UNIVERSITÀ DEGLI STUDI DI PARMA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Dottorato di Ricerca in Tecnologie dell'Informazione
XXII Ciclo

Luca Bombini

STUDIO DI TECNICHE DI VISIONE ARTIFICIALE IN TEMPO REALE E LORO IMPLEMENTAZIONE PER APPLICAZIONI INDUSTRIALI

DISSERTAZIONE PRESENTATA PER IL CONSEGUIMENTO
DEL TITOLO DI DOTTORE DI RICERCA

GENNAIO 2010

UNIVERSITÀ DEGLI STUDI DI PARMA

Dottorato di Ricerca in Tecnologie dell'Informazione

XXII Ciclo

STUDIO DI TECNICHE DI VISIONE ARTIFICIALE IN TEMPO REALE E LORO IMPLEMENTAZIONE PER APPLICAZIONI INDUSTRIALI

Coordinatore:

Chiar.mo Prof. Carlo Morandi

Tutor:

Chiar.mo Prof. Alberto Broggi

Dottorando: *Luca Bombini*

GENNAIO 2010

Ad Annalisa
e a chi è mio compagno
Proverbi 18:24, Bibbia

Sommario

Elenco delle figure

Elenco delle tabelle

Introduzione

La ricerca nel campo dei sistemi di visione artificiale, non semplici telecamere che trasmettono immagini ad un operatore, ma veri e propri sistemi di elaborazione dotati di visione e capaci di individuare un pezzo da scartare o di verificare la correttezza di un assetto, sono ormai luogo comune ed indispensabili per la competitività delle aziende. I maggiori utilizzatori di questi sistemi operano nel settore dei semiconduttori, in quello elettronico, automobilistico, aerospaziale, farmaceutico e medicale. "Oggi, grazie all'evoluzione delle tecnologie hardware e software, i sistemi di visione rivestono un ruolo di fondamentale importanza in molti stabilimenti in tutto il mondo, dove gestiscono la produzione di ogni tipo di prodotto realizzato in volumi, dalle patatine fritte ai circuiti integrati per computer", ricorda Robert Shilman, ancora oggi Presidente e CEO di Cognex azienda leader nel campo della visione artificiale. Dai primi sistemi, complessi e costosi, si è arrivati ai prodotti di oggi, dispositivi economici e facili da usare, grandi come un telefonino. Per questi ed altri motivi, oggi, la visione artificiale continua a ricevere sempre nuovi impulsi e la stragrande maggioranza delle aziende che produce sistemi industriali utilizza telecamere nei suoi impianti.

Come accennato pocanzi, le telecamere hanno raggiunto prezzi contenuti e prestazioni veramente elevate. L'eterogeneità e la molteplicità delle funzioni che una sola telecamera può svolgere ha spinto fortemente la ricerca in questo settore.

Favorita dalla continua evoluzione di questi sistemi, la mia attvità di Dottorato si è svolta prevalentemente nell'ambito della ricerca di applicazioni innovative basate su tecniche di elaborazione immagine per applicazioni industriali. Scopo del mio lavoro

è stato quello di applicare tecniche sempre più complesse a sistemi reali, studiandone le problematiche principali e curando gli aspetti di interfacciamento fra dati sensoriali elaboratati ed l'automazione della loro esecuzione.

Durante questi anni, il processo di accrescimento delle cognizioni tali da sviluppare alcuni sistemi industriali complessi è stato progressivo, prima, consolidando tecniche di elaborazione immagine e, successivamente, applicandole a scenari industriali. Lo scopo di questo percorso è stato quello di sviluppare una particolare sensibilità alla maggioranza di quelli che sono le principali problematiche nel campo industriale della elaborazione delle immagini ed acquisire competenze via via maggiori sulla implementazione reale di sistemi autonomi complessi.

L'attvità svolta durante il corso di Dottorato ha avuto anche l'obbiettivo di capire dove i sistemi di visione artificiale possono sostituire precedenti tecnologie migliorandone le prestazioni e come questi possano interfacciarsi in maniera efficiente con i dispositivi di automazione. In altre parole, non solo verranno proposti sistemi innovativi di elaborazione immagine che sostituiscono quelli attuali implementati con altre tecnologie ma verrà illustrato anche come il nuovo dispositivo si interfaccia con il sistema generale proponendo quindi tecniche di interfacciamento in tempo reale.

I capitoli che seguiranno illustrano il percorso menzionato pocanzi analizzando tre diversi progetti dove tecniche avanzate di visione assistono o sostituiscono preesistenti tecnologie comunemente in commercio aumentando le prestazioni.

In particolare nel Capitolo 1 viene mostrato un sistema di sola percezione installato su un'automobile sperimentale in grado di misurare le *performance* di guida dell'autista. Questa attività è stata condotta all'interno del progetto Europeo DRUID *(Driving under the Influence of Drugs, Alcohol and Medicines)* in collaborazione con la Società Italiana Psicologi Sicurezza Viaria *(SIP.SI.VI)* al fine di creare un modello comportamentale di guida in grado di riconoscere e classificare gli effetti sulla guida di droghe, alcol e farmaci. A tal fine, sono state sviluppate tecniche di visione artificiale in grado di misurare alcuni parametri caratteristici della guida: posizionamento nella corsia, riconoscimento di ostacoli improvvisi e linee di stop sulla carreggiata. Contemporaneamente, il sistema registra e sincronizza i dati odometrici del veicolo con quelli ottenuti in tempo reale dalla elaborazione delle immagini fornendo così

delle statistiche che delineano la performance di guida.

Nel Capitolo 2 vedremo come tecniche avanzate di visione artificiale possano sostituire le attuali tecnologie ad infrarosso inerenti alla sicurezza in apertura/chiusura di accessi carrai. In particolare l'attività è mirata alla sostituzione delle attuali fotocellule con un sistema capace di riconoscere oggetti di diverse dimensioni in aree pericolose. Per questo scopo è stata condotta una sperimentazione in tempo reale del sistema proposto includendo l'interfacciamento della centralina di automazione di un cancello automatico per i test.

Il Capitolo 3 mostra complesse tecniche di visione applicate alla navigazione autonoma di veicoli industriali adibiti al carico e scarico. Questo progetto è stato sviluppato con una azienda italiana che progetta, realizza e commercializza questi veicoli autonomi. In particolare, in questo capitolo verranno illustrati i risultati della prima fase dell'intero progetto: progettazione dell'architettura del sistema di visione artificiale, riconoscimento ostacoli ed interfacciamento con il sistema di navigazione del veicolo. La sperimentazione delle fasi successive è attualmente in corso.

Infine, nel Capitolo 4 vengono illustrate le personali conclusioni sul lavoro svolto e sulle possibilità rese concrete dall'applicazione delle innovative tecniche di visione nel campo delle applicazioni industriali.

Capitolo 1

Sistema di percezione

In questo capitolo viene mostrata la progettazione e lo sviluppo del sistema DPM *(Driver Performance Monitor)*: dispositivo capace di misurare le *performance* di guida di un autista. L'attività che ha portato alla creazione di questo dispositivo è stata condotta all'interno di un più ampio progetto europeo chiamato DRUID [2] *(Driving under the Influence of Drugs, Alcohol and Medicines)* in collaborazione con la Società Italiana Psicologi Sicurezza Viaria *(SIP.SI.VI)*. Scopo di questa attività è la creazione di un modello comportamentale di guida in grado di riconoscere e classificare gli effetti sulla guida di droghe, alcol e farmaci.

Per raggiungere questo obiettivo si è reso necessario ideare un sistema tale da misurare in maniera completamente autonoma alcuni parametri caratteristici della guida. Alcuni di questi sono: velocità media, posizionamento del veicolo nella corsia, tempi di reazione alla comparsa di ostacoli improvvisi e profilo della frenata in prossimità di uno stop. Allo stato attuale, l'Istituto Federale di Ricerca Autostradale tedesco *(BAST, Bundesanstalt für StraSSenwesen)* ha già lavorato in questo campo mettendo a disposizione un sistema installato su veicolo in grado di misurare solo alcuni dei parametri menzionati prima. Questo sistema utilizza sensori riflettometrici per calcolare la distanza del veicolo rispetto alla linea destra delimitante la corsia e dispositivi *wireless* installati su ostacoli improvvisi per ricavare i dati dei tempi di reazione alla frenata.

Scopo dell'attività che ho condotto è sostituire questa tecnologia utilizzata avente limiti sia nella misurazione che nella riproducibilità dei test nella guida reale in uno scenario quotidiano. La tecnologia proposta è quella della visione artificiale che permette di aggiungere nuovi parametri caratteristici ed aumentare contemporaneamente l'efficienza nella misurazione e la relativa precisione.

1.1 Analisi del problema

Scopo della sperimentazione è misurare le prestazioni di pazienti opportunamente classificati e monitorati. Onde non rappresentare pericolo per altre persone ed infrastrutture, il sito di test selezionato per la sperimentazione reale è quello del Centro ACI Guida Sicura di Vallelunga, nei pressi di Roma. Come è possibile vedere in Fig.1.1.b, questo centro di fama internazionale mette a disposizione un tracciato nel quale si è potuto testare il dispositivo prodotto in tutta sicurezza e professionalità. La Fig.1.1.a mostra il veicolo sperimentale utilizzato: una Mini Cooper della casa costruttrice BMW. La scelta di questi due primi elementi sono determinanti per la buona riuscita del progetto: il primo, deve consentire di creare un tracciato sperimentale tale da essere rappresentativo della guida reale (deve contenere rettilinei, curve di diverse ampiezze, incroci, etc.) ed il secondo, deve fornire il maggior numero di dati veicolari, rendendo necessaria la collaborazione con alcuni tecnici di BMW per la codifica di pacchetti del bus CAN contenenti i dati odometrici necessari.

Come abbiamo già detto, il sistema deve essere in grado di registrare e sincronizzare i dati odometrici del veicolo con quelli ottenuti in tempo reale dalla elaborazione delle immagini fornendo così delle statistiche che delineano la performance di guida.

In particolare, il sistema dovrà creare dei dati opportunamente strutturati ed organizzati tali da essere riconducibili a prove sperimentali differenti e da essere analizzati successivamente. A tal fine, sono state ideate quattro prove di guida differenti che il sistema dovrà gestire:

- **[1]** ***Road Tracking Test:*** misurazione della distanza del veicolo dalla linea destra di delimitazione della corsia disegnata a terra lungo un percorso definito.

(a) (b)

Figura 1.1: Veicolo sperimentale *a* ed una foto del circuito di test, autodromo di ACI Guida Sicura, Vallelunga *b*.

- **[2]** ***Speed Monitor Test:*** misurazione della velocità e di altri dati veicolari lungo un percorso predefinito e relativa sincronizzazione dei dati veicolari.

- **[3]** ***Gap Acceptance Test:*** misurazione della distanza da una linea di stop posizionata ortogonale alla direzione del moto tracciata sul mando stradale e relativa sincronizzazione dei dati veicolari.

- **[4]** ***Sudden Events Test:*** misurazione del tempo di reazione alla frenata a seguito della comparsa di un ostacolo improvviso nella corsia, misurazione della distanza dall'ostacolo e relativa sincronizzazione dei dati veicolari.

Per ottenere misurazioni conformi agli standard tecnici dettati dai diversi partner del progetto DRUID, la sperimentazione dovrà avvenire nelle ore diurne sia del mattino che del pomeriggio ed in condizioni sia di forte illuminazione come nelle ore estive, che di scarsa come nelle ore diurne dei mesi invernali. Questo fattore influenzerà notevolmente la scelta della telecamera e degli algoritmi sviluppati.

Un ulteriore fattore da considerare è che il dispositivo sarà utilizzato da personale non specializzato e tipicamente in condizioni non ottimali di guida. Queste caratteristiche implicano dei vincoli progettuali dell'architettura del sistema tali da garantirne una buona facilità d'uso ed una notevole robustezza. Questi problemi saranno analizzati nel paragrafo successivo.

1.2 Configurazione Hardware

In Fig.1.2 viene mostrato lo schema generale dell'architettura hardware del sistema DPM.

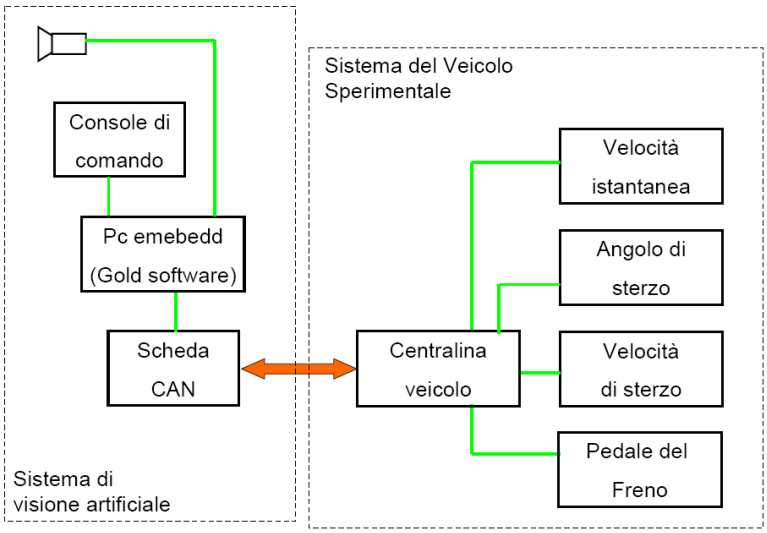


Figura 1.2: Schema generale.

Il sistema è diviso nelle seguenti parti fondamentali

- telecamera Guppy tecnologia CMOS a colori, (vedi Tab.1.1).
- PC compatto con disco tipo Compact Flash e sistema operativo Linux.
- monitor *touch screen* da 7 pollici.
- lettore dal bus CAN del veicolo.
- elettronica di alimentazione e protezione dispositivi.
- sistema esterno di sgancio ostacolo improvviso entro la carreggiata comandato mediante fotocellula di prossimità.

Queste parti sono ben visibili nella Fig.1.3. La collocazione della telecamera è stata studiata in modo tale da essere protetta dagli agenti atmosferici e da avere un campo visivo sufficiente per inquadrare sia le linee della corsia che un eventuale ostacolo in mezzo alla carreggiata. Il monitor *touch screen* è stato collocato frontalmente sul lato passeggero in modo tale da non recare disturbo a chi guida e da essere utilizzato come console di comando dall'operatore tecnico che seguirà il paziente durante la fase di test. L'elettronica di alimentazione e quella di protezione dei dispositivi, nonchè il lettore del bus CAN del veicolo sono stati posizionati nel baule come è possibile vedere in Fig.1.3.c . La dimensione di questi componenti non è banale ma in questo fase del progetto non si ci è occupati dell'ingegnerizzazione degli spazi dato che non ve ne era la necessità.

Infine, il sistema esterno di sgancio ostacolo improvviso entro la carreggiata comandato mediante fotocellula di prossimità è mostrato in Fig.1.3.d . Questo sistema rilascia l'ostacolo, vedi macchinina rossa, automaticamente dopo il passaggio del veicolo in prossimità della fotocellula. In questo modo è possibile regolamentare la comparsa di un ostacolo nella carreggiata.

Nella Tab. 1.1 vengono mostrate le principali caratteristiche della telecamera digitale utilizzata per la sperimentazione. Come si accennava nel paragrafo precedente, la scelta della telecamera è un punto critico per il buon svolgimento dell'intero progetto. In particolare questa dovrà funzionare sia in condizione di illuminazione diffusa intensa che in scarse condizioni di luminosità e questo indica che è necessario utilizzare una telecamera con una dinamica elevata. Inoltre, questo sensore dovrà essere dotato di sistemi di guadagno ed esposizione automatica onde sopperire a variazioni repentine di luminosità.

Un'altro fattore essenziale, è il numero massimo di *frame* a piena risoluzione che questa è in grado di acquisire. Secondo le specifiche di progetto, è necessario effettuare una misurazione ogni 10ms e pertanto la telecamera utilizzata è compatibile con queste specifiche.

Una volta selezionati i componenti ed individuate le connessioni fra il sistema CAN del veicolo ed quello di visione artificiale si è proseguito con l'installazione vera e propria di tutti i dispositivi onde ottenere un allestimento funzionale e robusto.

Specifiche tecniche telecamera Firefly MV			
Image Sensor Type	**Max Resolution**	**Pixel Size**	**Max Frame Rate**
1/3 progressive CMOS	752 x 480	6.0 m x 6.0 m	752x480 at 30 FPS

Tabella 1.1: Caratteristiche tecniche telecamera Guppy Allied.

(a) (b)

(c) (d)

Figura 1.3: Particolari dei sensori e dispositivi installati sul veicolo sperimentale.

La fase successiva è stata quella di acquisizione delle immagini per sviluppare e testare gli algoritmi di visione artificiale.

1.3 Algoritmo di percezione della guida

Le sequenze di immagini per la progettazione e lo sviluppo degli algoritmi sono state acquisite percorrendo sia strade comunali e provinciali di Parma che il percorso ufficiale adibito alla sperimentazione situato a Vallelunga, Roma. Avendo pertanto a disposizioni differenti *set* di immagini è stato possibile creare algoritmi robusti e ed efficienti nelle misurazioni.

In Fig.1.4 è possibile notare il diagramma di flusso degli algoritmi. Questo diagramma di flusso è valido sia per l'algoritmo di *Road Tracking Test*, sia per quello di *Gap Acceptance Test* che per quello di *Sudden Event Test*. Questo rende molto più flessibile la gestione di eventuali modifiche ed aggiunte di nuove specifiche di riconoscimento e misurazione.

L'algoritmo è composto dalle seguenti fasi principali:

- condizionamento delle immagini: questa fase racchiude tutte quelle operazioni radiometriche e morfologiche che permettono di estrarre l'oggetto da analizzare e classificare (linee bianche a terra o evento improvviso sul manto stradale).

- segmentazione: raggruppamento di pixel sulla base di caratteristiche puntuali e locali, come varianza, luminanza, tessitura; il risultato di questa elaborazione è un insieme di pixel contigui *(cluster)*, omogenei per qualche insieme di proprietà.

- etichettatura ed inseguimento: una volta etichettati i vari *cluster*, un algoritmo di inseguimento li valida in base a regole di persistenza temporale e coerenza del moto del veicolo.

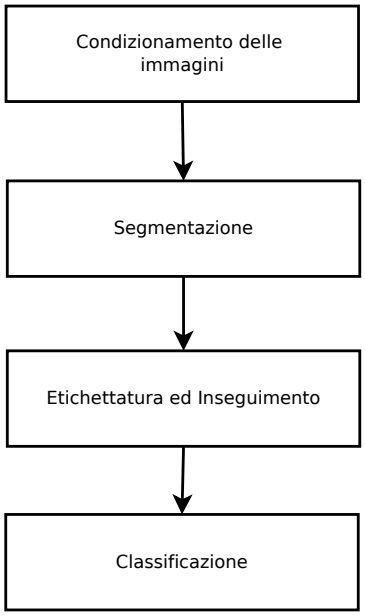
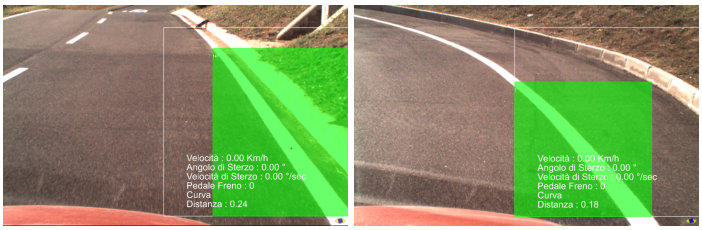


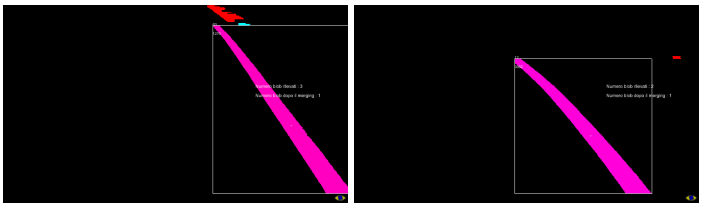
Figura 1.4: Schema generale algoritmi utilizzati nel sistema DPM.

- classificazione: questa fase si occupa di estrarre le caratteristiche morfologiche degli oggetti individuati calcolandone la distanza relativa al veicolo mediante algoritmi di *Inverse Prospective Mapping*.



(a) (b)

Figura 1.5: Road Tracking Test, immagini in ingresso e riconoscimento linea in tratto rettilineo *a* ed uno curvo *b*.

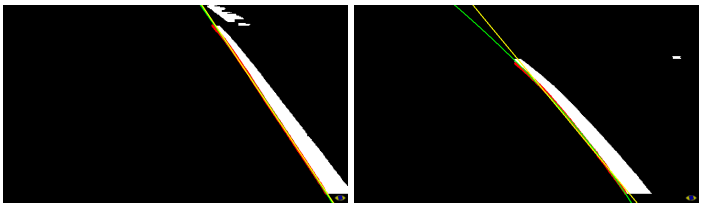


(a) (b)

Figura 1.6: Road Tracking Test, segmentazione ed etichettatura linea in tratto rettilineo *a* ed uno curvo *b*.

1.4 Logica di funzionamento del sistema

L'operatore potrà eseguire le varie prove della sperimentazione semplicemente interfacciandosi con il monitor *touch screen* montato frontalmente al sedile del passeggero. Premendo i bottoni di test sulla schermata potrà attivare le varie prove della sperimentazione. Il *download* dei dati ottenuti avverrà automaticamente mediante



(a) (b)

Figura 1.7: Road Tracking Test, inseguimento ed classificazione linea in tratto rettilineo *a* ed uno curvo *b*.

(a) (b) (c)

Figura 1.8: Sudden Event Test, immagini in ingresso inquadrante ostacolo improvviso*a*, segmentazione ed etichettatura ostacolo *b* ed inseguimento e classificazione *c*.

l'inserimento di una comune chiavetta USB nel PC posto nel baule del veicolo. Di seguito, in Fig.1.9 si riporta il diagramma dell'interfaccia.

