UNIVERSITÀ DEGLI STUDI DI PARMA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Dottorato di Ricerca in Tecnologie dell'Informazione
XXIV Ciclo

Elena Cardarelli

ELABORAZIONE DI IMMAGINI PER LA REALIZZAZIONE DI SISTEMI DI SUPPORTO ALLA GUIDA AUTOMATICA IN AMBITO URBANO, EXTRAURBANO ED OFF-ROAD

DISSERTAZIONE PRESENTATA PER IL CONSEGUIMENTO
DEL TITOLO DI DOTTORE DI RICERCA

GENNAIO 2012

UNIVERSITÀ DEGLI STUDI DI PARMA

Dottorato di Ricerca in Tecnologie dell'Informazione

XXIV Ciclo

ELABORAZIONE DI IMMAGINI PER LA REALIZZAZIONE DI SISTEMI DI SUPPORTO ALLA GUIDA AUTOMATICA IN AMBITO URBANO, EXTRAURBANO ED OFF-ROAD

Coordinatore:

Chiar.mo Prof. Marco Locatelli

Tutor:

Chiar.mo Prof. Alberto Broggi

Dottorando: *Elena Cardarelli*

Gennaio 2012

Sommario

Elenco delle figure

Elenco delle tabelle

Introduzione

Negli ultimi anni diversi interventi sono stati promossi per il miglioramento della sicurezza stradale, a partire dall'entrata entrata in vigore del Decreto Legge n.151 del 27 giugno 2003 che ha introdotto la patente a punti[1] e nuove regole in tema di codice della strada, tra cui sanzioni più severe per tutti gli automobilisti indisciplinati. Innumerevoli sforzi sono stati inoltre compiuti nei diversi campi di ricerca, in favore del miglioramento delle condizioni di guida, sia per quanto riguarda i dispositivi a bordo del veicolo (si pensi a sistemi quali ABS[2] o il più recente EPS[3]), che per quanto concerne la manutenzione delle infrastrutture stradali ed il continuo lavoro di potenziamento delle stesse (asfalto drenante, barriere di sicurezza, ecc..), soprattutto in ambito autostradale [1].

I dispositivi di supporto alla guida, in grado di sopperire alle disattenzioni del conducente o di avvisarlo di un'eventuale situazione di pericolo, diventano quindi

[1] La patente a punti è stata introdotta con il DL n. 151 del 27 giugno 2003, modificato in alcune parti prima di essere definitivamente convertito con la legge n. 214 del 1 agosto 2003. I punti sono sottratti facendo riferimento alla tabella allegata all'art. 126 bis del codice della strada. In pratica, tutti i titolari di patente italiana (o membri dell'Unione Europea con residenza in Italia e dunque con patente convertita) dal 30 giugno 2003 hanno ricevuto un "bonus" virtuale di 20 punti. Chi commette infrazioni al codice stradale, oltre ad una sanzione pecuniaria e ad eventuali sospensioni temporanee della patente di guida, è assoggettato alla decurtazione di un certo numero di punti, variabile a seconda della gravità dell'infrazione commessa. All'esaurimento dei punti la patente viene ritirata in modo definitivo. Inoltre, per i titolari di patente che per almeno due anni hanno mantenuto 20 punti, è previsto l'accreditamento di 2 punti, fino a raggiungere il tetto massimo complessivo di 30 punti.

[2] Antilock Braking System

[3] Electric Power Steering

sempre più indispensabili con l'evoluzione delle tecnologie a disposizione. Nell'ultimo decennio diversi settori di ricerca hanno focalizzato la loro attenzione sullo sviluppo di sistemi per autovetture, quali ad esempio ACC (Automatic Cruise Control) o ADAS (Advanced Driver Assistance Systems) e numerose risorse sono state investite per fornire sistemi sempre più efficienti, caratterizzati da elevate performance e costi ridotti.

La combinazione di questi sistemi ha reso possibile lo sviluppo concreto di prototipi, come i veicoli autonomi di Google [72], ARGO [17], TerraMax [83] e Braive [49], che hanno contribuito sia al miglioramento della sicurezza stradale, che alla nascita di una nuova concezione dei sistemi di trasporto, volta a favorire l'impiego di mezzi completamente autonomi per trasportare persone ed oggetti, da un luogo all'altro, in modo sicuro ed efficiente.

A tal proposito il Laboratorio di Visione Artificiale (VisLab) del Dipartimento di Ingegneria dell'Informazione dell'Università di Parma ha condotto, tra Luglio ed Ottobre 2010, VIAC (VisLab Intercontinental Autonomous Challenge) [15, 16, 23], una spedizione intercontinentale da Milano a Shanghai. Per dimostrare come sia possibile trasportare in modo autonomo persone e merci, anche lungo un percorso intercontinentale, sono stati equipaggiati quattro veicoli elettrici (Porter Piaggio) installando su ciascuno di essi un sistema di guida autonoma alimentato da pannelli solari. L'obiettivo principale del progetto è stato il testing delle performance in condizioni estreme: in particolare, sono state misurate le prestazioni dei sistemi di guida automatica in varie situazioni stradali (autostrada, urbano, extra-urbano, off-road, traffico intenso), in presenza di diverse condizioni metereologiche (sole, neve, pioggia, nebbia) e di illuminazione (mattina, pomeriggio, tramonto, sera). Con questo esperimento si è inoltre dimostrato come il consolidamento di nuove tecnologie permetta di migliorare l'affidabilità dei veicoli, in favore di una maggiore sicurezza dei passeggeri.

Tuttavia in ambito di sicurezza stradale quanto fatto finora non è abbastanza: secondo gli ultimi dati Istat [4], pubblicati il 9 Novembre 2011 e relativi all'anno 2010, in Italia ogni giorno si verificano mediamente 579 incidenti stradali, che causano la morte di 11 persone e il ferimento di altre 829. Nel complesso, nell'anno 2010 sono stati rilevati 211.404 incidenti stradali, che hanno causato il decesso di 4.090 persone

mentre altre 302.375 hanno subito lesioni di diversa gravità. La tabella 1 evidenza una diminuzione del numero degli incidenti (-1,9%), del numero dei morti (-3,5%) e del numero dei feriti (-1,5%) a partire dal 2008.

	Valori assoluti 2008	**Valori assoluti 2009**	**Valori assoluti 2010**	**Variazioni percentuali 2008/2009**	**Variazioni percentuali 2009/2010**
Incidenti	218.963	215.405	211.404	-1,6%	-1.9%
Morti	4.725	4.237	4.090	-10.3%	-3.5%
Feriti	310.745	307.258	302.735	-1.1%	-1.5%

Tabella 1: Incidenti stradali, morti e feriti anni 2008/2009 e 2009/2010. Il numero dei morti e feriti pubblicato dall'Istat il 13 novembre 2009 è stato rettificato a seguito della segnalazione dell'Ufficio di Statistica della Provincia di Rimini, che ha effettuato, durante il 2010, un controllo sul numero dei morti e feriti in incidenti stradali verificatisi nella Provincia per l'anno 2008. Il numero dei morti è stato rettificato da 4.731 in 4.725, mentre il numero dei feriti da 310.739 in 310.745. In particolare, le discrepanze tra il dato pubblicato dall'Istat e quello verificato dall'Ufficio di Statistica della Provincia sono da attribuirsi ad un errore nel numero di persone infortunate in incidenti stradali avvenuti nel Comune di Riccione.

Nonostante la sicurezza stradale sia determinata da diversi fattori legati alle tre entità uomo-macchina-ambiente, quali ad esempio l'affidabilità del veicolo, una corretta manutenzione, la guida in buone condizioni fisiche, l'ambiente in cui ci si muove, l'adeguatezza delle infrastrutture, osservando i dati statistici si scopre che il fattore umano è all'origine della maggioranza degli incidenti che avvengono in ambito urbano e extraurbano: nel 2010, secondo i dati Istat, più dell'80% dei sinistri è stato causato dal comportamento scorretto del conducente alla guida del veicolo. In particolare, il 17,1% del totale delle cause è rappresentato dal mancato rispetto delle regole di precedenza, il 17% dalla guida distratta o andamento indeciso, il 11,6% dall'eccesso di velocità e il 10,3% dal mancato rispetto della distanza di sicurezza: dispositivi di supporto alla guida in grado di sopperire a tutti questi comportamenti

diventano, quindi, sempre più indispensabili.

Il lavoro svolto nell'ambito di questa tesi di dottorato ha riguardato l'analisi, lo sviluppo e l'implementazione di algoritmi di visione artificiale per la realizzazione di sistemi di supporto alla guida automatica in ambito urbano, extraurbano ed *off-road*. Particolare enfasi è stata riservata al ruolo dei classificatori dimostrandone la loro versatilità.

Il primo sistema oggetto della ricerca si colloca nell'ambito del progetto che vede la collaborazione del *VisLab* con il settore sistemi elettronici di "Magneti Marelli". Il progetto ha come obiettivo la realizzazione di un sistema per riconoscere e classificare i principali segnali stradali europei appartenenti a tutte le categorie esistenti: pericolo, divieto, obbligo, fineobbligo, precedenza, stop e indicazione [21]. Il dispositivo, montato su un'autovettura, con l'ausilio di una singola telecamera digitale acquisisce, in tempo reale, immagini dell'ambiente stradale e comunica all'autista i risultati dell'elaborazione, mediante un'opportuna interfaccia. L'attività di ricerca svolta durante il dottorato si è concentrata sulla risoluzione di alcune problematiche riscontrate dall'analisi del sistema: l'algoritmo si è rivelato molto sensibile a rotazioni e traslazioni dei segnali nelle immagini acquisite e alcune componenti di rumore, introdotte dalle varie elaborazioni, compromettevano le prestazioni di classificazione In particolare il riconoscimento falliva nella maggior parte dei casi in cui il cartello si presentava ruotato, deformato, disallineato rispetto all'andamento della carreggiata, o anche solo ostruito in minima parte. Alla luce di queste problematiche, si è deciso di studiare un algoritmo alternativo, che non soffrisse di queste debolezze e che al contempo ne mantenesse i pregi. L'obiettivo principale è stato quindi il rendere più robusto il sistema nelle situazioni precedentemente citate, introducendo nuove funzionalità volte ad incrementare l'affidabilità dei risultati.

Il secondo sistema, trattato nell'ambito di questa tesi, è stato realizzato in collaborazione con "Caterpillar Inc." e riguarda la realizzazione di un dispositivo che permetta di rendere autonoma una ruspa che opera all'interno di una cava. Il progetto ha quindi due obiettivi principali: la ricostruzione del profilo del terreno, al fine di determinare le zone attraversabili, e l'individuazione di eventuali ostacoli all'interno di una specifica area di interesse. Al fine di rendere questo sistema commercializza-

bile si è resa necessaria la valutazione delle performance di riconoscimento in diverse condizioni di operabilità: in particolare, nell'ambito di questa tesi di dottorato, è stato studiato un metodo, basato su *machine learning*, che non si limita alla sola misura delle prestazioni, ma permette di prevedere il livello di affidabilità che il sistema sarà in grado di offrire.

Il terzo sistema oggetto di questa tesi si occupa dell'assistenza durante il cambio di corsia e permette di monitorare, tramite una singola telecamera, l'area relativa al punto cieco dell'autista (*blind spot*), ovvero la zona che non può essere vista né tramite lo specchietto retrovisore, né tramite quelli laterali. Quando il dispositivo rileva un veicolo in fase di sorpasso all'interno della zona critica, viene emesso un segnale acustico che avverte il conducente della pericolosità di cambiare corsia e del possibile rischio di collisione. Per questo specifico sistema l'attività di dottorato si è concentrata sullo studio di diverse metodologie per la rilevazione dei veicoli in fase di sorpasso e sull'implementazione di un algoritmo per affrontare questa problematica.

La ricerca condotta su questi tre sistemi ha dimostrato come i classificatori siano strumenti versatili che possono essere impiegati in diverse fasi dello sviluppo di un sistema. Nel riconoscimento della segnaletica stradale vengono utilizzati nell'ultima fase dello sviluppo, per determinare la classe di appartenenza di un campione specifico; nel sistema di riconoscimento ostacoli i classificatori permettono di prevedere le performance a partire da uno specifico insieme di caratteristiche estratte dalle immagini acquisite; nel sistema di rilevamento dei veicoli in sorpasso l'uso dei classificatori si dimostra cruciale nella prima fase dello sviluppo, per determinare la lista di possibili candidati da processare nei passi successivi.

Capitolo 1

Sistema per il riconoscimento segnaletica stradale

1.1 Stato dell'arte

Il riconoscimento della segnaletica stradale è un campo estremamente importante nel settore dei veicoli intelligenti. Ciò è dovuto alla rilevanza che i segnali stradali assumono durante la guida; essi definiscono un linguaggio visuale facilmente comprensibile, attraverso il quale si rappresenta lo stato corrente delle condizioni stradali, si mostra la presenza di particolari pericoli oppure si danno indicazioni sul comportamento da mantenere in strada. La loro struttura elementare li rende agevolmente individuabili, ma esistono numerose circostanze in cui la loro presenza può sfuggire e compromettere la sicurezza dei passeggeri.

Diversi approcci sono stati studiati e testati in letteratura per il riconoscimento automatico dei segnali stradali. Una delle principali differenze tra i metodi presentati consiste nella scelta di utilizzare immagini a toni di grigi oppure a colori. Per quanto riguarda la prima corrente, la motivazione principale che spinge all'utilizzo di immagini monocromatiche è legata a considerazioni sul fatto che il colore è sensibilmente influenzato da diversi fattori, tra cui l'illuminazione [71, 41, 75, 97, 24] e le diverse condizioni climatiche [41, 66] che rendono l'analisi cromatica poco affidabile.

Si dimostra quindi come possa avvenire un buon riconoscimento dei segnali considerando solamente le informazioni legate alla loro forma [80], ai loro contorni [56] o a particolari *features* [87] che li caratterizzano. Un vantaggio legato all'utilizzo di questi metodi sarebbe il minore onere computazionale dell'elaborazione di immagini a toni di grigio rispetto a quelle a colori, che rende l'algoritmo efficiente per l'utilizzo in ambito *automotive*; due grandi svantaggi relativi all'utilizzo di immagini a toni di grigio sono però l'alta dipendenza dal contrasto e alcuni malfunzionamenti legati ad un'eventuale asimmetria di illuminazione delle immagini acquisite.

Chi preferisce l'elaborazione di immagini a colori, invece, adduce motivazioni riguardanti la maggiore quantità di informazioni a disposizione, che riduce il numero di falsi positivi in fase di riconoscimento dei contorni, dovuti a oggetti inquadrati che hanno la stessa forma dei segnali; questo a costo di un maggiore onere computazionale, che però non compromette le prestazioni, come evidenziato in [79, 21]. Due sono i principali approcci seguiti da chi opta per l'elaborazione di immagini a colori: segmentazione basata sul colore, seguita da ricerca dei cartelli sulla base di forma e posizione oppure ricerca eseguita sulla sola base di colore e dimensione. Passo importante in entrambi i metodi risulta essere la segmentazione, che può essere eseguita utilizzando diversi spazi di colore: molti studi, come [31, 54, 89], sono stati sviluppati lavorando nello spazio di colore HSV al fine di rendere i sistemi robusti rispetto i cambi di illuminazione. La necessità di realizzare questi dispositivi a bordo di autovetture dotate di hardware con ridotte capacità, spinge i progettisti ad orientarsi verso approcci quanto più ridotti possibile, sia da un punto di vista della complessità di calcolo che di occupazione di memoria. Questa è la ragione per cui frequentemente non si lavora nello spazio di colore HSV, ma in spazi che non necessitano di trasformazioni gravose per la potenza di calcolo richiesta, come RGB [56, 9, 35, 102, 34, 53] e YUV [91, 89].

Non in tutti gli articoli si cerca di riconoscere più di una forma: in [11, 91, 52] si ricercano solamente le forme circolari; in [35, 34, 54] si rilevano solo triangoli, mentre in [68] si cercano, oltre ai triangoli, anche rettangoli e ottagoni. Il lavoro svolto nell'articolo [11] si basa su di una variante della *trasformata di Hough* per la ricerca di simmetria radiale. Alcuni lavori dimostrano inoltre come l'utilizzo di

pattern matching risulti essere un metodo veloce e robusto [91, 40]. In articoli come [10, 33, 31, 32, 70] sono stati studiati, come terzo possibile approccio, alcuni algoritmi genetici per valutare le forme presenti nell'immagine: sebbene questi algoritmi si rivelino abbastanza efficaci, risultano essere troppo lenti e quindi non adatti al contesto *automotive*. Un'ulteriore tecnica impiegata per il riconoscimento di segnaletica stradale è la visione stereoscopica [13, 69]: l'approccio descritto in [25] prevede l'individuazione di regioni complanari tra due immagini al fine di localizzare i segnali stradali.

1.2 Descrizione funzionale dell'algoritmo

Il sistema di riconoscimento dei segnali stradali oggetto della ricerca si compone essenzialmente di due parti: un sistema di acquisizione e uno di elaborazione. Il sistema di acquisizione si avvale di una singola telecamera montata all'altezza dello specchietto retrovisore per permettere l'acquisizione di immagini a colori relative all'area antistante il veicolo. L'elaborazione realizza invece le due operazioni principali necessarie per il riconoscimento dei segnali, ovvero la ricerca e la classificazione. La ricerca ha come obiettivo l'individuazione all'interno delle immagini di regioni di interesse (*bounding box*) compatibili con la presenza di un segnale stradale. La classificazione invece si occupa dell'estrazione del contenuto informativo al fine di determinare la categoria di appartenenza del segnale stradale.

La procedura di riconoscimento si basa sull'analisi cromatica e morfologica delle immagini a colori provenienti dalla telecamera, ed articolata in 5 fasi, illustrate e schematizzate in figura 1.1: *(a)* acquisizione delle immagini dall'ambiente stradale, *(b)* segmentazione basata sul colore, *(c)* riconoscimento delle forme, *(d)* combinazione dei dati elaborati ai due stadi precedenti, *(e)* classificazione mediante rete neurale. Per i segnali di fine prescrizione è stato sviluppato un procedimento *ad-hoc* [27], basato sull'analisi delle immagini a tono di grigio; l'analisi cromatica per questo specifico caso si è infatti dimostrata poco affidabile perché il colore bianco, caratteristico per questa tipologia di segnali, non rappresenta una *feature* distintiva per il loro riconoscimento: diversi elementi all'interno di un'immagine, come ad esempio l'asfalto

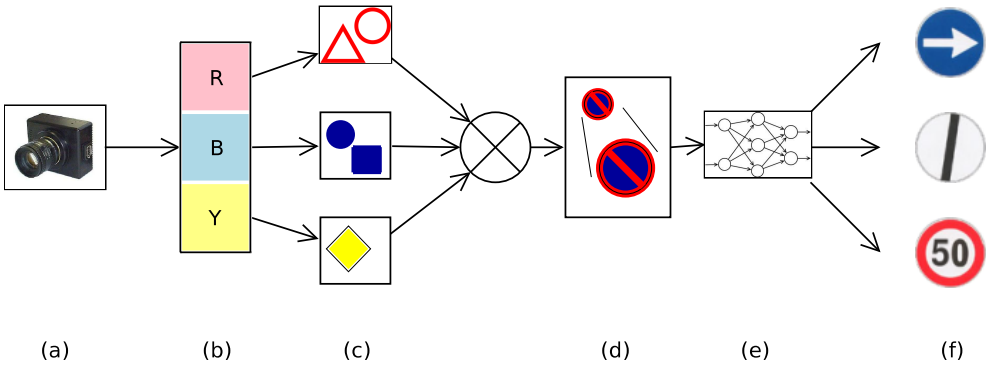
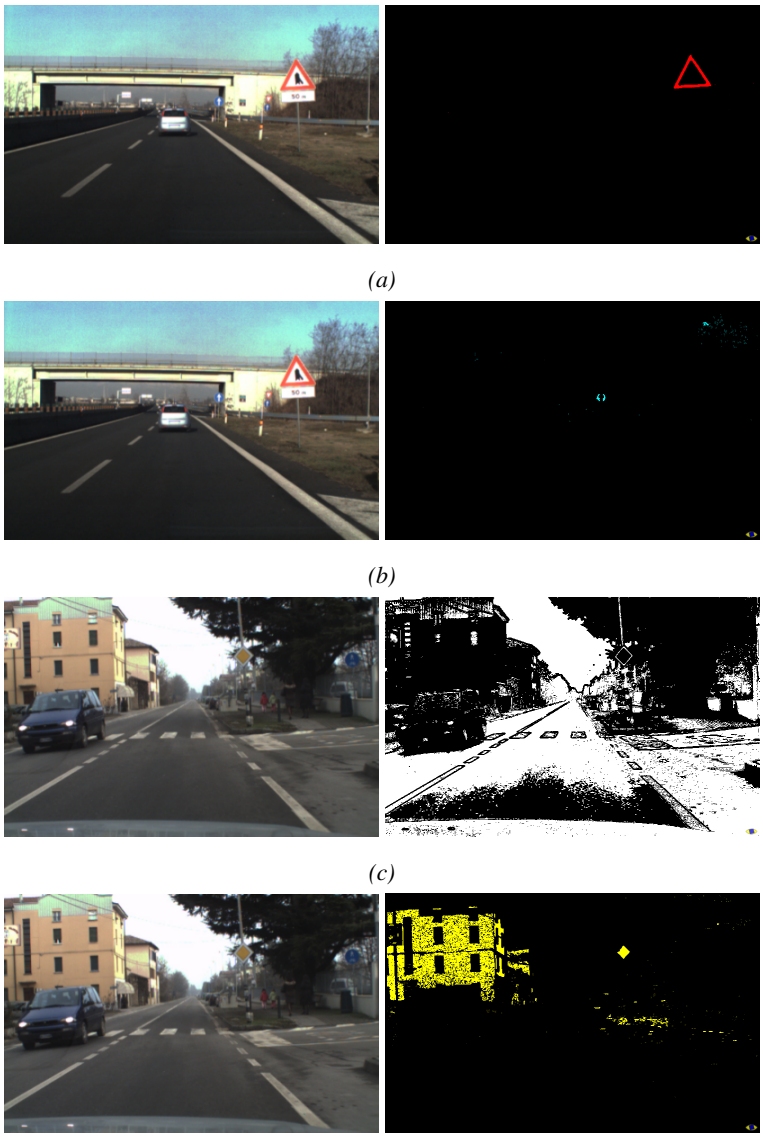


Figura 1.1: Schema a blocchi del sistema: *(a)* acquisizione, *(b)* segmentazione basata sul colore, *(c)* riconoscimento delle forme, *(d)* fusione e adattamento dei dati, *(e)* classificazione mediante rete neurale.

della strada o il cielo, possono apparire con una tonalità del tutto equivalente a quella del colore bianco caratteristico dei segnali di fine prescrizione.

