in [0, 1]$ che rappresenta la posizione del punto all'interno della fascia di blending.

La banda di blending è definita dall'intervallo:

$$u_R \in \left[x_{\text{split}} - \frac{w_b}{2}, x_{\text{split}} + \frac{w_b}{2} \right]. \quad (\text{D.2})$$

La normalizzazione, coerente con l'implementazione del programma, è:

$$t_0(u_R) = \frac{u_R - \left(x_{\text{split}} - \frac{w_b}{2} \right)}{w_b}. \quad (\text{D.3})$$

Per garantire che t resti nel dominio $[0, 1]$ anche fuori dalla banda di blending, si applica una funzione di saturazione (clipping):

$$t(u_R) = \text{clip}(t_0(u_R), 0, 1) = \max(0, \min(1, t_0(u_R))). \quad (\text{D.4})$$

In tal modo:

- se $u_R \leq x_{\text{split}} - \frac{w_b}{2}$, allora $t = 0$ (solo modello sinistro);
- se $u_R \geq x_{\text{split}} + \frac{w_b}{2}$, allora $t = 1$ (solo modello destro);
- nella regione intermedia t cresce in modo monotono da 0 a 1.

D.3 Definizione della funzione *smoothstep* usata nel programma

La funzione *smoothstep* è una funzione polinomiale di terzo grado frequentemente usata in grafica e interpolazione numerica perché garantisce:

- continuità della funzione;
- continuità della derivata prima agli estremi ($w'(0) = w'(1) = 0$);
- transizione morbida, evitando spigoli e discontinuità.

Nel programma la funzione è:

$$w(t) = t^2(3 - 2t), \quad t \in [0, 1]. \quad (\text{D.5})$$

Verifica delle proprietà ai bordi:

$$w(0) = 0, \quad w(1) = 1, \quad (\text{D.6})$$

$$w'(t) = 6t - 6t^2 = 6t(1 - t) \Rightarrow w'(0) = 0, \quad w'(1) = 0. \quad (\text{D.7})$$

La derivata nulla agli estremi implica che il peso passa da 0 a 1 senza “scalini” e senza variazioni brusche: ciò è particolarmente importante quando la mappa risultante è usata per campionare ROI di profondità (evitando salti improvvisi di coordinate).

D.4 Blending finale tra i due modelli affini

Una volta calcolato $w = w(t(u_R))$, le coordinate Depth finali sono ottenute per interpolazione pesata:

$$u_D = (1 - w) u_{D,L} + w u_{D,R}, \quad v_D = (1 - w) v_{D,L} + w v_{D,R}. \quad (\text{D.8})$$

Questa forma garantisce:

- continuità spaziale della mappa RGB→Depth;
- comportamento deterministico: a sinistra domina $\mathbf{A}_{\text{left}}$, a destra domina $\mathbf{A}_{\text{right}}$;
- transizione regolare nel corridoio di blending definito da w_b .

D.5 Parametri del programma: x_{split} e w_b

Nel programma, x_{split} viene determinato come media dei baricentri orizzontali delle selezioni di punti effettuate nelle due fasi di calibrazione (sinistra e destra):

$$x_{\text{split}} = \frac{1}{2} (\bar{u}_{\text{left}} + \bar{u}_{\text{right}}). \quad (\text{D.9})$$

La larghezza di blending viene scelta come frazione della larghezza dell'immagine RGB, ad esempio:

$$w_b = \alpha W_{\text{RGB}}, \quad (\text{D.10})$$

dove W_{RGB} è la larghezza in pixel dell'immagine RGB e α è un coefficiente (nel programma tipicamente $\alpha = 0.20$, quindi un blending esteso al 20% della larghezza).

D.6 Osservazioni pratiche e possibili limiti

La scelta della funzione *smoothstep* consente una fusione regolare tra due trasformazioni, ma non elimina completamente errori residui dovuti a:

- non-linearità ottiche o distorsioni non modellate dall'affinità;
- dipendenza dalla profondità (cambiamento di mapping con la distanza);
- incertezza nella selezione manuale dei punti.