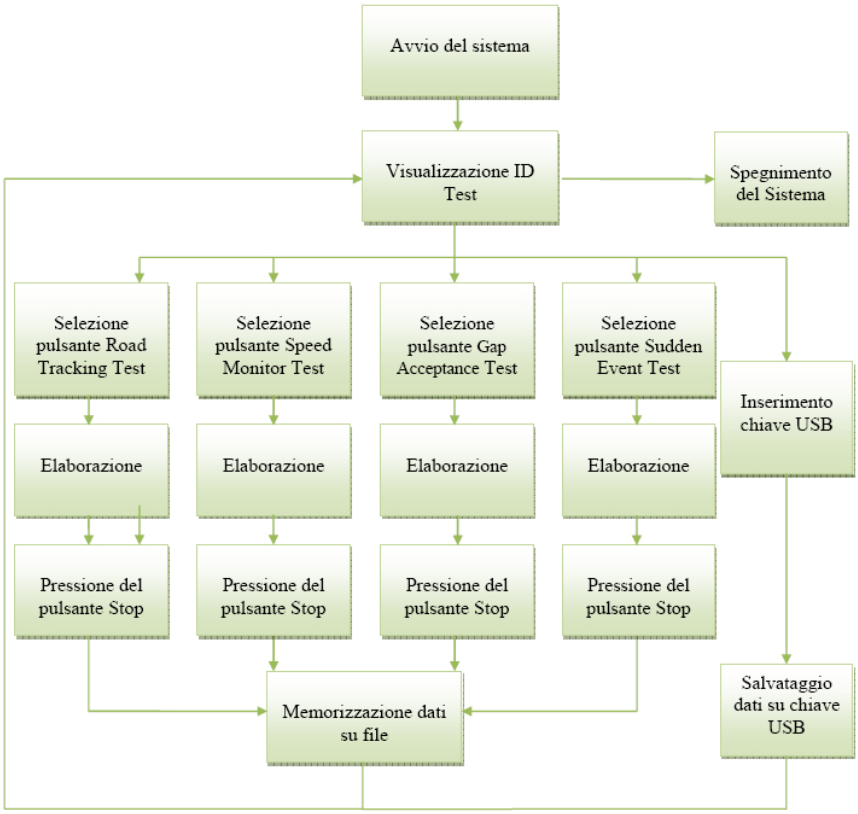


Figura 1.9: Schema interfaccia.

Una volta acceso il veicolo ed il sistema DPM, comparirà automaticamente a monitor il menu mostrato in Fig.1.10. Questo menu consente di selezionare una delle quattro prove menzionate nei paragrafi precedenti, visualizzare quello che la telecamera stà inquadrando ed usire dal menu spegnendo l'intero sistema.

Una volta premuto il tasto di un determinato test, comparirà l'interfaccia relativa a quel test con la possibilità di interromperlo in qualsiasi momento come mostrato in Fig.1.11 . In alto a destra della schermata compare un numero progressivo che

Figura 1.10: Menu principale del sistema DPM.

identifica univocamente il test effettuato. Per motivazioni di privacy del paziente e riservatezza dei dati, l'operatore associerà l'identificativo del test con quello del paziente. La prova termina quando viene premuto il tasto di STOP. In quell'istante, il sistema finisce la creazione di un file opportunamente indicizzato, contenente tutti i dati registrati in quel momento e le statistiche riepilogative della prova. Sarà possibile scaricare questo ed altri file di tutte le prove semplicemente inserendo una chiavetta USB nella relativa porta del PC posizionato nel baule. Una volta inserita, tutti i file vengono automaticamente scaricati: un messaggio a video mostra che questa operazione è conclusa ed è possibile estrarre la chiavetta.

Figura 1.11: Schermata del sistema DPM durante l'esecuzione di un test.

I file prodotti dal sistema sono raccolti in cartelle opportunamente indicizzate. Ogni cartella viene creata al momento dell'accensione del dispositivo ed ogni test di

quella sessione viene memorizzato in un file a se stante dentro la cartella. In questo modo, l'operazione di associazione paziente-test è molto rapida e semplice. Inoltre, in ogni file è contenuto la data e l'ora di registrazione dei dati, in modo tale da evitare qualsiasi tipo di errore fra paziente-test.

Di seguito si mostra un piccolo stralcio di un tipico file di test.

```
Sessione di Test Road Tracking ID= 2.
Sessione di Tue Oct 27 13:21:29 2009

Nota: Time [sec], Velocità [Km/h], Angolo di sterzo [grad],
Velocità di sterzo [grad/sec], Freno [On/Off],
Misura [metri], RECT/NO RECT.

0,00 0,00 1,33 0,00 1 NoDetect NO RECT
0,10 0,00 1,33 0,00 1 NoDetect NO RECT
0,20 0,00 1,33 0,00 1 NoDetect NO RECT
0,30 0,00 1,33 0,00 1 NoDetect NO RECT
0,40 0,00 1,33 0,00 1 NoDetect NO RECT
0,50 0,00 1,33 0,00 1 NoDetect NO RECT
0,60 0,00 1,33 0,00 0 0,19 NO RECT
0,70 0,00 1,33 0,00 0 0,19 NO RECT
0,80 0,00 1,33 0,00 0 0,19 NO RECT
0,90 0,00 1,33 0,00 0 0,19 NO RECT
1,00 0,00 1,33 0,00 0 0,19 NO RECT
...
...
...
299,10 34,08 -2,10 -29,83 0 0,15 NO RECT
299,20 34,06 -2,24 -24,30 0 0,09 NO RECT
299,30 34,00 -2,30 -12,13 0 0,03 NO RECT
299,40 33,95 -2,33 -6,31 0 -0,03 NO RECT
```

```
299,50 33,90 -2,37 -7,94 0 -0,05 NO RECT
299,60 33,85 -2,38 -0,55 0 -0,04 NO RECT
299,70 33,81 -2,38 0,00 0 -0,04 NO RECT
299,80 33,66 -2,38 0,00 0 -0,04 NO RECT
299,90 33,53 -2,38 0,00 0 -0,05 NO RECT
300,00 33,32 -2,38 0,00 0 -0,06 NO RECT


Sessione di Test Roat Tracking ID= 2.
Sessione di Tue Oct 27 13:21:29 2009

MEAN SM: 27,76    SD SM: 7,329

RECT_____________________________________________________
MEAN RT: 0,12    SDLP RT: 0,093
NO RECT__________________________________________________
MEAN RT: 0,22    SDLP RT: 0,173
```

Lo stralcio del file del test *Road Tracking Test* mostra tre parti fondamentali:

- *intestazione:* contiene l'identificativo del test, la data e l'ora di esecuzione. Inoltre, contiene un promemoria sul tipo dei dati che vengono riportati di seguito.
- *dati :* contiene i dati veri e propri. Sulla colonna di sinistra viene memorizzato il tempo espresso in secondi. Le due colonne più a destra rappresentano la misura del veicolo dalla linea di corsia a destra quando e la tipologia del percorso: rettilineo o non rettilineo.
- *statistiche finali:* contiene le statistiche del test, includendo velocità e distanza dalla linea media nonchè le relative deviazioni standard nel caso che il percorso sia rettilineo che non rettilineo.

Come è possibile notare, l'algoritmo non produce nessuna statistica inerente ai dati odometrici del veicolo come angolo di sterzo, velocità angolare e pressione del pedale del freno. Questi dati possono essere esaminati facilmente mediante un semplice foglio di calcolo elettronico in quanto il file creato è stato reso esportabile ai più comuni programmi di calcolo statistico come Excel ed OpenOffice.

1.5 Prestazioni qualitative del sistema

Il sistema DPM è in grado di registrare e sincronizzare i dati odometrici con quelli prodotti dagli algoritmi di visione artificiale sviluppati. Sono state condotte tre diverse campagne di test presso il circuito ACI Guida Sicura di Vallelunga nei mesi di Giugno, Ottobre, Dicembre del 2009. In queste occasioni, decine di pazienti, sottoposti ad alterazioni del proprio stato fisico si sono sottoposti ai test utilizzando il veicolo sperimentale. Il sistema si è dimostrato affidabile ed estremamente facile da utilizzare. Le situazioni di forte o di scarsa luminosità non hanno creato problemi significativi al sistema e neppure le temperature estive o quelle invernali. I dati prodotti sono al vaglio di un gruppo di psicologi che stanno cercando di costruire un modello comportamentale dei pazienti sottoposti a sperimentazione.

Alcuni problemi si sono riscontrati durante i periodi estivi con forte luminosità diffusa ed ombre che intersecavano le linee di demarcazione della carreggiata. Questo problema causa una temporanea mancata misurazione della distanza della linea. Tuttavia, elaborando dieci immagini al secondo, questa mancanza di rilevazione può essere considerata trascurabile.

Capitolo 2

Sistema di percezione con bassa automazione

La regolamentazione del traffico in accessi carrai, avviene perlopiù mediante dispositivi meccanici quali cancelli, barriere e pistoni pneumatici automatizzati. Questi dispositivi vengono pilotati mediante una centralina di comando che gestisce i segnali di apertura e chiusura e la sensoristica di sicurezza che ne garantisce la relativa certificazione.

Nonostante questi tipi di dispositivi siano in commercio da decine di anni, non sono stati compiuti significativi passi in avanti per quanto riguarda la loro sicurezza. Attualmente, coppie di fotocellule con tecnologia infrarossi fungono da rilevatori di presenza di eventuali pedoni/ostacoli in prossimità della zona di pericolo: nel caso di cancelli ad ante battenti, una prima coppia è tipicamente posizionata all'altezza del varco ed una seconda coppia poco dopo la linea di massima apertura del cancello. Queste fotocellule hanno lo scopo principale di interrompere il movimento di chiusura del cancello nel caso in cui sia presente un qualche oggetto sulla linea di rilevazione; di contro il movimento di apertura non viene, solitamente, influenzato da alcun evento.

Attualmente, questi sistemi di sicurezza presentano varie problematiche:

- **Dimensione degli oggetti:** le fotocellule hanno un'altezza fissa e rilevano un

ostacolo su una sola linea; di conseguenza oggetti troppo piccoli non vengono rilevati da questi sensori.

- **Staticitàă della rilevazione:** i sensori sono fissi, guardano su una sola linea a coordinate fisse, essi sono quindi adatti alla rilevazione di corpi in movimento di adeguate dimensioni; si può facilmente intuire come la presenza di oggetti o persone bloccati all'interno dell'area di lavoro non possa essere rilevata; di conseguenza essi si trovano in una situazione di pericolo in caso di apertura/chiusura del sistema.

- **Posizionamento dei sensori:** le fotocellule possono essere posizionate solamente al di fuori dell'area di azione del cancello, lavorano quindi esternamente ad essa e non direttamente sulla zona di pericolo: ancora una volta questo è un approccio coerente se si pensa a situazioni in cui si ha l'entrata o l'uscita di corpi nell'area di lavoro, ma non per oggetti che giàă si trovano al suo interno.

L'obiettivo di questo progetto è la sostituzione delle fotocellule con un sistema di visione artificiale che possa affrontare e risolvere i problemi appena analizzati e che sia in grado di far operare un cancello automatico in piena sicurezza.

2.1 Analisi del problema

Il primo passo affrontato nello sviluppo del progetto è stato la modellazione del problema attraverso la costruzione di un sistema di visione montato su un cancello automatico preesistente con lo scopo di acquisire un set di immagini da utilizzare per un primo test del progetto. Nella Fig.2.1 possiamo vedere una fotografia del cancello automatico selezionato ed un particolare del sistema di visione installato.

Sin dalle prime acquisizioni si sono resi subito evidenti i principali problemi che sono stati affrontati durante tutto lo sviluppo di questo progetto:

- **Forti variazioni luminosità:** essendo il sistema *outdoor* si è soggetti a forti variazioni di luminositàă della scena dovute al cambiamento, talvolta anche molto rapido, delle condizioni meteorologiche.

(a) (b)

Figura 2.1: Cancello automatico di test *a* ed un particolare dell'installazione delle telecamere *b*.

- **Stabilizzazione:** le telecamere montate sul cancello sono soggette a delle forti vibrazioni, con conseguente spostamento, talvolta rilevante, della scena, dovute principalmente all'arrivo a fine corsa delle ante del cancello e al successivo urto di queste contro i delimitatori presenti a terra; si rende quindi necessario un sistema di stabilizzazione dell'immagine che possa riportate la stessa nella posizione originale.

- **Cancello ad ante piene:** come è possibile notare dalla Fig. 2.1 il cancello utilizzato per l'acquisizione è dotato di ante cieche; è stato perciò necessario considerare la rilevante occlusione di parte dell'immagine delle due telecamere, ad opera delle ante stesse, in fase di apertura/chiusura del cancello.

- **Posizione delle camere:** le telecamere sono posizionate in alto e con un angolazione rispetto alla scena molto accentuata a causa delle domensioni dell'area da inquadrare com'è visibile in Fig. 2.1.

2.2 Setup Hardware

Per quanto riguarda l'hardware utilizzato per l'acquisizione delle sequenze video si è selezionata una telecamere a toni di grigio modello *Firefly MV* le cui specifiche tecniche sono riportate nella Tab. 2.1: Successivamente si sono selezionate ottiche *fish-eye*,

Specifiche tecniche telecamera Firefly MV			
Image Sensor Type	**Max Resolution**	**Pixel Size**	**Max Frame Rate**
1/3 progressive CMOS	752 x 480	6.0 m x 6.0 m	752x480 at 63 FPS

Tabella 2.1: Caratteristiche tecniche telecamera Firefly MV.

prodotte dalla *Tuss Vision, Inc.* i cui parametri di riferimento sono stati inseriti nella Tab. 2.2.

Specifiche tecniche ottiche fish-eye				
Modello	**Formato (inch)**	**Lunghezza Focale**	**Apertura Diaframma**	**Sistema**
BL2225M12	1/3	2.2mm	F2.5	Fixed

Tabella 2.2: Caratteristiche tecniche ottiche fish-eye.

Questo tipo di ottiche è dotato di una particolare struttura che permette di ottenere un angolo di campo anche superiore ai 180°; la scelta delle *fish-eye* è stata motivata dalla particolare posizione che le telecamere devono assumere per poter essere utili all'applicazione: in particolare esse devono inquadrare una zona piuttosto ampia da una posizione rialzata; la scelta delle ottiche fish-eye si quindi resa necessaria per poter inquadrare tutta la scena con entrambe le telecamere.

Durante le acquisizioni ci si è accorti di quanto rilevante fosse il problema delle variazioni di luminositàă citato in precedenza: in una giornata soleggiata le immagini acquisite con le telecamere viste sino ad ora risultavano quasi totalmente saturate anche utilizzando tecniche di gestione automatica dell'esposizione e dello *shutter*.

2.3 Progettazione del sistema

Nel capitolo precedente si è visto come la posizione di ripresa delle telecamere e l'occlusione delle immagini da parte delle ante del cancello siano problemi rilevanti.

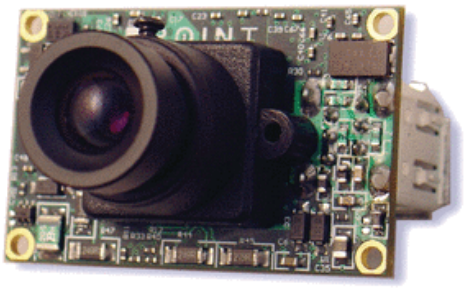
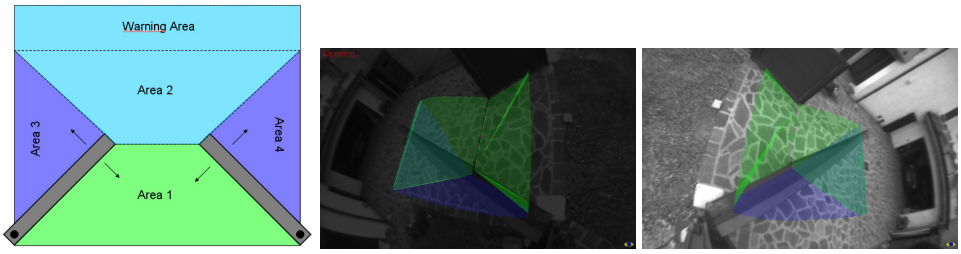


Figura 2.2: Telecamere Firefly MV.

Per la risoluzione di questi si è deciso di simulare virtualmente tre diversi sensori di rilevamento e di assegnare a ciascun sensore una zona di competenza. In particolare si considera in questo caso come sensore non una vera e propria telecamera fisica ma bensì un algoritmo di rilevazione ostacoli unito ad una porzione di immagine da analizzare. Questi sensori virtuali sono divisi secondo la seguente logica:

- **Sensore con elaborazione stereoscopica :** sensore che monitorizza una zona della scena inquadrata da entrambe le telecamere (*Area1*,*Area2*,*Warning Area*).
- **Sensore sinistro con elaborazione monoscopica sinistro:** sensore che monitorizza una zona della scena inquadrata solo dalla telecamera sinistra e non quella destra (*Area3*).
- **Sensore destro con elaborazione monoscopica :** sensore che monitorizza una zona della scena inquadrata solo dalla telecamera destra e non quella sinistra (*Area4*).

In Fig. 2.3.a vi è uno schema logico che descrive la suddivisione delle zone; le motivazioni che hanno portato a questa particolare divisione sono le seguenti:



(a) (b) (c)

Figura 2.3: Zone d'interesse dei sensori virtuali: schema logico *a*, in una immagine reale vista dall'alto *b* ed la stessa immagine vista da una telecamera *c*.

- **Area1:** parte dell'immagine visibile sia alla camera destra che alla camera sinistra, è situata in qualsiasi istante posteriormente rispetto alle ante del cancello.
- **Area2:** parte dell'immagine visibile da entrambe le camere ma situata sempre nella zona frontale rispetto alle ante del cancello.
- **Area3:** parte dell'immagine visibile solamente dal punto di vista della camera destra.
- **Area4:** parte dell'immagine visibile solamente da parte della camera sinistra.

I punti limite della zona denominata come *Area2* sono stati individuati come punti estremi della zona in cui entrambe le telecamere hanno visuale. In Fig. 2.3.a la zona più lontana dalle telecamere è stata segnalata come *Warning Zone* in quanto la zona è situata oltre alla posizione delle fotocellule dell'impianto in esame e di conseguenza la rilevazione in quella regione non sarebbe necessaria. Nonostante ciò si è deciso di effettuare la rilevazione degli ostacoli anche in questa regione aggiungendo il dimensionamento manuale di quest'area in modo da rendere il sensore di sicurezza ulteriormente efficace onde aggiungere funzionalità di anticipo/ritardo apertura/chiusura cancello a discrezione dell'utente.

2.3.1 Individuazione ante del cancello

Il problema successivo riguarda l'individuazione delle ante del cancello all'interno delle immagini, in modo da evitare che queste ultime influiscano sulla rilevazione degli oggetti nella scena; viste le difficoltà incontrate nell'identificazione di queste tramite ate nell'identificazione di queste tramite algoritmi di visione artificiale si è deciso di modellizzare il cancello come oggetto in tre dimensioni, quindi di disegnarne le ante sulla sua immagine semplicemente basandosi sull'angolazione che esse assumono ad ogni istante rispetto all'asse z del sistema di riferimento. Per effettuare questa operazione si necessitano di alcune informazioni riguardanti il sistema su cui il sensore è installato, in particolare:

- altezza, larghezza e spessore di entrambe le ante
- posizione esatta dei perni su cui le ante girano (rispetto al sistema di riferimento)
- posizione esatta delle camere su i tre assi
- inclinazione di entrambe le ante rispetto all'asse z del sistema di riferimento

I dati misurabili sono disponibili da subito all'interno del sistema, per esempio le dimensioni dell'anta e la sua posizione sono note al momento dell'installazione del cancello stesso; mentre l'inclinazione va misurata ad ogni istante di acquisizione della telecamera poiché in caso di movimento di apertura o chiusura essa cambia ad ogni istante. Questo necessità non pone limiti dal punto di vista dell'applicazione pratica, infatti nel sistema reale un semplice *encoder* di posizione applicato ai motori può restituire l'esatto angolo di apertura/chiusura dell'anta.
Il problema si è presentato invece in fase di simulazione: il movimento indotto dai motori sulle ante del varco è una funzione continua nel tempo che ha però un andamento non facilmente predicibile in quanto soggetta a continue ondulazioni che ne rendono difficile la discretizzazione in ambiente simulato. Per aggirare il problema si sono suddivisi i movimenti di apertura e chiusura in movimenti di ampiezza inferiore; inserendo la posizione esatta del cancello all'inizio di ciascuna di queste sezioni di

movimento si è ottenuta una discreta approssimazione dell'andamento generale del moto, senza per arrivare ad eliminare del tutto questo problema.

Un battente del cancello, sia esso di destra o di sinistra, è definita dalla tupla [lunghezza,altezza,spessore, offset x, offset y, alfa], in cui gli offset sono da intendersi come spostamento dell'anta rispetto al centro del sistema di riferimento (nel nostro caso l'angolo in basso a sinistra della zona di osservazione) ed α è definito come angolo di rotazione sull'asse z (l'altezza) dell'anta stessa. Tutti questi dati sono da intendersi in coordinate mondo e a partire da questi dati si ricavano le coordinate dei punti costituenti il cancello, come mostrato dal diagramma di flusso rappresentato in Fig. 2.4.

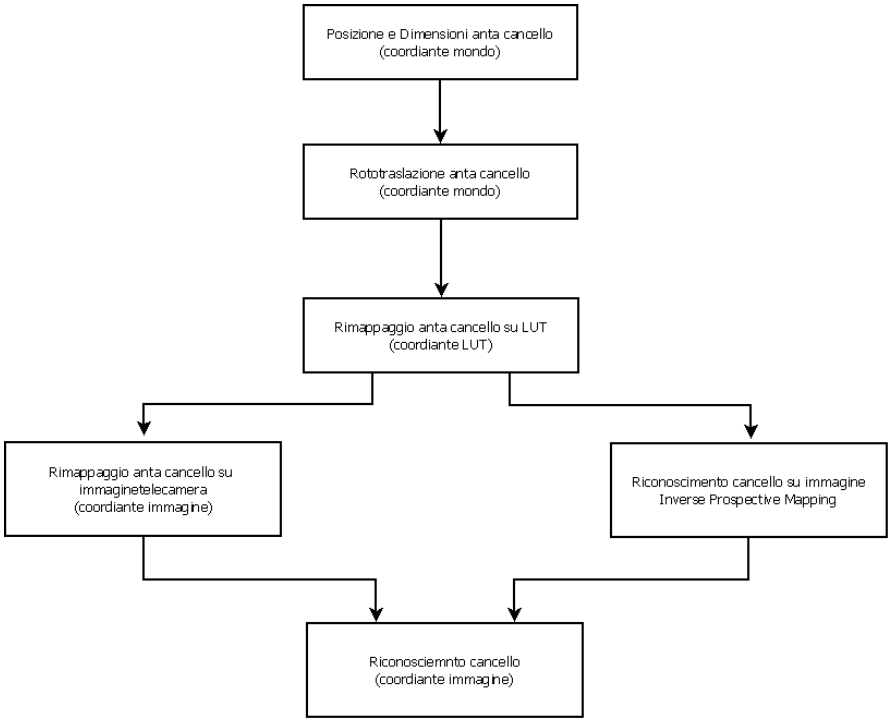


Figura 2.4: Diagramma di flusso sulla trasformazione da coordinate mondo a coordinate immagine.