1.2.1 Segmentazione del colore

La segmentazione del colore ha come obiettivo la ricerca all'interno delle immagini delle regioni con caratteristiche cromatiche compatibili con quelle dei segnali stradali: si limita quindi la ricerca dei colori al rosso, giallo, blu e bianco (figura 1.2). Per garantire il corretto funzionamento dell'algoritmo sotto diverse condizioni di illuminazione, si prevede innanzitutto di eseguire sulle immagini in ingresso una procedura di equalizzazione cromatica per attenuare il contributo della sorgente luminosa, affinché gli oggetti all'interno delle immagini appaiano con il loro colore originale. In particolare si riconduce il problema di equalizzazione alla rimappatura dei livelli RGB. La figura 1.3 mostra gli effetti dell'equalizzazione sulla segmentazione del colore: in questo caso i vantaggi più evidenti si hanno sul canale giallo, ove si apprezza una notevole diminuzione del rumore. Osservando attentamente i numerosi segnali stradali, risulta evidente come sia possibile definire per ognuno di essi un colore principale, quello cioè presente con percentuale più alta rispetto al totale, ed un



(a)

(b)

(c)

(d)

Figura 1.2: Filtraggio cromatico.

(a) *(b)*

Figura 1.3: Effetti dell'equalizzazione cromatica sulla segmentazione del colore. Sono esposti i risultati del filtraggio dei tre canali: (a) prima; (b) dopo l'equalizzazione.

colore secondario, che ne permette l'associazione con una distinta categoria. Si consideri, ad esempio, la classe di cartelli che rappresentano la fine di un obbligo: essi hanno sfondo blu e sono attraversati da una banda rossa obliqua; è quindi ragionevole assegnare loro come colore principale il blu, e come secondario il rosso.

Questa suddivisione permette quindi di fare alcune considerazioni preliminari circa la categoria di appartenenza dei candidati individuati: ad esempio per i segnali di divieto, pericolo, precedenza, stop il rosso rappresenta il colore primario ed il bianco quello secondario (costituiscono un'eccezione i cartelli di divieto di sosta il cui colore secondario è il blu). Il blu è colore primario per i segnali di obbligo, fine obbligo ed indicazione. Il giallo risulta essere colore primario per i segnali che indicano una strada con diritto di precedenza e secondario per quelli temporanei, come quelli di lavori in corso. Al termine di questo processo si dispone quindi di una lista di regioni di interesse caratterizzate da un colore dominante ed eventualmente da uno secondario che ne permettono una prima suddivisione in classi.

1.2.2 Riconoscimento delle forme

La fase successiva si occupa dell'analisi morfologica: avendo già operato una prima pre-classificazione, basata sui colori primario e secondario, si ricercano forme specifiche per ogni categoria di segnali precedentemente rilevata. Ad esempio se si considera un *bounding box* avente colore primario giallo, è noto che l'unico segnale che risponde a questa proprietà è il cartello di diritto di precedenza (figura 1.4), pertanto su di esso non si cercheranno corrispondenze con cerchi o triangoli, bensì solo con rombi.

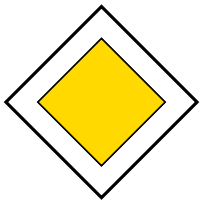


Figura 1.4: Segnale di strada con diritto di precedenza.

Il riconoscimento della forma avviene tramite *pattern matching* tra immagini binarizzate (per i *bounding box* dal colore primario rosso e giallo) o, alternativamente, mediante elaborazioni locali sui bordi dell'immagine originale (per i *bounding box* dal colore primario blu). Il *pattern matching* consiste nel calcolare il valore di correlazione del *bounding box* in esame con ognuna delle forme campione (riportate in figura 1.5), ottenendo per ogni forma un valore di *match* che indica la percentuale di somiglianza tra il contenuto del *bounding box* e la forma stessa; il valore più elevato stabilisce quale sia la forma riconosciuta, cioè quella presente con maggiore probabilità all'interno del riquadro. Una procedura differente viene eseguita, invece, per i *bounding box* caratterizzati da colore primario blu, in quanto ricorrendo al solo *pattern matching* risulta difficile distinguere le forme circolari dai quadrati con angoli arrotondati. Pertanto, nei *bounding box* di colore blu si ricercano i pixel accesi a partire dai quattro vertici spostandosi verso il centro in direzione radiale e incrementando di volta in volta il raggio della ricerca (figura 1.6), così da determinare, in modo univoco, se si tratta di una forma quadrata (poca distanza dai vertici) o cir-

(a) *(b)* *(c)* *(d)* *(e)* *(f)*

Figura 1.5: Immagini campione utilizzate per il *pattern matching*. La prima *(a)* identifica segnali di divieto; la seconda *(b)* serve per riconoscere segnali di obbligo ed eccezionalmente anche quello di stop; la terza *(c)* identifica i segnali di divieto d'accesso; *(d)* e *(e)* riconoscono rispettivamente i segnali di pericolo e precedenza; *(f)* serve per i segnali di diritto di precedenza.

colare (distanza maggiore dai vertici). Al termine di questa fase, ogni *bounding box* possiede un'informazione precisa riguardante la forma dell'oggetto contenuto. Dalla combinazione dei risultati relativi all'analisi cromatica e morfologica è quindi possibile stabilire la categoria di appartenenza dei segnali individuati e determinare il ramo della rete neurale verso cui instradare il *bounding box*.

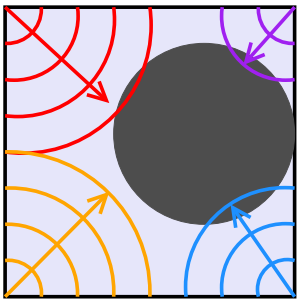


Figura 1.6: Procedimento utilizzato per il riconoscimento delle forme blu, nell'algoritmo originale.

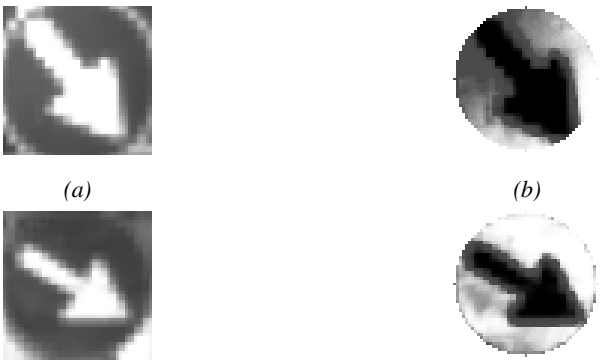
1.3 Processo di adattamento delle immagini

Per stabilire la tipologia del segnale inquadrato ed il suo contenuto informativo si rendono necessarie alcune elaborazioni sui candidati, al fine di renderli il più possibile somiglianti con i campioni utilizzati per addestrare i classificatori, nel caso specifico le reti neurali.

La fase di adattamento delle immagini ha appunto come obiettivo la definizione degli ingressi per la rete neurale, a partire dai risultati ottenuti dalle fasi di segmentazione e riconoscimento delle forme. In particolare si vuole che due *bounding box*, relativi a segnali della stessa tipologia, producano gli stessi ingressi per il classificatore, così da facilitare la corretta estrazione del loro contenuto informativo. A partire dalle immagini relative ai cartelli individuati si costruiscono quindi dei modelli, che oltre a rappresentare gli ingressi della rete neurale, possono essere sfruttati per definire il set di addestramento (*training set*).

Si prevede quindi di ritagliare dall'immagine a colori originale tutte le regioni di interesse, candidate a contenere un segnale stradale. Ogni ritaglio è poi ricampionato e convertito in un'immagine a toni di grigio. Si ottiene così un'immagine 50×50 *pixel* per ogni *bounding box* individuato, indipendentemente dalla sua dimensione e quindi dalla distanza del segnale stradale dalla telecamera. Al fine di delimitare l'area da cui estrarre il contenuto informativo si definisce, in base alla forma del segnale inquadrato da ciascun *bounding box*, una maschera che permette di scartare tutti i *pixel* dell'immagine che non appartengono al cartello, ma allo sfondo su cui è posto. In particolare per i segnali che possiedono un bordo (divieto e pericolo) si provvede inoltre a ridurre la dimensione del *bounding box*, in modo che l'applicazione della maschera permetta di escludere oltre allo sfondo, tutti i contributi apportati dal bordo. L'area filtrata tramite l'applicazione della maschera sui campioni 50×50 pixel ha quindi una forma che varia a seconda della categoria di appartenenza del cartello inquadrato e una dimensione sempre fissa (figura 1.7).

Un fattore da tenere in considerazione nella determinazione degli ingressi per il classificatore è la luce: sotto diverse condizioni atmosferiche e a diversi orari del giorno uno stesso soggetto può mostrarsi in modi totalmente differenti; può inoltre



(a) *(b)* *(c)* *(d)*

Figura 1.7: Esempi di applicazione della maschera per filtrare il contenuto informativo del cartello. L'area circolare, filtrata su due immagini differenti ha sempre la stessa dimensione.

trovarsi esposto alla luce diretta del sole oppure essere parzialmente oscurato dall'ombra di un altro soggetto. Cartelli della stessa tipologia, se illuminati in modo differente, possono quindi generare risultati differenti; la semplice conversione delle immagini a colori in toni di grigio non è sufficiente a garantire che i dati prodotti dalla segmentazione collimino con quelli di addestramento, in quanto:

- i toni di grigio relativi a primo piano e sfondo possono differire, anche di poco, da quelli utilizzati nella fase di addestramento;

- colori differenti mostrano, dopo la conversione, lo stesso tono di grigio.

Per evitare che queste condizioni compromettano drasticamente le prestazioni si utilizzano tecniche di calibrazione del contrasto (*contrast stretching*), volte a modificare l'istogramma relativo all'immagine contenente il segnale da classificare, in modo da renderlo il più possibile somigliante a quello appartenente ai modelli utilizzati in fase di addestramento.

1.3.1 Problematiche riscontrate

Dall'analisi sperimentale dell'algoritmo è stato riscontrato che i dati in ingresso alla rete neurale presentano contributi rumorosi in presenza di:

- segnali stradali ruotati sia verticalmente che orizzontalmente (figura 1.8 *a*-*b*);
- disallineamenti e traslazioni dovuti a *bounding box* non perfettamente centrati sul segnale inquadrato (figura 1.8 *c*-*d*).

(a) *(b)*

(c) *(d)*

Figura 1.8: Problematiche riscontrate nell'algoritmo preesistente. *(a)* Segnale ruotato orizzontalmente. *(b)* Segnale ruotato verticalmente. *(c)* e *(d)* Esempi di disallineamenti dei *bounding box* non centrati rispetto il cartello.

In questi casi le immagini non si limitano a rappresentare il solo contenuto informativo, ma contengono elementi appartenenti allo sfondo e/o al bordo del cartello

che compromettono le prestazioni della rete neurale. Ciò accade perché la maschera utilizzata per filtrare le componenti di rumore (al fine di evidenziare il solo contenuto informativo), considera sempre una forma di dimensione fissa, senza valutare le singole caratteristiche dei cartelli inquadrati.

Sulla base di queste considerazioni è stato quindi studiato un metodo alternativo per eliminare i contributi rumorosi all'interno delle immagini ed agevolare quindi la fase di riconoscimento.

1.3.2 Descrizione funzionale del nuovo algoritmo

L'algoritmo implementato per far fronte alle problematiche precedentemente esposte prevede innanzitutto un'estensione della regione di interesse al fine di risolvere i problemi legati a disallineamenti e traslazioni dei *bounding box* rispetto al cartello: si considera quindi per ciascun *bounding box* individuato il 5% in più della sua area. La regione così ottenuta è ritagliata e ricampionata in modo da ottenere, per ogni *bounding box* un'immagine a colori 32×32 *pixel*: in questo modo per uno stesso tipo di cartello rilevato in *frame* differenti si ottengono in uscita immagini del tutto simili, indipendente dalla distanza del segnale dalla telecamera. La scelta di ridurre le dimensioni dei campioni da porre in ingresso alla rete neurale è giustificata dal fatto che migliorando la fase di ritaglio e riducendo il rumore nelle immagini si può pensare di sottocampionare ulteriormente i dati senza che un'eventuale perdita di informazioni influenzi le prestazioni di riconoscimento: lavorando su ritagli sottocampionati, ma comunque più precisi e meno rumorosi, a parità di prestazioni di classificazione rispetto al metodo preesistente, si elaboreranno comunque immagini di dimensioni minori, favorendo un alleggerimento del carico computazionale.

Si procede poi con l'individuazione dei bordi del cartello che delimitano l'area da cui estrarre il contenuto informativo. In particolare si cercano, all'interno di ciascun ritaglio, le transizioni di colore che:

- per i cartelli di divieto e pericolo segnano il passaggio dal bordo rosso alla regione interna del segnale (bianca o gialla) su cui si trova la prescrizione da seguire (figura 1.9 *a*-*b*);

- per le altre tipologie di segnali separano i bordi del cartello dallo sfondo (figura 1.9 *c-d*).

(a) *(b)* *(c)* *(d)*

Figura 1.9: Estrazione del contenuto informativo. In verde sono evidenziate le aree da individuare per l'estrazione del contenuto informativo.

Il primo passo per l'individuazione dei punti di transizione riguarda la trasformazione dei ritagli di input a colori in in un'immagine a toni di grigio da cui estrarre l'informazione relativa ai bordi del cartello. In questa fase dell'elaborazione l'attenzione è rivolta verso la massimizzazione del rapporto tra le informazioni utili estratte e il rumore introdotto: nel passaggio da un'immagine a colori (RBG) ad una monocromatica si perde infatti parte del contenuto informativo originale; si deve quindi prestare attenzione nella scelta tra le informazioni da mantenere o viceversa scartare. In altre parole, dopo l'elaborazione si dovranno riconoscere in campo scalare (immagine monocromatica) quelle variazioni che prima erano determinate in un campo vettoriale (immagine a colori).

Disponendo delle informazioni preliminari sul colore dominante del *bbox* di volta in volta analizzato, si utilizzano tecniche di esaltazione del contrasto: in particolare, al fine di facilitare la ricerca dei bordi, si evidenzia la differenza cromatica tra lo sfondo presente nel ritaglio e la componente primaria di colore associata al *bounding box*. La scelta della formula per l'esaltazione di colore (dallo spazio RGB dell'immagine in ingresso) ha richiesto uno studio sperimentale approfondito legato all'alta variabilità dell'ambiente stradale in cui si collocano i segnali. In particolare, si è dovuto tener conto dello sfondo sul quale si può presentare il cartello, che in seguito alla perdita di informazione causata dall'estrazione di una singola componente di colore, potrebbe erroneamente uniformarsi al colore (e quindi all'area) del segnale stesso,

con conseguente perdita di informazione sui contorni. Trascurando i casi particolari di segnali posti su sfondo dello stesso colore, che costituiscono indubbiamente situazioni di difficile analisi, esistono altri casi in cui un'erronea scelta della funzione di trasformazione dell'immagine originale in immagine monocromatica può causare l'assenza di contrasto tra il cartello e lo sfondo. Ad esempio, selezionando il solo canale rosso della terna RGB, nel caso di un cartello dal colore primario rosso posto su sfondo di foglie gialle autunnali si perderebbero completamente le informazioni sui bordi del segnale stesso. Ricordiamo infatti che sia il colore rosso sia il colore giallo presentano valori alti per quanto riguarda la componente rossa nello spazio di colore RGB (figura 1.10): pertanto, estraendo dall'immagine originale solo il valore della prima componente si annullerebbe totalmente il contrasto tra le due zone; di conseguenza si perderebbero anche le informazioni sui bordi originali degli oggetti inquadrati e, in questo modo, elementi rossi ed elementi gialli risulterebbero indistinguibili nell'immagine trasformata.

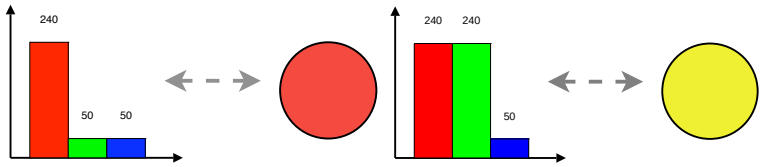


Figura 1.10: Studio comparativo dei colori rosso e giallo. Si vuole evidenziare come entrambe le tinte presentino alti valori della componente rossa, nello spazio di colore RGB.

Nella scelta del metodo di esaltazione del contrasto è inoltre importante considerare l'effetto delle condizioni di illuminazione sulla scena inquadrata: queste infatti possono influenzare il processo di estrazione della componente primaria; uno stesso cartello può trovarsi esposto alla luce diretta del sole oppure essere parzialmente oscurato, e sotto diverse condizioni atmosferiche e a diversi orari del giorno può apparire con colori totalmente diversi all'interno delle immagini. Nell'immagine a toni

di grigio il bordo di un cartello può presentarsi quindi poco contrastato rispetto lo sfondo su cui è posto, rendendo critica la fase di ricerca dei bordi (figura 1.11).

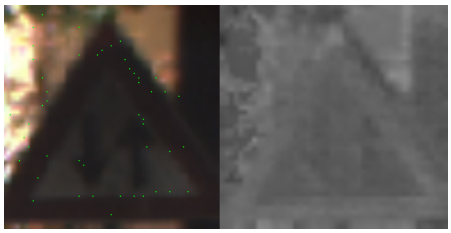


Figura 1.11: Esempio di cartello in ombra. Nell'immagine a toni di grigio i bordi del segnale non sono distinguibili.

Sono stati effettuati quindi vari test, utilizzando tecniche differenti per l'estrazione del colore primario: il metodo che sperimentalmente è risultato rispondere meglio nella maggior parte dei casi prevede il calcolo della distanza euclidea tra ciascun *pixel* del ritaglio a colori e la componente primaria di colore associata al cartello inquadrato. In particolare per la stima del colore primario (rosso per i cartelli di divieto e pericolo, blu per quelli di indicazione e di obbligo) non si utilizza un valore fisso per tutti i *bounding box* individuati, ma si considerano di volta in volta le caratteristiche cromatiche del singolo cartello inquadrato, rendendo quindi l'intero processo di estrazione indipendente dalle condizioni di illuminazione. Nel caso dei segnali di divieto e pericolo il colore primario medio è calcolato come la media dei *pixel* appartenenti al bordo rosso del cartello: si ritaglia, secondo i limiti definiti dal *bounding box*, l'immagine binaria del rosso definita in fase di segmentazione e si utilizza il ritaglio ottenuto (figura 1.12 *b-c*) come maschera per individuare i *pixel* appartenenti al bordo, dei quali si calcola il valor medio. Per i cartelli di obbligo e indicazione la componente primaria di colore è stimata calcolando il valore medio dei *pixel* appartenenti al blu del cartello. Il procedimento utilizzato per ottenere questo risultato è analogo al caso dei segnali di divieto e pericolo, con la differenza che la maschera utilizzata per selezionare i *pixel* deriva dall'immagine binaria del blu (figura 1.12 *h-k*), invece che del rosso.

Dati i vettori RGB p_i e m che rappresentano rispettivamente l'i-esimo *pixel* dell'immagine di ingresso e il colore primario medio stimato per il cartello inquadrato, l'i-esimo *pixel* dell'immagine di uscita equivale alla distanza euclidea d_i, calcolata secondo la seguente formula:

$$d_i = \frac{p_i \cdot m}{p_i \times m} \tag{1.1}$$

Il risultato ottenuto, chiamato immagine della distanza euclidea (figura 1.12 *c-f-i-l*), mostra quanto il valore di ogni *pixel* dell'immagine originale si discosta dalla componente cromatica primaria del cartello, calcolata tenendo conto della tinta con cui il colore primario (rosso o blu) appare all'interno dell'immagine.

A questo punto è stato possibile individuare l'area relativa al contenuto informativo ricercando radialmente, a partire dal centro di ciascun *bounding box*, i punti corrispondenti ad una transizione chiaro-scuro per i segnali con bordo rosso (figura 1.13 *a-b*) e, viceversa, scuro-chiaro per quelli senza bordo rosso (figura 1.13 *c-d*).

L'insieme di questi punti delimita la regione di interesse per il riconoscimento: si calcola così la *trasformata di Hough* per trovare la forma (circolare, triangolare o rettangolare) che meglio approssima questi punti. In particolare per i segnali di forma circolare (divieto e obbligo), l'applicazione della *trasformata di Hough* sui punti individuati, permette di stimare le coordinate del centro e i valori dei semiassi (maggiore e minore) dell'ellisse che meglio approssima la forma del segnale inquadrato (figura 1.14 *b-d*). Per i cartelli di forma triangolare (pericolo) e rettangolare (indicazione) si utilizza la *trasformata di Hough* per individuare le rette i cui punti di intersezione corrispondono ai vertici del poligono (triangolo o rettangolo) che approssima i punti individuati (figura 1.14 *f-h*).