Nonostante ciò, la tecnica dual-affine con blending morbido rappresenta un compromesso efficace tra semplicità computazionale e riduzione dell'errore spazialmente variabile, soprattutto in scenari dove un singolo affine globale non è sufficientemente accurato sull'intero campo visivo.

Appendice E

Funzione di *clipping* e variabile normalizzata t nel blending dual-affine

E.1 Contesto: perché servono t e clip nel blending

Nel modello *dual-affine* adottato nel programma, la mappatura RGB→Depth viene descritta da due trasformazioni affini distinte: una valida prevalentemente nella parte sinistra del campo visivo e una nella parte destra. Per evitare discontinuità (salti improvvisi) nella mappa risultante, le due trasformazioni non vengono commutate bruscamente, ma fuse tramite un peso di blending continuo.

L'idea è assegnare a ciascun pixel RGB una variabile scalare che rappresenti la sua posizione relativa rispetto a una *fascia di transizione*. Tale variabile, indicata con t , deve assumere valori compresi tra 0 e 1:

- $t = 0$ indica che il pixel deve essere trattato interamente con il modello sinistro;
- $t = 1$ indica che il pixel deve essere trattato interamente con il modello destro;

- $t \in (0, 1)$ indica che il pixel cade nella zona di fusione e quindi la mappatura sarà una combinazione dei due modelli.

Affinché questo comportamento sia garantito in modo deterministico, è necessario introdurre un'operazione di saturazione dei valori di t nell'intervallo $[0, 1]$. Tale operazione è comunemente detta *clipping* (o *clamp*) ed è implementata tramite la funzione $\text{clip}(\cdot)$.

E.2 Definizione della fascia di blending

Sia u la coordinata orizzontale (asse x) di un punto in immagine RGB. Si definiscono due parametri:

- x_{split} : coordinata (in pixel RGB) che rappresenta il centro ideale della transizione tra modello sinistro e destro;
- b_w : larghezza (in pixel RGB) della fascia di blending.

La fascia di blending è l'intervallo:

$$u \in \left[x_{\text{split}} - \frac{b_w}{2}, x_{\text{split}} + \frac{b_w}{2} \right]. \quad (\text{E.1})$$

All'esterno di tale fascia, l'algoritmo deve selezionare in modo netto uno dei due modelli (sinistro o destro); all'interno, deve effettuare una fusione progressiva.

E.3 Costruzione della variabile normalizzata t

Per trasformare la coordinata u (espressa in pixel) in una variabile adimensionale compresa idealmente tra 0 e 1, si introduce una normalizzazione lineare:

$$t_0(u) = \frac{u - \left(x_{\text{split}} - \frac{b_w}{2}\right)}{b_w}. \quad (\text{E.2})$$

Questa definizione ha una proprietà fondamentale: agli estremi della fascia di blending si ottengono direttamente i valori limite desiderati:

$$t_0\left(x_{\text{split}} - \frac{b_w}{2}\right) = 0, \quad t_0\left(x_{\text{split}} + \frac{b_w}{2}\right) = 1. \quad (\text{E.3})$$

Tuttavia, per valori di u esterni alla fascia, $t_0(u)$ risulta minore di 0 (a sinistra) o maggiore di 1 (a destra). In tali condizioni è necessario imporre che il peso di blending non assuma valori non validi, altrimenti l'interpolazione tra modelli genererebbe risultati non controllati.