Nel dettaglio il primo passo del diagramma è rappresentato dalla rotazione dell'anta sull'asse z del sistema di riferimento e dalla sua traslazione lungo gli assi x e y. Ognuno degli otto vertici costituenti il parallelepipedo rappresentante l'anta è costituito in coordinate mondo da un vettore $[Xp, Yp, Zp]$, esso può quindi essere in primo luogo ruotato e secondariamente traslato attraverso un'apposita matrice di rotazione e un vettore di traslazione:

$$\begin{bmatrix} x_i \\ y_i \\ zi \end{bmatrix} = \begin{bmatrix} x_p & y_p & z_p \end{bmatrix} \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \tag{2.1}$$

Dopo quest'operazione si ha una definizione corretta dei punti costituenti l'anta in coordinate mondo e per poter disegnare il cancello sull'immagine IPM si sfruttano le proprietà della *Lookup Table* (LUT). In particolare sono state eseguite due azioni distinte: per prima cosa si sono ricavate le coordinate dei punti sul piano (x,y) attraverso le equazioni:

$$x = -z_c \frac{x_i - x_c}{z_i - z_c} + x_c \tag{2.2}$$

$$y = -z_c \frac{y_i - y_c}{z_i - z_c} + y_c \tag{2.3}$$

dove x_c, y_c e z_c sono gli *offset* rispetto al sistema di riferimento della camera che sta riprendendo la scena.

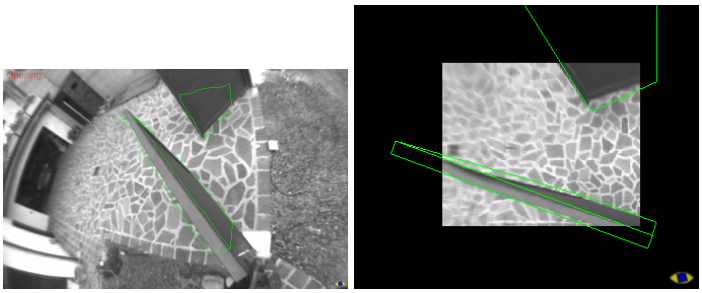
Successivamente si è potuto passare in coordinate LUT attraverso un'altra coppia di equazioni (da cui è possibile ricavare le coordinate LUT del punto in questione):

$$x_l = (x_{max} - x_{min} - x) \frac{length_{ipm}}{x_{max} - x_{min}} \tag{2.4}$$

$$y_l = (y_{max} - y) \frac{height_{ipm}}{y_{max} - y_{min}} \tag{2.5}$$

dove $[x_{max}, x_{min}, y_{max}, y_{min}]$ sono le dimensioni massime e minime della LUT sul piano (x,y) in coordinate mondo, mentre $[length_{ipm}, height_{ipm}]$ sono le dimensioni di

altezza e larghezza della LUT (in coordinate LUT). Si è quindi ottenuta una definizione completa delle coordinate delle ante del cancello nel sistema di riferimento LUT. Questi dati possono essere utilizzati per la rilevazione delle ante sull'immagine IPM, rendendo di fatto possibile la rimozione dei disturbi causati dalla presenza di queste all'interno della sopracitata immagine IPM. Come ultimo passo, sempre sfruttando le proprietà della LUT, si sono potute ricavare le coordinate immagine dei punti cercati attraverso un'apposita funzione presente nel pacchetto *Stereobox* di *GOLD*[18]. I risultati di quanto appena esposto sono visibili nelle immagini 2.5.



(a) (b)

Figura 2.5: Immagini acquisita dalla telecamera sinistra *a* e output dell'algoritmo (in verde) di riconoscimento ante nell'immagine vista dall'alto *b*.

2.4 Algoritmo di rilevazione ostacoli

Individuate ed isolate le diverse zone di interesse all'interno della scena ed eliminato il disturbo introdotto dalla presenza dei battenti sulle immagini, si è passati all'implementazione degli algoritmi necessari all'individuazione degli ostacoli. In precedenza si è accennato alla volontàà di simulare tre diversi sensori di sicurezza basandosi su soli due sensori fisici ed assegnando ad ognuno di essi una delle regioni di pericolo individuate in precedenza; la realizzazione dei tre sensori si concretizza qui attraverso la realizzazione di due diversi algoritmi di rilevazione ostacoli applicati a tre diverse regioni; in questo modo ogni regione ha a disposizione un sensore per l'indi-

viduazione degli ostacoli e la diversitàă dei sensori permette di sfruttare al meglio le peculiaritàă di ogni singola zona.

Il diagramma di Fig. 2.6 rappresenta l'interazione fra i vari algoritmi implementati all'interno del progetto. Nelle zone contrassegnate come *Area3* e *Area4* viene

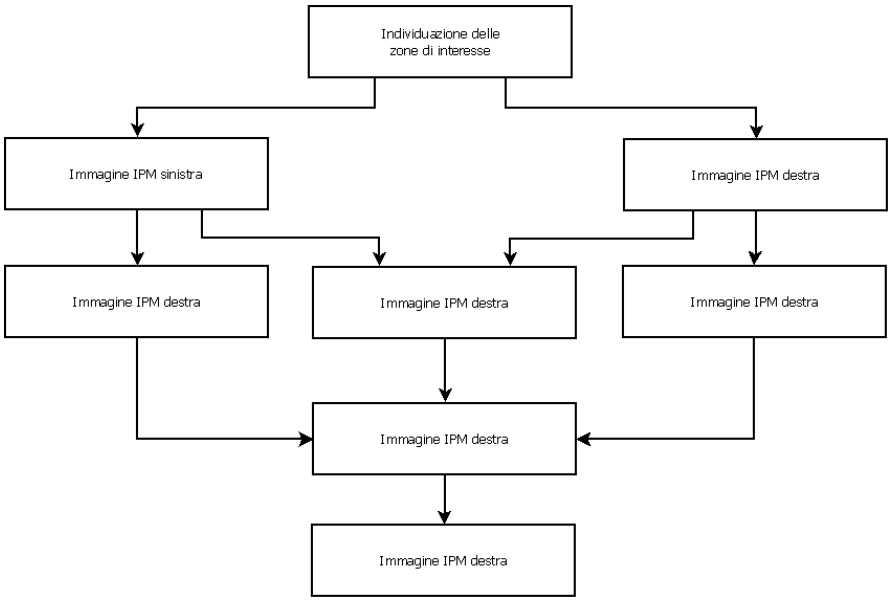


Figura 2.6: Interazione algoritmi di riconoscimento ostacoli implementati.

eseguito un algoritmo di tipo monoscopico poichè in esse una delle due telecamere non ha visuale e quindi un algoritmo di tipo stereoscopico non sarebbe applicabile; al contrario in *Area1* e *Area2* entrambe le telecamere hanno piena visibilitàă ed è stato quindi possibile applicare un algoritmo di tipo stereoscopico. Da notare inoltre che le zone *Area3* e *Area4* esistono solo in fase di apertura o chiusura del varco; infatti quando il cancello risulta completamente fermo, quindi in caso di cancello totalmente aperto o totalmente chiuso, le regioni *Area1*, *Area2* e *Warning Area* occupano l'intera immagine mentre le rimamenti aree risultano avere valore nullo.
Il primo passo di questo algoritmo è la generazione delle immagini IPM da quelle

acquisite dalla telecamera destra, sinistra, immagine di *background* destra e immagine di *background* sinistra. Per immagine IPM s'intende il risultato di un'operazione di *Inverse Perspective Mapping*[19] applicata ad una delle immagini viste in precedenza; l'IPM crea un'immagine con un dominio in due coordinate partendo da un'immagine proveniente da un dominio con tre coordinate; sostanzialmente serve ad eliminare la prospettiva da un'immagine creando un'immagine 2D di un mondo intrinsecamente 3D.

2.4.1 Rilevazione ostacoli con algoritmo monoscopico

L'algoritmo che si occupa della rilevazione degli ostacoli nelle zone mono è costituito principalmente da due parti: nella prima si riceve come input l'immagine IPM della telecamera e se ne restituisce un'immagine binarizzata contenente gli ostacoli presenti sulla scena; nella seconda si riceve come input l'immagine binarizzata e si effettua un'analisi delle informazioni in essa contenute con lo scopo di individuare eventuali ostacoli.

Individuazione delle immagini contenenti ostacoli

Nella prima parte dell'algoritmo vengono generate delle immagini binarizzate contenti le informazioni sugli ostacoli presenti sulla scena dall'algoritmo rappresentato in Fig. 2.7.
Il primo passo di questo algoritmo consiste nell'acquisizione dell'immagine di *background* che verrà utilizzata per l'individuazione degli ostacoli sulla scena.
L'immagine di *background* viene acquisita in un primo momento come prima disponibile delle telecamere a partire dall'istante in cui entrambe le telecamere sono attive; questo metodo di acquisizione si è rilevato però problematico a causa dei molteplici difetti intrinseci ad un algoritmo di rilevazione come quello implementato.

Un'immagine come quella di Fig. 2.8 evidenzia problemi dell'agoritmo in questione dovuti ad un'immagine di *background* non corretta o comunque non sufficien-

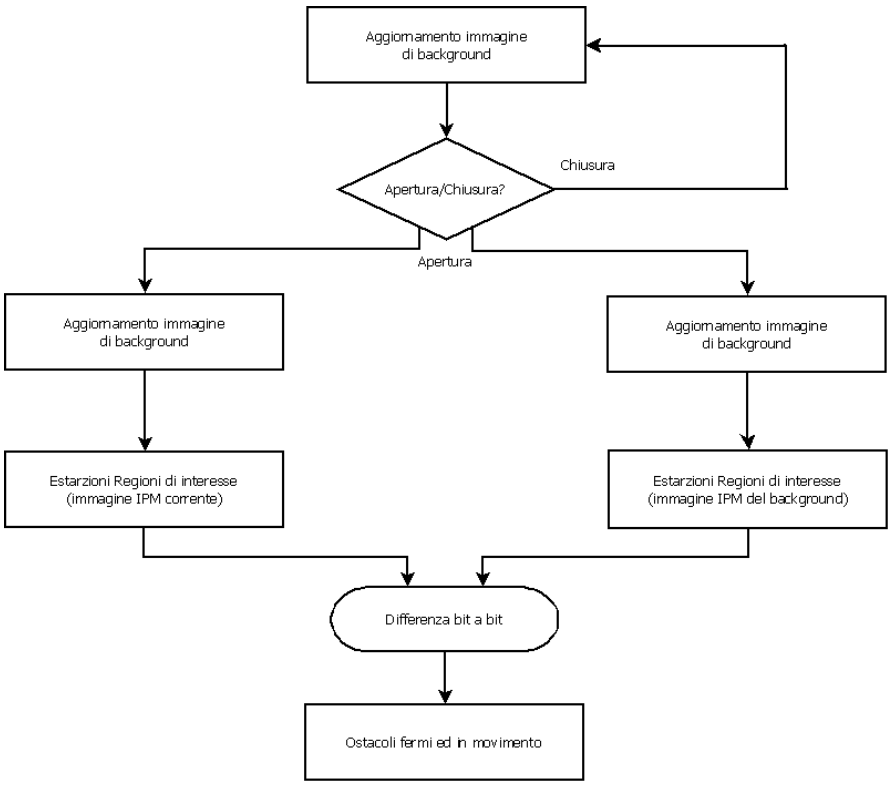
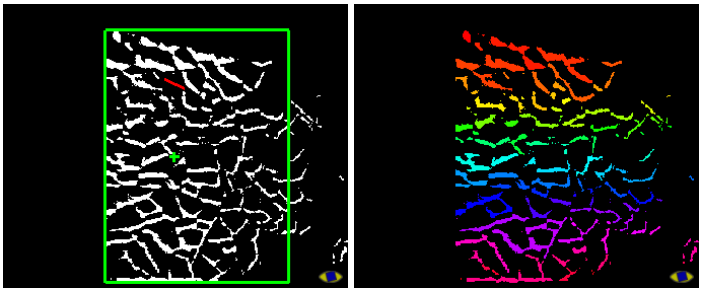


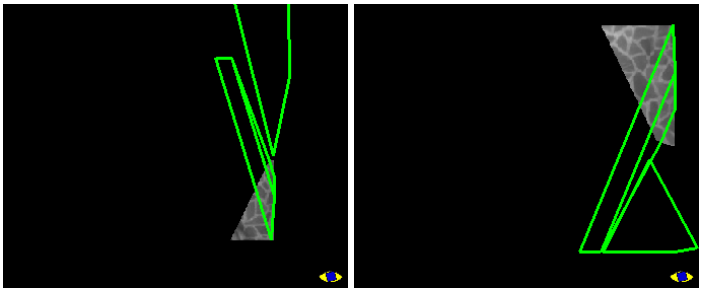
Figura 2.7: Algoritmo di rilevazione monoscopica, generazione immagini.



(a) (b)

Figura 2.8: Algoritmo monoscopico: problemi di *background*, rilevazione ostacolo *a* e immagine etichettata *b*.

temente aggiornata.
Come soluzione si è proposto di acquisire come nel caso precedente l'immagine di *background* come primo frame disponibile ed inoltre, per ogni frame successivo, di continuare ad acquisire l'immagine finchè il sensore deputato alla visione stereoscopica non segnali la presenza di ostacoli sulla scena , o in caso di mancanza di ostacoli, fino al momento in cui non sia segnalato un comando di apertura o, in chiusura del varco automatico.
In questo modo si ha la certezza di acquisire in ogni caso l'immagine di *background* di ciascuna telecamera nell'ultimo istante possibile; si segnala inoltre che i sensori funzionante tramite algoritmo mono, cioè quelli deputati alla gestione di *Area3* e *Area4*, rimangono spenti fino all'istante prima dell'ultima acquisizione dell'immagine di sfondo.
Il passo successivo nell'algoritmo è l'eliminazione dell'immagine dei battenti dalle immagini IPM delle telecamere destra/sinistra e dalle immagini IPM di *background* destra/sinistra. Per fare ciò si sono usati i dati elaborati nel capitolo precedente in cui, partendo dalla conoscenza dei dati dimensionali dei battenti e dalla conoscenza dell'esatta posizione di questi all'interno del sistema di riferimento , si erano individuati tutti i vertici del parallelepipedo costituente il battente sia nell'immagine originale che nell'immagine IPM.
Come punto di partenza per la creazione del metodo di eliminazione dei battenti si è usata la classe *PatchSewer* presente all'interno del pacchetto *GOLD*[18]: essa è stata normalmente implementata per l'applicazione di *patch* nere in determinate parti dell'immagine ed è stata modificata in modo da sostituire le semplici *patch* nere con intere porzioni di una seconda immagine che viene passata come parametro al metodo. In particolare è stato creato un *overloading* del metodo *SewPatch* funzionante per *patch* triangolari o rettangolari ed è contraddistinto da un funzionamento molto semplice: i pixel dell'immagine di destinazione vengono scorsi uno ad uno, coloro i quali risultassero appartenti all'area interna alla figura geometrica passata come parametro vengono sostituiti con i pixel nella stessa posizione nella seconda immagine che viene passata al metodo *SewPatch*.



(a) (b)

Figura 2.9: Immagine IPM sinistra*a*,destra *b*, zone monoscopiche, e disegno tridimensionale dei battenti in colore verde.

Il passo successivo è rappresentato dall'estrazione delle zona di interesse sia dall'immagine IPM delle telecamera che dalle immagini IPM di *background*. Per ottenere ciò è necessario definire le aree viste in precedenza. Nel caso dell'algoritmo in esame si devono estrarre la regione denominata *Area3* dalle immagini IPM (telecamera/*background*) di destra mentre al contrario si deve estrarre la regione *Area4* delle immagini IPM (telecamera/*background*) di sinistra. Lo scopo è stato raggiunto mediante un nuovo *overloading* di alcuni metodi della giàă citata classe *PatchSewer*, in particolare è stato creato un nuovo metodo che presi in ingresso una serie di punti rappresentanti una figura geometrica (triangolo/rettangolo) copia una zona d'immagine dell'immagine presa come input in una nuova immagine appositamente inizializzata come completamente nera, con un funzionamento analogo a quanto visto per il metodo precedente.
I risultati di questa elaborazione sono visibili nella Fig. 2.9, in cui in verde vengono evidenziati per chiarezza gli elementi costituenti il cancello mentre la porzione di immagine visibile è quella corrispondente all'area in cui si applica l'algoritmo mono.
Una volta ottenute queste immagini si è effettuata una sottrazione pixel a pixel tra l'immagine IPM attuale e quella impostata come *background*, evidenziando in questo modo le modifiche avvenute sulla scena rispetto alla situazione iniziale; questo permette di rilevare eventuali oggetti pervenuti sulla scena che essendo l'immagine binarizzata da un apposito filtraggio a soglia risaltano stagliandosi in bianco su

sfondo nero come visibile nell'immagine 2.10.a.

(a) (b) (c)

Figura 2.10: Rilevazione ostacolo in area mono, immagine differenza IPM *a*, *blob* rilevato in verde *b*, *output* nell'immagine originale *c*.

Analisi delle immagini e classificazione ostacoli

L'algoritmo generale di analisi delle immagini binarizzate è riportato nelle Fig. 2.11, il quale è composto da:

- **Acquisizione immagine binarizzata:** l'immagine viene binarizzata tramite un filtro a soglia il cui valore di soglia gestito tramite uno slider all'interno dei pannelli di *GOLD*[18].
- **Erosione:** ha lo scopo di eliminare componenti di piccola entità e di aumentare l'apertura fra componenti non connesse.
- **Espansione:** ha un effetto contrario rispetto all'erosione, infatti aumenta la dimensione di componenti di piccola entità e chiude le apertura fra componenti non connesse.
- **Analisi:** raccoglie dati sulle componenti in questione e li utilizza per il loro filtraggio e per la stampa di informazioni sugli ostacoli presenti sulla scena.

L'erosione, l'espansione e l'analisi, essendo operazioni comuni a tutti gli algoritmi di analisi immagini presenti nell'applicazione, verranno trattate nel paragrafo 2.4.3.

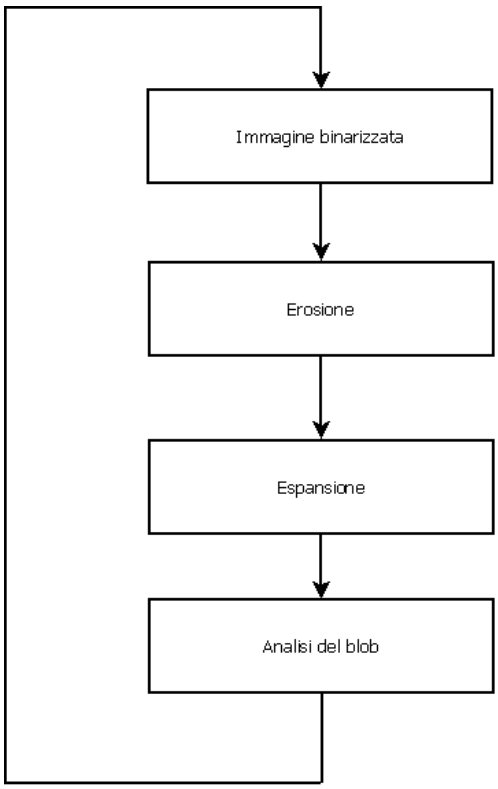


Figura 2.11: *Preprocessing* su immagine binarizzata.

I sensori virtuali monoscopici forniscono come output al sistema l'eventuale presenza di un oggetto e alcuni altri dati statistici sulla posizione e dimensione degli ostacoli. In particolari questo algoritmo non permette il calcolo del punto di contatto a terra degli oggetti, nè la loro altezza a causa delle limitazioni introdotte dalla visione monoscopica che essendo priva di prospettiva inibisce una visione 3D della scena. Per contro si sono riusciti a ricavare altri dati importanti per il sistema:

- **Rettangolo circoscritto:** è il rettangolo che racchiude completamente le componenti (*blob*) rilevate dal sistema.

- **Baricentro:** è il dato più importante dell'elaborazione, rappresenta il punto in

cui si suppone sia situato l'oggetto.

- **Area:** rappresenta il numero di pixel bianchi contenuti all'interno del *blob*, risulta utile per dare un'idea della grandezza dell'ostacolo.

Una particolare importanza è rivestita dalle informazioni sul baricentro del *blob*, il cui algoritmo di calcolo è rappresentato in Fig.2.10.b.

Per ogni pixel all'interno del *blob* viene controllato per prima cosa il valore di luminosità, se esso corrisponde ad un pixel bianco vengono aumentati due accumulatori con il valore della coordinata x e y rispettivamente del punto; una volta controllati tutti i pixel all'interno del *blob* , vengono generate le coordinate del baricentro, denominati *baricentre.y* e *baricentre.y* in Fig.2.10.b, dividendo il valore degli accumulatori per il numero di pixel bianchi contenuti nel 2.10.a. Questa operazione viene effettuata durante la prima analisi dei *blob* ma anche in tutte le successive modifiche alla struttura dei *blob*, modifiche dovute a unione o estensione di uno o più *blob*.

2.4.2 Rilevazione ostacoli con algoritmo stereoscopico

Nelle zone *Area1* e *Area2* (vedi Fig.2.3) viene applicato un algoritmo che sfrutta la visione stereoscopica per l'individuazione degli ostacoli sulla scena. L'algoritmo è diviso sostanzialmente in due parti:

- Estrazione regioni di interesse
- Preelaborazione e analisi del contenuto

Estrazione delle regioni di interesse

Il diagramma di flusso di Fig.2.12 rappresenta la prima fase dell'algoritmo.