Il poligono individuato dalla *trasformata di Hough* delimita la regione di interesse relativa al contenuto informativo: si utilizzano quindi i dati ottenuti per definire una maschera che permette escludere tutti i punti che, non trovandosi all'interno dell'area racchiusa dal poligono, rischiano di portare contributi rumorosi. La maschera viene applicata all'immagine a toni di grigio ottenuta a partire dal ritaglio a colori originale. In particolare nella conversione da colori a toni di grigio viene dato un peso maggiore al canale rosso ed uno minore al canale blu, questo per evitare di avere lo stesso

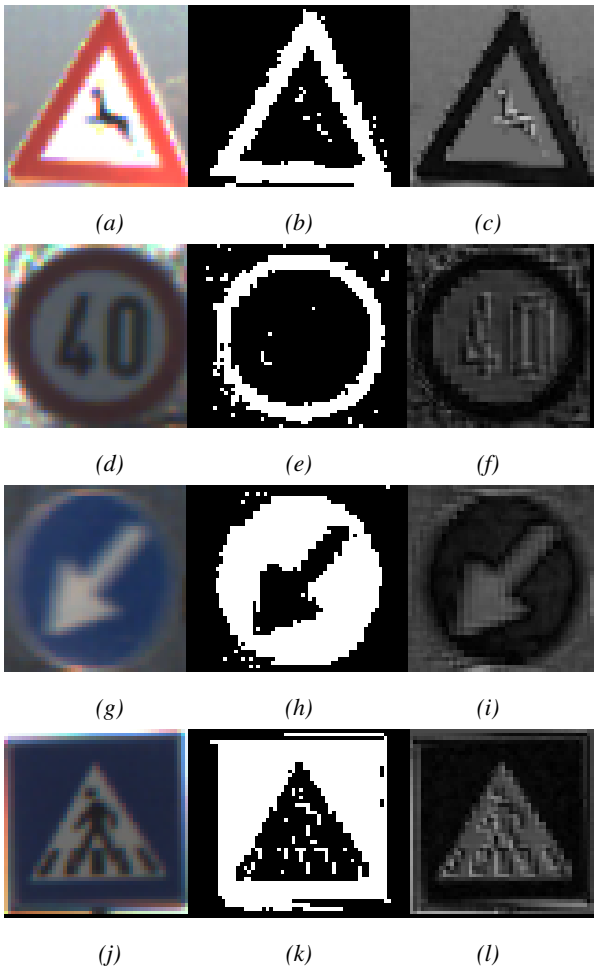


Figura 1.12: Elaborazione dell'immagine della distanza euclidea. In particolare per i cartelli di divieto e pericolo (*c*-*f*) l'immagine della distanza euclidea mostra quanto il valore di ogni *pixel* dell'immagine originale si discosta dal valore medio dei *pixel* appartenenti al bordo del segnale. Per i cartelli di obbligo e indicazione (*i*-*l*) l'immagine della distanza euclidea mostra invece quanto il valore di ogni *pixel* dell'immagine originale si discosta dal valor medio dei *pixel* che rappresentano lo sfondo blu del cartello.

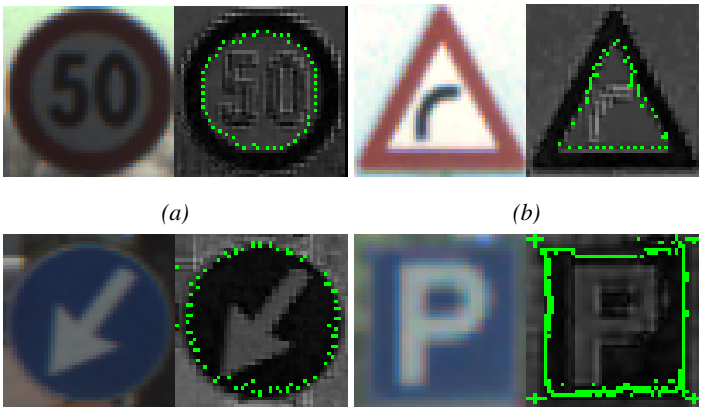


Figura 1.13: Ricerca dei punti di transizione per segnali di: *(a)* divieto, *(b)* pericolo, *(c)* obbligo, *(d)* indicazione. Per migliorare l'affidabilità del risultato, per i segnali di obbligo e di indicazione si ricercano radialmente, a partire dal centro, due punti di transizione invece che uno solo.

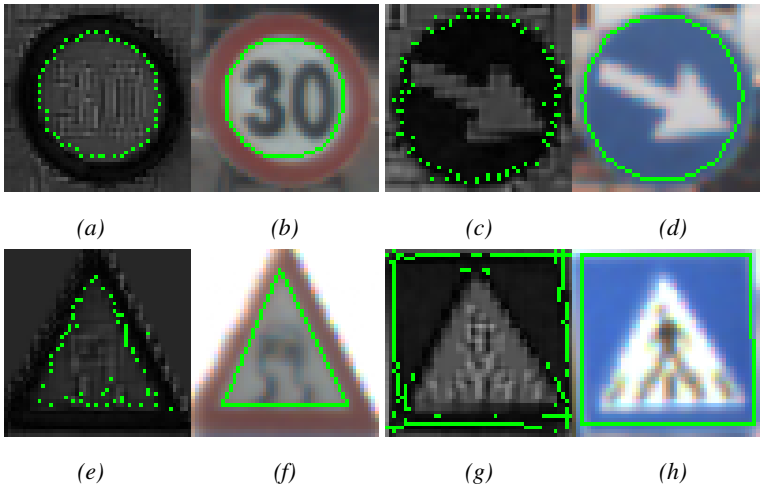


Figura 1.14: Individuazione, tramite l'applicazione della *trasformata di Hough*, della forma che approssima i punti di transizione trovati.

tono di grigio per colori diversi. I miglioramenti sono netti, come risulta evidente valutando la figura 1.15.

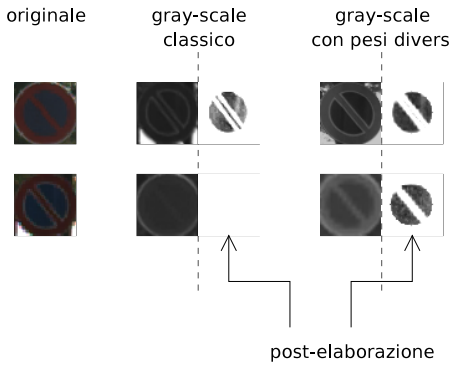


Figura 1.15: Utilizzo di pesi differenti nella conversione in scala di grigi.

1.4 Ottimizzazione del contrasto

Il passo finale per l'elaborazione degli ingressi del classificatore consiste nella calibrazione del contrasto (*contrast stretching*), al fine di rendere l'immagine contente il segnale da classificare il più possibile somigliante ai modelli utilizzati in fase di addestramento.

La tecnica utilizzata per il miglioramento del contrasto prevede l'applicazione di un algoritmo di trasformazione che permette di espandere l'intervallo dei livelli di grigio di ciascuna immagine verso tutto l'intervallo dei valori possibili oppure verso un *range* di valori desiderati (figura 1.16). Si modifica quindi l'istogramma dell'immagine per variare il *range* dei valori di intensità dei pixel; l'approccio si articola in due passaggi: inizialmente si recuperano i valori massimi all'interno dell'istogramma, coi quali si effettua, nella fase successiva, una specifica rimappatura.

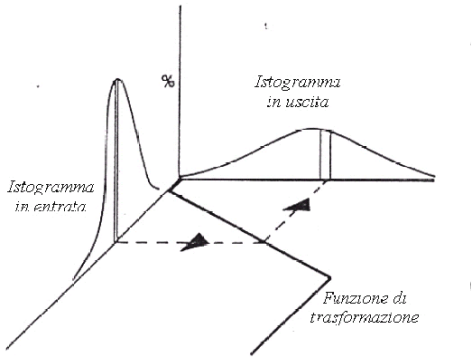


Figura 1.16: Modello per la calibrazione del contrasto.

1.4.1 Tecniche esplorate

Poiché la forma degli istogrammi relativi ai segnali stradali varia da categoria a categoria, è giustificabile l'idea di aver sviluppato più di un metodo. In particolare per ridistribuire i valori dell'istogramma sono stati applicate tecniche di *stretching* che utilizzano sia funzioni di trasformazione lineari che non lineari.

Lineare

Si parla di *contrast stretching* lineare quando la funzione di trasformazione è una retta del tipo:

$$y = ax + q \tag{1.2}$$

Tale retta mette in relazione i livelli di grigio x dei pixel dell'immagine originale con i livelli di grigio y dell'immagine trasformata. Il contrasto aumenta o diminuisce in funzione del coefficiente angolare a. Per costruire la funzione di calibrazione del contrasto si considerano, inoltre, due valori che rappresentano il valore massimo e mimino che i pixel dell'immagine rimappata potranno assumere; nel caso del *contrast stretching* lineare questi due valori definiscono il *range* entro cui la retta varierà. In particolare, in un immagine codificata ad 8 bit l'intervallo dei livelli di grigio rappresentabili nell'immagine rimappata è $(0, 255)$.

Oltre al caso lineare semplice uno dei più diffusi è quello lineare a tratti o *piecewise* (figura 1.17): in questo caso la funzione di trasformazione è rappresentata da una spezzata, costituita da un insieme di segmenti con diversi coefficienti angolari. In questo modo i livelli di grigio dell'immagine possono essere redistribuiti in modo differenziato a seconda degli obiettivi. In particolare, nell'esempio mostrato in figura 1.17, il primo tratto della funzione ha pendenza minore di uno e permette quindi di contrarre i valori dell'istogramma (in corrispondenza dell'intervallo evidenziato in verde sull'asse delle ascisse), in modo da rimanere al di sotto di una soglia (evidenziata dall'intervallo in giallo sulle ordinate). Il secondo tratto permette invece di espandere i valori dell'istogramma (che appartengono all'intervallo evidenziato in blu) verso un *range* più esteso (evidenziato in arancione).

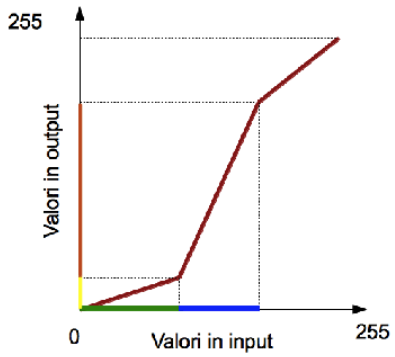


Figura 1.17: Esempio di grafico lineare a tratti.

Non lineare

Lo *stretching* non lineare si basa sull'utilizzo di curve anziché rette. Questa tecnica, nota anche come equalizzazione dell'istogramma, implementa una funzione mono-

tona non lineare che permette di appiattire l'istogramma dell'immagine in uscita, distribuendo uniformemente le intensità. L'equalizzazione classica, visivamente, tende a ridurre il contrasto nei livelli molto alti e molto bassi dell'immagine originale, aumentando invece quello dei livelli intermedi. È comunque possibile utilizzare diverse curve di equalizzazione al fine di ottenere risultati differenti: ad esempio le curve logaritmiche possono essere impiegate per esaltarne i toni bassi dell'immagine e per comprimere i toni alti; viceversa, le curve esponenziali tendono ad esaltare i toni alti e a comprimere quelli bassi. Facendo una panoramica dei casi, le curve non lineari utilizzate per il *contrast stretching* sono molteplici: parabole, funzioni esponenziali, logaritmiche, curve gaussiane, ecc.

Le curve non lineari principalmente analizzate in questo progetto sono state le funzioni paraboliche poiché grazie alle loro proprietà di trasformazione permettono sia la compressione che l'esaltazione dei toni di grigio al variare della curvatura. In particolare l'equazione della parabola utilizzata è quella passante per due punti:

$$y = y_1 + \frac{(y_2 - y_1)(x - x_1)^2}{(x_2 - x_1)^2} \tag{1.3}$$

dove i valori dell'equazione sono quelli dei due punti estremi del tratto parabolico, rispettivamente $(x1, y1)$ e $(x2, y2)$, mentre x è il valore d'ingresso della trasformazione e y l'uscita trasformata.

Nella precedente equazione la parabola viene determinata solo dai due estremi, avendo una concavità fissa. Mentre nell'implementazione di un tratto di parabola parametrica è possibile modificare la sua concavità in funzione delle esigenze di trasformazione (figura 1.18).

É anche possibile implementare, come nello *stretching* lineare, funzioni a tratti costituite sia da parti lineari che non lineari. In questo caso si parla di funzione di trasferimento mista come mostrato in figura 1.19.

1.4.2 Modelli utilizzati

Poiché gli istogrammi presentano andamenti diversi a seconda della categoria di appartenenza del segnale, l'approccio preesistente prevede l'uso di funzioni diverse sia

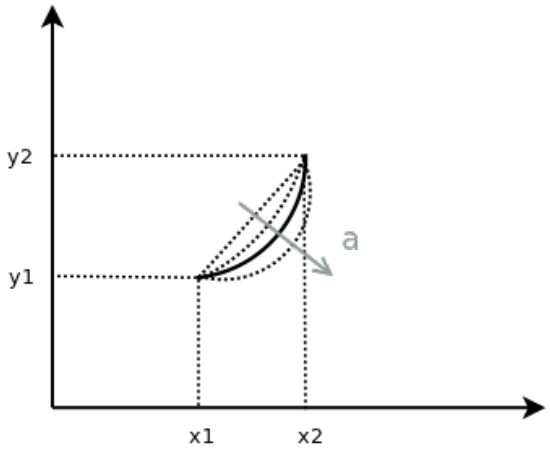


Figura 1.18: Tratto parabolico che ha due estremi fissi e concavità dipendente dal valore di *(a)*.

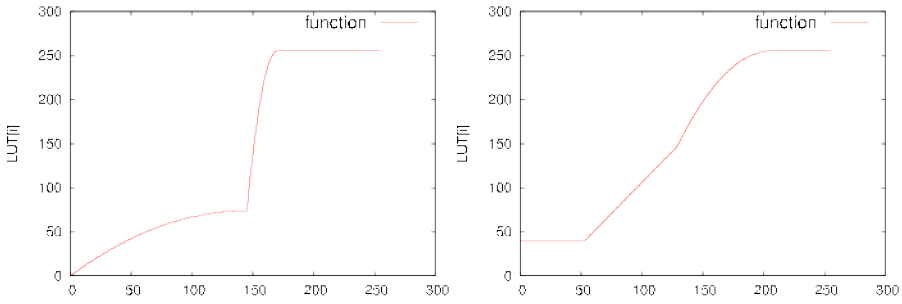


Figura 1.19: Modelli di funzione di trasferimento misto.

per la ricerca del picco di minimo e di massimo all'interno dell'istogramma, che per la redistribuzione dei valori. Nel seguito vengono analizzati i tre metodi utilizzati e successivamente le funzioni di *contrast stretching* applicate alle categorie.

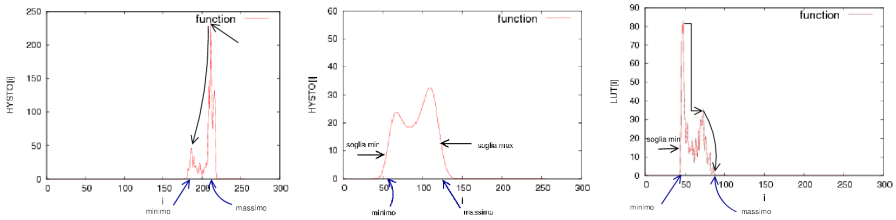
1. Nel primo metodo vengono recuperati due valori di picco, il primo mediante una semplice ricerca del massimo elemento in un *array*, il secondo, invece, mediante l'utilizzo di una macchina a stati. A partire dal livello più elevato, corrispondente al tono di grigio di valore 255, si procede a ritroso verso i livelli inferiori analizzando la cresta dell'istogramma alla ricerca di due decrementi consecutivi (figura 1.20 *a*). Il punto di scollinamento immediatamente precedente ai decrementi viene considerato come secondo valore di massimo. La ricerca è tale da scartare i massimi locali di un intorno piccolo; questo metodo funziona bene con le categorie dei segnali di stop, di obbligo e di fine obbligo.

2. Il secondo algoritmo si differenzia dal primo nel metodo di ricerca dei due massimi[1]. Inizialmente, viene fissata una soglia con un valore convenzionale e si cerca nell'istogramma un punto di valore superiore a tale soglia. Se nessuno dei punti dell'istogramma soddisfa tale condizione si procede decrementando la soglia di un passo predefinito e si ricomincia la ricerca. Il metodo è simmetrico, ovvero la ricerca viene effettuata su due fronti: dal basso verso l'alto e viceversa. L'intento principale è di trovare una regione compatta all'interno dell'istogramma, delimitata dai due valori di picco appena trovati, e ciò accade in particolar modo per i segnali di divieto, in cui l'informazione in primo piano si staglia nettamente rispetto allo sfondo, molto più chiaro (figura 1.20 *b*).

3. L'ultimo metodo analizzato è assai simile al precedente, in quanto il punto di massimo relativo ai toni scuri viene trovato con lo stesso procedimento. Cambia radicalmente il metodo di ricerca del punto di massimo relativo ai toni chiari: a partire dal primo massimo rilevato, si procede valutando ogni livello successivo finché non si scende al di sotto di una soglia ulteriore (figura 1.20

[1] Questi due valori non sono sempre massimi locali o globali dell'istogramma ma vengono indicati in questo modo perché spesso sono associati agli indici bassi e alti dell'istogramma.

c). Il vantaggio offerto da questo metodo risiede nel fatto che si pone maggiore importanza ai toni di grigio intermedi, ignorando eventuali gruppi di pixel troppo chiari. L'algoritmo ben si adatta ai segnali di divieto di sosta, in cui i livelli di grigio corrispondenti ai colori rosso e blu, sono molto somiglianti. In questo caso l'informazione è contenuta in almeno due colori, entrambi differenti dallo sfondo.



(a) *(b)* *(c)*

Figura 1.20: Metodi di ricerca dei valori massimi dell'istogramma.

Come descritto in precedenza, per la rimappatura dei valori dell'istogramma sono state principalmente utilizzate funzioni di trasferimento misto, con tratti lineari e non lineari. I dati dinamici, fondamentali per la costruzione della funzione di trasferimento, sono i due punti di minimo (*min*) e massimo (*max*) ovvero i due picchi dell'istogramma che rappresentano rispettivamente il bianco e il nero. Questi punti vengono forniti proprio dalla fase di ricerca dei massimi che, come descritto in precedenza, è personalizzata per ogni categoria. Successivamente avendo questi due valori, *min* e *max*, si passa alla fase di costruzione della funzione di *contrast stretching*.

La figura 1.21 mostra la funzione di trasferimento utilizzata per i segnali di divieto: l'implementazione del tratto parabolico nella parte più alta del grafico permette di comprimere i toni alti del grigio più velocemente al valore massimo. Questo permette di risolvere parzialmente il problema relativo alla presenza di contributi rumorosi dovuti al ritaglio impreciso o ai bordi del cartello: se i toni di grigio che rappresentano questi contributi rumorosi occupano la regione più alta del grafico, i loro valori vengono rimappati ai toni di grigio più vicini al bianco.

Applicando invece un tratto parabolico anche alla parte bassa del grafico si ottiene la compressione dei toni di grigio scuro della regione più bassa e l'esaltazione della regione intermedia. Questa funzione si adatta ai cartelli in cui lo sfondo è bianco e il contenuto nero: non si vuole infatti rischiare di convertire in un nero più scuro le parti sporche dell'immagine e compromettere quindi il risultato; sullo sfondo bianco il rischio di fare il contrario è invece minimo. Le immagini in ingresso al classificatore appaiono quindi migliori solo nel caso in cui gli elementi rumorosi presentino toni di grigio chiari: questo limita le performance di classificazione e da qui è nata la necessità di irrobustire la fase precedente che si occupa del ritaglio, piuttosto che utilizzare la calibrazione del contrasto per risolvere il problema dei contenuti rumorosi.

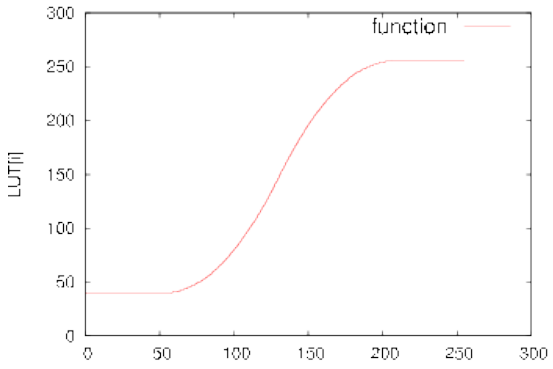


Figura 1.21: Funzione di trasferimento per segnali di divieto.

Le altre funzioni di *contrast stretching* utilizzate per ogni categoria sono:

- La funzione 1.22 *a* per i cartelli di indicazione e pericolo.
- La funzione 1.22 *b* per i cartelli di sosta e pericolo temporaneo.
- La funzione 1.22 *c* per i cartelli di stop e fine obbligo.
- La funzione 1.22 *d* per i cartelli di obbligo

Un esempio completo di funzionamento del *contrast stretching* è mostrato in figura 1.23: i toni di grigio del cartello di divieto individuato dall'algoritmo (figura 1.23 *a*) vengono rimappati tramite la funzione di trasferimento mista (figura 1.23 *b*) al fine di ottenere il campione (figura 1.23 *c*) da porre in ingresso alla rete neurale. Le figure 1.23 *c-d* illustrano gli istogrammi del cartello rispettivamente prima e dopo la calibrazione del contrasto. Mediante la trasformazione si vedono gli effetti desiderati, una concentrazione dei valori su i due estremi dell'istogramma e un appiattimento dei valori intermedi. La nuova distribuzione dà ai toni chiari una concentrazione maggiore: si vede mediante l'ordinata che va fino a 600. Essendo il colore dello sfondo in maggioranza, la necessità è di *pulire* il più possibile l'immagine al fine di render il suo istogramma somigliante a quello del modello di addestramento (figura 1.23 *f*).

1.4.3 Risoluzione problematiche nella calibrazione del contrasto

Come descritto in precedenza il metodo preesistente utilizza il *contrast stretching* sia per migliorare il contrasto delle immagini da porre in ingresso al classificatore che per eliminare il rumore introdotto dal ritaglio non preciso del cartello: in particolare quest'ultima scelta non si è dimostrata una strategia vincente come illustrato nelle figure 1.24 *a-b*.

Nel primo caso, come mostrato in dettaglio in figura 1.25 *b*, il risultato del *contrast stretching* è un segnale il cui contenuto informativo è poco contrastato rispetto allo sfondo ed, inoltre, nella parte inferiore dell'immagine è presente un contributo di rumore proveniente dal bordo del cartello; analizzando l'istogramma dell'immagine rimappata (figura 1.25 *b*) si nota sia la mancanza di un picco significativo in corrispondenza dei toni scuri che la concentrazione di numerosi picchi in corrispondenza dei valori alti: questi due fattori rendono l'immagine poco contrastata.

Nel secondo caso, come mostrato in dettaglio in figura 1.26 *b*, la calibrazione del contrasto non ha successo e viene generata un immagine rumorosa in cui il contenuto informativo non si trova su sfondo bianco ma grigio: dall'analisi dell'istogramma relativo all'immagine rimappata (figura 1.26 *b*) si riscontra infatti la presenza di numerosi picchi significativi in corrispondenza dei toni di grigio intermedi che rendono l'immagine risultante prevalentemente grigia.