E.4 Funzione clip: definizione e interpretazione

La funzione di clipping (o saturazione) forza un valore reale a rimanere entro un intervallo prefissato. Nel caso in esame l'intervallo è $[0, 1]$:

$$\text{clip}(x, 0, 1) = \max(0, \min(1, x)). \quad (\text{E.4})$$

Applicando tale operazione a $t_0(u)$ si ottiene:

$$t(u) = \text{clip}(t_0(u), 0, 1). \quad (\text{E.5})$$

Ne derivano tre casi:

- se $u \leq x_{\text{split}} - \frac{b_w}{2}$, allora $t_0(u) \leq 0$ e quindi $t(u) = 0$;
- se $x_{\text{split}} - \frac{b_w}{2} < u < x_{\text{split}} + \frac{b_w}{2}$, allora $0 < t_0(u) < 1$ e quindi $t(u) = t_0(u)$;
- se $u \geq x_{\text{split}} + \frac{b_w}{2}$, allora $t_0(u) \geq 1$ e quindi $t(u) = 1$.

In termini operativi, **clip** garantisce che fuori dalla fascia di blending il sistema non esegua alcuna fusione: a sinistra il contributo del modello destro deve essere nullo, mentre a destra il contributo del modello sinistro deve essere nullo.

E.5 Dal parametro t al peso di blending w : collegamento con *smoothstep*

Nel programma, una volta calcolato $t(u) \in [0, 1]$, il peso di fusione non viene scelto come interpolazione lineare ($w = t$), ma mediante la funzione *smoothstep*:

$$w(t) = t^2(3 - 2t), \quad t \in [0, 1]. \quad (\text{E.6})$$

Questa scelta è motivata dal fatto che *smoothstep* produce una transizione più regolare rispetto a quella lineare. In particolare:

$$w(0) = 0, \quad w(1) = 1, \quad (\text{E.7})$$

e la derivata prima si annulla agli estremi:

$$w'(t) = 6t(1-t) \Rightarrow w'(0) = 0, \quad w'(1) = 0. \quad (\text{E.8})$$

L'annullamento della derivata agli estremi riduce l'effetto di “spigolo” nella transizione tra i due modelli, evitando variazioni improvvise del mapping RGB→Depth in prossimità dei bordi della fascia di blending.

E.6 Blending finale tra le due trasformazioni affini

Indicando con $(u_{D,L}, v_{D,L})$ le coordinate Depth prodotte dal modello sinistro e con $(u_{D,R}, v_{D,R})$ quelle prodotte dal modello destro, la mappa finale è ottenuta mediante interpolazione pesata:

$$u_D = (1-w)u_{D,L} + wu_{D,R}, \quad v_D = (1-w)v_{D,L} + wv_{D,R}. \quad (\text{E.9})$$

Grazie alla definizione di t e alla saturazione con `clip`, questa formulazione garantisce:

- a sinistra della fascia: $t = 0 \Rightarrow w = 0$ e quindi $(u_D, v_D) = (u_{D,L}, v_{D,L})$;
- a destra della fascia: $t = 1 \Rightarrow w = 1$ e quindi $(u_D, v_D) = (u_{D,R}, v_{D,R})$;
- dentro la fascia: $w \in (0, 1)$ e quindi fusione continua dei due modelli.

E.7 Nota conclusiva

La coppia {normalizzazione in t + saturazione con `clip`} è un elemento essenziale per rendere controllabile la fusione dual-affine: consente di definire in modo geometrico una regione di transizione e garantisce che l'algoritmo si comporti

correttamente anche per punti esterni alla fascia di blending. La successiva applicazione di *smoothstep* produce un peso di blending regolare e adatto a applicazioni metrologiche, dove discontinuità o brusche variazioni nel mapping possono degradare la stima delle ROI di profondità e, di conseguenza, delle altezze degli oggetti.