L'algoritmo inizia con l'acquisizione delle immagini IPM provenienti dalla telecamere destra e sinistra rispettivamente; successivamente il sistema controlla se il cancello è in movimento o meno in quanto le operazioni da eseguire sono diverse nei

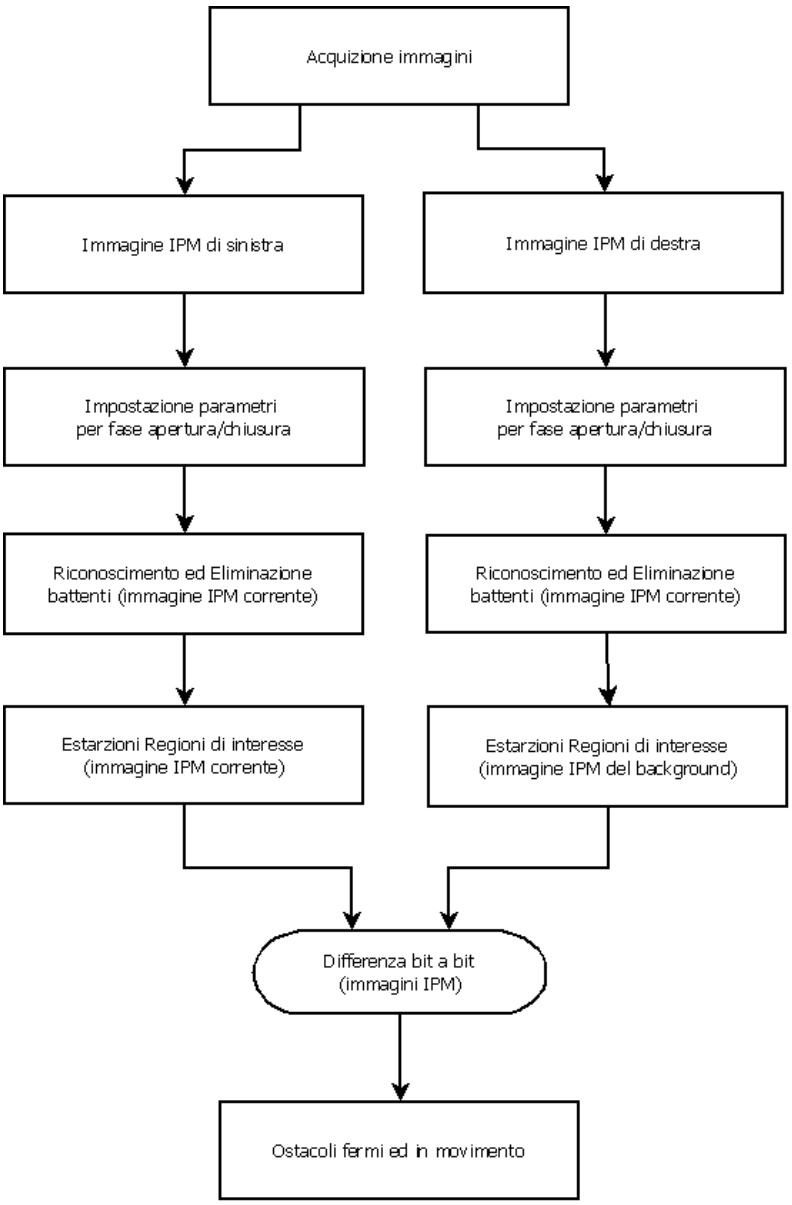
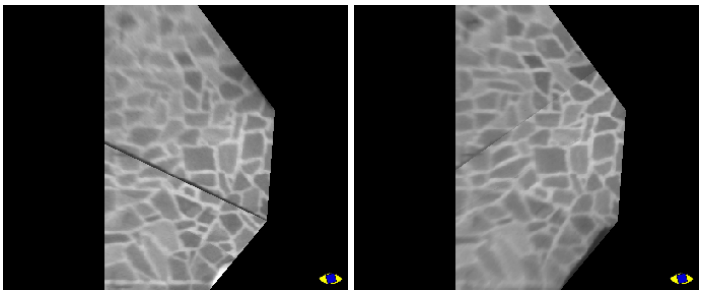


Figura 2.12: Algoritmo stereoscopico: generazione delle immagini binarizzate.

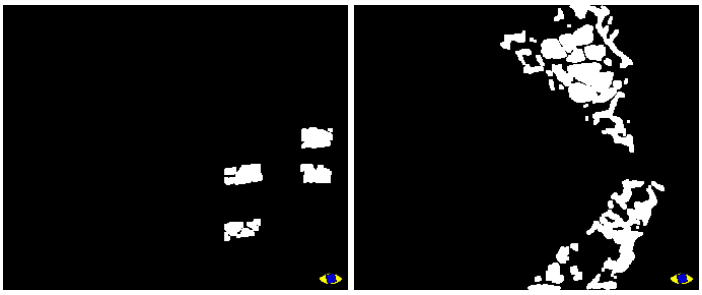
due casi. In caso di cancello fermo si ha il solo mascheramento della *Warning Area* su entrambe le immagini: per far questo si è fatto uso dei metodi presenti nella classe *Patch Sewer* ed in particolare si è sfruttato il metodo *BlackPatchSew* che, dati quattro punti costituenti un rettangolo, permette di applicare una *patch* nera nei punti interni al rettangolo stesso.
Nel caso in cui il cancello sia in movimento si eseguono due passi distinti: per prima cosa si eliminano le proiezioni dei battenti sulle immagini IPM destra e sinistra. Per far questo si sono usati i vertici dei parallelepipedi costituenti le ante del cancello e si sono eliminati tramite un'*overloading* del metodo *SewPatch* della classe *PatchSewer* come giàă descritto nella sezione precedente a riguardo dell'algoritmo monoscopico. Successivamente si sono estratte dalle due immagini IPM le aree di interesse. Nel caso dell'algoritmo stereoscopico le aree di interesse sono state contrassegnate come *Area1* e *Area2* ma vengono estratte entrambe solamente in caso di chiusura. In particolare durante il moto di apertura estratta solamente la regione denominata *Area2* in quanto un oggetto situato in *Area1* non viene considerato a tutti gli effetti un ostacolo poichè in quella zona non è in pericolo durante l'apertura.
L'estrazione delle regioni di interesse avviene come descritto per l'algoritmo monoscopico: si è creato un nuovo metodo nella classe *PatchSewer* che presi in ingresso una serie di punti rappresentanti una figura geometrica (triangolo/rettangolo) copia una zona d'immagine dell'immagine presa come input in una nuova immagine appositamente inizializzata come completamente nera, con un funzionamento analogo a quanto visto per il mascheramento dei battenti.

In Fig.2.13 si illustra proprio questo risultato, con l'evidenziazione del mascheramento appena spiegato. L'ultimo passo dell'algoritmo consiste in una differenza pixel a pixel dell'immagini IPM destra con l'immagine IPM sinistra; così facendo si possono ottenere tutti gli oggetti sulla scena non appartenenti al terreno, cioè tutti gli ostacoli presenti nell'area di interesse, come visualizzato nella Fig.2.14.



(a) (b)

Figura 2.13: Immagine IPM ricavata dalla telecamera sinistra *a* e da quella destra *b*.



(a) (b)

Figura 2.14: Differenza fra immagine IPM destra e sinistra inquadrante ostacolo di piccole dimensioni *a* e di grandi dimensioni *b*.

Analisi delle immagini

La seconda parte dell'algoritmo di rilevazione ostacoli ha lo scopo di ricevere in ingresso un'immagine binarizzata e produrre in uscita una serie di informazioni sugli oggetti presenti sulla scena.

Il funzionamento di questo algoritmo è raffigurato in Fig.2.11; esso è costituito da un operazione di erosione , da un'espansione e infine dall'analisi vera e propria degli oggetti presenti sulla scena. Queste operazioni, essendo comuni ad entrambi gli algoritmi, verranno trattate più approfonditamente nella sezione successiva.
Più interessante è l'analisi dell'output fornito dall'algoritmo stereoscopico, esso è costituito da

- **Punto di contatto a terra:** supponendo che non vi siano oggetti staccati dal suolo nelle immagini questo dato rappresenta le coordinate *(x,y)* del punto in cui un ostacolo tocca il terreno.

- **Rettangolo Circoscritto:** è il rettangolo che racchiude completamente le componenti (*blob*) rilevate dal sistema.

- **Area:** rappresenta il numero di pixel bianchi contenuti all'interno del *blob*, risulta utile per dare un'idea della grandezza dell'ostacolo.

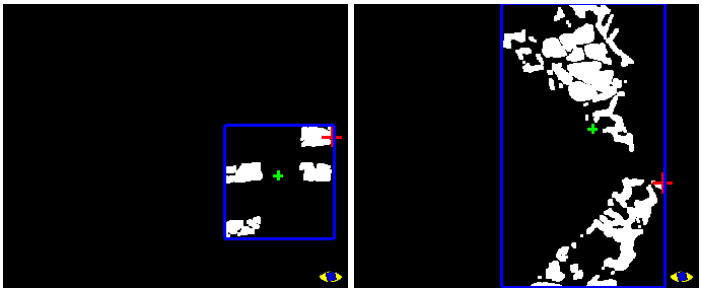
Una grande importanza è rivestita dal punto di contatto a terra: esso è l'output più significativo per sistemi di visione di questo tipo e permette in primo luogo di capire esattamente dove si trova l'ostacolo in modo da poter attivare una politica di gestione del cancello coerente con la posizione degli ostacoli presenti sulla scena. Per il calcolo di questo dato si sono seguite due strade:

- **Metodo delle rette incidenti:** per ogni coppia di *blob* viene calcolata l'orientamento generale del *blob* , se i due *blob* appartengono allo stesso oggetto il punto di contatto a terra è il punto di intersezione fra le rette rappresentati l'orientazione delle sagome degli oggetti nelle immagini *IPM*.

- **Punto più vicino alle telecamere:** vista l'orientazione delle telecamere il punto di contatto a terra di un qualsiasi oggetto di altezza finita sarà il punto appartenete al *blob* dell'ostacolo che si trova più vicino alle telecamere.

Il metodo delle rette incidenti calcola, se esiste, la retta rappresentante l'andamento generale del *blob*; lo fa cercando, tra tutti i pixel bianchi all'intero del *blob* , quello che risulta essere più lontano dal centro del *blob* stesso. La retta passante tra questo punto ed il centro del *blob* saràà la retta rappresentante l'andamento medio dell'orientazione dei pixel componenti il *blob*; l'equazione della retta è scritta nella classica forma $y = mx + q$.
Il passo successivo di questo algoritmo è la classificazione delle rette in orizzontali e verticali: se la coordinata x del punto usato per l'intersezione risulta essere tra l'inizio e la fine delle coordinate x in cui il *blob* è definito allora la retta ha un andamento verticale, in caso contrario si dice che essa ha un andamento orizzontale. Questa classificazione risulta molto utile nel passo successivo dell'algoritmo durante il confronto fra *blob*: infatti presi due *blob* contigui, se essi hanno rette rappresentanti la pendenza con verso opposto e le rette stesse si intersecano all'interno dell'immagine allora l'intersezione è il punto di contatto a terra dell'oggetto rappresentato nei *blob*. Per discriminare facilmente quali rette abbiano verso opposto si considera la classificazione fatta in precedenza, quindi rette con andamento orizzontale sono considerate opposte a rette con andamento verticale.
Il metodo delle rette incidenti ha però evidenziato alcuni limiti durante i test effettuati: in primo luogo risulta computazionalmente dispendioso per il sistema, inoltre si sono incontrate difficoltàă nel definire le politiche per cui due rette erano da considerare effettivamente di verso opposto, a causa anche della particolare posizione delle camere che si ha in questo caso. Per questi motivi si è deciso di implementare un secondo metodo per il calcolo del punto di contatto a terra, basato sulla distanza del *blob* dalle camere. Questo secondo algoritmo può essere descritto nel seguente modo: per ogni pixel all'interno del *blob* l'algoritmo cerca quello il più distante dalle telecamere; la distanza è definita come somma delle distanze euclidee fra il pixel e i punti rappresentati le due telecamere all'interno del sistema di riferimento del sistema.

Data la particolare orientazione delle telecamere ogni oggetto presente sulla scena si troveràà sempre frontalmente ad esse e nell'immagine IPM eventuali ostacoli verranno stirati sull'immagine stessa a partire dal punto di contatto a terra in direzione uscente rispetto al verso delle telecamere: questo permette di assicurare che il punto meno distante dalle telecamere è sicuramente il punto di contatto a terra dell'oggetto.



(a) (b)

Figura 2.15: Evidenziazione *blob*, oggetto di piccole dimensioni *a* e di grandi dimensioni *b*, punto di contatto evidenziato in rosso.

Questo è dimostrato anche dalle immagini di Fig.2.15 in cui il punto di contatto a terra di una persona viene riconosciuta.

2.4.3 Operazioni morfologiche di visione artificiali comuni

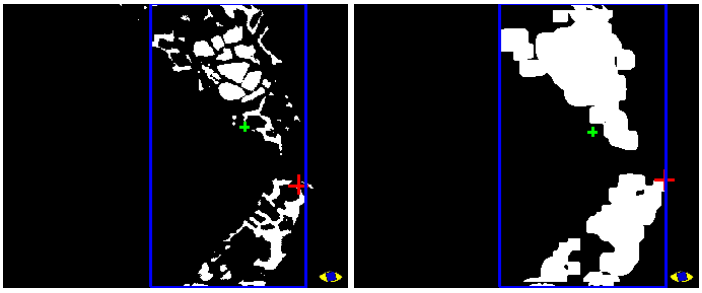
Erosione

Con il nome di *erosione*[21] si identifica un'operazione di matematica morfologica che ha lo scopo di erodere i bordi dei *blob* al fine di eliminare i *blob* di piccole dimensioni e diminuire la dimensione dei rimanenti. La matematica morfologica è ampiamente trattata nella letteratura di settore e non viene quindi trattata in questo contesto.
L'implementazione effettuata di questo operatore viene invece riportata in quanto differisce dalle implementazioni comuni.

Nell'algoritmo tutti i pixel dell'immagine vengono testati, per ognuno di questi che risulta essere bianco (quindi con valore di luminosità 255) viene creato un *bounding box* di dimensione scelta dall'utente; successivamente viene creato un accumulatore contenente la somma di tutti i pixel all'interno del *bounding box*, se esso risulta essere maggiore di una certa soglia il pixel di partenza viene impostato nero (quindi con valore di luminosità pari a zero). Il *bounding box* generato non é altro che una matrice di ordine dispari centrata nel pixel in esame.

Espansione

Con il nome di *espansione*[21] si identifica l'operatore di morfologia matematica che ha lo scopo di aumentare le dimensioni di un *blob* in base all'elemento strutturante usato per effettuare l'*espansione*; come nel caso dell'*erosione* questo un tema trattato nella letteratura di settore , motivo per cui non la trattazione teorica non viene effettuata in questa sede.
I pixel dell'immagine in ingresso vengono testati uno ad uno, per tutti quelli che risultano neri (luminositá pari a zero) viene creato un *bounding box*. Il valore di tutti i pixel all'interno del *bounding box* viene accumulato in una variabile , una volta esauriti viene fatto un confronto fra il valore così ottenuto e una soglia: se l'accumulatore risulta essere maggiore della soglia il pixel centrale del *box* viene impostato a valore di luminositá massimo. Come nel caso precedente il *bounding box* non è altro che una matrice di ordine dispari centrata nel pixel in esame la cui dimensione viene scelta dall'utente.
Dalla Fig.2.11 si nota come i due operatori appena esposti vengono utilizzati sempre in modo sequenziale: l'esecuzione di un'*erosione* seguita da un'*espansione* è conosciuta con il nome di *apertura* e ha lo scopo di separare *blob* debolmente collegati e rinforzare *blob* isolati. L'effetto di un'operazione di *apertura* è rappresentata nella Fig.2.16. Qui possiamo vedere rappresentata a destra un'immagine IPM binarizzata contenente un ostacolo senza l'applicazione di nessun operatore morfologico, a sinistra la stessa immagine a cui è stata applicata un'apertura: i *blob* appaiono mag-



(a) (b)

Figura 2.16: Effetti di un operazione di apertura, rilevazione senza *a* e con *b* apertura dell'immagine

giormente definiti e distinti, sono inoltre spariti alcuni *blob* di dimensioni minori.

2.4.4 Classificazione degli ostacoli

L'analisi delle informazioni contenute nei *blob* è l'operazione più importante dell'intera applicazione in quanto consente di estrarre da immagini preelaborate le informazioni necessaria al funzionamento del sistema. In particolare le informazioni estratte dall'immagine sono:

- Numero di *blob*
- Numero di pixel presenti nel *blob*
- Coordinate estremo destro, sinistro, alto e basso del *blob*
- Primo vertice e dimensione dei lati del quadrato circoscritto
- Posizione del baricentro del *blob*
- Pendenza e intercetta della retta raffigurante l'inclinazione
- Distanza del *blob* dalle telecamere

- Eventuale punto di contatto a terra del *blob* (per algoritmo stereo)

Per fare questo ad ogni istante viene applicato l'algoritmo di Fig.2.17.

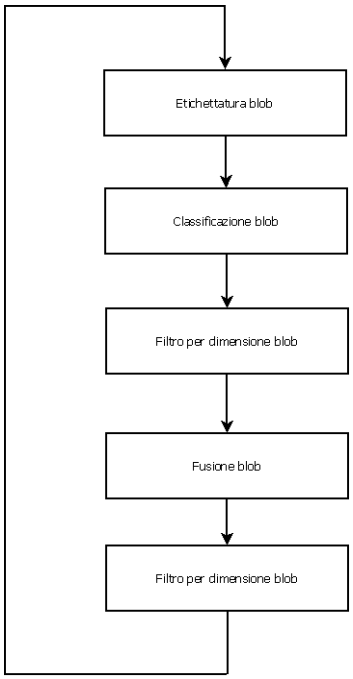


Figura 2.17: Classificazione ostacolo: algoritmo generico

Esso consta di cinque passi principali, inizialmente viene effettuata una etichettatura dei *blob*, dopodiché vengono eseguiti in sequenza il riconoscimento dei *blob*, seguito da un filtraggio basato sull'area minima dei *blob* e dall'unione dei *blob* secondo politiche che verranno descritte nel proseguo, infine un secondo filtraggio sull'area completa l'algoritmo. Verranno ora analizzati uno ad uno i passi appena descritti.
Il primo passo dell'algoritmo è descritto come *Blob Labelling*; esso è un passo preliminare al riconoscimento dei *blob* ed ha lo scopo di produrre un immagine in cui i *blob* siano etichettati in modo che l'analisi possa distinguere i pixel appartenenti ad un *blob* piuttosto che ad un altro.
L'etichettatura (labelling) prende in input un immagine binarizzata, testa pixel a pixel

l'immagine ed assegna ad ogni gruppo di pixel bianchi contigui un livello di grigio in una scala che va da 1 a 255. In uscita si ha quindi un immagine a toni di grigio in cui le zone dell'immagine originale appartenenti ad uno stesso *blob* sono rappresentate con lo stesso livello di grigio; per comodità dell'utente la stessa immagine può essere convertita a colori utilizzando il pattern RGB, come visibili nelle immagini di Fig.2.10.a.
Dopo aver etichettato l'immagine tramite la funzione appena vista è possibile effettuare il riconoscimento dei *blob*.

Durante il riconoscimento dei *blob* ogni pixel dell'immagine etichettata viene preso in esame: il proprio valore di luminosità ne definisce l'appartenenza ad un un *blob* piuttosto che ad un altro per cui per ogni pixel viene esaminato questo valore, se esso appartiene ad un *blob* già creato semplicemente vengono aggiornate le statistiche del *blob* in questione considerando l'aggiunta del nuovo pixel. Vengono quindi aggiornati dati come il numero di pixel presenti, il baricentro e le coordinate del rettangolo circoscritto. Viceversa se il tono di grigio del pixel non viene riconosciuto in nessuno dei blob già esistenti viene creato un nuovo *blob* e contemporaneamete vengono generate tutte le statistiche ad esso associate. Il passo successivo nello svolgimento dell'algoritmo di classificazione degli ostacoli è un'operazione filtraggio basato sull'area di ciascun *blob*: ogni *blob*, come visto durante la prima fase dell'analisi, ha al suo interno una statistica in cui sono espressi i numeri di pixel da cui è costituito e di conseguenza la sua area. Il filtraggio basato sull'area minima elimina dall'immagine tutti i *blob* il cui valore di area è inferiore ad una soglia fissata dall'utente, aggiornando di conseguenza la lista dei *blob* e le relative statistiche. Questo primo filtraggio ha lo scopo di eliminare tutte le componenti dell'immagine che sono state generate da rumore presente nell'immagine oppure da un qualche errore nell'algoritmo di preparazione delle immagini stesse; queste componenti vanno eliminate in quanto rappresenterebbero solamente un problema per il proseguo dell'algoritmo e per il processo decisionale in genere.
A seguito del filtraggio si ha una situazione in cui vi sono molti *blob* nell'immagine di dimensione contenute, al fine di distinguere quelli con contenuto informativo da quelli che rappresentano solamente rumore si è implementato il *blob merging* (unione

dei *blob*). In questa operazione ogni *blob* viene analizzato in relazione agli altri *blob* dell'immagine e in base ad alcune politiche definite a priori l'algoritmo di *merging* unisce alcuni *blob* invece di altri. L'algoritmo in se non risulta interessante, viceversa lo sono le politiche attraverso le quali i *blob* vengono uniti. Le politiche di unione dei *blob* implementate sono tre, raggruppabili in due gruppi distinti:

- **Unione per intersezione** : vengono uniti *blob* i cui rettangoli circoscritti si intersecano.
- **Unione per contiguitá** : vengono uniti i *blob* sulla base di regole di vicinanza dei baricentri.

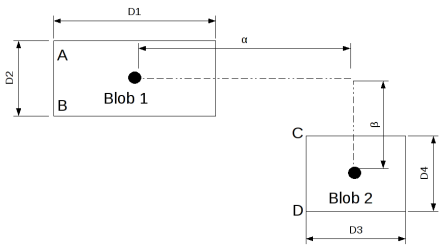


Figura 2.18: Politiche di *merging*, schema

In Fig.2.18 sono rappresentati due *blob* con le variabili utili a comprendere le politiche di unione che verranno esposte. La politica di unione per intersezione ha lo scopo di unire tutti i *blob* i cui lati del rettangolo circoscritto si intersecano, la relazione logica che permette questa unione è:

$$(A.x > C.x) \text{ U } (A.x < C.x + D3) \text{ U } (A.y > C.y) \text{ U } (A.y < C.y + D4)$$
$$(A.x > C.x) \text{ U } (A.x < C.x + D3) \text{ U } (A.y + D2 > C.y) \text{ U } (A.y + D2 < C.y + D4)$$
$$(C.x > A.x) \text{ U } (C.x < A.x + D1) \text{ U } (C.y > A.y) \text{ U } (C.y < A.y + D2)$$
$$(C.x > A.x) \text{ U } (C.x < A.x + D1) \text{ U } (C.y + D4 > C.y) \text{ U } (C.y + D4 < A.y + D2)$$

Nel momento in cui una di queste relazioni diventi vera per due *blob* si ha l'unione dei *blob* stessi. La seconda tipologia di politiche è stata chiamata unione per contiguitèă:

essa prevede l'unione in base a regole di vicinanza tra i centri dei *blob*. In questa tipologia sono definite due politiche:

Metodo A: distanza tra le coordinate del baricentro in entrambe le direzioni proporzionale alla dimensione del rettangolo.

Metodo B: distanza euclidea fra i baricentri proporzionale alla diagonale maggiore del più grande dei due *blob*.

La relazione matematica a cui è affidata l'esecuzione del primo tipo di unione è:

$$alfa < (D1 + D2)/2 + theta$$
$$beta < (C1 + C2)/2 + theta$$

essa lega la distanza sull'asse x fra i due baricentri alla dimensione del lato parallelo all'asse medesimo, così come per l'asse y.
La seconda relazione del sottogruppo unioni per contiguitàà è implementata nella seguente espressione:

$$\sqrt{alfa^2 + beta^2} < maxdiag$$

dove per *maxdiag* si intende la diagonale di dimensione maggiore fra le diagonali dei due *blob*.
Come ultimo passo nell'algoritmo di analisi si effettua una seconda sogliatura in base alla dimensione dei *blob*: questa seconda sogliatura viene effettuata poiché dopo l'unione i *blob* di effettivo interesse avranno dimensioni molto maggiori rispetto a quelli imputabili ad errori nell'algoritmo; di conseguenza una seconda operazione di filtraggio può utilizzare una soglia molto piú alta senza il pericolo di eliminare *blob* contenenti informazioni utili.