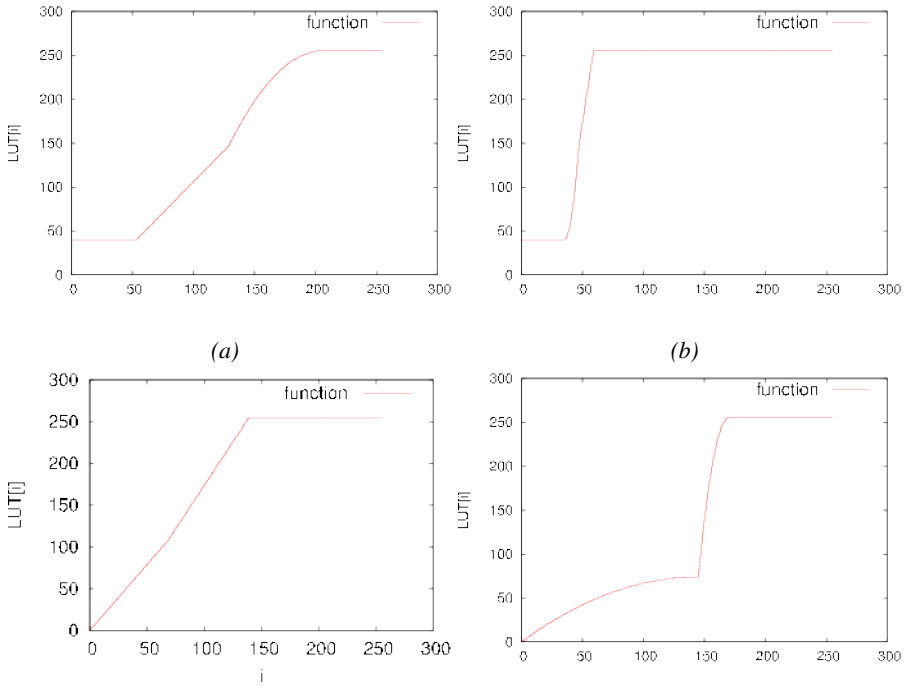


Figura 1.22: Funzioni di *contrast stretching* ottimizzate per *(a)* cartelli di indicazione e pericolo; *(b)* segnali di sosta e pericolo temporaneo; *(c)* cartelli di stop e fine obbligo; *(d)* segnali di obbligo.

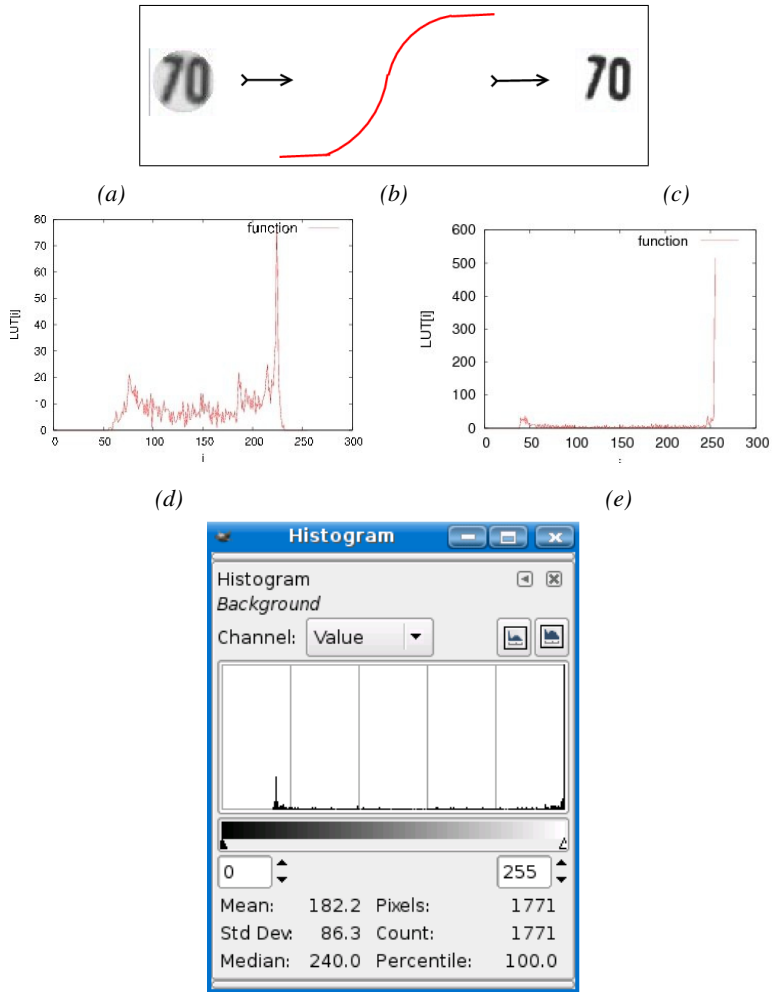
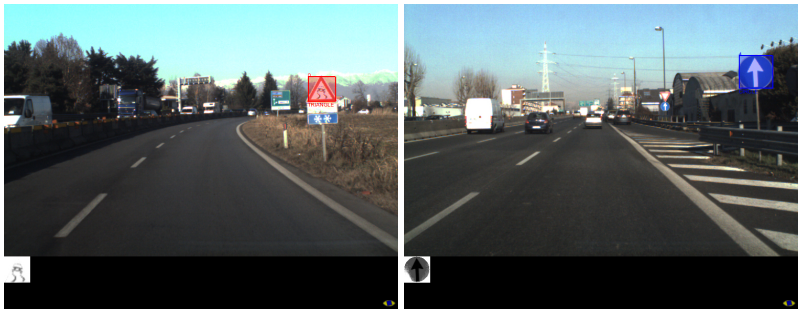
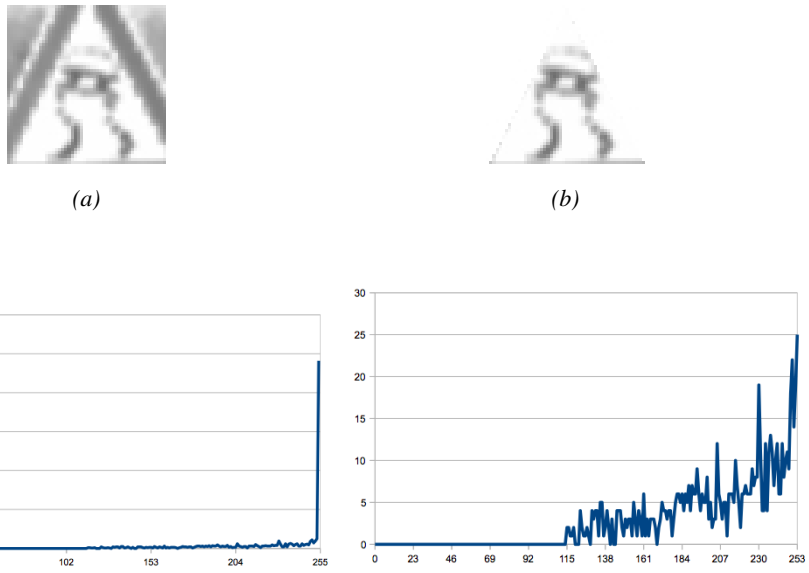


Figura 1.23: Esempio di *contrast stretching*; *(a)* immagine in ingresso al processo; *(b)* funzione di trasferimento con compressione dei toni di grigio alti e bassi; *(c)* immagine di uscita con contrasto calibrato; *(d)* istogramma a monte di *contrast stretching*; *(e)* istogramma a valle del *contrast stretching*; *(f)* istogramma originale preso dal modello di training set.



(a) *(b)*

Figura 1.24: *(a)-(b)* Esempi in cui il metodo preesistente di ricerca dei massimi all'interno dell'istogramma genera risultati rumorosi nella calibrazione del contrasto.



(a) *(b)*

(c) *(d)*

Figura 1.25: Input ed output dell'algoritmo preesistente e relativi istogrammi, ottenuti dall'applicazione del *contrast stretching* su di un segnale di pericolo.

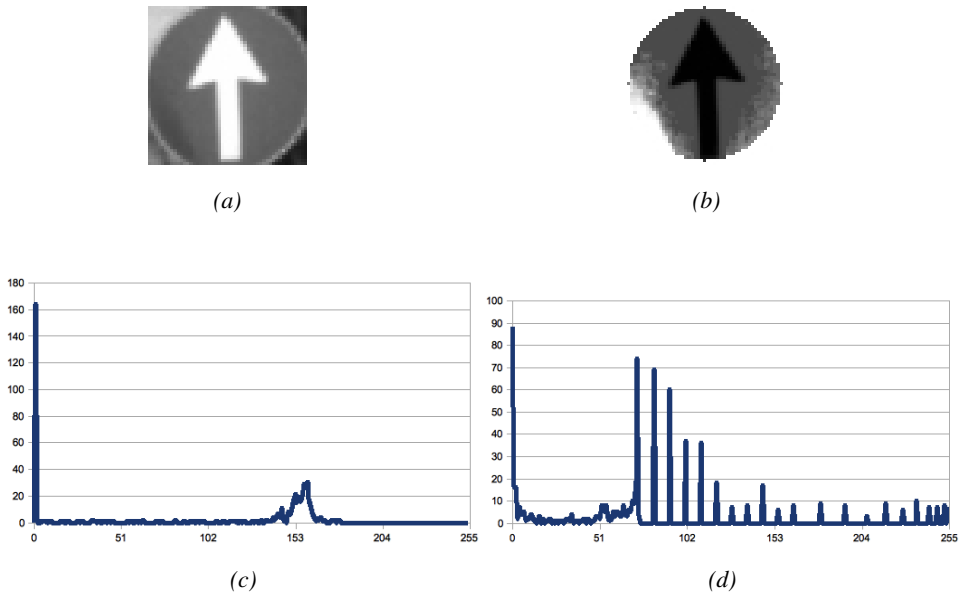


Figura 1.26: Input ed output dell'algoritmo preesistente e relativi istogrammi, ottenuti dall'applicazione del *contrast stretching* su di un segnale di obbligo.

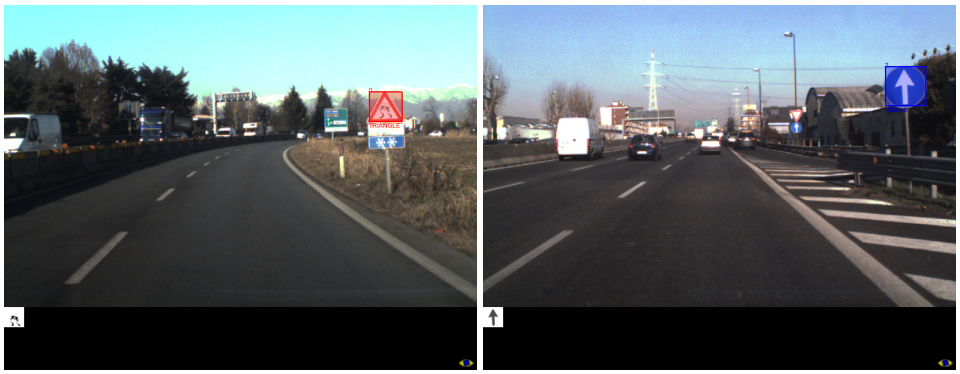
Da qui è nata la necessità di intervenire su due fronti: migliorare la fase di ritaglio delle immagini (come illustrato nel paragrafo precendente) e, nel contempo, irrobustire la calibrazione del contrasto. A tal proposito è stato sviluppato un nuovo algoritmo per eseguire il contrast stretching che, a differenza dell'approccio preceesistente, utilizza un unico metodo di ricerca dei massimi per tutti i cartelli: si calcola il valore medio dell'istogramma, e a partire da questo valore si ricerca il massimo assoluto, ovvero il picco che rappresenta il bianco. Il secondo massimo, corrispondente al nero viene invece ricavato a partire dal picco del bianco: questo perché dall'analisi delle sequenze di immagini acquisite è stato riscontrato che, per ciascun *bounding box* individuato, si mantiene pressoché costante il rapporto tra i *pixel* neri e bianchi del cartello inquadrato.

Con questa metodologia di ricerca si alleggerisce computazionalmente la procedura di *contrast stretching*, limitandola alla sola calibrazione del contrasto: infatti, grazie alle funzionalità introdotte, non ci di deve più preoccupare dell'eliminazione di eventuali componenti di rumore dovute a ritagli imprecisi.

I miglioramenti introdotti sono significativi, come mostrato nelle figure 1.28 *b* e 1.29 *b*: mantenendo le stesse funzioni di trasformazione dell'algoritmo precedente e modificando il metodo di ricerca dei massimi, si nota come il contenuto informativo di queste immagini sia ben contrastato rispetto il bianco dello sfondo, il quale non contiene più componenti di rumore. Gli istogrammi ottenuti (figure 1.28 *d* e 1.29 *d*) sono infatti ben distribuiti, ciascuno caratterizzato da due picchi significativi in corrispondenza dei valori che rappresentano, rispettivamente, il bianco e il nero; inoltre le componenti di grigio intermedie risultano attenuate, motivo per cui il bianco dello sfondo del cartello di uscita è uniforme e non presenta contributi rumorosi.

1.5 Classificazione

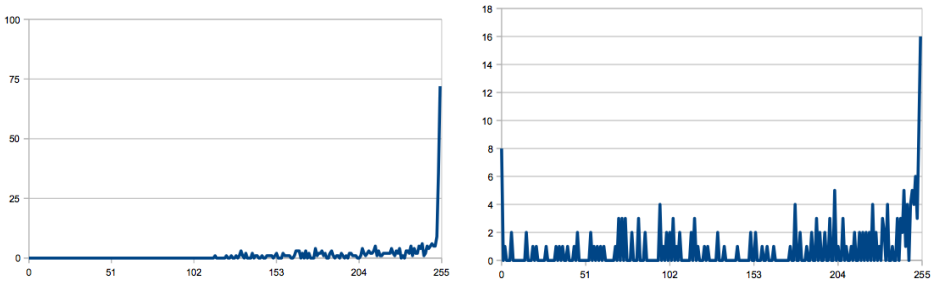
L'ultima fase del processo di riconoscimento dei segnali stradali consiste appunto nella classificazione, ovvero nell'identificazione precisa del segnale, all'interno di una specifica categoria.



(a) *(b)*

Figura 1.27: *(a)-(b)* Correzione del contrasto eseguita secondo il nuovo metodo di ricerca dei massimi.

(a) *(b)*



(c) *(d)*

Figura 1.28: Input ed output relativi al *contrast stretching* per il segnale di pericolo.

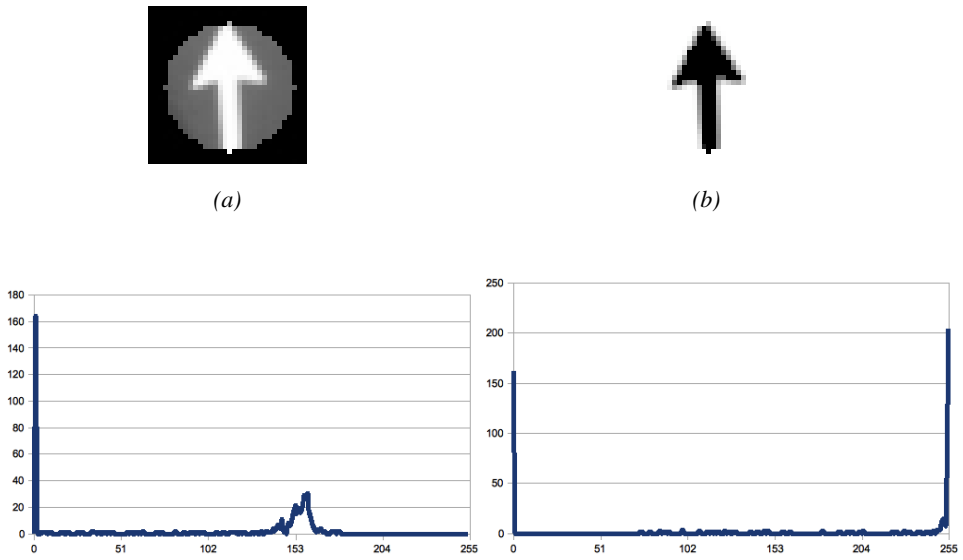


Figura 1.29: Istogrammi relativi al *contrast stretching* secondo il metodo corrente.

La classificazione si differenzia principalmente in base al numero di segnali che si deve riconoscere. Quando il numero di segnali da riconoscere è ridotto si fa uso indistintamente di *pattern matching* e di reti neurali; quando invece occorre riconoscere un vasto elenco di segnali, anche di categorie diverse, quasi tutti i lavori si orientano sull'utilizzo di reti neurali [96, 81, 48, 10].

1.5.1 Reti Neurali

Le Reti Neurali costituiscono un ottimo strumento per affrontare i problemi di classificazione, in quanto il loro funzionamento di base è simile all'apprendimento degli esseri umani, grazie all'accumulazione nel cervello di informazioni relative all'esperienza passata. Una rete neurale artificiale è costituita da uno strato di neuroni al livello di input della rete stessa e da uno o più livelli intermedi; si nota che, in realtà, l'ultimo strato (quello di output) fa parte convenzionalmente degli strati intermedi. Neuroni di strati diversi possono essere connessi tra loro osservando svariate topologie ed ogni connessione è caratterizzata da un peso, il cui ruolo è fondamentale nella generazione del risultato della rete stessa. La corretta classificazione di una classe si basa, quindi, sulla corretta impostazione del peso di ciascun collegamento presente all'interno della rete neurale: è ovviamente impossibile generare in un'unica iterazione dei pesi tali da garantire il corretto riconoscimento delle classi in input. A tal proposito sono quindi effettuate più iterazioni, al termine delle quali si varia il peso di ciascun collegamento in modo da migliorare via via le prestazioni di riconoscimento, minimizzando una funzione di errore. Ciascuna iterazione prende il nome di epoca. Per una spiegazione dettagliata del funzionamento di una rete neurale artificiale si rimanda alla consultazione dell'appendice A.

Nell'ambito di questo progetto per l'utilizzo delle reti neurali ci si affida ad una libreria esterna, la *Lightweight Neural Network Plus*, [29]. Il principale pregio di questa libreria è quello di permettere l'esecuzione dell'addestramento *off line*, separatamente dall'esecuzione.

Addestramento

La fase di addestramento, comune alla realizzazione di tutte le reti neurali, può essere schematizzata nel modo seguente:

- *Recupero dei modelli*: si considerano come modelli di addestramento i segnali stradali di dimensione 50×50 e 32×32 pixel ottenuti dal processo di adattamento delle immagini precedentemente descritto.

- *Progetto del training set*: per ciascuno dei modelli acquisiti si costruiscono due set di addestramento distinti: uno per i modelli di dimensione 50×50 e uno per i campioni di 32×32 pixel; questi verranno utilizzati per addestrare reti neurali diverse.

- *Adattamento del codice*: si adatta il *software* di addestramento fornito dalla libreria alla categoria di riferimento, stabilendo il numero di ingressi e uscite della rete.

- *Esecuzione*: si minimizza la funzione di errore agendo sui parametri *learning rate* e *momentum*; il prodotto finale è un file `.net` contenente la memoria della rete.

1.5.2 Applicazione delle reti neurali

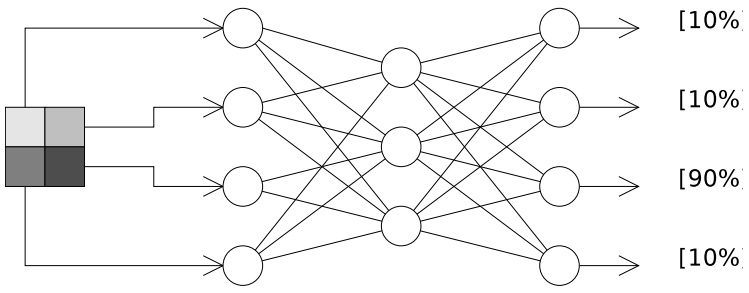


Figura 1.30: Definizione degli ingressi e delle uscite della Rete Neurale

Il primo passo per l'applicazione delle reti neurali al processo di riconoscimento di segnaletica stradale ha riguardato la definizione degli ingressi e delle uscite. In particolare, per il sistema implementato, l'ingresso della rete consiste in un *array* di pixel corrispondenti all'immagine a toni di grigio di dimensione fissa, ottenuta tramite il processo di adattamento delle immagini: se si utilizza il metodo preesistente si ottengono, come ingresso per la rete, immagini 50×50 pixel, ovvero 2500 unità, le quali definiscono un eguale numero di neuroni appartenenti allo strato di ingresso della rete neurale. Con il nuovo processo di elaborazione degli ingressi per il classificatore l'immagine di input ha invece dimensione 32×32 pixel, che equivalgono a 1024 neuroni per lo strato di ingresso della rete neurale. L'uscita che si richiede per la rete neurale è costituita da un altro *array* contenente le percentuali di somiglianza dell'input fornito con ognuno dei segnali appartenenti alla categoria di riferimento. Il numero di neuroni contenuti nello strato di uscita dipende, pertanto, dal numero di cartelli presenti nella suddetta categoria (figura 1.30). Si è inoltre scelto di non definire nodi di uscita dedicati ai casi di falsi positivi: ci si aspetta infatti che, in presenza di questa tipologia di campioni, la rete fornisca in uscita un valore molto basso, indice della poca somiglianza tra l'ingresso ed un cartello stradale.

Allo scopo di migliorare la capacità di riconoscimento si è scelto di utilizzare reti neurali multilivello con funzionamento *feed forward*; in particolare, sono state addestrate sei reti neurali diverse, una per ogni categoria di riferimento: divieto, obbligo, indicazione, pericolo, sosta, stop. Inoltre, limitatamente ai segnali di divieto, obbligo e pericolo, le reti neurali sono state utilizzate per confrontare le prestazioni tra il nuovo metodo di elaborazione degli ingressi e quello preesistente. Le reti neurali sono state addestrate tramite l'algoritmo di *back propagation* e, a supporto del *training set*, è stato introdotto un *validation set* al fine di limitare il problema dell'*overfitting* (appendice A).

Per le diverse classi di segnali stradali sono state addestrate e testate reti neurali con topologie differenti: in particolare si è mantenuto fisso il numero dei neuroni di ingresso, ponendolo uguale al numero dei pixel delle immagini di ingresso (50x50 per il metodo preesistente e 32×32 per la nuova tecnica introdotta). Variando il numero e la dimensione dei livelli intermedi della rete, per ciascuna categoria di riferimento

è stata scelta la rete con prestazioni migliori: si è visto come l'uso di pochi neuroni in genere non fornisce risultati soddisfacenti, mentre troppi nodi intermedi rischiano di generare situazioni di *overfitting*.