Bibliografia

- [1] D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *International Journal of Computer Vision*, vol. 47, no. 1–3, pp. 7–42. [Online]. Available: <https://vision.middlebury.edu/stereo/taxonomy-IJCV.pdf>
- [2] J. D. Giglierano, “Lidar basics for mapping applications,” U.S. Geological Survey, Open-File Report 2007–1285, 2007, pDF open access. [Online]. Available: <https://pubs.usgs.gov/of/2007/1285/pdf/Giglierano.pdf>
- [3] S. Foix, G. Alenyà, and C. Torras, “Lock-in time-of-flight (tof) cameras: A survey,” *IEEE Sensors Journal*, vol. 11, no. 9, pp. 1917–1926, 2011.
- [4] Y. Zhang, B. Zhang, C. Shen, H. Liu, J. Huang, K. Tian, and Z. Tang, “Review of the field environmental sensing methods based on multi-sensor information fusion technology,” *International Journal of Agricultural and Biological Engineering*, vol. 17, no. 2, pp. 1–13, Apr. 2024. [Online]. Available: https://www.researchgate.net/publication/380755289_Review_of_the_field_environmental_sensing_methods_based_on_multi-sensor_information_fusion_technology
- [5] M.-M. C. J. S. L. S. Tao Zhou, Deng-Ping Fan, “Rgb-d object recognition: A survey,” *Computational Visual Media*, 2020, available online: <https://link.springer.com/content/pdf/10.1007/s41095-020-0199-z.pdf>.
- [6] H. Zeng, B. Yang, X. Wang, J. Liu, and D. Fu, “Rgb-d object recognition using multi-modal deep neural network and ds evidence

- theory,” *Sensors*, vol. 19, no. 3, p. 529, 2019. [Online]. Available: <https://www.mdpi.com/1424-8220/19/3/529>
- [7] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, and X. Zabulis, “Bop: Benchmark for 6d object pose estimation,” in *European Conference on Computer Vision (ECCV)*, 2018, accessed: 2026-01-07. [Online]. Available: https://openaccess.thecvf.com/content_ECCV_2018/papers/Tomas.Hodan_PESTO_6D_Object_ECCV_2018_paper.pdf
- [8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, 2015, preprint available Open Access. [Online]. Available: <https://arxiv.org/abs/1506.08640>
- [9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: <https://www.deeplearningbook.org>
- [10] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in Neural Information Processing Systems*, 2012. [Online]. Available: https://papers.nips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf
- [11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. [Online]. Available: <https://arxiv.org/abs/1311.2524>
- [12] J. e. a. Redmon, “You only look once: Unified, real-time object detection,” *CVPR*, 2016. [Online]. Available: <https://arxiv.org/abs/1506.02640>
- [13] A. e. a. Bochkovskiy, “Yolov4: Optimal speed and accuracy of object detection,” *arXiv*, 2020. [Online]. Available: <https://arxiv.org/abs/2004.10934>
- [14] Z. e. a. Ge, “Yolox: Exceeding yolo series in 2021,” *arXiv*, 2021. [Online]. Available: <https://arxiv.org/abs/2107.08430>

- [15] Orbbec Technology Group, “Coordinate systems (itof camera),” [https://doc.orbbec.com/documentation/Orbbec%20Femto%20Bolt%20Documentation/Coordinate%20Systems%20\(iToF%20Camera\)](https://doc.orbbec.com/documentation/Orbbec%20Femto%20Bolt%20Documentation/Coordinate%20Systems%20(iToF%20Camera)), 2024, accessed: 2026-01-07.
- [16] OpenCV Development Team, “Camera calibration and 3d reconstruction,” <https://docs.opencv.org>, 2024, accessed: 2026-01-07.
- [17] P. Vandewalle, J. Kovacevic, and M. Vetterli, “Reproducible research in signal processing,” *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 37–47, 2009. [Online]. Available: <https://infoscience.epfl.ch/record/136640/files/VandewalleKV09.pdf>
- [18] M. Zollhöfer, “Commodity rgb-d sensors: Data acquisition,” <https://arxiv.org/pdf/1902.06835.pdf>, 2019, arXiv preprint arXiv:1902.06835, accessed 2026-01-08.
- [19] H. Sarbolandi, D. Lefloch, and A. Kolb, “On the accuracy of depth sensors for 3d vision,” *Sensors*, vol. 15, no. 8, pp. 18 419–18 445, 2015. [Online]. Available: <https://arxiv.org/pdf/1505.05459.pdf>
- [20] C. Zhang and Z. Zhang, “Calibration between depth and color sensors for commodity depth cameras,” https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/hot3d11_ChaZhang.pdf, 2011, microsoft Research, accessed 2026-01-08.
- [21] Istituto dell’Enciclopedia Italiana Treccani. (n.d.) Reti neurali. Immagine tratta da: Dizionario di Medicina Treccani, consultato il 8 aprile 2026. [Online]. Available: [https://www.treccani.it/enciclopedia/reti-neurali_\(Dizionario-di-Medicina\)/](https://www.treccani.it/enciclopedia/reti-neurali_(Dizionario-di-Medicina)/)
- [22] S. D. University of California, “Artificial neural networks – a tutorial,” <https://cenl.ucsd.edu/Jclub/Artificial%20Neural%20Networks%20-%20A%20Tutorial.pdf>, 2013, accesso PDF con panoramica storica e motivazioni.