2.5 Collegamento al sistema reale

Dopo la fase di sperimentazione e test degli algoritmi in laboratorio con sequenze di immagini acquisite dal sistema reale si è reso necessario implementare un dispositivo

elettronico in grado di interfacciare il calcolatore con il sistema di automazione del cancello automatico.

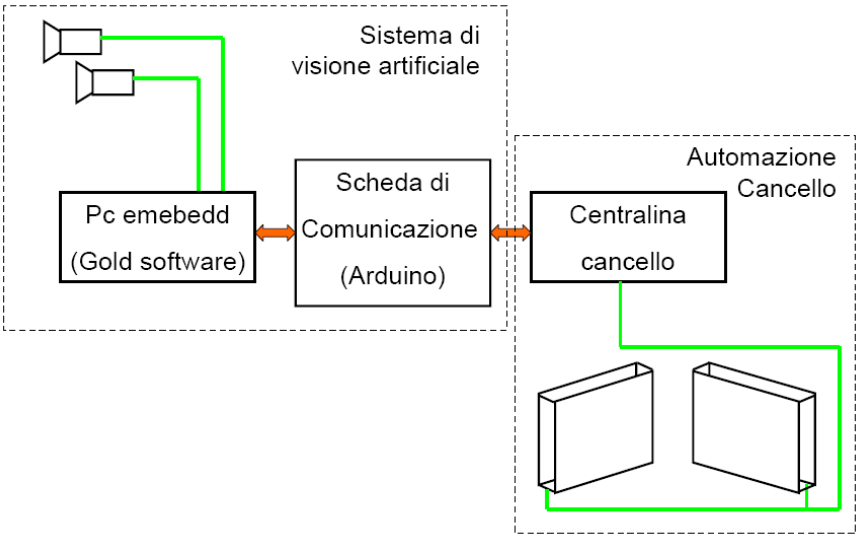


Figura 2.19: Architettura generale.

Per far questo si è reso necessario l'utilizzo di un certo numero di dispositivi hardware, preesistente o progettati ad-hoc ed si sono creati dei plugin software per la connessione del software *GOLD*[18] con l'hardware. Si presenta lo schema di comunicazione creato: in Fig.2.20 viene rappresentato il sistema di automazione e ed il loro interfacciamento. Il sistema è così composto:

- Pc equipaggiato con gold:
- Centralina
- Scheda di comunicazione gold/centralina

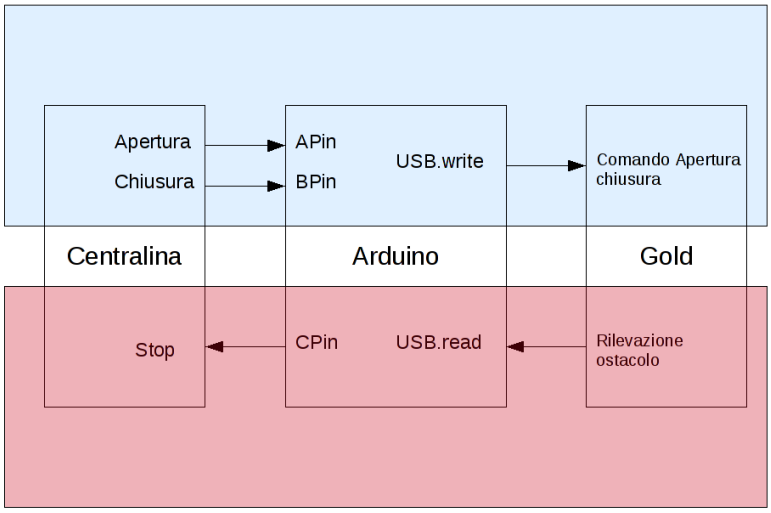


Figura 2.20: Schema input/output comunicazione fra i diversi sistemi.

PC equipaggiato con gold

Il sensore di sicurezza implementato in questa tesi è, come giàă detto, basato su un'applicazione realizzata per *GOLD*[18], in particolare per *GOLD*[18] versione 4.1 . Come output verso la centralina il software produce ad ogni frame delle camere un bit di informazione: 0 per assenza ostacolo, 1 per presenza ostacolo. Come si può vedere la logica è molto semplice, questo è dovuto al fatto che si va a sostituire una coppia di fotocellule e che quindi l'unico dato comunicabile alla centralina era la presenza o meno dell'ostacolo sulla scena.

Arduino

La scheda in Fig.2.21 è il device *Arduino*[20], prodotto dall'omonima ditta e disponibile su *www.arduino.cc*.

Arduino[20] è una piattaforma elettronica open-source sia dal lato hardware che dal lato software , può interagire con altri dispositivi attraverso la connessione di questi ai pin digitali ed analogici presenti sulla scheda. In particolare durante il progetto è

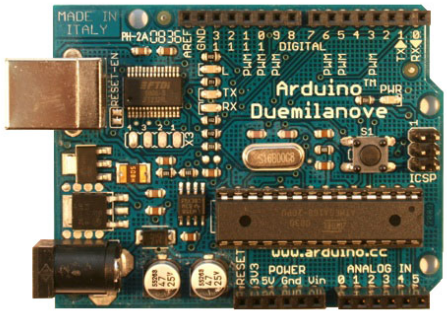


Figura 2.21: Arduino Duemilanove.

stato utilizzato il dispositivo *Arduino Duemilanove*, il cui nome deriva dal primo anno in cui è stato commercializzato, come evidenziato in Fig.2.21. La versione *Duemilanove* è basata sul controllore ATmega169 , è dotata di 14 digital input/output pins, 6 input analogici ,un processore con frequenza 16 MHz e un connettore USB; le principali caratteristiche di *Arduino*[20] sono racchiuse nella Tab. 2.3.

Centralina cancello

Il varco automatico utilizzato per le sperimentazioni è dotato di una centralina modello *Label B02*, visibile nell'immagine di Fig.2.22.

Questa centralina non è altro che un sistema embedded che ha lo scopo di gestire tutte le funzioni che il cancello ha o potrebbe avere; in particolare si occupa della gestione dei motori, dell'acquisizione dei comandi di apertura, di chiusura e della rilevazione di ostacoli tramite le fotocellule.
La centralina ha due modalità di funzionamento principali: normale e apprendimento; la fase di apprendimento è generalmente riservata agli installatori e comprende tutti gli strumenti necessari a calibrare la centralina una volta installata su un cancello, mentre la modalità normale riguarda l'utilizzo quotidiano del varco: nel corso di questa tesi ci si è occupati solamente della modalitàă classica in quanto, essendo il varco giàă installato, il sistema era giàă tarato.

Specifiche Tecniche Arduino	
Dato	**Valore**
Controller	ATmega168
Operting Voltage	5 V
Input Voltage	7-12 V
Input Voltage Limit	6-20 V
Digital I/O Pins	15
Analog Input Pins	6
DC Current I/O Pins	40 ma
DC Current 3.3 V Pin	50 ma
Flash Memory	16 Kb
SRAM	1 Kb
EEPROM	512 bytes
Clock Speed	16

Tabella 2.3: Specifiche tecniche del dispositivo Arduinio.

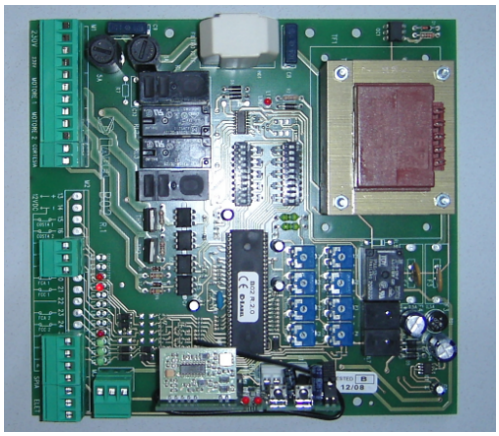


Figura 2.22: Centralina utilizzata per i test.

Nelle Fig.2.22 è presentato il layout della centralina utilizzata: come si può vedere sono presenti principalmente due morsettiere, una di bassa tensione e una a 220 V, cioè una che si occupa di segnali logici o di comando e l'altra che si occupa dell'alimentazione degli attuatori. La morsettiera d'interesse è nel nostro caso quella a bassa tensione, nella Tab. 2.4 sono riportati le funzioni di tutti i morsetti presenti.

Specifiche Tecniche Centralina Automazione Cancello

Morsetto	Collegamento	Morsetto	Collegamento
1	F.C. Open Mot. 1	11	Fotocellula 2
2	F.C. Close Mot. 1	12	Stop
3	F.C. Open Mot. 2	13	Comune 12 Vdc
4	F.C. Close Mot. 2	14	Apre uomo presente
5	Ingresso Costa	15	Chiude uomo presente
6	Comune 12 Vdc	16	Uscita 24 Vac
7	Start	17	Uscita Elettroserratura
8	Start Pendonale	18	Comune 12/24 Vac
9	Comune 12 Vdc	19	Uscita 24 Vac
10	Fotocellula 1		

Tabella 2.4: Schema di collegamento centralina utilizzata per i test.

Tra i vari morsetti presenti si è scelto di utilizzare :

- **10-13**, **Contatto Fotocellula**: In questo morsetto si è inserito il dato in uscita dal sensore di sicurezza creato in sostituzione alla fotocellula prima presente;
- **14-13**, **Apre uomo presente**: Al fine di ottenere un comando univoco di apertura si è collegato un normale interruttore a bottone per comandare l'apertura del cancello; il segnale viene inviato, oltre che alla centralina, al sensore di sicurezza tramite l'apposita scheda di comunicazione.
- **15-13**, **Chiude uomo presente**: Come sopra, per ottenere un comando univo di chiusura si è inserito un interruttore fra questi due morsetti, il segnale vie-

ne inviato, oltre che alla centralina, al sensore di sicurezza tramite l'apposita scheda di comunicazione.

2.6 Test Sperimentali

Al fine di valutare le prestazioni qualitative dell'algoritmo si è deciso di testare l'applicazione creata su un sistema reale acquisendo, tramite le telecamere viste in precedenza, una serie di video da elaborare poi *offline*.

2.6.1 Prima fase di test

Nella prima fase di test si è utilizzato il setup hardware riportato in Tab. 2.5.

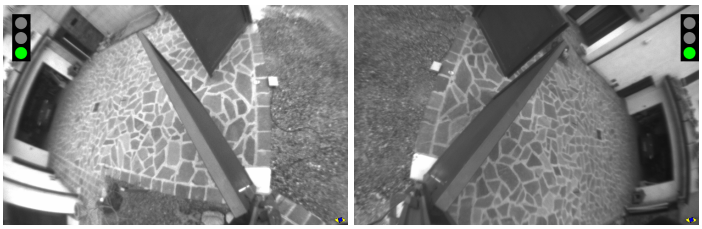
Setup installazione [mm]			
Dato	**x**	**y**	**z**
Dimensioni battente	1550	1460	80
Posizionamento telecamera destra	-80	-240	2100
Posizionamento telecamera sinistra	3350	-260	2100
Dimensioni area di visione	4.25	3.25	nd

Tabella 2.5: Parametri setup hardware, prima fase di test.

Tutti i dati riportati in Tab. 2.5 sono riferiti all'estremo basso destro dell'area di rilevazione e sono riportati in metri; durante il test inoltre sono state utilizzate le telecamere della serie *FireFly MV* con ottiche di tipo *fish-eye* giàă descritte nel nella sezione introduttiva di questo capitolo.
Per poter valutare in modo qualitativo il funzionamento del sistema sono stati eseguiti i seguenti test:

- **Apertura senza ostacoli:** sistema inizialmente in quiete a cui viene indirizzato un comando di apertura senza nessun ostacolo sulla scena;
- **Chiusura senza ostacoli:** sistema inizialmente totalmente aperto a cui viene indirizzato un comando di chiusura senza nessun ostacolo sulla scena;

- **Presenza pedone:** sistema inizialmente in quiete a cui viene indirizzato un comando di apertura con un pedone inizialmente in movimento nella scena, successivamente con pedone fermo in tutte le varie aree di rilevazione viste in precedenza;
- **Ostacoli fissi:** presenza sulla scena di ostacoli fissi rappresentati da scatole di varie dimensioni, cancello fermo ad inizio scena e successivamente in movimento;
- **Presenza auto:** cancello inizialmente in quiete, dopo un movimento di apertura si assiste al passaggio di un auto in ingresso;



(a) (b)

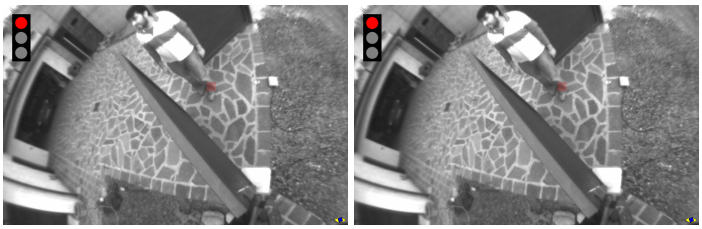
Figura 2.23: Apertura senza ostacoli, telecamera sinistra *a* e telecamera destra *b*.

Per quanto riguarda i primi due test hanno dato entrambi risultati positivi, con il sensore creato che non disturba la normale attivitàă del varco in assenza di ostacoli sulla scena; in particolare, come visibile in Fig.2.23, esso si dimostra robusto rispetto all'individuazione di falsi positivi anche in caso di illuminazione diversa fra telecamera destra e telecamera sinistra. Nella stessa immagine si può vedere un semaforo verde stampato sul bordo esterno: esso rappresenta l'output del sensore, cioè l'azione che intraprenderebbe il sensore se fosse posto in sostituzione alle fotocellule. Ovviamente un semaforo verde significa assenza di ostacoli, viceversa , come visibile nelle immagini che saranno riportate in seguito, un semaforo rosso significa ostacolo presente ed è quindi un segnale di stop per il varco automatico. La difficoltà principale incontrata durante questo test è stata rappresentata da una corretta simulazione

del movimento compiuto dal cancello durante la fase di apertura/chiusura: esso si è presentato come molto ondulato e quindi di difficile discretizzazione.
Il test successivo ha riguardato lo stesso movimento di apertura/chiusura visto sino ad ora ma con l'aggiunta di un pedone nell'area di collisione; per rendere più interessante il test si è sistemato il pedone in due posizioni diverse:

- Pedone fermo o in movimento nelle regioni *Area1* e *Area2*
- Pedone fermo nelle regioni *Area3* e *Area4*

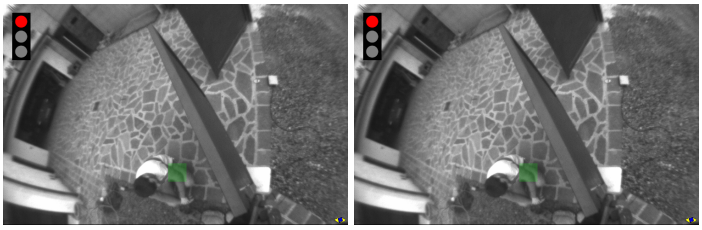
Il primo dei due casi è riportato nelle immagini di Fig.2.24. Nelle immagini riportate



(a) (b)

Figura 2.24: Pedone presente, rilevazione corretta *a* e errore nella stima della posizione *b*.

in Fig.2.24 si può vedere come la rilevazione della presenza di un ostacolo sia corretta, con semaforo rosso accesso in entrambe. In queste immagini viene stampato a video anche la posizione di contatto a terra dell'oggetto rilevato: si può vedere come nell'immagine di sinistra questa stima sia sostanzialmente corretta mentre nell'immagine di destra il punto di contatto risulta essere spostato verso il battente destro del varco. Questo avviene poiché la simulazione del movimento del cancello non è perfetta e di conseguenza il sistema associa il rumore generato dalla presenza del battente ad un estensione dell'ostacolo presente, portando quindi ad un errore nella stima del punto di contatto. Questa caratteristica non risulta problematica poiché, come visto nel test precedente, in caso di assenza di ostacolo il movimento del battente non risulta in falsi positivi.
In Fig.2.25 sono riportate le immagini provenienti dalla seconda parte del test.



(a) (b)

Figura 2.25: Pedone in *Area3*, telecamera sinistra (a) e telecamera destra (b).

L'ostacolo in questo caso è un pedone situato nella zona denominata *Area3* , cioè nella regione in cui viene applicato un algoritmo di *background subtraction*. L'ostacolo viene rilevato correttamente come dimostrato dal semaforo rosso acceso, inoltre viene stampato a video la posizione del baricentro dell'oggetto stesso. Da notare come la telecamera di destra, raffigurata nella Fig.2.25.b, abbia la visuale completamente occlusa dal battente e come quindi la rilevazione dell'ostacolo sia in questo caso affidata alla sola telecamera di sinistra, la cui visuale è contrassegnata in Fig.2.25.b.

(a) (b)

Figura 2.26: Presenza ostacoli fissi, rilevazione in *Area1 a* e in *Area2 a*.

Nel test seguente , come visibile in Fig.2.26, si sono poste delle scatole di dimensioni ridotte nelle aree individuate come *Area1* e *Area2* nell'immagine di Fig.2.3.
Il test ha dato esito positivo durante tutta l'esecuzione, come dimostrato dal semaforo rosso accesso in entrambe le immagini di Fig.2.26. Dalle stesse immagini emergono altre due situazioni di rilievo, nella Fig.2.26.a viene rilevato un solo ostacolo, mentre

nella Fig.2.26.b viene rilevata solamente la scatola situata in posizione più arretrata delle due. Nel primo caso il problema è dovuto alle politiche di *merging* viste nella sezione precedente, gli oggetti infatti sono molto vicini e di conseguenza vengono accumulate durante l'analisi degli ostacoli. La seconda situazione è invece dovuta alle politiche di apertura del cancello: la prima scatola viene a trovarsi nella regione *Area1* che durante il movimento di apertura del varco non viene considerata come area di pericolo e di conseguenza la scatola presente in quell'area viene ignorata.

Nell'ultimo dei test qui esposti si è provato un tipico caso d'uso: l'arrivo di un automobile in entrata con cancello inizialmente chiuso. Anche in questo caso, come visibile nelle immagini di Fig.2.27 , il passaggio dell'ostacolo viene rilevato correttamente dal sensore di sicurezza come dimostrato dal semaforo rosso presente nella Fig.2.27.a; nell'immagine (b) si è voluto dimostrare come l'automobile non venga rilevata come ostacolo una volta uscita dall'area di pericolo; nella Fig.2.27.b essa ha oltrepassato quest'area e viene quindi stampato a video un semaforo verde.

(a) (b)

Figura 2.27: Presenza automobile, oggetto rilevato durante il passaggio *a* e successivamente ignorato *b*.

2.6.2 Seconda fase di test

Durante la seconda fase di test è stata cambiata la locazione con conseguente cambio di cancello. I dati di *setup* di questa seconda fase di test sono riportati nella Tab. 2.6

Tutti i dati riportati sono stati rilevati prendendo come centro del sistema di riferimento l'estremo basso destro dell'area di rilevazione;

Setup installazione [mm]			
Dato	**x**	**y**	**z**
Dimensioni battente	1.55	1.46	0.06
Posizionamento telecamera destra	-0.12	-0.35	2.30
Posizionamento telecamera sinistra	3.12	-0.38	2.30
Dimensioni area di visione	4.0	3.0	nd

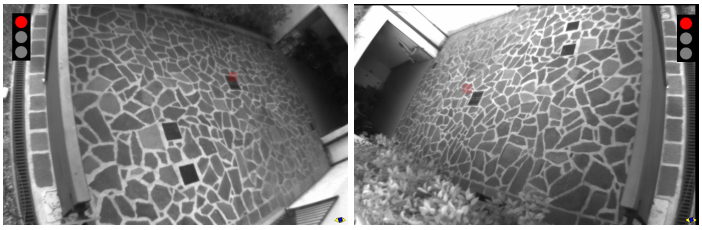
Tabella 2.6: Parametri setup hardware, seconda fase di test.

Per quanto riguarda le scena provate si sono realizzati tutti i test visti in precedenza con aggiunta di alcune situazioni come:

- **Presenza motocicletta:** situazione tipica in cui un soggetto a bordo di una motocicletta cerca di uscire ed entrare dal varco;
- **Presenza bicicletta:** bicicletta posta come ostacolo nella regione di interesse;
- **Presenza animali:** è stato posto un animale di piccole dimensioni nell'area di interesse al fine di valutare la precisione dell'algoritmo per ostacoli di dimensioni ridotte e in movimento veloce;
- **Presenza di ostacoli multipli:** diverse persone poste all'interno della scena al fine di valutare la capacità dell'algoritmo di disambiguare tra soggetti diversi;

I filmati acquisiti durante la seconda giornata di test si sono rivelati densi di problematiche e di fatto inutilizzabili a causa di un errore compiuto durante la fase di setup. Nello specifico, al fine di risolvere il problema dei forti cambiamenti di luminosità visti nel capitolo 1 , si è deciso di imporre alle telecamere la gestione automatica dello *shutter* e dell'esposizione, senza però accoppiare le telecamere in modo che usassero entrambe gli stessi valori.

A causa delle forti diversità di *background* fra le due telecamere esse hanno utilizzato parametri di *shutter* e esposizione diversi con i risultati visibili in Fig.2.28. Dalle immagini si nota una forte differenza di luminositàă fra l'immagine sinistra,



(a) (b)

Figura 2.28: Seconda fase di test, diversa luminosità fra immagine sinistra (a) e immagine destra (b).

rappresentata in (a), e l'immagine destra, rappresentata in (b) ; questa differenza porta il sistema ad individuare lo sfondo come un ostacolo, creando così un falso positivo permanente all'interno del sistema, come dimostrato dal semaforo rosso rappresentato nelle immagini di Fig.2.28.

2.6.3 Prestazioni

Inizialmente si analizzano le prestazioni dell'algoritmo in termini di tempo computazionale richiesto, i dati rilevati sono riassunti nella Tab. 2.7. I dati elencati in Tab. 2.7 sono riferiti all'esecuzione dell'algoritmo su un pc con le caratteristiche riportate in Tab. 2.8.
Sulla piattaforma usata per i test l'algoritmo si è quindi dimostrato in grado di operare in real-time ad una frequenza di acquisizione di 10 Hz.
Per quanto riguarda le prestazioni dell'algoritmo sviluppato non vengono presentati dati statistici poichè, a causa del numero limitato di prove che si sono effettuate, questi risulterebbero sicuramente incorretti; per contro si vuole invece fornire una valutazione qualitativa dell'algoritmo nel capitolo *Conclusioni*.