Per irrobustire il processo di riconoscimento si applica inoltre una procedura di *tracking* sui candidati classificati al fine di stabilizzare l'identificazione dei segnali tra i vari *frame* e nel contempo filtrare i casi di falsi positivi.

1.5.3 Tracking

L'impiego di tecniche di riconoscimento basate su conoscenza pregressa ha permesso di migliorare notevolmente i risultati, consentendo di risolvere problemi relativi all'errata classificazione di cartelli o alla presenza di falsi positivi rilevati per un solo *frame*. Nell'ambito del riconoscimento di segnaletica stradale l'obiettivo principale del *tracking* è quello di determinare dove si trova nel *frame* corrente un cartello precedentemente individuato e sfruttare questa informazione per stimare la posizione futura del candidato.

Prima di eseguire la classificazione, tutte le regioni rilevate nell'immagine corrente sono comparate con quelle classificate al ciclo precedente: il *tracking* viene quindi eseguito limitatamente alle regioni classificate con successo.

Si assume quindi di lavorare a regime e di avere a disposizione, ad ogni ciclo, una lista che contiene i dati relativi ai *bbox* classificati nei cicli precedenti:

- **id**: identificativo del *bounding box*;
- **nDetection**: valore che definisce quanti sono i *frame* in cui il *bounding box* è stato classificato;
- **oldness (vecchiaia)**: da quanti cicli non compare un *bbox* riconosciuto in passato; una nuova rilevazione porta *oldness* a zero, ringiovanendo il *bbox*;
- **size**: dato relativo alle dimensioni del *bbox* (ovvero il lato del quadrato che lo rappresenta);
- **lastDetectedCenter**: centro dell'ultimo *bbox* rilevato in un *frame*;

- **velocity**: stima della velocità di spostamento (in pixel) del centro del *bbox* nell'immagine;
- **extimatedCenter**: è il centro stimato per il *bounding box*.

All'inizio di ogni ciclo si aggiornano i dati relativi all'*oldness* e al centro stimato per ogni *bounding box* (calcolato come somma tra il centro del *bbox* stimato al ciclo precedente e la velocità) contenuto nella lista che descrive la "storia passata". Per poter effettuare i confronti e calcolare correttamente le stime è necessario capire dove si trova, al *frame* corrente, un *bounding box* classificato in passato e confrontarlo con ogni *bounding box* rilevato nel ciclo corrente; per questa ragione sono stati valutati tutti i possibili accoppiamenti tra *bounding box* correnti e *bounding box* classificati nel passato. Ad ogni coppia è assegnato un "voto", stabilito sulla base del rapporto tra il quadrato dell'area di sovrapposizione dei due *bbox* e il prodotto tra le rispettive aree. La formula per l'assegnazione del voto è la seguente:

$$voto = \frac{A_{overlap}}{A_{currentBbox} \times A_{pastBbox}} \tag{1.4}$$

La scelta dei *match* è fatta scegliendo via via le coppie di *bounding box* con voto più alto; alla fine, se i *bounding box* del *frame* corrente non sono stati accoppiati, significa che non erano stati riconosciuti al ciclo precedente e quindi i loro dati devono essere aggiunti alla lista che verrà passata al prossimo ciclo. La presenza di un *match* indica invece che un elemento classificato nel *frame* precedente è di nuovo presente in quello corrente, quindi nella lista devono essere opportunamente aggiornati i suoi dati:

- il numero di rivelazioni;
- la dimensione del *bbox*;
- il centro (sia stimato che effettivo);
- la velocità, calcolata come rapporto tra lo spostamento del centro del *bbox*, rispetto al suo ultimo centro rilevato, e la sua *oldness* ($\frac{spostamento}{tempo}$);

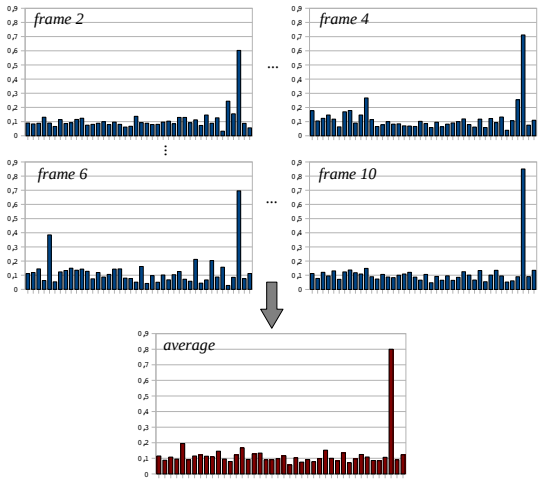


Figura 1.31: Applicazione del *tracking* alle reti neurali; gli output relativi al riconoscimento di uno stesso segnale rilevato per 10 *frame* sono rappresentati su diversi istogrammi di cui si calcola la media pesata al fine di limitare la variabilità degli output.

- l'*oldness*, che è portata a zero.

Gli elementi con *oldness* maggiore o uguale di due sono eliminati dalla lista perché un elemento non riconosciuto per due *frame* consecutivi è ritenuto obsoleto.

Sono invece visualizzati i *bbox* rilevati più di due volte ($nDetection \geq 2$). In questo modo se l'algoritmo non riconosce nel *frame* corrente un segnale, grazie al tracking implementato, si riesce comunque a stimare la posizione del cartello se in due dei tre *frame* precedenti era stato individuato il relativo *bbox*. Inoltre il segnale non riconosciuto può essere riassociato correttamente al ciclo successivo. Questo metodo permette inoltre di eliminare falsi positivi rilevati per un solo *frame*, per i quali *nDetection* non raggiunge il valore 2.

Il risultato del *tracking* contribuisce inoltre alla determinazione della classe di appartenenza del segnale individuato: a partire dai *match* tra candidati correnti e precedenti stabiliti tramite il *tracking* si calcola, in base a ciascun accoppiamento, la

media pesata tra l'output che la rete neurale fornisce ricevendo in ingresso il campione precedente e l'output che la rete neurale fornisce ricevendo in ingresso il campione corrente. In questo caso le reti neurali sono quindi interpretate come funzioni di trasferimento caratterizzate da n output $O_0 \dots O_n$; si considerano questi output rappresentati su di un istogramma sulle cui ascisse si trovano le possibili classi di appartenenza del campione in ingresso e sulle ordinate un voto che rappresenta la percentuale di somiglianza tra il campione di ingresso e la specifica classe in ascissa. Ad ogni frame si calcola la media tra l'output della rete neurale corrente e l'output di quella precedente $A_0 \dots A_n$, usando come peso w; questo peso equivale alla dimensione di una regione sull'istogramma dei voti, graficamente centrata sulla classe di appartenenza stabilita per il campione al passo precendente: questa regione permette di dare maggior peso alle classi più vicine a quella precedentemente scelta e meno peso a quelle più lontane.

$$\begin{aligned} A'_i &= A_i + O_i * w \\ W' &= W + w \end{aligned} \tag{1.5}$$

La classe assegnata al candidato è quindi quella che nell'istogramma presenta il voto maggiore:

$$class_id = \max_i A_i \tag{1.6}$$

Quando l'area delimitata dal *bbox* di un segnale esce dai limiti dell'immagine o quando un candidato è stato rilevato come *ghost* per più di cinque frame, il processo di riconoscimento prevede di assegnare una classe al candidato rilevato soltanto se il picco dell'istogramma dei voti A_{id}/W supera una certa soglia ξ, altrimenti si considera il candidato come falso positivo.

Un esempio di applicazione del *tracking* alle reti neurali è mostrato in figura 1.31: l'istogramma identificato con *average* rappresenta la media pesata tra tutti gli output ottenuti applicando la rete neurale ad uno stesso cartello rilevato in *frame* differenti.

1.5.4 Classificatore *AdaBoost*

Per estendere le funzionalità di classificazione si è scelto di addestrare un classificatore di tipo *AdaBoost* allo scopo di confrontare le prestazioni di riconoscimento dei

due metodi di elaborazione degli ingressi ed effettuare inoltre un'analisi comparativa rispetto le reti neurali.
Il classificatore *AdaBoost* prevede di utilizzare, per definizione, un insieme di classificatori deboli (*weak classifier*) tenuti a fornire un voto in output; tale voto ha la funzione di indicare quale sia la classe riconosciuta come corretta dal classificatore stesso. La modalità di votazione è per maggioranza: ogni classificatore debole genera un voto per ogni classe e nella fase finale, sommando i voti ottenuti da ciascuna classe, è possibile determinare l'etichetta vincitrice ovvero quella che, secondo il classificatore, si avvicina maggiormente alla classe di appartenenza dell'input. Esistono inoltre più modalità di generazione del voto per ciascun *weak classifier*, con possibilità di astensione.
Come per le reti neurali, anche per il classificatore *AdaBoost* utilizzato nell'ambito di questa tesi si fa riferimento ad una libreria esterna, la *multiboost* sviluppata da Norman Casagrande [28, 14]. Questa libreria, come la *Lightweight Neural Network Plus*, consente di eseguire separatamente la fase di addestramento da quella di esecuzione e, analogamente alle reti neurali, si è scelto di porre in ingresso al classificatore *AdaBoost* l'*array* di pixel corrispondenti all'immagine a toni di grigio di dimensione fissa (50x50 o 32×32 pixel), ottenuta tramite il processo di adattamento delle immagini. In uscita il classificatore restituisce la somma dei voti che ogni *weak classifier* assegna a ciascuna tipologia presa in considerazione compatibilmente con la categoria di appartenenza del segnale individuato.
Per l'implementazione dell'algoritmo di *boosting* le specifiche prevedono l'uso delle *features di Haar* come classificatori deboli. L'algoritmo di classificazione si basa sulla versione *AdaBoost.*MH discreta (per maggiori dettagli si rimanda all'appendice B).
Il confronto tra le prestazioni delle reti neurali ed il classificatore *AdaBoost* è stato limitato ai segnali di divieto, obbligo e pericolo, ma può essere esteso alle altre categorie di cartelli esistenti.

1.6 Presentazione dei risultati

1.6.1 Analisi del rumore all'interno delle immagini

Dall'analisi delle sequenze di immagini acquisite è emersa l'affidabilità del nuovo algoritmo introdotto: a differenza di quello preesistente si è dimostrato particolarmente robusto in presenza di cartelli ruotati:

- orizzontalmente (figura 1.32 *a*);
- verticalmente (figura 1.32 *c*);
- sia orizzontalmente che verticalmente (figura 1.32 *d*).

In particolare si nota come i ritagli di dimensione 32×32 *pixel* (figura 1.32 *c-f-i*), ottenuti dall'applicazione del nuovo algoritmo, non presentano contributi di rumore, a differenza delle immagini di dimensione 50×50 *pixel* risultanti dal metodo preesistente (figura 1.32 *b-e-h*)

Inoltre i disallineamenti e le traslazioni dovuti a *bounding box* non perfettamente centrati sul segnale inquadrato (figura 1.33 *a-c-g*) non comportano più la generazione di rumore all'interno delle immagini da porre in ingresso al classificatore (figura 1.33 *c-f-i*). In particolare quando il *bounding box* occupa un'area minore (figura 1.33 *a*) o maggiore (figura 1.33 *d*) rispetto quella del segnale, le immagini ottenute tramite l'applicazione del nuovo algoritmo (figura 1.33*c-f-i*) non presentano contributi rumorosi a differenza di quelle risultanti dall'approccio preesistente (figura 1.33 *b-e-h*).

L'algoritmo implementato ha inoltre dimostrato di funzionare correttamente anche nel caso in cui le condizioni di illuminazione rendono critica l'individuazione dell'area relativa al contenuto informativo, ad esempio quando i cartelli si presentano ombreggiati (figura 1.34 *a-b*) o in controluce (figura 1.34 *c-d*).

1.6.2 Analisi sulle prestazioni di classificazione

Nell'ambito di questa tesi sono state addestrate e testate diverse reti neurali al fine di valutare le performance di riconoscimento dell'algoritmo di identificazione dei segnali stradali e confrontare le performance tra il nuovo metodo di elaborazione degli

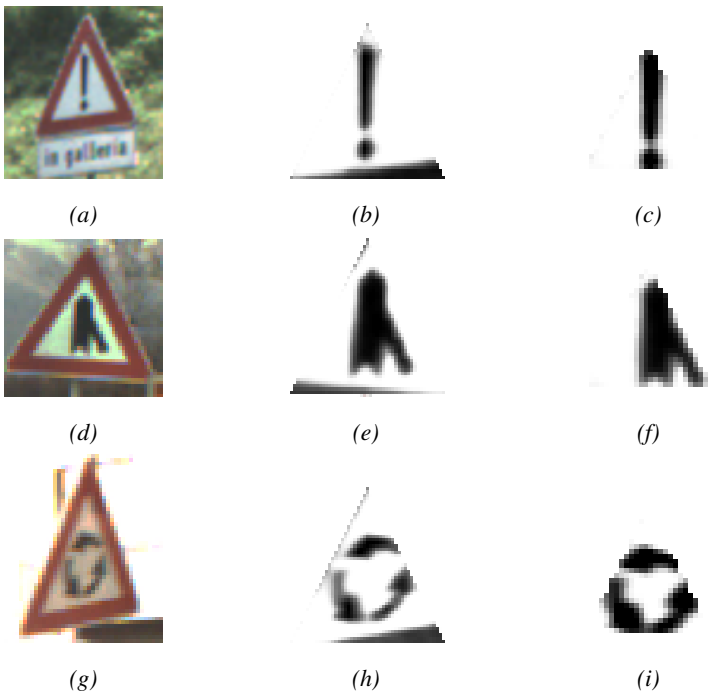
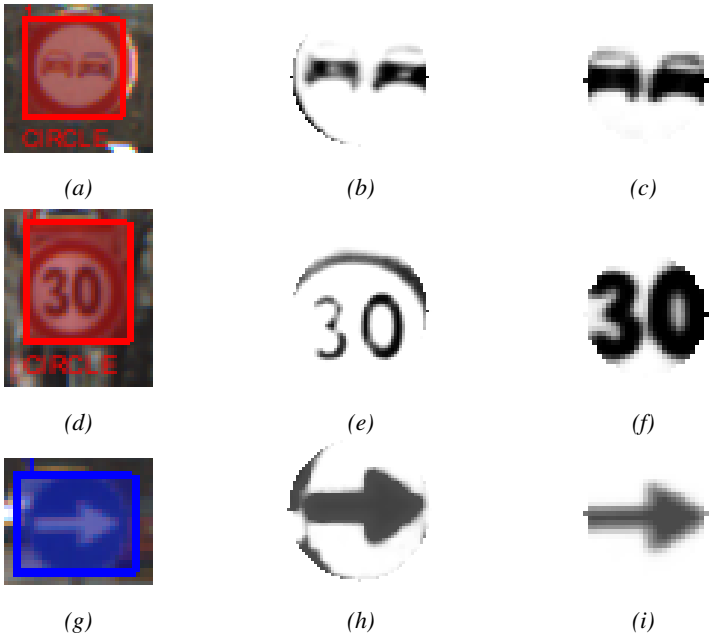


Figura 1.32: Confronto, in presenza di segnali ruotati, tra i risultati ottenuti applicando il preesistente approccio e il nuovo algoritmo per l'elaborazione degli ingressi del classificatore.

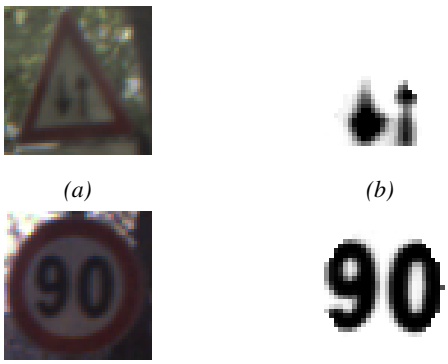


(a) *(b)* *(c)*

(d) *(e)* *(f)*

(g) *(h)* *(i)*

Figura 1.33: Confronto, in presenza di *bounding box* disallineati, tra approccio preesistente e nuovo metodo per l'elaborazione degli ingressi per il classificatore.



(a) *(b)*

(c) *(d)*

Figura 1.34: Casi limite e relative immagini 32×32 da porre in ingresso al classificatore. In particolare: *(a-b)* segnale in ombra , *(c-d)*, cartello in controluce.

ingressi per il classificatore e quello preesistente; sono state inoltre valutate le prestazioni di riconoscimento del classificatore *AdaBoost*, comparando i risultati ottenuti con quelli relativi alle reti neurali. Le categorie di reti neurali artificiali, utilizzate per condurre la ricerca, sono principalmente di due tipi: ad uno e due livelli di neuroni nello strato intermedio. L'analisi dei risultati è avvenuta nella fase finale di entrambi i metodi di apprendimento, mediante l'utilizzo delle curve ROC, grafici a dispersione che rappresentano l'andamento dei corretti riconoscimenti in funzione degli errori di riconoscimento, al variare di una certa soglia di funzionamento. La soglia di funzionamento stabilisce il limite al di sopra del quale il risultato del classificatore è ritenuto affidabile: se la soglia è bassa significa che molti candidati, anche quelli poco somiglianti ad un cartello stradali, vengono comunque considerati validi, quindi si otterranno molti rilevamenti corretti, ma anche molti errori; quando invece la soglia è alta pochi candidati vengono considerati validi, quindi si riduce il numero di errori, ma anche di rilevamenti corretti. Le curve ROC descrivono graficamente le performance di vari classificatori, permettendo di confrontali senza definire esplicitamente diverse soglie per compararli. La zona di valutazione interessante per tali curve è il cosiddetto punto di ginocchio, cioè la zona in cui siamo disposti ad accettare un nu-

mero piuttosto rilevante di corretti riconoscimenti a discapito di un numero modesto di errori di riconoscimento. Inoltre il confronto tra diverse curve ROC generate per ciascuna categoria di rete andrà effettuato esattamente in quella zona, poiché è quella dalla quale ci si aspettano i migliori risultati a fronte di un livello di funzionamento accettabile. Per la valutazione delle reti sono considerati errori di riconoscimento sia le regioni erroneamente associate a segnali stradali che i candidati classificati in modo sbagliato.

Come si è accennato in precedenza, i segnali sono stati suddivisi con criterio tra gli insiemi di addestramento e di validazione, anche in base alla loro qualità (dipendente principalmente dalla nitidezza, dal fatto che l'immagine appaia o meno intera oppure distorta). In modo particolare sono stati redatti complessivamente quattro insiemi che definiscono la qualità del segnale e che rappresentano rispettivamente:

1. I modelli dei segnali in cui l'immagine è ottima, contraddistinta quindi da buona nitidezza e in cui il segnale compare dritto o quasi.

2. I modelli in cui è possibile riconoscere distintamente il segnale al quale si riferiscono, ma la qualità dell'immagine è scarsa e il segnale rappresentato è parzialmente oscurato, saturato e comunque difficile da classificare per un umano. Per questo insieme soltanto i casi di segnali classificati erroneamente contribuiscono alla determinazione dell'errore sulle statistiche, mentre i casi di falsi negativi non vengono considerati: si ritiene infatti che, limitatamente a questo insieme, un'errata classificazione sia più grave rispetto ad una mancata individuazione del segnale.

3. I modelli in cui non è nemmeno possibile intuire quale fosse il segnale dal quale sono stati tratti.

4. I modelli dei quali non è possibile intuire quale fosse il segnale d'origine, ma che presentano alcune regolarità (*pattern*) tipiche di certe tipologie di segnale, pur non potendo riconoscerlo.

Di questi, al *training set* sono stati designati esclusivamente i modelli di segnale di qualità migliore, appartenenti alla prima categoria; il *validation set* inve-

ce è stato formato con segnali appartenenti a tutte e quattro le categorie precedenti, a patto di non avere classi presenti nell'insieme di validazione e assenti in quello d'addestramento.

Definizione dell'insieme di addestramento

L'abilità di un classificatore nel riconoscere con successo un particolare soggetto, dipende strettamente da come esso viene addestrato. Se il soggetto in questione si ripresenta ogni volta uguale a se stesso, è ragionevole addestrare il classificatore fornendogli un solo esempio per ogni individuo da riconoscere. Purtroppo l'ambiente stradale è mutevole e imprevedibile e i segnali si posso presentare ogni volta con notevoli differenze. La destrezza con cui il classificatore riconosce i segnali risulta inevitabilmente compromessa, se lo si addestra con un solo esempio per individuo. La possibilità di fornire da esempio al classificatore molte varianti degli stessi individui, incrementa la capacità di reintegrare le informazioni in presenza di rumore. La ricerca di un buon *training set* diventa dunque indispensabile.

Come descritto in precedenza, per quanto riguarda la fase di addestramento si è scelto di utilizzare, come modelli per il *training set*, i *pattern* ottenuti dal processo di adattamento delle immagini per l'elaborazione degli ingressi per il classificatore. Si utilizzano quindi le immagini dei cartelli inquadrati all'interno delle sequenze video acquisite per addestrare, su casi reali, sia le reti neurali che i classificatore *AdaBoost*. Rilevante è inoltre il metodo con il quale i modelli dei segnali sono stati suddivisi tra *training set* e *validation set*, evitando la presenza di modelli appartenenti alla stessa sequenza di *frame* sia nel *training set* che nel *validation set*. Poiché ogni singolo segnale sulla strada non viene rilevato unicamente su un *frame*, ma su più *frame* successivi, gli stessi possono anche comparire in numero relativamente elevato e se la suddivisione dei segnali tra gli insiemi di addestramento e validazione avviene in modo completamente casuale, è possibile che copie dello stesso segnale appartengano contemporaneamente sia al *training set* che al *validation set*. Validando quindi una rete, giunta al termine dell'addestramento, con gli stessi modelli con i quali essa è stata addestrata, potrebbe agevolare la rete stessa al riconoscimento di un maggior numero di segnali, producendo, per questo, risultati falsati. Nel corso della sperimen-

tazione descritta si è evitato, per quanto possibile, di incorrere in questa evenienza, dedicando quindi intere sequenze al solo insieme d'addestramento oppure a quello di validazione.

Per confrontare le prestazioni tra il nuovo metodo di elaborazione degli ingressi per il classificatore e quello preesistente sono stati definiti due *training set* diversi per ogni categoria di riferimento, costituiti rispettivamente da immagini di 32×32 pixel e 50×50 pixel. Si definisce inoltre, a partire dalle immagini ottenute, un set di validazione per evitare il problema critico dell'*overfitting* che porta la rete a specializzarsi troppo sugli esempi forniti, perdendo quindi il suo potere di generalizzazione.