- [23] Oel de la Rosa Colon. Principales eventos y personajes en la evolución como disciplina de la ia. [Online]. Available: <https://www.timetoast.com/timelines/principales-eventos-y-personajes-en-la-evolucion-como-disciplina-de-la-ia>
- [24] L. Bertene. (2024, Sept.) Frank rosenblatt e o perceptron: A invenção que revolucionou o aprendizado de máquina. [Online]. Available: <https://compraco.com.br/en/blogs/atualidades-e-noticias/frank-rosenblatt-e-o-perceptron-a-invencao-que-revolucionou-o-aprendizado-de-maquina>
- [25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, pp. 533–536, 1986, introduzione dell’algoritmo di retropropagazione. [Online]. Available: <https://www.cs.toronto.edu/~hinton/absps/naturebp.pdf>
- [26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, capitoli introduttivi con rassegna storica delle reti neurali. [Online]. Available: <https://www.deeplearningbook.org/contents/intro.html>
- [27] A. Minini. Le reti neurali informatiche. [Online]. Available: <https://www.andreaminini.com/ai/le-reti-neurali-informatiche>
- [28] MathWorks, “Define persistent variable,” <https://www.mathworks.com/help/matlab/ref/persistent.html>, documentazione ufficiale MATLAB, consultato il 25 febbraio 2026.
- [29] —, “canusegpu - verify supported gpu is available for computation,” <https://it.mathworks.com/help/matlab/ref/canusegpu.html>, documentazione ufficiale MATLAB, consultato il 25 febbraio 2026.

Ringraziamenti

Desidero ringraziare innanzitutto il mio relatore Minto Riccardo, per la guida, la competenza e la disponibilità, offerte durante la stesura di questa tesi. Il suo supporto e i suoi suggerimenti sono stati fondamentali per portare a termine questo lavoro.

Ringrazio i miei genitori, per il loro sostegno costante, i sacrifici e la fiducia che non mi hanno mai fatto mancare. Un grazie speciale va anche a mia sorella, per essermi stata accanto con affetto, pazienza e incoraggiamento in ogni momento. Alla mia morosa va il mio ringraziamento più sincero, per aver condiviso con me questo percorso con amore, comprensione e presenza. Grazie per aver saputo alleggerire i momenti più difficili e per aver creduto in me anche quando io facevo fatica a farlo. Ringrazio di cuore anche la sua famiglia, per l'affetto, la disponibilità e il sostegno dimostrati lungo questo cammino.

Un pensiero speciale va poi ai miei amici, che con la loro vicinanza, leggerezza e amicizia sincera hanno reso questi anni indimenticabili. Grazie per le risate, i consigli, i momenti condivisi e per avermi aiutato a non perdere mai la motivazione.

A tutti voi, grazie di cuore per aver contribuito, ognuno a modo proprio, al raggiungimento di questo traguardo.