Tempi di esecuzione [msec]			
Dato	**Min**	**Max**	**Avg**
Main Loop	153.4	269.6	177.6
Applicazione	90.1	157.2	102.0

Tabella 2.7: Sintesi prestazioni computazionali.

Specifiche Elaboratore			
Dato	**CPU [Hz]**	**RAM [Gb]**	**HD [rpm]**
ASUS W5A	Intel Centrino 1.7	1 DDR2	5400

Tabella 2.8: Pc utilizzato per i test.

Capitolo 3

Sistema di percezione con elevata automazione

L'automazione dei processi produttivi industriali ricopre un ruolo fondamentale nello sviluppo economico di un'azienda. Questo settore racchiude sia tecnologie meccatroniche che informatiche sviluppate nelle ultime decine di anni. Oggi, i processi produttivi industriali usano in larga scala diverse tipologie di sensori fra i quali quelli di visione artificiale sia per funzioni di controllo qualità sia per manipolazione di oggetti. Il progetto del quale si vanno a spiegare le attività svolte nel corso di Dottorato, si spinge ulteriormente in là nelle applicazioni industriali della visione artificiale.

L'obiettivo di questo progetto è la sostituzione delle attuali tecnologie di guida applicate ai veicoli AGV *(Automatic Guided Vehicle)* con un sistema di visione artificiale in grado di sopperire alle attuali difficoltà implementative e strutturali delle odierne tecnologie utilizzate. Questo progetto in collaborazione con una ditta italiana che progetta e produce questi sistemi ha una durata pluriennale e durante il periodo di Dottorato è stata svolta un prima fase di questo ambizioso progetto.

3.1 Analisi del problema

Esistono diverse tecnologie di guida autonoma per i veicoli AGV, comunemente chiamati navette. Le due principali categorie nelle quali si dividono sono:

- **Guida magnetica:** Un cavo elettrico viene affogato nel terreno. Mediante sensori magnetici, la navetta è in gradi di seguire questa pista magnetica e referenziare la sua posizione all'interno dello stabilimento. I principali svantaggi di questo sistema sono la scarsa flessibilità e la difficoltà nel creare le condizioni magneti ottime per il sistema nelle moderne pavimentazioni industriali.

- **Guida laser:** Mediante un laser installato a bordo della navetta in una posizione elevata, e il posizionamento di specchi montati ad una certa quota per tutto il percorso, è possibile ricavare la posizione assoluta all'interno del capannone industriale della navetta seguito della lettura di un numero sufficiente di riflessioni. Comunicando ad un server di gestione della navigazione questa posizione è possibile pianificare la navigazione. I principali vantaggi di questo sistema sono la precisione e la flessibilità nella riconfigurazione dei percorsi. Tuttavia presenta diversi svantaggi come l'impossibilità di percorrere piani inclinati, l'installazione di specchi riflettenti lungo tutto il percorso.

I sistemi guidati mediante tecnologia al laser sono i più evoluti e rappresentano lo stato dell'arte dei dispositivi di guida autonoma per veicoli industriali. In Fig.3.1 è possibile vedere una immagine di quello che sarà il sistema laser che il progetto cercherà di sostituire con quello di visione artificiale.

In particolare, dopo una prima sperimentazione diretta sul campo, le principali problematiche dei sistemi LGV sono:

- Installazione degli specchi riflettenti.

- Incapacità di navigazione *outdoor* a causa di mancanza di specchi riflettenti.

- Incapacità di navigazione su piani inclinati a causa della mancanza di rilevazione degli specchi riflettenti.

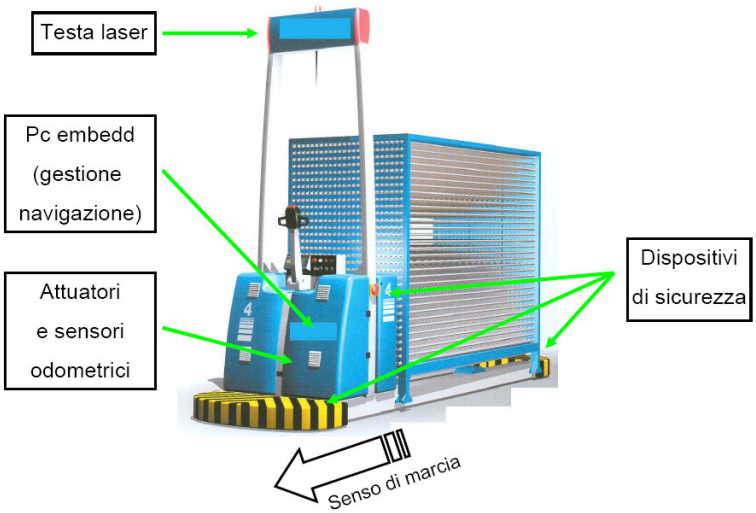


Figura 3.1: Architettura di un moderno LGV *Laser Guided Vehicle*.

- Incapacità di navigazione in ambienti con differenze di temperature a causa della condensa che si forma sul dispositivo laser.
- Limitazione sulle dimensioni di oggetti trasportati dalla navetta a causa delle occlusioni delle rilevazioni laser.

Questi problemi possono essere risolti mediante tecniche di visione artificiale con opportuni accorgimenti architetturali e nuove tecniche di visione artificiale. Nelle sezioni successive si và a spiegare quella che è stata la prima fase del progetto: implementare un sistema in grado di riconoscere ostacoli e trasmetterli al sistema di guida autonoma della navetta.

3.2 Setup Hardware

In questa sezione si và a descrivere quella che è l'architettura proposta per la navigazione autonoma dei veicoli industriali ed alcuni accorgimenti ingegneristici nel campo della visione artificiale.

Nella Fig.3.2 viene mostrato il sistema di controllo e azionamento della navetta e il sistema di visione artificiale. Per quanto concerne il sistema di visione, si è previsto un dispositivo frontale di acquisizione immagini composto da tre telecamere ed un dispositivo posteriore composto da una sola telecamera per guidare il veicolo nelle fasi di carico/scarico laddove è necessaria una elevata precisione. Di seguito ci concentreremo solo sul sistema frontale, sulla elaborazione delle immagini acquisite e sulla trasmissione dei dati ottenuti.

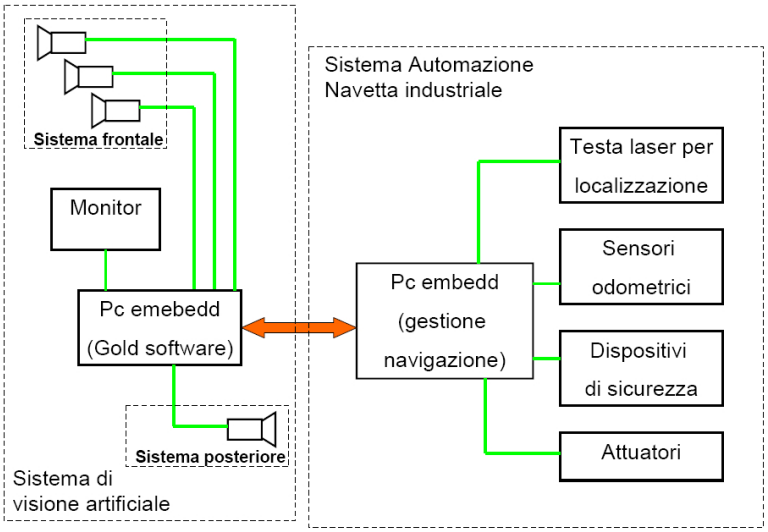


Figura 3.2: Schema a blocchi del sistema.

3.2.1 Architettura del sistema

In Fig.3.3 è possibile vedere dove i dispositivi del sistema di visione artificiale sono stati installati. Il posizionamento delle telecamere frontali e posteriori è determinato dal miglior compromesso fra esigenze costruttive della navetta e miglior inquadratura della scena frontale e posteriore.

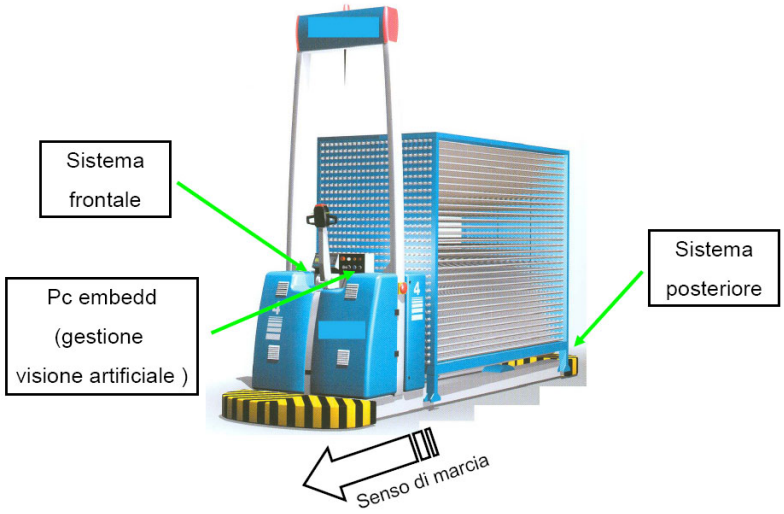


Figura 3.3: Architettura del sistema di visione montato sulla navetta.

La navetta LGV è stata equipaggiata con un PC industriale della azienda AXIOM[1], modello IPC910H. La Tab. 3.1 ne riassume le principali caratteristiche.

Il PC in Fig.3.1.a è stato dotato di una memoria compactflash di dimensione 8GB, montata su scheda PCIexpress. Su questa si è installato un sistema operativo (OS) GNU/Linux. In Fig.3.1.b è possibile osservare una delle telecamere utilizzate.

3.2.2 Realizzazione del sistema stereoscopico

Dopo una prima analisi generale degli obiettivi di progetto e delle caratteristiche da considerare per la realizzazione di un sistema stereoscopico, nella seconda fase di progetto si è passati alla realizzazione pratica e al test delle soluzioni inizialmente previste.

[1]Per maggiori informazioni in merito a questa azienda, fare riferimento alla pagina internet `http://www.axiomtek.it/`

(a) (b)

Figura 3.4: PC Industriale Axiom IPC910H *a* ed telecamera Modello FireFly MV *b*.

Realizzazione di un sistema trinoculare

Per quanto concerne il sistema di visione frontale si è optato per un sistema stereoscopico. I vantaggi di questo sistema rispetto a quello monoscopico sono noti in letteratura. Dopo aver eseguito uno studio teorico delle possibili *baseline* da impiegare nella realizzazione del sistema stereoscopico, si è deciso di testare nella fase sperimentale le seguenti configurazioni:

La scelta di testare queste tre differenti configurazioni deriva dalla necessità di trovare un compromesso per un buon funzionamento dell'algoritmo (come la rilevazione degli ostacoli a terra e in movimento) sia durante la navigazione a bassa che ad alta velocità.

Tuttavia, data la necessità di navigare in zone sia *indoor* che *outdoor*, che presentano le più svariate caratteristiche di larghezza del percorso e velocità di navigazione, si è implementato il sistema trinoculare illustrato in Fig.3.6.a.

In questa fase progettuale, il vantaggio di avere un sistema trinoculare è legato al fatto di aver a disposizione a tempo di esecuzione, tre differenti baseline tra cui scegliere per poi procedere con l'esecuzione dell'algoritmo (che risulta inalterato). visto che è necessario ancora capire quale sia la *baseline* più adatta a soddisfare i vincoli di progetto. Questo tipo di architettura è già stata utilizzata da altri gruppi di ricerca e si trova ben documentata in letteratura.

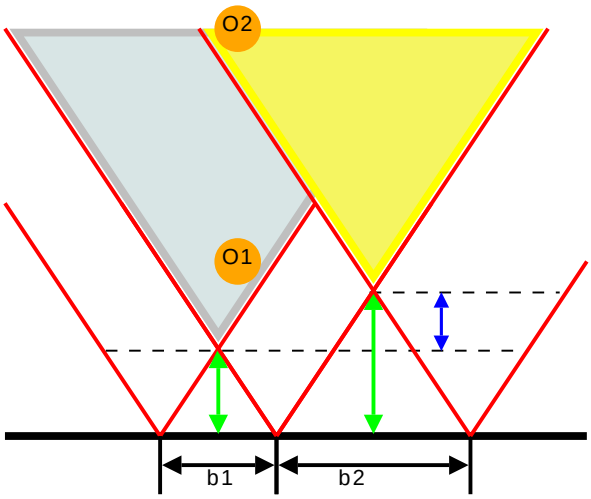


Figura 3.5: Distanze di rilevazione degli ostacoli O1 e O2 nel caso di baseline corta (b1) e lunga (b2); In verde vengono evidenziate le distanze del cono visivo delle rispettive baseline, mentre in blu la differenza tra queste

(a) (b)

Figura 3.6: BarraTrinoculare *a* installata sulla navetta ed uno degli scenari di test *b*.

Caratteristiche PC IPC910H	
Nome	**Valore**
Processore	Intel Core 2 Duo CPU 2,33GHz a 64bit
Memoria	$2 \times$ 2GB RAM, tipo DDR2
Porte USB	6×2.0
Porta VGA	1
Porte COM	4
FireWire	3 porte FireWire a 6pin
Ethernet	$2 \times$ 10BaseT/100BaseTX/1000BaseTX
Alimentazione	DC 18V - 24V

Tabella 3.1: Principali caratteristiche del PC montato nel vano motore

3.2.3 Fase di acquisizione immagine

La fase di acquisizione delle immagini è un punto estremamente critico per i sistemi avanzati di visione artificiale. Si consideri inoltre, che nello scenario industriale, si possono configurare situazioni di forte luminosità sia diretta che riflessa e cambiamenti repentini di illuminazione naturale ed artificiale. Per tali motivi segue una dissertazione delle principali tecniche di regolazione dei parametri caratteristici di acquisizione delle immagini delle telecamere selezionate.

La funzionalità 12-to-10 bit Companding

Questa funzionalità realizza una compressione non lineare dei 12 bit dell'immagine acquisita in una immagine a 10bit; a questo proposito si consiglia di fare riferimento alle figure 3.7 e 3.8.

Per abilitare questa modalità è necessario impostare il registro **ADC Resolution control (0x1C)** a 0x3 (I valori ammissibili sono in decimale 2 e 3; 2 è il valore di default).

FireFly MV Camera	
Nome	**Valore**
Modello	FireFly MV-03MTC
Firmware	0.9.2.12
Tipo	CMOS, RoHS 0.3MP Color, 1/3”
Risoluzione	752×480
Sensore	$6\mu\text{m} \times 6\mu\text{m}$

Tabella 3.2: Principali caratteristiche del sensore video impiegato per la realizzazione del sistema.

Test effettuati		
Baseline	**D**$_{max}$	**Precisione**
200	$<$ 6000	300
350	7000 - 8000	300
500	9200	300

Tabella 3.3: Baseline testate per la scelta della configurazione HW della testa stereo; i dati si intendono espressi in *mm*

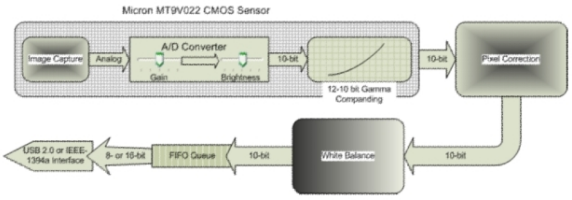


Figura 3.7: Flusso di dati nella telecamera Firefly MV.

La funzionalità HDR

Come indicato dal *Technical Application Note TAN2008002* (High Dynamic Range Modes Available in Firefly MV-Revised May 30, 2008), esistono due metodi

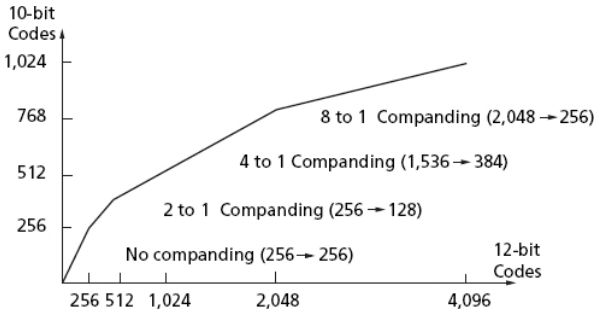


Figura 3.8: Schema di compressione della funzionalità 12-to-10 bit Companding.

principali per abilitare la modalità HDR:

- 12-to-10 bit Companding
- Controllo manuale dell'HDR

Il primo metodo risulta completamente automatico una volta abilitato, mentre il secondo permette di avere un controllo maggiore della telecamera, andando a modificare un certo numero di parametri.

Leggere e scrivere i registri del sensore

Visto che per i sistemi operativi GNU/Linux la Point Grey Research non ha messo a disposizione alcuna API per la gestione delle funzionalità della telecamera, è necessario fare uso delle **libdc1394** (passando dalla versione 1 alla versione 2 di queste librerie, alcune funzioni per il controllo dei registri della telecamera sono leggermente modificate, ma non di molto). Il principio di funzionamento per leggere e scrivere i registri del sensore video tramite alcuni registri della telecamera è il seguente:

1. Si scrive l'indirizzo del registro del sensore che si vuole leggere, all'interno del registro della telecamera **0x1A00**[2]

[2]Gli indirizzi dei registri, così come i valori in essi contenuti, vengono espressi in esadecimale.

2. Per leggere o scrivere il registro indicato al passo precedente, bisogna rispettivamente leggere o scrivere il registro **0x1A04**

Controllo manuale dell'HDR

Per ottenere un controllo manuale della dinamica della telecamera, è possibile modificare la risposta del sensore video variando i voltaggi d'integrazione della sorgente luminosa, durante il periodo di esposizione del sensore. In Fig.3.9 viene mostrata una sequenza di controllo del livello di saturazione dei pixel del sensore, mentre in Fig.3.10 ne viene mostrata la corrispettiva risposta in cui i punti di rottura (*knee points*), vengono univocamente determinati dalla combinazione dei tempi di integrazione e dai rispettivi voltaggi.

Per abilitare l'HDR manuale, è necessario impostare a 1 il bit 6 del registro **Pixel Operation Mode (0x0F)** (partendo a contare i bit da 0 e quindi il 7^o bit nel complessivo); è altresì necessario inserire dei valori corretti per i registri degli shutter se **NON** si vuole sfruttare il calcolo automatico dei *knee point*.

Secondo quanto scritto sul datasheet della Micron, l'intensità della luce ha un andamento confrontabile col reciproco dei tempi di esposizione parziali t_1, t_2 e t_3, dove t_1 è il tempo di esposizione più lungo, t_2 appena appena più piccolo e così via per t_3; in questo modo l'intervallo di valori delle lunghezze d'onda molto basse gode di una sensibilità maggiore rispetto all'intervallo di valori delle lunghezze d'onda maggiori (vedi Fig.3.10).

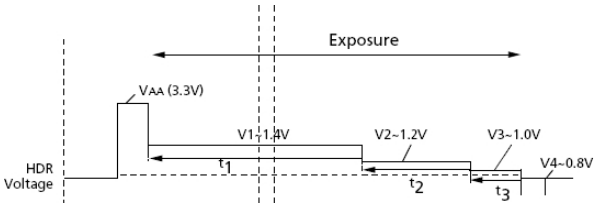


Figura 3.9: Esempio di una sequenza di applicazione dei voltaggi di integrazione della funzionalità manual HDR.

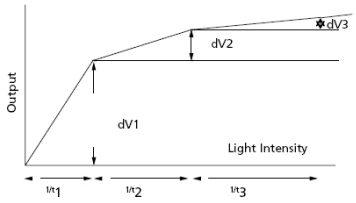


Figura 3.10: Risposta del sensore video in modalità HDR manuale, in funzione della lunghezza d'onda.

È possibile specificare il calcolo dei *knee point* o in modalità manuale (comportamento predefinito) o in modalità automatica.

Calcolo Automatico dei Knee Point

Per abilitare il calcolo automatico dei *knee point* è necessario impostare ad 1 il bit 8 del registro **Shutter Width Control (0x0A)** (partendo a contare i bit da 0 e quindi il 9^o bit nel complessivo).

Nel calcolo automatico dei *knee point* i tempi di esposizione vengono calcolati dalla logica interna al sensore nel seguente modo:

$$t_1 = t_{INT} - t_2 - t_3 \tag{3.1}$$

$$t_2 = t_{INT} - \left(\frac{1}{2}\right)^{R0x0A,bits3:0} \tag{3.2}$$

$$t_3 = t_{INT} - \left(\frac{1}{2}\right)^{R0x0A,bits7:4} \tag{3.3}$$

dove t_{INT} rappresenta lo shutter totale della telecamera, specificato all'interno del registro **Total Shutter Width (0x0B)** e che viene determinato dall'unità AEC (Auto Exposure Control). Quando la funzionalità AEC viene abilitata, allora anche il calcolo automatico dei *knee point* **deve** essere abilitato. I valori predefiniti per i tempi di integrazione sono per t_2 $\left(\frac{1}{16}\right)$ di t_{INT} e per t_3 $\left(\frac{1}{64}\right)$ di t_{INT}.

Calcolo Manuale dei Knee Point

Quando invece il calcolo dei *knee point* viene effettuato manualmente (come predefinito), allora devono essere programmati i registri degli shutter[3] e va specificato il tempo di integrazione massimo all'interno del registro **Total Shutter Width (0x0B)**:

$$t_1 = R0x08, bits14:0 \tag{3.4}$$

$$t_2 = (R0x09, bits14:0) - (R0x08, bits14:0) \tag{3.5}$$

$$t_3 = t_{INT} - t_1 - t_2 \tag{3.6}$$

3.2.4 Controllo dell'esposizione della telecamera

Dai test effettuati sulla dinamica della telecamera, si è potuto constatare come questa, in associazione con una esposizione statica del sensore video, non basti a garantire sempre l'acquisizione di immagini contenenti sufficiente tessitura, utile all'algoritmo di visione artificiale. Per questo motivo si è pensato di prendere in considerazione l'utilizzo di un algoritmo che permetta di variare dinamicamente l'esposizione del sensore video, agendo sui parametri *Shutter* e *Gain*. Le possibili soluzioni a nostra disposizione sono:

- Utilizzo del plugin AutoExposure di GOLD che, rispetto all'AutoExposure proprietario della telecamera, vedi 3.2.4, garantisce il sincronismo di *Gain* e *Shutter* per entrambe le telecamere controllate dall'algoritmo.
- Utilizzo della funzionalità AutoExposure implementata dal *firmaware* della telecamera.
- Realizzazione di un secondo plugin di AutoExposure in GOLD, simile a quello natio, ma che sfrutta l'algoritmo di esposizione implementato nella Firefly MV.