La tabella 1.1 mostra le caratteristiche del database utilizzato per l'addestramento ed il testing dei classificatori. È possibile notare come data una categoria, il numero di *pattern* usati per il *training set* ed il *validation set* è diverso a seconda che si utilizzino le immagini 32×32 pixel o 50×50 pixel: ciò accade perché nell'elaborazione preesistente la calibrazione del contrasto non funziona sempre, ma fallisce quando i massimi trovati non sono sufficientemente distanti. Il metodo preesistente quindi, data l'immagine di ingresso relativa ad un cartello reale, non garantisce sempre l'elaborazione di un'immagine d'uscita da utilizzare come ingresso al classificatore o come modello per l'addestramento.

	Divieto		**Pericolo**		**Obbligo**	
	32×32	**50×50**	**32×32**	**50×50**	**32×32**	**50×50**
Trainign Set	2395	2365	1387	1204	791	560
Validation set	6292	5404	2272	2092	2913	3083
Numero Classi	20		25		11	

Tabella 1.1: Caratteristiche del database utilizzato per l'addestramento e il testing dei classificatori.

Nelle seguenti immagini sono riportate spesso, in didascalia, le parole *set9, set10 e set13*: esse rappresentano una nomenclatura più formale per indicare l'insieme dei segnali su cui le reti sono state sperimentate e al quale le curve ROC si riferiscono.

In particolare il *set9* è costituito da tutti i modelli del *validation set*; al *set10* appartiene la totalità dei segnali del *validation* e *training set*, mentre il *set13* si riferisce ai modelli in cui l'immagine è ottima.

Processo di specializzazione di una rete

Una prima sperimentazione condotta sulle reti neurali ha riguardato la stima del momento in cui una rete inizia a specializzarsi, limitando e successivamente perdendo la capacità di riconoscere segnali non appartenenti all'insieme sul quale è stata addestrata. Il principio secondo il quale la prova è stata condotta ha previsto di disabilitare il *validation set* in modo da evitare qualsiasi uscita forzata prima della convergenza verso l'errore minimo.

Si è poi incrementato di volta in volta il numero di epoche d'addestramento ad intervalli regolari di 50.000, fino al raggiungimento del numero massimo di 250.000. Per rendere sensata la prova è stato necessario caricare in input, passo dopo passo, la rete precedentemente addestrata: noto che, all'inizio dell'addestramento, la distribuzione di pesi sui collegamenti è del tutto casuale, non sarebbe stato possibile ottenere risultati veritieri semplicemente facendo ripartire daccapo la prova e limitandosi ad aumentare il numero di epoche. La rete scelta per condurre la prova è stata quella caratterizzata da 65 neuroni nello strato intermedio (quindi ad un solo livello interno), addestrata sul formato 32×32 dei segnali di pericolo. Aumentando il numero di iterazioni i risultati della rete sembrano migliorare riconoscendo sempre meglio l'insieme di campioni: in realtà la rete inizia a specializzarsi, ovvero si riduce l'errore sul *training set*, mentre quello sul *validation set* inizia a crescere.

Una rete poco addestrata e conseguentemente poco specializzata riconosce correttamente un limitato numero di campioni, mantenendo relativamente basso anche il numero di errori. Man mano che aumenta il numero di epoche, la rete tende a migliorare le proprie performance di riconoscimento, ma superato un certo limite di epoche di addestramento tende a specializzarsi sempre di più e, per questo, sarà disposta a riconoscere un maggior numero di campioni nel *training set* a fronte di peggiori performance sul *validation set*. Per affrontare questo problema si prevede

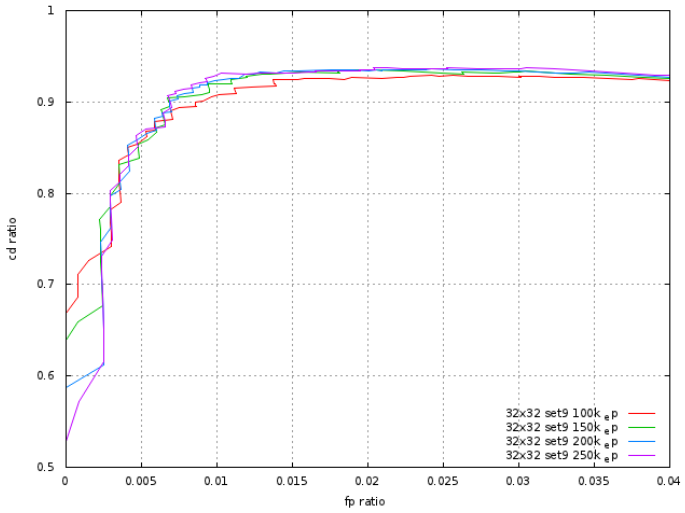


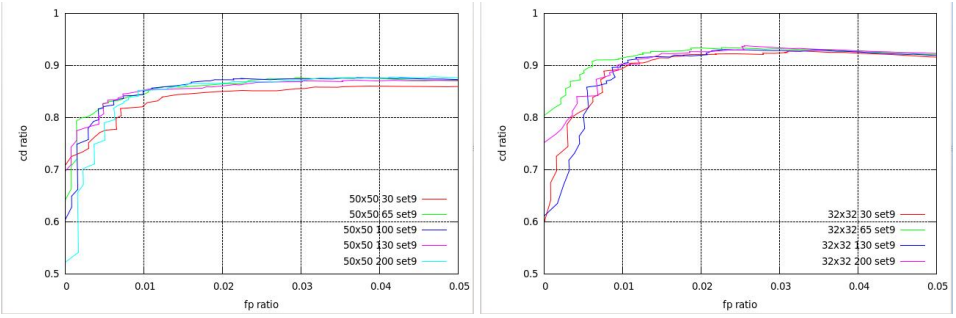
Figura 1.35: Esempio di overfitting sulla rete di pericolo ad un singolo livello intermedio con 65 neuroni, addestrata sul formato 32×32 dei modelli di alta e bassa qualità.

di interrompere l'addestramento quando l'errore sull'insieme di validazione tende a crescere oltre una certa soglia.

Addestramento di reti ad un livello di neuroni nello strato intermedio

Nel presente paragrafo si analizzeranno i risultati ottenuti mediante l'addestramento di reti neurali con un solo strato di neuroni al livello intermedio. In particolare, per le categorie di pericolo, divieto e obbligo sono stati utilizzati per l'addestramento sia i modelli di dimensione 50×50 pixel, ottenuti dal preesistente algoritmo, che i campioni grandi 32×32 pixel, ottenuti tramite le nuove funzionalità introdotte.

In figura 1.36 sono mostrati i risultati ottenuti per il riconoscimento dei segnali di pericolo: analizzando, per ciascuna funzione, la zona in corrispondenza del ginocchio si evince che le migliori prestazioni si ottengono con 100 neuroni nello strato intermedio, per le reti addestrate con i modelli 50×50 pixel e con 65 neuroni per le reti addestrate utilizzando le nuove funzionalità introdotte. Dal confronto diretto tra le migliori reti relative all'algoritmo preesistente e a quello introdotto (figura 1.37), emerge che il nuovo algoritmo permette di ottenere risultati di classificazione migliori.



(a) *(b)*

Figura 1.36: Risultati ottenuti dall'addestramento delle reti neurali ad un solo livello intermedio per il riconoscimento dei segnali di pericolo, tramite l'utilizzo di modelli in formato 50×50 e 32×32.

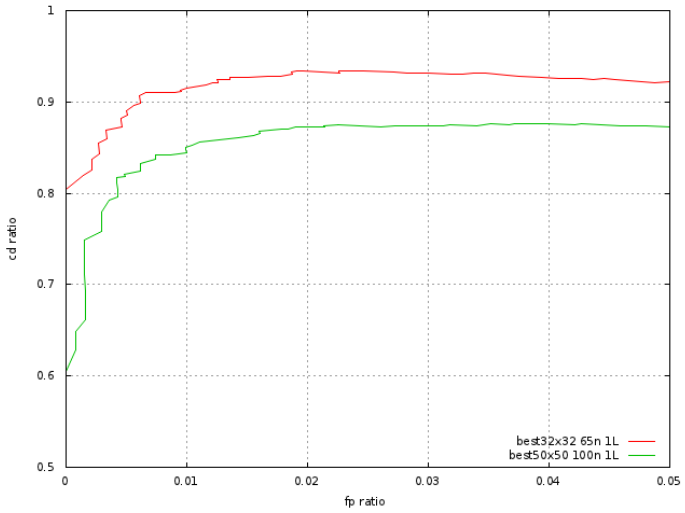


Figura 1.37: Confronto tra le migliori reti di pericolo ad un solo livello addestrate con i modelli in formato 50×50 e 32×32.

Anche per il riconoscimento dei segnali di divieto e di obbligo si evidenzia un significativo miglioramento grazie alle nuove funzionalità sviluppate: come mostrato nelle figure 1.38 e 1.39, per entrambe le categorie di segnali, le reti addestrate sui nuovi modelli comportano prestazioni di classificazione migliori.

Addestramento di reti a due livelli di neuroni nello strato intermedio

Per quanto riguarda l'addestramento di reti a due livelli, nelle figure 1.40 e 1.42 sono mostrate le curve ROC relative al riconoscimento dei segnali di pericolo e di divieto; confrontando, per entrambe le categorie, le migliori reti addestrate sui modelli di dimensione 50×50 pixel e 32×32 pixel (figura 1.41 e 1.43) si dimostra come, anche per queste tipologie di reti, il nuovo algoritmo comporti prestazioni migliori.

Medesimo discorso per la classificazione dei segnali di obbligo (figura 1.44) in cui l'addestramento sui nuovi modelli a 32×32 pixel genera risultati migliori rispetto quelli ottenuti sui campioni di dimensione 50×50 pixel.

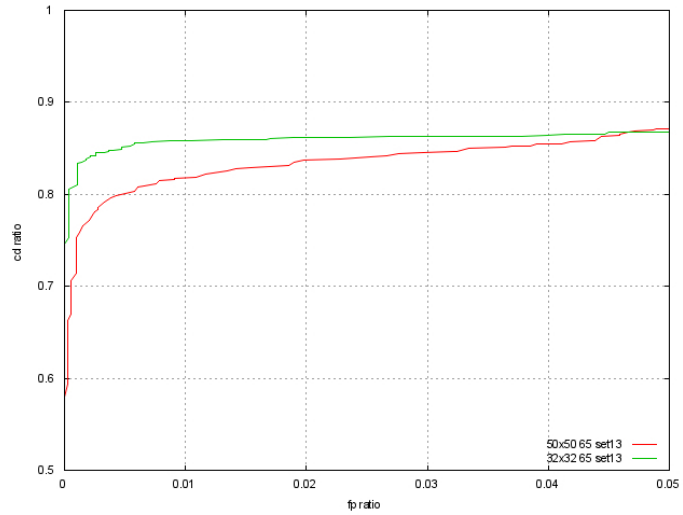


Figura 1.38: Confronto tra le migliori reti di divieto ad un solo livello addestrate con i modelli in formato 50×50 e 32×32.

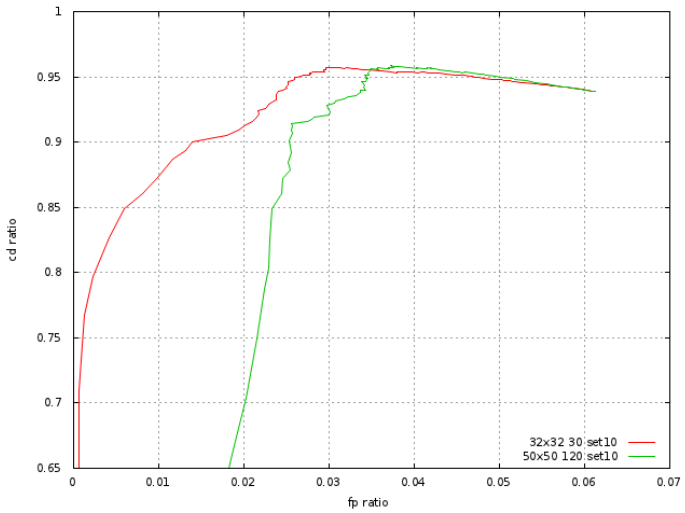
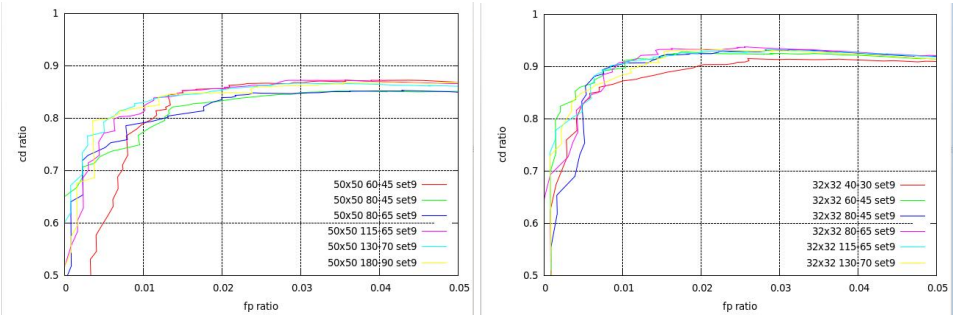


Figura 1.39: Confronto tra le migliori reti di obbligo ad un solo livello addestrate con i modelli in formato 50×50 e 32×32.



(a) *(b)*

Figura 1.40: Risultati ottenuti dall'addestramento delle reti neurali a due livelli intermedi per il riconoscimento dei segnali di pericolo, tramite l'utilizzo di modelli in formato 50×50 e 32×32.

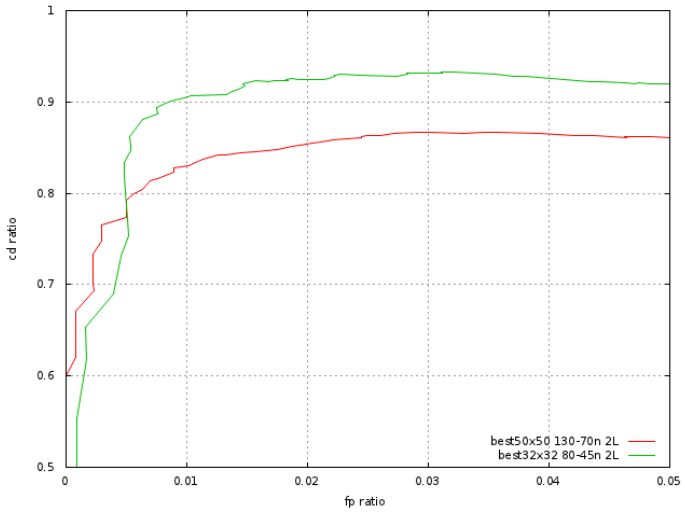
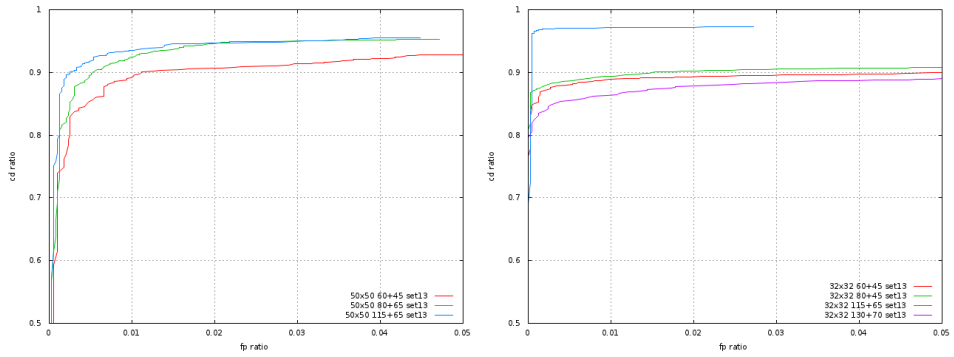


Figura 1.41: Confronto tra le migliori reti di pericolo a due livelli addestrate con i modelli in formato 50×50 e 32×32.



(a) *(b)*

Figura 1.42: Risultati ottenuti dall'addestramento delle reti neurali a due livelli intermedi per il riconoscimento dei segnali di divieto, tramite l'utilizzo di modelli in formato 50×50 e 32×32.

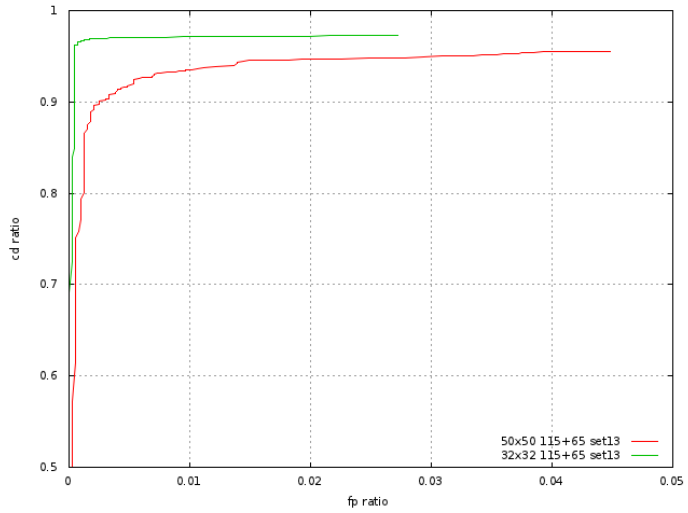


Figura 1.43: Confronto tra le migliori reti di divieto a due livelli addestrate con i modelli in formato 50×50 e 32×32.

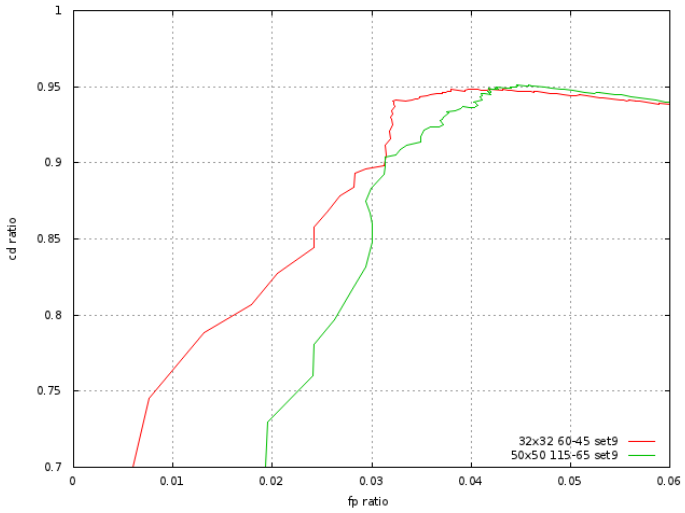


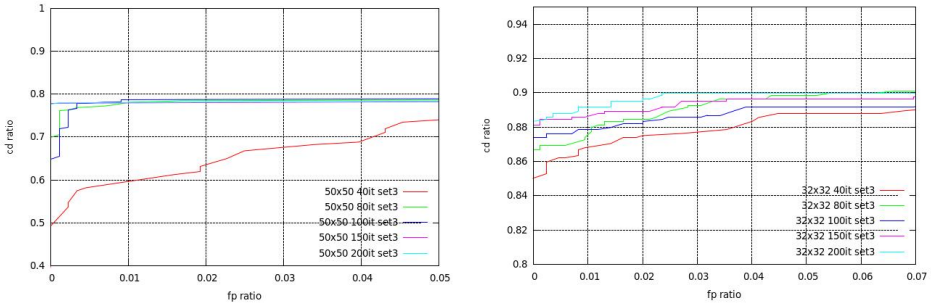
Figura 1.44: Confronto tra le migliori reti di obbligo a due livelli addestrate con i modelli in formato 50×50 e 32×32.

Classificazione mediante ADA-boost

Come detto in precedenza, un'ulteriore sperimentazione ha riguardato l'analisi delle prestazioni di riconoscimento dell'algoritmo tramite il classificatore *AdaBoost*: in questo caso l'addestramento ha permesso sia di ottenere un confronto tra il nuovo algoritmo introdotto e quello preesistente, che, al contempo, un'analisi comparativa tra il classificatore *AdaBoost* e le reti neurali.

I risultati ottenuti nell'ambito il riconoscimento dei segnali di pericolo sono mostrati in figura 1.45: considerando le migliori curve ROC addestrate rispettivamente sui modelli 50×50 pixel e 32×32 pixel (figura 1.46), si dimostra come il nuovo algoritmo risulta essere, anche in questo caso, più performante rispetto quello preesistente.

Lo stesso risultato è stato ottenuto anche per i segnali di divieto e di obbligo come mostrato in figura 1.47.



(a) *(b)*

Figura 1.45: Risultati ottenuti dall'addestramento del classificatore *AdaBoost* per il riconoscimento dei segnali di pericolo, tramite l'utilizzo di modelli in formato 50×50 e 32×32.

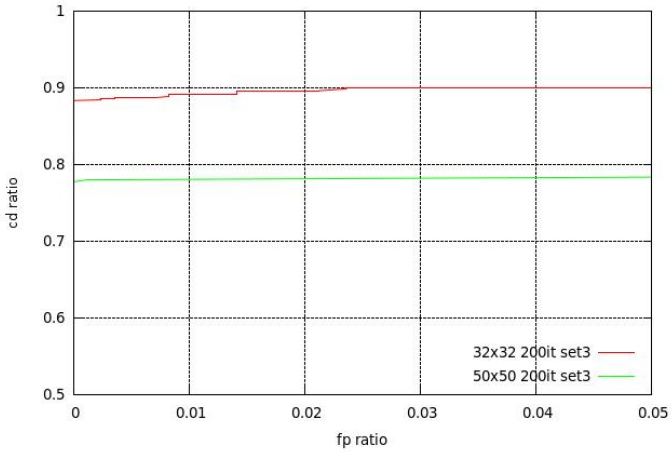


Figura 1.46: Confronto tra le migliori curve ROC ottenute addestrando il classificatore *AdaBoost* per riconoscere i segnali di pericolo con i modelli in formato 50×50 e 32×32.