[3] I registri degli shutter sono **Shutter Width 1 (0x08)** e **Shutter Width 2 (0x09)**.

Algoritmo di AutoExposure in GOLD

La gestione dell'esposizione del sensore video è sostanzialmente un problema di controlli automatici, risolvibile mediante la progettazione di un controllore PID *Proporzionale-Integrativo-Derivativo*[4]), come quello mostrato in Fig.3.11.

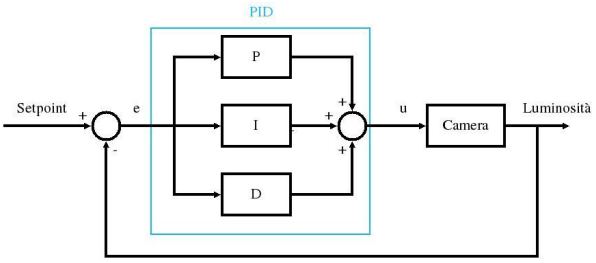


Figura 3.11: Controllore Proporzionale Integrale Derivativo.

Il PID regola l'uscita in base a:

- Il valore del segnale d'errore (tramite l'azione del regolatore *Proporzionale P*)
- I valori passati del segnale d'errore (tramite l'azione dell'*Integratore I*)
- Quanto *veloce* vogliamo che sia il nostro controllore a raggiungere il segnale di setpoint (tramite l'azione del *Derivatore D*)

I controllori PID sono di immediata applicazione e facilmente configurabili; è per questo che si prestano bene per un loro uso in ambito industriale, salvo però presentare alcune forti limitazioni: è necessario ricalibrare il controllore ogni volta che vi siano dei cambiamenti nei parametri del processo; possono innescarsi delle forti escursioni della variabile controllata (effetto *Windup*) a causa dell'azione integrale; non sono adatti per controllare sistemi multivariabili.

Nel caso del framework GOLD, utilizzato per questo progetto, è stato realizzato uno specifico plugin di AutoExposure, basato sulle regole di taratura di *Ziegler-Nichols*. Queste regole permettono di estrapolare i coefficienti dei regolatori P,I,D,

[4]Una breve spiegazione del loro principio di funzionamento è reperibile su `http://it.wikipedia.org/wiki/Controllori_PID`.

Costanti regolatore PID			
Tipo	$\mathbf{K}_P$	$\mathbf{K}_I$	$\mathbf{K}_D$
P	$0.5 \times \mathrm{K}_c$	-	-
PI	$0.45 \times \mathrm{K}_c$	$\frac{T_c}{1.2}$	-
PID	$0.6 \times \mathrm{K}_c$	$\frac{T_c}{2}$	$\frac{T_c}{8}$

Tabella 3.4: Stima delle costanti del regolatore PID.

senza la determinazione di un modello matematico del processo. Ciò che bisogna determinare sono la *costante critica* K_c e la *pulsazione critica* T_c, mediante delle misure sperimentali. Qui di seguito vengono esposte le regole per la rilevazione di queste due costanti.

Regole di Ziegler-Nichols

1. Il processo viene fatto controllare da un controllore esclusivamente proporzionale (si settano a zero i valori di K_I e K_D)

2. Il guadagno K del regolatore proporzionale viene gradualmente aumentato fintanto che non si innescano delle oscillazioni sostenute[5] della variabile controllata; definiamo tale valore di K come K_c, il valore della *costante critica*

3. Ora sarà possibile misurare anche il valore della *pulsazione critica* T_c (generalmente compreso entro i valori di 0.7 – 0.8 s)

4. Sulla base dei valori determinati, si considera la Tab. 3.4 per la determinazione delle costanti Proporzionali, Integrali e Derivative

[5] È necessario fare molta attenzione che tali oscillazioni non si esauriscano nel breve termine, altrimenti la rilevazione di K_c e T_c porta alla configurazione di un PID potenzialmente instabile, che può ancora generare delle oscillazioni sulla variabile controllata.

In sostanza è possibile realizzare un regolatore PID tarato secondo questo metodo, solo che aumentando troppo il valore di K_c, si velocizza si la risposta del controllore, ma è altrettanto probabile che si inneschino dei fenomeni ondulatori come sfarfallio delle immagini a causa dell'utilizzo di una costante critica inadatta per l'ambiente ripreso; è per questo che si preferisce utilizzare valori di K_c molto bassi in modo tale da poter garantire il corretto funzionamento del controllore per il numero maggiore di situazioni probabili, ma a discapito di una minore velocità di inseguimento del segnale di setpoint. Ad esempio un PID caratterizzato da valori di (K_c,T_c) pari a (0.05 , 0.7), nel caso peggiore considerando un gradino come segnale d'ingresso al sistema telecamera (un semplice esempio di funzionamento è ottenibile coprendo la telecamera con una mano ed inquadrando una una zona molto luminosa; poi togliendo la mano e misurando quanto tempo passa prima che l'immagine si stabilizzi, ovvero non presenti oscillazioni sostenute della luminosità), può impiegare fino a 5 - 6 s prima che si stabilizzi, mentre con una configurazione molto spinta del controllore (K_c molto alto) di (2.35 , 1), la transizione stabile risulta essere immediata, oppure può innescare un fenomeno ondulatorio la cui durata non è nota a priori.

PID a parametri variabili (Gain Scheduling)

Come già accennato nella sezione 3.2.4, la taratura del PID viene effettuata in base a una corretta modellazione del sistema da controllare, oppure seguendo regole empiriche di controlli automatici, rispetto ad una determinata condizione operativa (es. nel nostro caso, il livello di luminosità della scena ripresa). Al variare di quest'ultima, mentre i parametri della funzione di trasferimento del sistema da controllare cambiano valore, quelli dell'algoritmo di regolazione rimangono sempre invariati, causando così un degrado delle prestazioni, o peggio ancora, un errato controllo del sistema che può dare origine a fenomeni non desiderati (es. le oscillazioni di luminosità nelle immagini acquisite dalla telecamera). L'idea quindi che nasce è quella di cercare di rilevare questi mutamenti e riadattare dinamicamente i parametri del PID, come schematizzato in Fig.3.12.

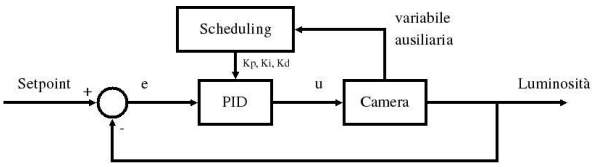


Figura 3.12: PID a parametri variabili, noto anche come *Gain Scheduling*.

A questo punto basterebbe calcolarsi un numero finito di configurazioni del nostro PID, spostando tutto il problema sulla determinazione della condizione di luminosità dello scenario inquadrato tale per cui si renda necessario un cambio di configurazione e sopratutto, verso quale direzione (nel caso si volessero impiegare tre o più PID).

Non avendo però a disposizione un esposimetro (basterebbe anche l'utilizzo di una telecamera per la lettura della luce, a patto che questa goda di una dinamica tale da non fare saturare il sensore video, cosa che nel nostro caso non è possibile) o il modello matematico del sensore video in uso dalla telecamera, abbiamo provato a basarci sull'andamento della varianza della derivata della luminosità media dell'immagine, calcolata sugli ultimi dieci frame.

Purtroppo una soluzione di questo tipo non risulta efficace, anche nel caso di due solo configurazioni (Fast PID e Slow PID), in quanto la sola informazione della varianza della derivata della luminosità media non risulta sufficiente a capire il tipo di ambiente in cui ci si trova (se poco o tanto luminoso). Per questo motivo la soluzione finale adottata per il progetto è quella seguente.

Algoritmo di AutoExposure proprietario della telecamera

La funzionalità AutoExposure proprietaria della telecamera rappresenta una soluzione vantaggiosa per risolvere il problema delle variazioni di luce: in questo modo si elimina il problema della progettazione di un controllore PID illustrato in , in quanto si andarà ad utilizzare l'unità di controllo riportata in Fig.3.8, con la differenza rispetto al plugin di GOLD, vedi 3.2.4, di non poter sincronizzare lo *shutter* e il *gain* tra le telecamere. Questo problema però è stato risolto medicando opportunamente i se-

Sensore Video MT9V022			
Reg 0x	**Descrizione**	**Valori Legali**	**Default 0x**
36	Valore Massimo del Gain	1-63	40
AF	AEC/AGC abilitato/i	0,3	3
BA	Valore in uscita dall'unità AGC	1-63	RO
BB	Valore in uscita dall'unità AEC	1-2047	RO
BD	Valore Massimo dello Shutter	1-2047	1E0

Tabella 3.5: Elenco dei principali registri del sensore video MT9V022, di interesse per la funzionalità AutoExposure

gnali inviati alle telecamere per la loro sincronizzazione come descritto nel paragrafo successivo.

Algoritmo di AutoExposure proprietario della telecamera modificato

Per poter utilizzare la funzionalità AutoExposure del sensore video, superando i vincoli della sincronizzazione, sia temporale (medesimo istante di acquisizione delle immagini), sia hardware (stesso valore di *shutter* e *gain* per le telecamere), è stata apportata una modifica ai driver DCam2 di GOLD per poter scavalcare i compiti del firmware della telecamera, andando a modificare il valore di alcuni registri del sensore video (vedi Tab. 3.5).

Per poter abilitare la funzionalità AutoExposure, è necessario scrivere il valore 0x3 nel registro 0xAF (ciò permette di abilitare le unità AGC e AEC) e successivamente limitare i valori massimi di gain e shutter, rispettivamente scrivendo 0x3F (convertito in decimale rappresenta 63 in valore relativo) e 0xC8 (convertito in decimale rappresenta 200 in valore relativo)[6] nei registri 0x36, 0xBD; in questo modo garantiamo che la durata temporale dello shutter della telecamera, non ecceda il li-

[6]Per ulteriori informazioni in merito a questi valori, si consiglia di fare riferimento al datasheet del sensore video MT9V022.

mite di 100ms imposto dall'acquisizione delle immagini a 10Hz (il guadagno viene invece settato al valore massimo disponibile, ma si può abbassare[7]).

Il sincronismo tra lo *shutter* e il *gain* delle telecamere viene invece ottenuto grazie all'implementazione di un plugin in GOLD, il cui compito è quello di andare a leggere i valori di *gain* e *shutter* dalla telecamera Master e di impostare tali valori all'interno delle telecamere Slave; il campionamento viene effettuato con l'arrivo di un nuovo segnale di trigger, mentre durante la fase di inizializzazione delle telecamere, viene stabilita qual'è la telecamera di riferimento per la lettura dei valori da impostare nelle restanti telecamere.

Sincronizzazione delle telecamere

Volendo individuare gli ostacoli tramite tecniche di elaborazione steroscopica, è necessario assicurarsi che vi sia sincronismo nell'acquisizione dei fotogrammi; in questo modo, dopo la rimozione della distorsione introdotta dalle ottiche e la rettificazione delle immagini, si può procedere con la ricerca dei punti omologhi orizzontali, al fine di ottenere una immagine di disparità con le corrispettive distanze del mondo reale corrette.

Le telecamere del dispositivo trinoculare vengono sincronizzate mediante la ricezione di un segnale di trigger.

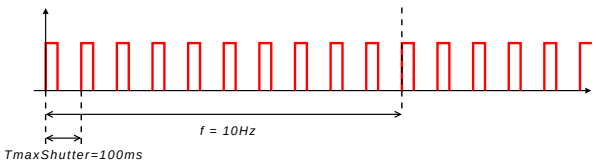


Figura 3.13: Esempio di trigger a 10Hz

[7] Alzare il guadagno massimo del AutoExposure significa poter utilizzare un numero di configurazioni (*gain*, *shutter*) maggiore e avere la possibilità di amplificare il più possibile il segnale acquisito quando siamo in presenza di luoghi poco luminosi, anche se ciò potrebbe causare un aumento del rumore nell'immagine.

La frequenza di campionamento utilizzata attualmente per lo sviluppo degli algoritmi di ricerca ostacoli è pari a 10Hz, il che significa che lo *shutter* della telecamera deve assumere valori inferiori a 100ms per evitare che venga perso il successivo segnale di campionamento.

Le soluzioni per la generazione del segnale di sincronismo sono principalmente due:

- Impiego di un generatore di forme d'onda esterno (es. Arduino [20] o dispositivi similari)
- Impiego della porta seriale RS232 per la generazione del trigger mediante framework GOLD

La soluzione che è stata adottata è la seconda, permettendo così una gestione più centralizzata dell'intero sistema di visione artificiale ed una riduzione dei costi.

3.3 Algoritmo di riconoscimento ostacoli

L'algoritmo di *Obstacle Detection* sviluppato considera come dato di partenza il contenuto informativo della mappa di disparità ricavata con algoritmi di stereoscopia noti in letteratura. Una volta ottenuta questa mappa è possibile ottenere una mappa tridimensionale della scena inquadrata mediante un'operazione di trasformazione IPM *Inverse Prospective Mapping*. [8]

L'algoritmo considera due distinti punti di vista della scena tridimensionale rispetto al sistema di riferimento adottato, e proiettando su opportuni piani d'interesse l'insieme dei punti ottenuti si è in grado di ottenere due differenti profili della scena contenenti delle informazioni che opportunamente elaborate permettono di ricavare una stima del profilo del terreno e a partire da essa localizzare gli eventuali ostacoli presenti.

[8] Trasformazione geometrica che permette di ottenere dalle coordinate immagine di un punto le coordinate mondo del corrispondente punto reale noti i parametri di calibrazione delle telecamere.

I sottosistemi principali che compongono l'algoritmo, e che successivamente vengono descritti in modo dettagliato, sono illustrati in Fig. 3.14.

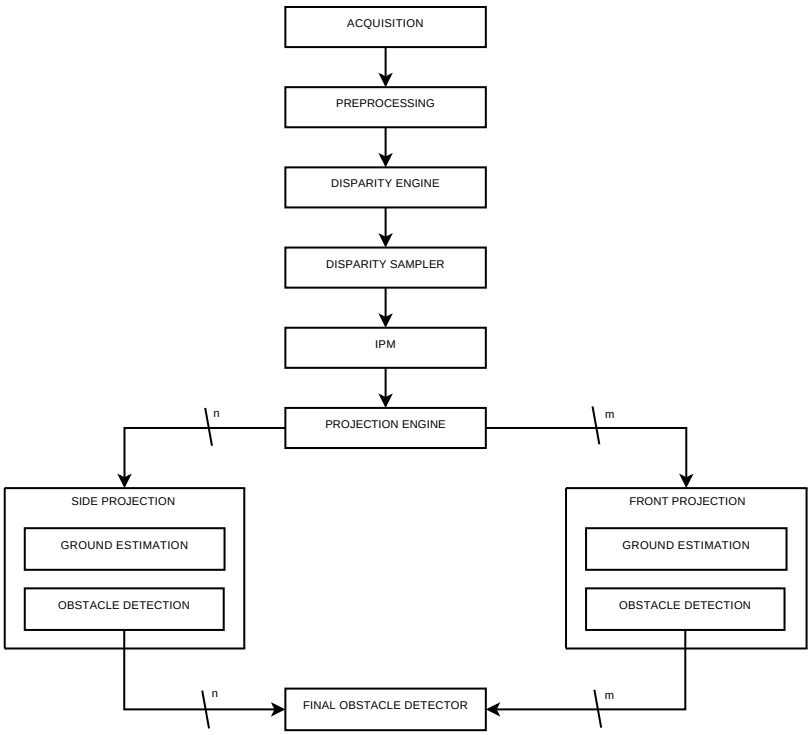


Figura 3.14: Schema a blocchi dell'algoritmo di riconoscimento ostacoli.

La fase iniziale prevede l'acquisizione delle immagini provenienti dal sistema stereoscopico, una successiva operazione di rettificazione, ed il calcolo della mappa di disparità (vedi Fig.3.15.e).

Da un'analisi delle mappe di disparità calcolate nelle varie sequenze acquisite, si è notato che tipicamente risultano essere mappe sparse[9] di punti a disparità valida

[9]Mappe sparse stanno ad indicare una percentuale non elevata di punti presenti

ed è stato riscontrato inoltre al loro interno la presenza di rumore, ovvero punti i cui valori di disparità non risultano essere corretti.

Poichè l'approccio si basa sulla ricostruzione tridimensionale dei punti presenti nella mappa di disparità, quest'ultima deve essere necessariamente priva di disturbi o zone dal contenuto informativo errato e contenere una quantità sufficiente di regioni di punti corretti, per cui si è reso necessario rimuovere tali disturbi dall'immagine e far si che risulti sufficientemente densa[10].
La soluzione è stata quella di utilizzare un filtro che eliminasse dall'immagine le zone troppo rumorose, cioè zone ad alta varianza dei valori di disparità, e che allo stesso tempo uniformasse quelle regioni dal corretto contenuto informativo. Queste operazioni sono descritte nel paragrafo 3.3

Una volta ottenuta un'immagine di disparità sufficientemente densa e corretta, le coordinate di ciascun punto vengono convertite da immagine a mondo attraverso un'operazione di *Inverse Perpective Mapping*.

Nelle sequenze di elaborazioni riportate di seguito è possibile vedere le varie fasi dell'algoritmo di ricerca ostacoli:

- *a*,*b* Immagini acquisite dalle telecamere di sinistra e destra
- *c*,*d* Filtro di Sobel applicato alle immagini separatamente
- *e*,*f* Immagini di disparità e relativa trasformazione IPM nell'immagine originale
- *g*,*h* Punti appartenenti ad ostacoli visti dall'alto in coordinate mondo e vista tridimensionale degli ostacoli discretizzata spazialmente.

I punti tridimensionali saranno dunque il punto di partenza della parte principale dell'algoritmo che permette di ottenere una stima dell'occupazione della scena e di localizzare gli eventuali ostacoli.

[10]Per densa non si intende una mappa completamente coperta di punti dalla disparità nota, bensì una mappa con una percentuale comunque significativa di punti

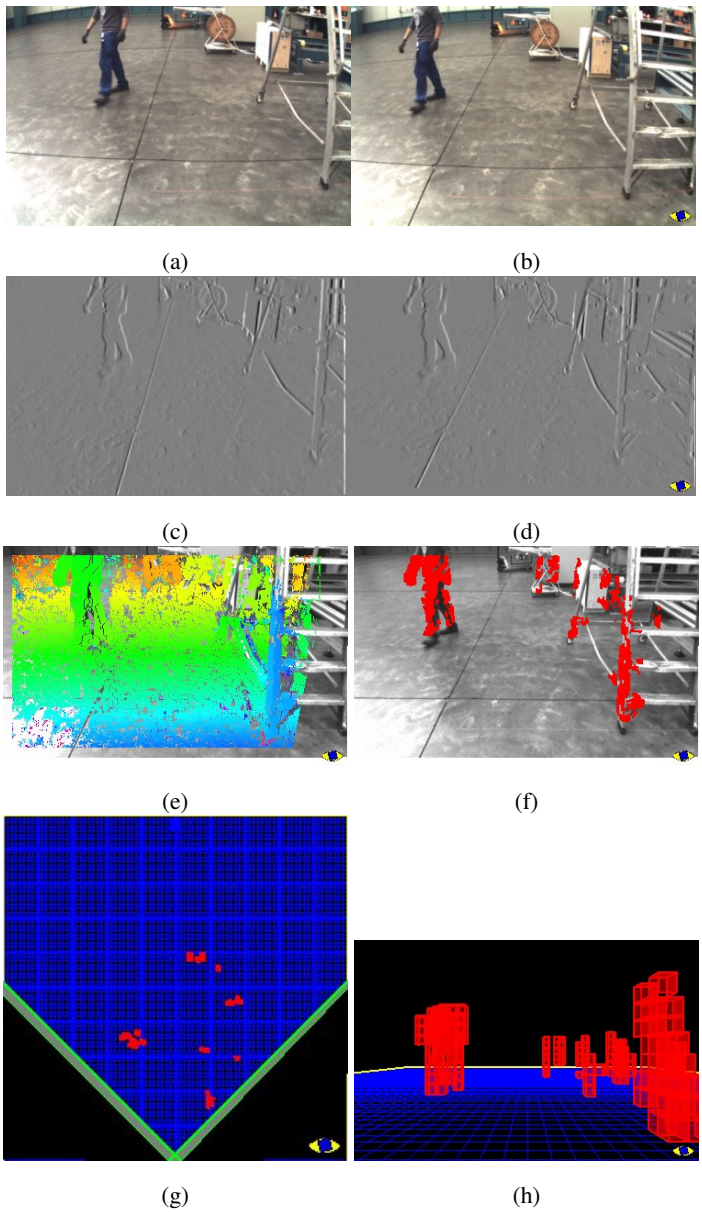
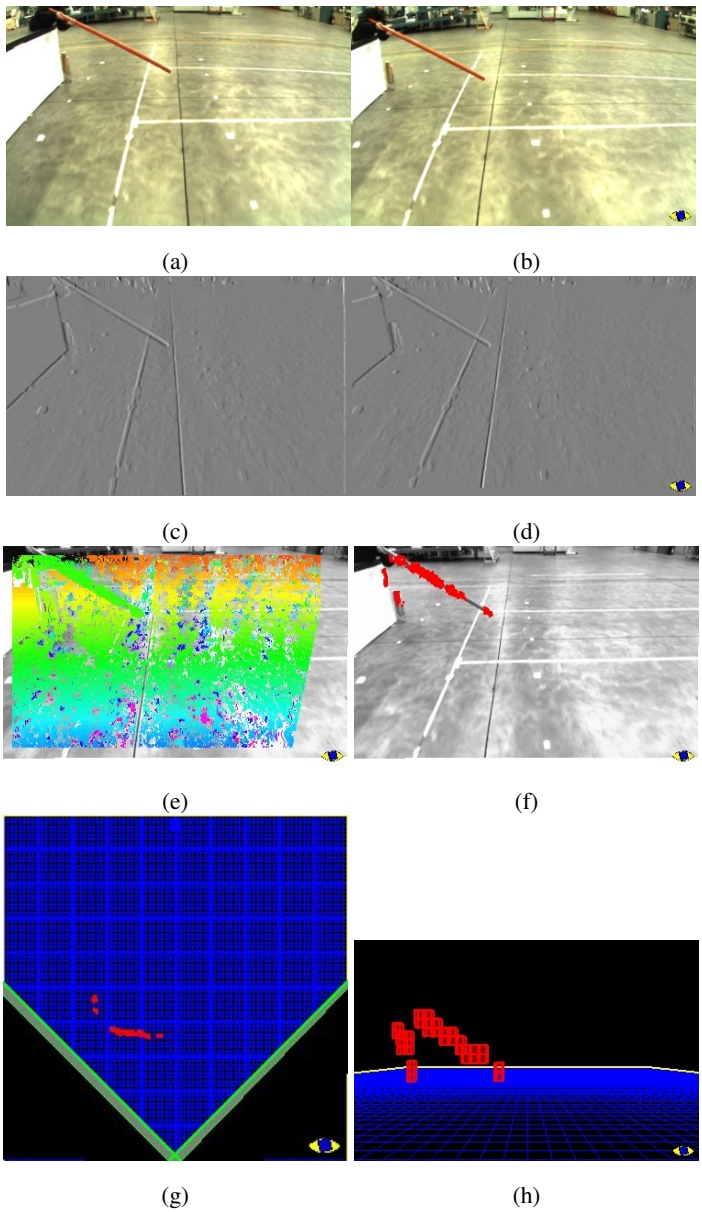
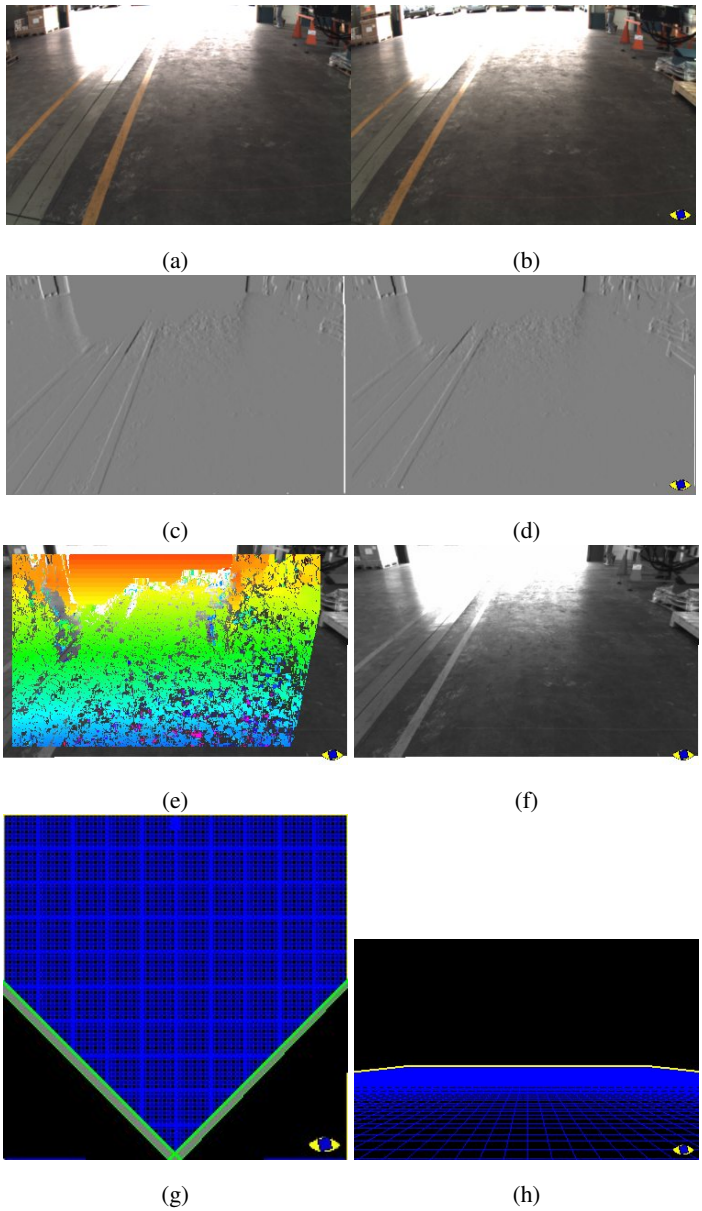


Figura 3.15: Fasi dell'algoritmo di riconoscimento ostacoli. Scena inquadrante una persona.



(a) (b) (c) (d) (e) (f) (g) (h)

Figura 3.16: Fasi dell'algoritmo di riconoscimento ostacoli. Scena inquadrante una ostacolo sottile e sospeso.



(a) (b) (c) (d) (e) (f) (g) (h)

Figura 3.17: Fasi dell'algoritmo di riconoscimento ostacoli. Scena inquadrante una riflesso intenso.

Filtro di Projection Engine

Noto l'insieme dei punti tridimensionali estratti è stato necessario applicare un approccio in grado di percepire la conformazione della scena nell'area presa in considerazione.

In base a come sono state posizionate le telecamere, all'interno delle immagini l'area inquadrata è occupata principalmente dal pavimento nel caso *indoor*, o terreno nel caso *outdoor*, e dagli eventuali ostacoli presenti, di conseguenza ci si aspetta che la maggior parte dei punti estratti appartenga alla superficie del terreno stesso.
Ragionando in quest'ottica si è pensato quindi di considerare due punti di vista opportunamente definiti rispetto al sistema di riferimento adottato, in modo tale da ottenere mediante le proiezioni dei punti su piani associati i due profili differenti della scena, andando ad eliminare l'informazione su una dimensione.
Dai primi risultati ottenuti e dalle osservazioni ricavate durante l'analisi delle due proiezioni costruite, si è notato che le informazioni contenute nelle immagini presentano caratteristiche significative ma non risultano sufficientemente corrette e precise per ottenere una stima della conformazione della scena, quindi si è reso necessario apportare delle modifiche opportune che permettessero di incrementare la qualità e la quantità di informazioni estraibile riguardo il profilo del terreno e la posizione degli ostacoli.
L'idea è stata quella di non considerare più un singolo piano di proiezione ma si è pensato di suddividere il volume d'interesse in differenti zone, e di proiettare i punti in esse contenuti su piani di proiezione separati.
Questa suddivisione permette quindi di effettuare stime più precise perchè la zona che si prende in considerazione ha un volume minore, di conseguenza la quantità di disturbi diminuisce e l'insieme di pixel omogenei che seguono il profilo del terreno o compongono ostacoli risulta più delineato all'interno dell'immagine della proiezione.
Parametro fondamentale risulta essere quindi il numero di zone in cui dividere l'area e di conseguenza la precisione che si vuole ottenere: più alto è il numero di zone considerate e maggiore sarà la precisione, ma d'altro canto una suddivisione eccessiva dell'area porta ad un carico computazionale elevato in quanto aumenta il numero di

immagini che si dovranno elaborare per ottenere la stima del terreno e gli ostacoli presenti nella zona.
In fase di implementazione sono state utilizzate differenti misure di profondità nella suddivisione dell'intera area d'interesse, in particolare valori pari a cinquanta, venticinque e dodici centimetri; si è notato che con questi valori si ottiene un carico computazionale accettabile, ed al diminuire del valore di profondità la precisione aumenta.
Dalle immagini ottenute ci si è resi conto chiaramente che il risultato in entrambe le tipologie di proiezioni nelle zone considerate fosse migliorato sia in termini di precisione che di qualità e che fosse realmente utilizzabile come ipotizzato, difatti tramite queste suddivisioni ciascuna immagine risulta nella maggior parte dei casi meno affetta da disturbi ed inoltre è possibile ricostruire la conformazione della scena ricomponendo le singole parti.

Le Fig.3.17 e quella Fig.3.16 dimostrano la robustezza dell'algoritmo proposto in condizioni di forti riflessi e l'affidabilità nel riconoscimento di ostacoli di piccole dimensioni anche quando questi eludono i sistemi di sicurezza automatica delle odierne navette LGV.

3.4 Trasmissione dati e protocollo di comunicazione con il veicolo

Il sistema di visione artificiale e quello di navigazione della navetta LGV sono collegati mediante un cavo ethernet e comunicano mediante protocollo TCP/IP. La sincronizzazione dei due orologi di sistema che servono per la generazione dei timestamp sono gestiti autonomamente dai servizi di ntpd.

La comunicazione è di tipo master-slave dove il sistema di navigazione funge da slave ed interroga il sistema di visione, master, fornendo quelli che sono i dati odometrici disponibili (velocità istantanea, angolo di sterzo e velocità di sterzo). Il sistema di visione, restituisce in risposta, la lista degli ostacoli individuati precedentemente elaborata.

Protocollo Generale	
@ msg id timestamp [data] *	
@	inizio del messaggio
msg	tipo di messaggio
id	identificatore del messaggio
timestamp	orario di spedizione messaggio
[*data*]	dati scambiati
*	fine del messaggio

Tabella 3.6: Protocollo per lo scambio dei dati fra i due sistemi

In Tab. 3.6 viene riportato il formato generale delle stringhe utilizzate nel protocollo per lo scambio dei dati.

Di seguito si riporta un esempio semplice di ricezione e risposta dati da lato master.

```
MyClass::On_Execute()
{
  // ACQUISIZIONE IMG

  // PREPROCESSING

  // DSI ENGINE

  // IPM

  // PROJECTION ENGINE
  m_ProjectionEngine.Run( points );
  m_finalObstacles = m_ProjectionEngine.GetFinalObstacles();
```

```
  // SORT OSTACOLI e CONTEGGIO

  // SOCKET
   if( m_socket_enable )
  {
    if( !m_client.getOdometry( m_odometry ) )
    {
      cout << "********************    ODOM NOT READY" << endl;
    }else
    {
      cout << "RECEVED MES: " << m_odometry.msg << endl;
    }

    m_obstacles_mes.cicle++;
    if(count_obst > 0) m_client.setObstacles( m_obstacles_mes, m_finalObst

    else{
      m_obstacles_mes.data.clear();
      m_client.setObstacles( m_obstacles_mes );
    }
  }
}
```

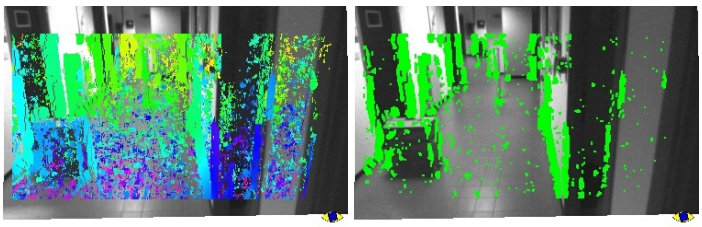
3.5 Prestazioni del sistema

In questa sezione vengono riportati i risultati di precisione nella misura ed le prestazioni in termini di tempi di esecuzioni ottenuti da alcune delle prove effettuate eseguendo l'algoritmo sviluppato in uno scenario noto e controllato.

Precisione dei risultati

La valutazione della precisione nelle misure finali delle posizioni degli ostacoli è stata eseguita posizionando il sistema all'interno di un corridoio, ed inquadrando una scena fissa nell'area di circa dieci metri di profondità e due di larghezza.
In particolare sono stati posizionati a varie distanze dalla testa stereo ostacoli dalle forme e dimensioni differenti. L'obiettivo di queste prove è stato quello di ottenere un riscontro significativo sull'errore fra misura reale e calcolata dall'algoritmo sulle posizioni e sulle dimensioni di ciascun ostacolo, e soprattutto sulla correttezza della conformazione dell'ambiente di test.

In Fig.3.19.b viene riportata una mappa dall'alto relativa alla struttura del corridoio ed al posizionamento degli ostacoli. Invece in Fig.3.18.a è illustrata la mappa di disparità ottenuta dalle due immagini sinistra e destra relative all'area inquadrata e l'insieme di punti validi ottenuti dal filtro di campionamento Fig.3.18.b, valori in ingresso all'algoritmo di proiezione implementato.



(a) (b)

Figura 3.18: Mappa di disparità e punti ottenuti mediante il filtro campionatore.

Di seguito, nelle due tabelle sono riportate per ciascun tipo di ostacolo le misure reali, quelle ottenute dall'algoritmo ed il conseguente errore; questa tipologia di suddivisione è stata considerata sia per le posizioni X ed Y rispetto all'origine di riferimento, che per le dimensioni di ogni singolo ostacolo.

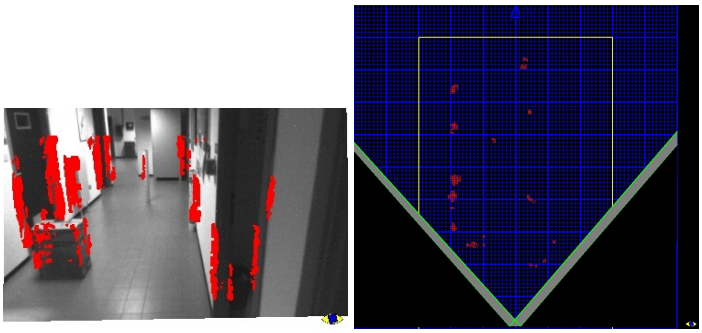
	Misure [mm]					
OSTACOLO	**reale**	**algoritmo**	**errore**	**reale**	**algoritmo**	**errore**
Porta A	3100	3040	60	+1900	+1750	150
Porta B	6350	6130	220	+1900	+1790	110
Porta C	6500	6650	150	-457	-455	2
Porta D	1800	1875	75	-457	-500	43
Ostacolo 1	2600	2500	100	+1000	+1000	0
Ostacolo 2	4000	4000	0	-400	-500	100
Ostacolo 3	4700	4600	100	+1775	1550	225
Ostacolo 4	5800	5750	50	+600	+625	25
Ostacolo 5	8000	8125	125	-357	-340	17

Tabella 3.7: Risultati ottenuti sulla posizione nel mondo di tutti gli ostacoli presenti

	Misure [mm]					
OSTACOLO	**reale**	**algoritmo**	**errore**	**reale**	**algoritmo**	**errore**
Porta A	1080	1100	20	125	130	5
Porta B	1080	1250	170	125	160	35
Porta C	1080	ND	ND	125	ND	ND
Porta D	1080	ND	ND	125	250	125
Ostacolo 1	300	625	325	520	500	20
Ostacolo 2	115	200	85	115	200	85
Ostacolo 3	480	ND	ND	125	200	75
Ostacolo 4	115	160	45	115	125	10

Tabella 3.8: Risultati ottenuti sulle dimensioni di tutti gli ostacoli presenti

Dai risultati ottenuti si può certamente sottolineare che l'algoritmo presenta una buona precisione nelle misure degli ostacoli, indifferentemente dalla distanza a cui si trovano dalla navetta LGV.
In particolare è possibile notare che salvo qualche sporadico caso in cui si presentano dei lievi errori nel calcolo della mappa di disparità, l'errore nelle misure risulta dell'ordine dei dieci centimetri in media ovunque.
Inoltre, gli ostacoli risultano ben delineati e correttamente riconosciuti senza la presenza di falsi negativi nella *Occupancy Map* relativa alla scena vista dall'alto.
Questa tipologia di output e di correttezza dei risultati rimane invariata al cambiare di uno dei parametri importanti dell'algoritmo, ovvero il passo fra un piano di proiezione e il successivo che determina il grado di suddivisione del volume di interesse.



(a) (b)

Figura 3.19: Risultato algoritmo ostacoli *a* e Occupancy Map *b*

.

3.5.1 Tempi di elaborazione

La valutazione dei tempi di esecuzione dell'algoritmo è stata svolta inserendo dei singoli *timer* per ciascuna fase dell'algoritmo, ed eseguendo l'algoritmo per un numero di immagini consecutive elevato significativo.
I motivi principali di questa scelta sono i seguenti:

Tempi di elaborazione [msec]	
Fase algoritmo	**Tempo**
Acquisition	0.005
Preprocessing	5
Disparity	10
Disparity Sampler	3
IPM Disparity Points	0.5
Projection Engine(step=125mm)	8
Projection Engine(step=250mm)	7
Projection Engine(step=500mm)	5

Tabella 3.9: Tempi di elaborazione delle singole fasi dell'algoritmo.

- è stato deciso di prendere in considerazione un numero elevato di frame in modo tale da verificare la robustezza in sé dell'algoritmo in presenza di ostacoli o meno, e soprattutto per rilevare dei valori medi significativi dei tempi relativi a ciascuna fase dell'algoritmo;

- è stato infine deciso di valutare come si comporta l'algoritmo in termini computazionali e se si riscontrano differenze significative, al variare del passo dei piani di proiezione utilizzato.

I risultati ottenuti dalle prove effettuate sono riportati nelle tabelle seguenti.

Dai tempi ottenuti in entrambe le situazioni analizzate, è possibile osservare che ciascuna operazione non impiega tempi particolarmente elevati, e nel totale l'applicazione impiega nel complesso un tempo ben inferiore al limite imposto di cento millisecondi.
Il motivo principale della bontà di questo risultato è dovuto al fatto che ove possibile in fase di implementazione, è stata sfruttata la caratteristica multiprocessore del pc e sono state eseguite in parallelo la maggioranza delle operazioni tramite l'utilizzo di *thread*.

Un'altra considerazione riguarda i tempi di elaborazione ottenuti nell' elaborazione delle proiezioni dei punti della mappa di disparità al variare del passo dei piani di proiezione utilizzato. Come è visibile in Tab. 3.9, al diminuire del passo, si nota un leggero incremento del tempo di elaborazione dei punti, ma nel complesso in media questo peggioramento non si nota nel tempo totale di esecuzione dell'algoritmo. Pertanto è possibile concludere che il sistema è in grado di lavorare in modo real-time.

Capitolo 4

Conclusioni

Scopo del mio lavoro è stato quello di sviluppare ed applicare tecniche di visione artificiale complesse a sistemi industriali reali, studiandone le problematiche principali e curando gli aspetti di interfacciamento fra i dati sensoriali elaborati ed l'automazione della loro esecuzione.

Durante questi anni di Dottorato, il processo di accrescimento delle cognizioni tali da sviluppare alcuni sistemi industriali complessi è stato progressivo, consolidando le tecniche avanzate di elaborazione immagine prima e applicandole a scenari industriali, dopo. Questo percorso mi ha permesso di sviluppare una particolare sensibilità alla maggioranza di quelli che sono le principali problematiche nel campo della visione industriale ed acquisire competenze via via maggiori sulla implementazione reale di sistemi autonomi complessi.

È importante sottolineare che durante questa attività ho avuto l'opportuinità di studiare diverse soluzioni architetturali adottate per integrare i componenti che formano un sistema autonomo industriale, la loro evoluzione, i loro pregi ed i loro difetti. Essendo le competenze richieste per la realizzazione di questi sistemi varie ed eterogenee, includendo l'intelligenza artificiale, l'elettronica, la meccanica e le telecomunicazioni; questo mi ha portato ad acquisire conoscenze trasversali indispensabili nel campo dell'automazione industriale avanzata.

L'eterogeneità dei sistemi industriali che sfruttano tecniche di visione artificiale

proposti in questa dissertazione, dimostrano come questo specifico campo sia ancora estremamente aperto e vasto. Infatti nei capitoli precedenti si è fatto riferimento a sistemi di percezione e misura applicati a veicoli commerciali, di seguito ad un sistema innovativo di sicurezza per cancelli automatici ed infine ad un sistema complesso che unisce algoritmi complessi di elaborazione immagini per veicoli industriali ed la loro navigazione autonoma.

Questo percorso di studi mi ha così dato la capacità di analizzare sistemi industriali avanzati, riconoscendo le principali problematiche e possibilità, nonchè di sviluppare possibili soluzioni mediante nuovi metodi sperimentali di visione artificiale considerando sia gli aspetti algoritmici che quelli implementativi.

Il lavoro svolto durante il periodo di Dottorato ha contribuito alla ricerca nel settore della visione artificiale mediante lo studio di nuovi algoritmi applicati a problemi complessi industriali e non, sperimentando queste soluzioni su piattaforme reali avvalendosi della collaborazione tecnica delle stesse industrie oggetto delle varie collaborazioni.

Bibliografia

[1] http://www.vislab.it

[2] http://www.druid-project.eu

[3] Broggi, A., Cappalunga, A., Caraffi, C., Cattani, S., Ghidoni, S., Grisleri, P., Porta, P.P., Posterli, M., Zani, P.: TerraMax Vision at the Urban Challenge 2007. IEEE Trans. on Intelligent Transportation Systems (2008). Submitted

[4] Norén, J.: Warning systems design in a glass cockpit environment. Thesis dissertation, University of Linkopings (2008)

[5] Broggi, A., Medici, P., Porta, P.P.: StereoBox: a Robust and Efficient Solution for Automotive Short Range Obstacle Detection. EURASIP Journal on Embedded Systems – Special Issue on Embedded Systems for Intelligent Vehicles (2007). ISSN 1687-3955

[6] Lin, Q., Fredrik, Tjärnströ, Roll, J., Wass, B.: A Far-Infrared based Night-Vision System with Detection. In: Procs. VISION 2008 Intl. Conference. Versailles-Satory, France (2008)

[7] Roger Y. Tsai: A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology Using off-the-shelf TV Cameras and Lenses. IEEE Journal of Robotics and Automation **3**, 323–344 (1987)

[8] Wei, G.Q., Ma, S.D.: Implicit and explicit camera calibration: Theory and experiments. IEEE Trans. Pattern Anal. Mach. Intell. **16**(5), 469–480 (1994).

[9] Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(11), 1330–1334 (2000).

[10] Ziraknejad, N., Tafazoli, S., Lawrence, P.: Autonomous stereo camera parameter estimation for outdoor visual servoing. Machine Learning for Signal Processing, 2007 IEEE Workshop on pp. 157–162 (2007).

[11] Kwon, H., Park, J., Kak, A.: A new approach for active stereo camera calibration. Robotics and Automation, 2007 IEEE International Conference on pp. 3180–3185 (2007).

[12] Broggi, A., Caraffi, C., Porta, P.P., Zani, P.: The Single Frame Stereo Vision System for Reliable Obstacle Detection used during the 2005 Darpa Grand Challenge on TerraMax. In: Procs. IEEE Intl. Conf. on Intelligent Transportation Systems 2006, pp. 745–752. Toronto, Canada (2006)

[13] Dickmans, E.D., Mysliwetz, B.D.: Recursive 3-D Road and Relative Ego-State Recognition. IEEE Trans. on Pattern Analysis and Machine Intelligence **14**, 199–213 (1992)

[14] Schiehlen, J., Dickmanns, E.: Design and control of a camera platform for machine vision. In: Procs. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, pp. 2058–2063 (1994)

[15] Cardani, B.: Optical image stabilization for digital cameras. IEEE Control System Magazine **26**(2), 21–22 (2006)

[16] Bombini, L., Cerri, P., Grisleri, P., Scaffardi, S., Zani, P.: An Evaluation of Monocular Image Stabilization Algorithms for Automotive Applications. In: Procs. IEEE Intl. Conf. on Intelligent Transportation Systems 2006, pp. 1562–1567. Toronto, Canada (2006)

[17] European Machine Vision Association (2008) European Vision Technology Market Statistics 2008. Frankfurt/Main, Germany

[18] M. Bertozzi and A. Broggi. GOLD: a Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection. In *IEEE Trans. on Image Processing 1998*, chapter 7, pages 62–81, Gen 1998.

[19] P. Medici Tecniche ibride basate su movimento e stereoscopia per il riconoscimento ostacoli in ambiente automobilistico. In *PhD Thesis*, dissertation, University of Parma (Gen 2009).

[20] Http://www.arduino.cc

[21] P. Medici Artificial Vision Course. In *Teaching slides*, University of Parma (2009).