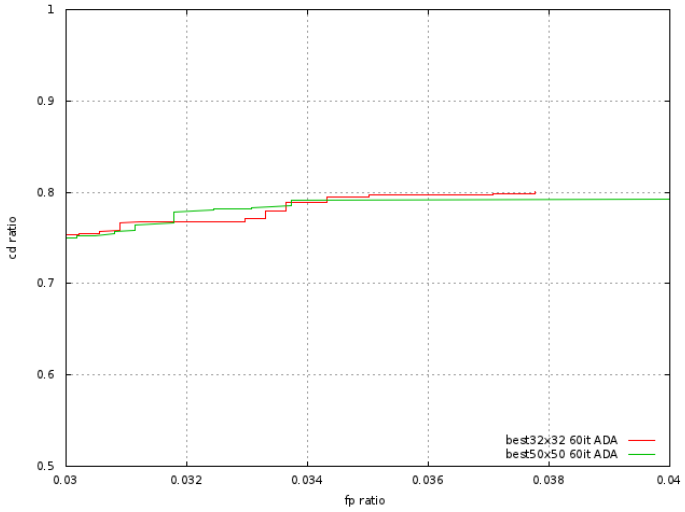


Figura 1.47: Confronto tra le migliori curve ROC ottenute addestrando il classificatore *AdaBoost* per riconoscere i segnali di obbligo con i modelli in formato 50×50 e 32×32.

Analisi comparativa tra reti neurali e ADA-boost

Per determinare la tipologia di classificatore da utilizzare nell'applicazione finale sono state confrontate le prestazioni di riconoscimento tra le migliori curve ROC ottenute addestrando sia il classificatore *AdaBoost* che le reti neurali ad uno e due livelli intermedi. Dall'analisi effettuata è emerso che le funzionalità introdotte hanno permesso di ottenere migliori risultati di riconoscimento e per questa ragione i classificatori scelti per l'applicazione finale sono tutti addestrati con modelli di dimensione 32×32 pixel.

In particolare dal confronto tra le migliori curve ROC ottenute nell'ambito del riconoscimento dei segnali di pericolo (figura 1.48) si evince che il miglior classificatore per questa categoria di cartelli è la rete neurale addestrata sui nuovi modelli e costituita da un singolo livello intermedio composto da 65 neuroni.

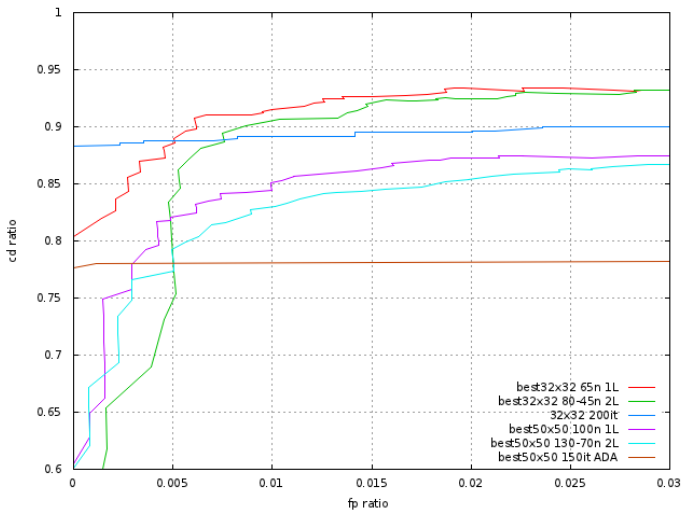


Figura 1.48: Confronto tra le migliori curve ROC per riconoscere i segnali di pericolo.

Per quanto invece riguarda il riconoscimento dei segnali di divieto in figura 1.49 sono mostrate le migliori curve ROC ottenute dall'analisi precedentemente descritta:

in questo caso il classificatore scelto per l'applicazione finale è la rete neurale addestrata sui campioni di dimensione 32×32 pixel e costituita da due livelli intermedi, composti, rispettivamente, da 115 neuroni nel primo strato e 65 nel secondo.

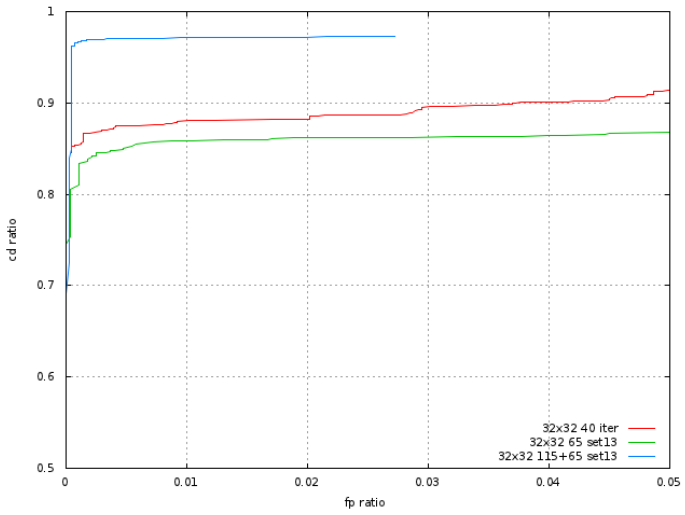


Figura 1.49: Confronto tra le migliori curve ROC per riconoscere i segnali di divieto.

I migliori risultati ottenuti per la classificazione dei segnali di obbligo sono mostrati in figura 1.50: il classificatore scelto per l'applicazione finale è la rete neurale ad un solo livello intermedio, composta da 40 neuroni e addestrata sui nuovi modelli. Si noti come per questa categoria di segnali le prestazioni del classificatore *AdaBoost* sui modelli 32×32 e 50×50 siano del tutto paragonabili tra loro.

Analisi delle reti di divieto con diversi insiemi di addestramento

Un'ulteriore analisi ha riguardato lo studio del miglior insieme di addestramento: al variare del *training set* è stata quindi misurata la capacità di una stessa rete di riconoscere i campioni su cui è stata addestrata.

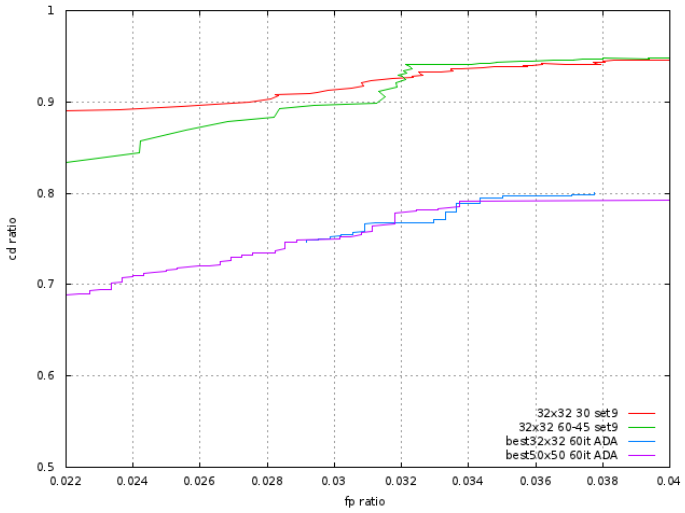


Figura 1.50: Confronto tra le migliori curve ROC per riconoscere i segnali di obbligo.

In particolare si è scelto di utilizzare la rete dell'applicazione finale scelta per il riconoscimento dei segnali di divieto: quella addestrata sui modelli di dimensione 32×32 pixel e costituita da un primo livello con 115 neuroni e da un secondo livello con 65 neuroni, .

In figura 1.51 sono riportate le curve ROC ottenute relative alla rete precedentemente citata, addestrata con modelli di immagini sintetiche, reali ed una combinazione di entrambi.

Nei test precedenti gli insiemi di addestramento e validazione sono stati costruiti manualmente, cercando di inserire nel *training set* più varianti possibili per uno stesso cartello, per tenere conto delle condizioni sotto cui un segnale può apparire nelle immagini (ombra, visibilità parziale, illuminazione intensa ecc.). La procedura si è dimostrata molto laboriosa e talvolta ha influenzato negativamente le performance della rete, come mostrato in figura 1.52: una rete addestrata con 623 modelli selezionati manualmente è stata comparata con una rete a 696 modelli selezionati casualmente dal calcolatore e sono emersi i vantaggi dell'uso di un *training set* selezionato

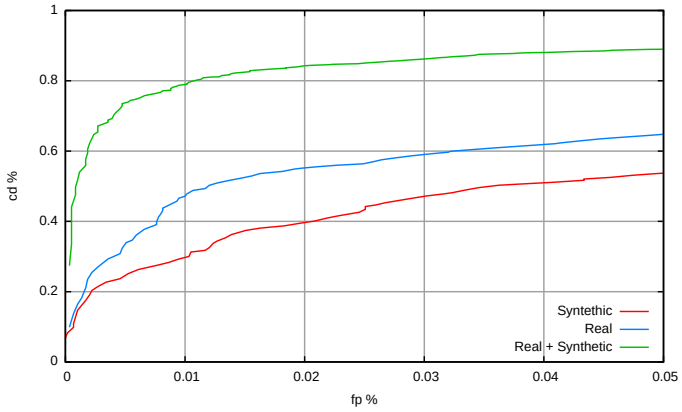


Figura 1.51: Cuve ROC relative alla valutazione di una stessa rete su diversi insiemi di addestramento composti rispettivamente da immagini sintetiche (curve rossa), casi reali (curva blu) ed una combinazione di entrambi (curva verde).

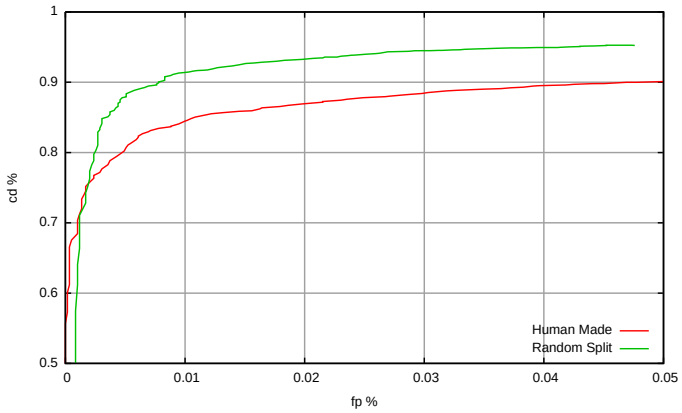


Figura 1.52: Rete di divieto a 115+65 neuroni, addestrata utilizzando due *training set*, uno selezionato da un supervisore umano (in rosso) e l'altro casualmente generato (in verde). Approssimativamente entrambi gli insiemi contengono lo stesso numero di modelli e rappresentano circa il 10% dell'insieme di validazione.

casualmente. La dimensione ottima del *training set* rimane quindi una questione aper-

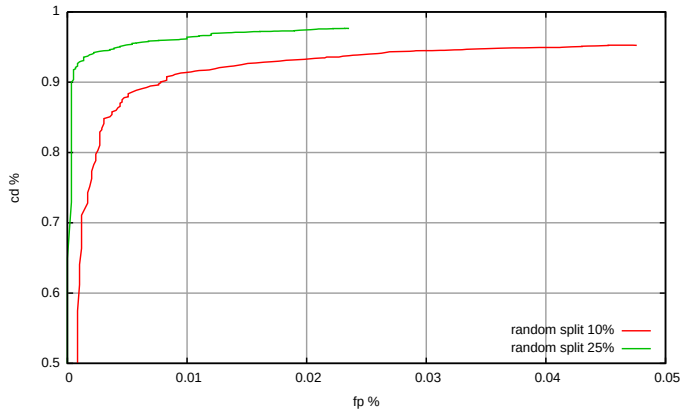


Figura 1.53: Rete di divieto a 115+65 neuroni, addestrata utilizzando un insieme addestramento generato casualmente e formato, rispettivamente dal 10% (in rosso) e 25% (in verde) dei pattern disponibili.

ta: in figura 1.53 sono mostrate le performance della stessa rete di divieto addestrata utilizzando, rispettivamente, il 10% and 25% dei segnali disponibili: si è scelto di mantenere il numero dei segnali limitato in modo da evitare un'eccessiva specializzazione. Un altro aspetto che giustifica questa scelta è dovuto al fatto che attualmente la grande quantità di modelli acquisiti per l'addestramento rimane comunque un valore limitato rispetto la variabilità con cui un segnale può apparire all'interno delle immagini: si preferisce quindi collezionare più dati possibili da destinare all'insieme di validazione e certificazione, piuttosto che aumentare la dimensione dell'insieme di addestramento.

Riepilogo delle performance

Visto il miglioramento introdotto dal nuovo algoritmo si è deciso di adottare i modelli di formato 32×32: le topologie di rete scelte per l'applicazione finale sono mostrate in tabella 1.2.

Net Name	# Livelli Nascosti	Geometria	# Output
Divieto	2	115+65	37
Indicazione	1	50	10
Obbligo	1	40	26
Pericolo	1	65	42
Sosta	1	40	3
Stop	1	80	3

Tabella 1.2: Geometria delle reti neurali adottate.

Un breve riepilogo delle performance delle varie reti addestrate è mostrato nella tabella 1.3.

Nome della Rete	Rilevazioni Corrette sui modelli di qualità	Rilevazioni Corrette su tutti i modelli	Falsi Positivi / Negativi su tutti i modelli
Divieto	83.3%	79.9%	5.9%
Indicazione	87.9%	97.3%	0.5%
Obbligo	94.1%	92.2%	3.2%
Pericolo	76.4%	76.6%	3.2%
Sosta	95.1%	94.1%	2.7%
Stop	95.7%	94.5%	1.1%

Tabella 1.3: Breve riepilogo delle performance di riconoscimento.

1.6.3 Analisi delle prestazioni computazionali

Di seguito si riportano i risultati dell'analisi delle prestazioni, in termini di tempi di calcolo necessari per l'esecuzione delle varie funzionalità. In particolare si è scelto di valutare separatamente l'incidenza sui tempi di computazione delle funzioni principali dell'algoritmo:

- *Segmentazione*: comprende le fasi di equalizzazione cromatica e filtraggio dei colori;

- *Forme*: fase di etichettatura delle componenti connesse e di ricerca delle forme;
- *Fine Limite*: procedura sviluppata *ad hoc* per il riconoscimento dei segnali di fine prescrizione.

Poiché alcune voci, come `Forme` e `Fine Limite`, dipendono fortemente dal numero di candidati presenti nelle immagini, che può essere assai diverso da frame a frame, si è scelto di misurare i tempi medi di elaborazione prendendo come riferimento due frame, il 5816 e il 13619 (visualizzati in figura 1.54); in questi frame la presenza di numerosi elementi, con colori e forme compatibili con quelle di un cartello stradale, rende l'elaborazione costosa dal punto di vista computazionale. I risultati della valutazione sono mostrati nella tabella 1.4. La macchina utilizzata per la misurazione delle prestazioni monta un processore Intel Pentium 4 2.8GHz, con 2 GByte di RAM su sistema operativo *Linux*.

Appaiono subito evidenti i tempi eccessivi di elaborazione nelle fasi di segmentazione e ricerca delle forme: sorge quindi la necessità di operare delle ottimizzazioni in questi due ambiti.

(a) *(b)*

Figura 1.54: *(a)* Frame 5816 e *(b)* frame 13659, presi come riferimento per la valutazione dei tempi di esecuzione dell'algoritmo di riconoscimento dei segnali stradali.

Tempo di elaborazione		
	Frame 5816	**Frame 13659**
Segmentazione	7ms	10ms
Ricerca forme	7ms	12ms
Fine Limite	3ms	3ms

Tabella 1.4: Tempi delle funzioni principali dell'algoritmo di riconoscimento dei cartelli di fine prescrizione, in riferimento ai frame 4324 e 4630.

I tempi di elaborazione per il riconoscimento dei cartelli stradali sono stati invece misurati valutando, separatamente, il tempo totale di classificazione impiegato utilizzando il metodo attuale e quello preesistente: i risultati di questa analisi sono riportati in tabella 1.5. Appare subito evidente come i tempi computazionali dell'algoritmo attuale siano aumentati, in favore di un risultato più accurato che implica un migliore riconoscimento dei cartelli presenti nell'immagine. In realtà, se si confrontano i tempi di esecuzione totale dell'algoritmo, l'aumento percentuale è di circa 10%, e rispetta ampiamente i tempi di elaborazione imposti dalle specifiche di utilizzo dell'applicazione.

Tempo di elaborazione per la fase di classificazione		
	Frame 5816	**Frame 13659**
Metodo preesistente	2ms	1ms
Nuovo metodo	5ms	4ms

Tabella 1.5: Confronto, con riferimento ai frame 4324 e 4630, tra i tempi di esecuzione della fase di classificazione, utilizzando l'algoritmo attuale e quello preesistente.

1.7 Conclusioni e sviluppi futuri

Sulla base delle prove sperimentali effettuate, si può certamente affermare che il sistema realizzato risponde con successo alle numerose problematiche riscontrate, sebbene appaia chiaro come alcune soluzioni progettuali meritino ulteriori sviluppi. Il sistema, come descritto, si articola su vari passaggi successivi e appare evidente come il fallimento nel funzionamento di una singola parte, provochi il degrado sensibile delle prestazioni generali. Le funzionalità introdotte hanno permesso di incrementare notevolmente l'affidabilità del sistema nel riconoscere i segnali, soprattutto in condizioni limite, quando ad esempio nelle fasi di segmentazione e ricerca delle forme non si riesce ad eliminare perfettamente il bordo dei segnali di divieto, oppure quando un cartello si presenta ruotato all'interno delle immagini. Assai convincente risulta inoltre la fase di calibrazione del contrasto che ha permesso di migliorare la qualità delle immagini in ingresso al classificatore, facilitando il compito di quest'ultimo e rendendolo meno gravoso dal punto di vista computazionale. La metodologia introdotta per l'elaborazione degli ingressi al classificatore può inoltre sfruttata in fase di riconoscimento delle forme al posto del *pattern matching*, riducendo ulteriormente il carico computazionale della fase di classificazione.

Gli sviluppi futuri del progetto possono procedere in diverse direzioni. Un primo sforzo può essere volto al miglioramento dei metodi sviluppati per l'equalizzazione cromatica: il sistema si dimostra infatti poco robusto in condizioni di controluce, quando l'elevata luminosità dell'ambiente genera una sorgente di luce predominante che incide direttamente sul dispositivo di acquisizione rendendo sottoesposti tutti gli altri soggetti. Per sopperire a questo problema si può pensare di intervenire sia dal punto di vista hardware, ricorrendo a metodologie avanzate come come HDR [2], che dal punto di vista software, sviluppando una procedura *ad hoc* per il rilevamento delle situazioni di controluce: si potrebbe cosí interrogare il sistema per sapere se è necessario modificare i parametri della telecamera, agendo quindi sul guadagno ed il tempo di integrazione (otturatore) per la correzione delle immagini in condizioni di controluce, oppure proseguire con la normale acquisizione, dato che siamo in una

[2] High Dynamic Range

situazione normale.

Sviluppi plausibili del sistema possono inoltre indirizzarsi verso l'introduzione di nuove funzionalità, come, ad esempio il riconoscimento dei pannelli integrativi, oppure l'estensione delle categorie dei segnali riconosciuti. L'introduzione di una seconda telecamera può permettere di godere dei vantaggi dell'analisi stereo: dalla rilevazione di superfici per scartare riconoscimenti errati, al calcolo della distanza tra gli oggetti riconosciuti e l'autovettura.

Una volta ottenuto un sistema robusto ed affidabile, si può passare alla fase di implementazione su autovettura, ponendo l'accento sulla comunicazione dei risultati al guidatore. Questo può avvenire mediante segnalazione visiva, utilizzando uno schermo a colori su cui mostrare tutti i cartelli riconosciuti. Nel caso di riconoscimenti dubbi si mostrerebbe il riquadro dell'immagine reale contenente il segnale non classificato. La disponibilità del valore contenuto nei segnali di limite di velocità lascia aperte tante prospettive: l'inserimento di un congegno di vibrazione posto sul pedale in caso di superamento del limite, oppure un segnalatore acustico o, in alternativa, un semplice avvertimento lampeggiante. Soluzioni così originali necessitano di un sistema estremamente affidabile. Si pensi, ad esempio, ad un segnale di limite di velocità condizionato da un pannello indicante la scritta "*in caso di nebbia*": lo sviluppo di un metodo di ricerca per riconoscere il pannello integrativo potrebbe aiutare a far capire che il limite non è valido in qualunque condizione. Rimane ovvio che dispositivi così *invasivi* devono poter essere disattivati in qualunque istante a discrezione del guidatore.

In ambito industriale sono già presenti alcuni sistemi attinenti al progetto sviluppato, integrati su alcuni modelli di auto (*FORD Focus, OPEL Insignia, BMW Serie 3*): questi dispositivi si limitano però al solo riconoscimento dei divieti e limiti di velocità.

Capitolo 2

Sistema di percezione in ambito off-road

2.1 Stato dell'arte

Diversi lavori in letteratura si concentrano sul riconoscimento di ostacoli ed innumerevoli sono gli ambiti applicativi in cui questi studi hanno trovato riscontro, come ad esempio il mantenimento della distanza di sicurezza tra veicoli, il rilevamento di ostacoli da evitare lungo la carreggiata, la realizzazione di sistemi ACC, pre-crash e stop & go. Per questa tipologia di sistemi la scelta del sensore da utilizzare si dimostra un passo importante: grazie all'accuratezza fornita nelle misurazioni alcuni lavori si basano sull'elaborazione di dati provenienti da radar e laser scanner; seppur precisi questi sensori dimostrano alcune limitazioni. In particolare l'affidabilità del radar dipende fortemente dal RCS (Radar Cross Section) [1] dell'oggetto da rilevare: poiché le superfici metalliche sono ottimi riflettori, un veicolo ha maggiore probabilità di essere individuato rispetto ad un pedone; questo aspetto rischia di rendere il sistema poco versatile, limitandone le performance. Per quanto invece riguarda i laser scanner, è stato riscontrato che in determinate condizioni climatiche, come ad esem-

[1] Area equivalente radar, è una variabile aleatoria (dimensionalmente una superficie) che permette di ottenere la potenza totale reirradiata dal bersaglio

pio pioggia e nebbia, questa tipologia di sensori dimostra prestazioni limitate, dovute alla presenza di echi rumorosi nei segnali elaborati. Inoltre quando il veicolo sobbalza il raggio emesso colpisce il terreno o i punti del cielo rendendo l'acquisizione dei dati inaffidabile.

Per queste ragioni molti lavori si orientano sull'uso di telecamere, preferendo l'elaborazione di immagini provenienti da singola telecamera (visione mono) o da coppie stereo (visione stereoscopica). La scelta di elaborare singole immagini è principalmente dettata da ragioni computazionali. L'uso della visione mono per la localizzazione degli oggetti si basa spesso sulla ricerca ed elaborazione delle loro caratteristiche distintive, come ad esempio le simmetrie [61], le ombre [37], il colore [50], i bordi [73] o le forme [47, 82, 77, 78, 22, 19]. Ampiamente utilizzate sono anche le tecniche basate sullo studio del movimento degli oggetti, sia nel visibile [18] che nell'infrarosso [20].

Per quanto invece riguarda la visione stereo, la motivazione principale che spinge i progettisti ad affidarsi a questo tipo di approcci è essenzialmente legata all' affidabilità del risultato fornito e alla possibilità di ricostruire con precisione l'informazione tridimensionale della scena; generalmente si utilizzano immagini a toni di grigio, ma alcuni approcci lavorano su immagini a colori. Essendo più costosa dal punto di vista computazione spesso si rende necessario raggiungere un compromesso tra precisione e massima distanza di rilevazione.

Nel settore industriale l'impiego di sistemi basati sulla visione stereo ha permesso la realizzazione di prototipi sia in ambito *automotive* che *off-road* [36], raggiungendo un livello di consolidamento abbastanza avanzato. Questo tipo di sistemi si è infatti dimostrato particolarmente performante in ambienti consistenti e controllabili, ma la necessità di ottenere prodotti industriali finiti e commercializzabili richiede il loro impiego anche in ambienti più complessi, fortemente variabili ed imprevedibili. La presenza di numerose variabili d'ambiente incontrollate rende questi sistemi molto complessi, ma nel contempo permette agli sviluppatori di ottenere soluzioni versatili ed adattabili ad un ampio spettro di applicazioni. Questi sistemi permettono inoltre di valutare il loro funzionamento in retroazione, al fine di misurare l'errore dell'uscita fornita rispetto al risultato atteso ed eventualmente correggerla. Nella valutazione

delle performance di un sistema di visione artificiale uno sviluppatore esperto è in grado prevedere intuitivamente, dall'analisi qualitativa delle immagini, il risultato che il sistema fornirà in uscita: la capacità di tradurre questa regola empirica, condizionata principalmente dall'esperienza dell'osservatore, in una tecnica parametrizzabile e valutabile quantitavamente, rende l'analisi del sistema esauriente, permettendo di evidenziarne potenzialità e difetti tramite curve caratteristiche e limiti.

2.2 Descrizione funzionale dell'algoritmo

L'attività oggetto della ricerca è stata condotta in collaborazione con "Caterpillar Inc." e riguarda lo sviluppo e la realizzazione di un sistema di percezione per ruspe che operano all'interno di cantieri ed in generale in ambito *off-road* [26]. Il dispositivo, basato su un sistema di acquisizione stereo, ha come obiettivo la ricostruzione di una mappa tridimensionale relativa all'attraversabilità del terreno nella zona di operabilità del veicolo. Il lavoro si concentra quindi sulla ricostruzione del profilo del terreno attraversabile e sulla rilevazione di eventuali ostacoli lungo il percorso. Lavorando in ambito *offroad* la definizione di ostacolo non è quindi univoca, ma dipende strettamente dall'inclinazione del terreno che il veicolo sta attraversando: si considera quindi oggetto tutto ciò che si eleva dal profilo del terreno con deviazioni sufficientemente significative. Per individuare gli ostacoli si ricercano quindi angoli significativi tra veicolo ed oggetti.

L'approccio sviluppato si articola in 4 fasi principali:

1. *Elaborazione punti 3D*. Per l'esecuzione dell'algoritmo è essenziale la conoscenza dell'informazione 3D dei punti che occupano l'area di interesse: in generale questi punti devono essere sufficientemente densi e distribuiti per garantire un'adeguata risoluzione della mappa del terreno. Qualsiasi algoritmo di ottimizzazione necessita infatti di un ampio insieme di dati (punti 3D) ben distribuiti sul dominio, al fine di evitare la localizzazione di minimi locali: maggiore è il numero di punti di cui si conoscono le coordinate mondo e più accurato sarà il risultato.

(a) *(b)* *(c)*

Figura 2.1: Esempio di elaborazione della disparità: *(a)* immagine originale *(b)* immagine rettificata e dedistorta *(c)* mappa di disparità.

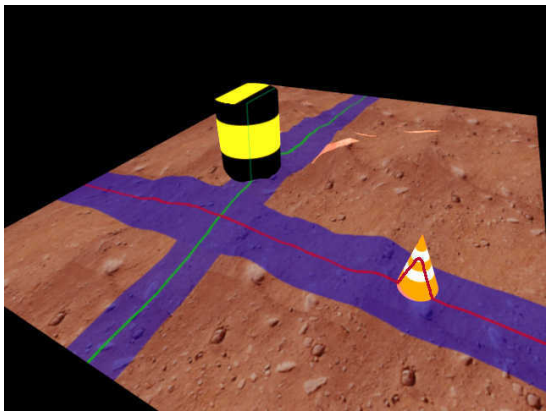
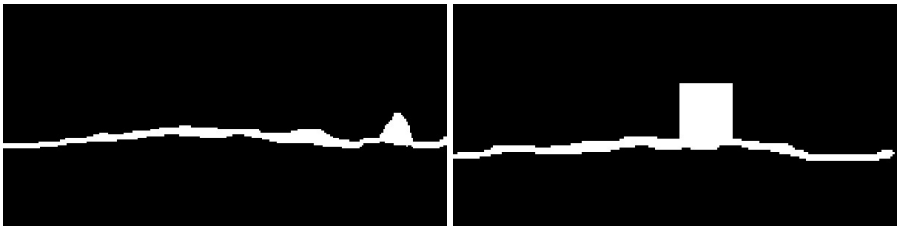


Figura 2.2: Scenario sintetico in cui si mostrano le tracce (rossa e verde) risultanti dalla proiezione dei punti 3D appartenenti all'area di interesse sui due piani verticali (laterale lungo la traccia rossa e longitudinale lungo quella verde) che suddividono il mondo in due sezioni perpendicolari tra di loro.



(a) (b)

Figura 2.3: Esempio di proiezioni 2D relative allo scenario mostrato in figura 2.2; *(a)* proiezione frontale relativa alla sezione descritta dalla linea rossa, *(b)* proiezione laterale, relativa alla sezione descritta dalle retta verde.

Per l'estrazione dell'informazione 3D ci si affida all'elaborazione dell'immagine di disparità [42]; il primo passo consiste nella eliminazione della distorsione introdotta dall'ottica e nella rettificazione delle immagini (figura 2.1 *a*-*b*): questa preelaborazione permette di avere sempre i punti omologhi lungo rette epipolari orizzontali, anche quando la posizione attuale delle camere non rispecchia la tipica configurazione di una coppia stereo. In figura 2.1 *c* è mostrata la mappa di disparità ottenuta: ogni pixel dell'immagine codifica l'informazione relativa alla distanza tra i punti omologhi nelle due immagini acquisite in ingresso (destra e sinistra). La presenza di punti mancanti nella mappa indica che per quei pixel specifici (nell'immagine destra) non è stato possibile individuarne l'omologo (su quella sinistra), ad esempio perché non c'è sufficiente tessitura o perché l'omologo si trova al di fuori dei limiti dell'immagine. Sfruttando i principi di visione stereoscopica e triangolazione, per ciascun pixel, di cui è noto il valore di disparità, è possibile ricavare le sue coordinate mondo a partire dalle sue coordinate immagine. Più la mappa di disparità è densa e maggiore sarà il numero di punti mondo ricostruiti a partire dai pixel nelle immagini.

2. *Proiezioni 2D*. Si rappresentano i punti 3D ricavati al passo precedente su di un sistema di riferimento tridimensionale e si considera questo spazio suddivi-

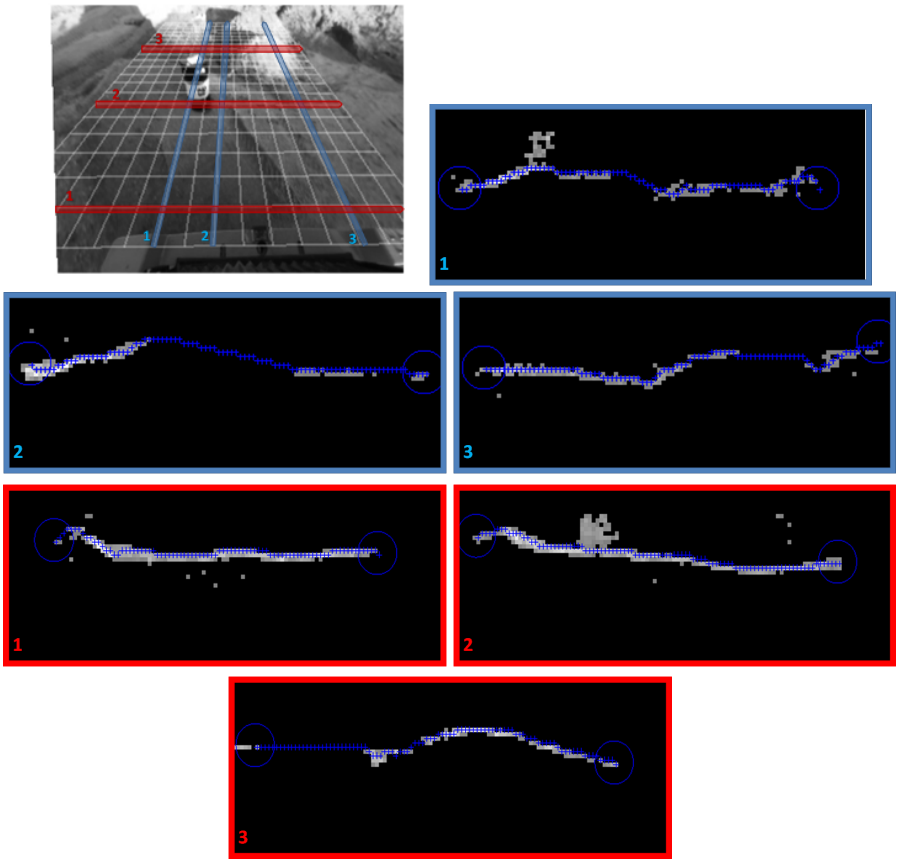


Figura 2.4: Dall'alto al basso: immagine destra con evidenziate le proiezioni longitudinali e laterali analizzate; 3 proiezioni laterali con il relativo cammino (evidenziato con crocette blu) individuato dall'algoritmo basato sulle *Ant Colonies*; 3 proiezioni frontali con il cammino (evidenziato con crocette blu) risultante dall'applicazione dell'algoritmo basato sulle *Ant Colonies*. Notare come in figura 1 blu e 2 rosso gli elementi non appartenenti al profilo del terreno vengono discriminati tramite l'uso delle *Ant Colonies*. Notare indolore come in 1-2-3 blu a 3 rosso le formiche sono in grado di collegare le parti disconnesse, sopperendo alla mancanza di punti, per ricostruire il profilo del terreno.

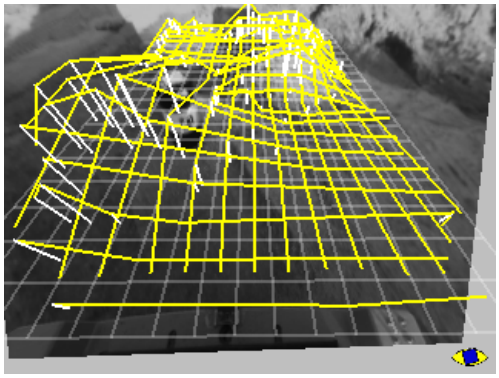


Figura 2.5: I migliori cammini individuati dalle formiche sono approssimati tramite linee spezzate (evidenziate in giallo): il profilo ricostruito combinando le proiezioni longitudinali e laterali è riportato sull'immagine di input.

so longitudinalmente e lateralmente in diverse sezioni con larghezza, altezza e spessore di dimensione fissa; si ottengono così un insieme di piani paralleli in due direzioni (frontale e laterale) su cui si proiettano i punti 3D, compatibilmente con la loro posizione nel mondo. Le proiezioni ottenute sono riportate su immagini binarie (figura 2.3) che rappresentano viste bidimensionali di diverse porzioni di mondo (figura 2.2): questo passaggio è particolarmente importante perchè permette di passare dalla risoluzione di un problema tridimensionale ad uno bidimensionale, riducendone quindi la complessità.

3. *Ricostruzione dell'andamento del terreno*. Ogni proiezione rappresenta quindi il profilo del terreno in corrispondenza di una specifica porzione di mondo. Questa rappresentazione include quindi tutto ciò che si trova in quella specifica sezione di mondo (figura 2.2): terreno, oggetti verticali, buche, accumuli di terra ecc. Analizzando le immagini relative alle proiezioni si vuole determinare la curva che rappresenta l'andamento della superficie attraversabile su cui il veicolo può muoversi. Per tale scopo si utilizza un approccio basato sulle *Ant Colonies* [39, 30]: per ogni proiezione 2D si definisce una colonia costituita da

un insieme di agenti che, muovendosi pixel per pixel, individuano la curva che meglio approssima l'andamento del terreno (figura 2.4). In particolare l'output di questo processo consiste in una lista di punti che viene semplificata tramite une linea spezzata (figura 2.5): questa retta rappresenta un'approssimazione dell'andamento del terreno e permette di stabilire quali punti della proiezione appartengono all'area attraversabile e quali invece rappresentano ostacoli e/o contributi di rumore.

4. *Ricostruzione mappa 3D*. A questo punto si dispone di un insieme di profili che rappresentano viste longitudinali e laterali del terreno attraversabile in specifiche porzioni dell'area di interesse. Combinando insieme i vari profili si ricostruisce una griglia (figura 2.5) costituita da punti (x,y) di cui si conosce l'altezza corrispondente z. Il numero di questi punti dipende dalla larghezza delle proiezioni e dalle loro distanze. Data la mappa 3D è possibile stimare la z del terreno di ogni punto (x,y) appartenente all'area d'interesse, tramite alcune interpolazioni sulle altezze della griglia note.

5. *Individuazione degli ostacoli*. Ogni punto 3D estratto tramite l'elaborazione della disparità viene classificato compatibilmente con la sua posizione rispetto la mappa del terreno precedentemente ricostruita: tutti punti 3D che si trovano ad una certa distanza dalla superficie che rappresenta il profilo stimato vengono considerati ostacoli: i punti vicini vengono poi raggruppati ed etichettati. L'ultima fase dell'algoritmo prevede l'esecuzione del tracking sugli ostacoli al fine di rendere più stabile il loro riconoscimento.

2.3 Analisi delle prestazioni

Il sistema è stato testato all'interno di cantieri utilizzando una ruspa Cat 980H. L'algoritmo si è dimostrato affidabile, come mostrato in figura 2.6, garantendo un ottimo *trade-off* tra accuratezza e complessità di calcolo: in particolare il sistema è in grado di elaborare 10 frame per secondo con immagini 320x240 pixel.

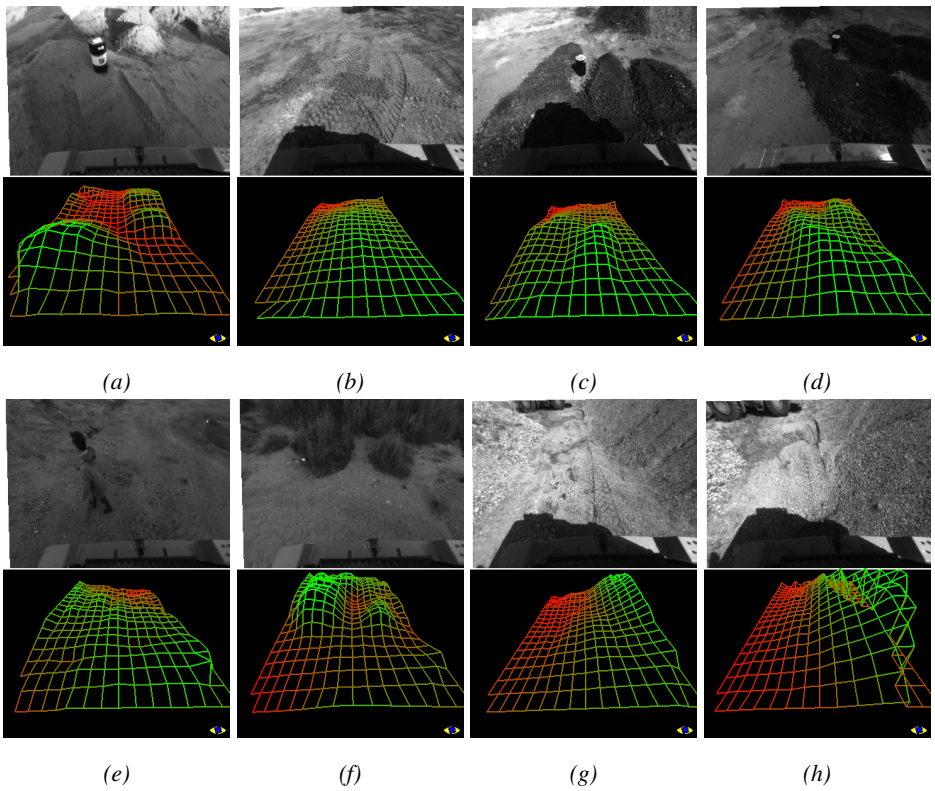
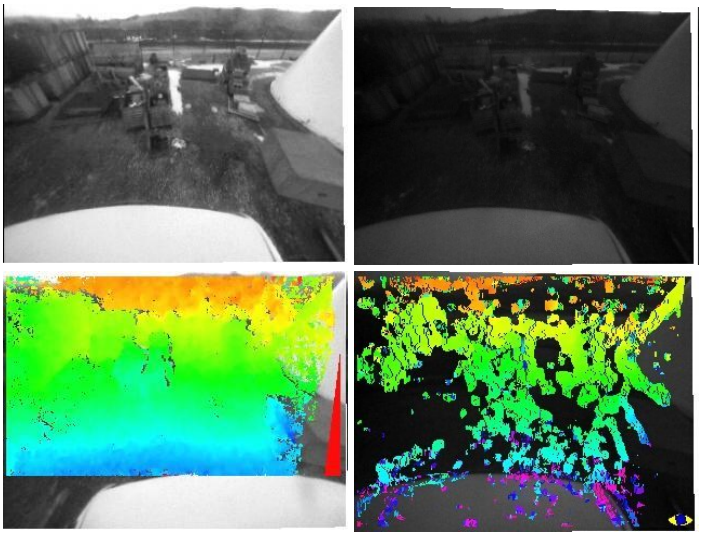


Figura 2.6: Esempi di ricostruzione del profilo del terreno; *(a)* tipico scenario con accumuli di terra, polvere e oggetti comuni; *(b)* rilevazione di terreno piano; *(c)(d)* presenza di accumuli di terreno e barili; *(e)* caso di terreno piano con pedone; *(f)* scenario con arbusti; *(g)(h)* presenza di un canyon. La mappa si limita ad individuare il modello del profilo del terreno, filtrando tutti gli altri contributi: gli oggetti sono costituiti da tutti i punti che descrivono elementi che si elevano da questa mappa.

Dall'analisi delle performance è emersa la dipendenza del sistema dalla mappa di disparità: sia la fase di ricostruzione del terreno che quella di riconoscimento degli ostacoli si dimostrano infatti fortemente influenzate dalla densità della mappa di disparità (figura 2.8); le mappe dense, con pochi punti errati, garantiscono quindi una migliore rilevazione degli oggetti rispetto a quelle sparse e/o rumorose. In figura 2.7 si mostra come diverse condizioni di illuminazione incidano sulla qualità dell'immagine di disparità e quindi sulle performance generali del sistema.



(a) *(b)*

Figura 2.7: Esempi di effetti generati dalle diverse condizioni di illuminazione sulla mappa di disparità. Nella colonna *(a)* sono mostrate le immagini acquisite dalla telecamera destra in diverse condizioni di illuminazione e la relativa mappa di disparità. La colonna *(b)* mostra le stesse scene acquisite al mattino: la mappa di disparità è decisamente meno densa e molto più rumorosa.

Visto che la procedura di riconoscimento degli ostacoli risulta essere strettamente correlata alla valutazione soggettiva della qualità della mappa di disparità, è possibile individuare numericamente questo legame per permettere al sistema di effettuare,

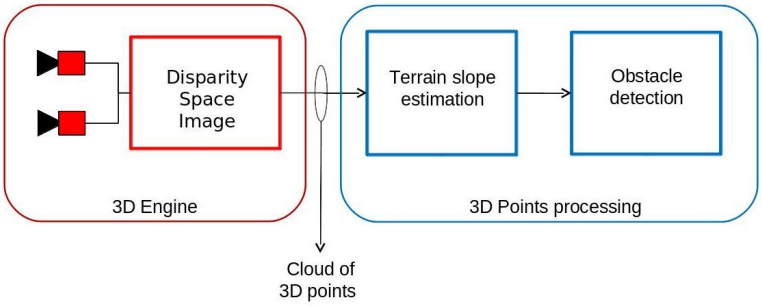


Figura 2.8: Diagramma a blocchi del sistema stereo sviluppato per la ricostruzione del profilo del terreno ed il riconoscimento degli ostacoli.

dalla sola analisi della disparità, una previsione sulla qualità del risultato che sarà in grado di fornire in uscita. E' sorta quindi la necessità di tradurre questa relazione qualitativa tra i risultati di riconoscimento e le caratteristiche della mappa di disparità in un legame quantificabile numericamente.

Alcuni studi sulla valutazione qualitativa delle immagini stereoscopiche sono già presenti in letteratura: metriche basate su PNSR (*Peak-to-Noise-Ratio*), SNR (*Signal-to-Noise-Ratio*) [103] o tecniche come [100] non si dimostrano indicatori affidabili per stabilire, nell'ambito del sistema oggetto della ricerca, quanto la qualità della mappa di disparità incida sulle prestazioni dell'applicazione oggetto della ricerca.

Per questo progetto per l'individuazione della funzione che mette in relazione diretta la mappa di disparità con le performance generali dell'algoritmo si prevede di misurare la precisione nella stima del profilo del terreno e nel riconoscimento ostacoli al variare della densità della mappa di disparità. Si valuta quindi come il sistema risponde alla variazione delle condizioni di illuminazione e delle condizioni climatiche che influenzano sia la densità che la correttezza della nuvola di punti 3D estratta tramite la disparità.

Formalmente si ricerca quindi una funzione lineare che mette in relazione un insieme di caratteristiche (*features*) dell'immagine di disparità con il risultato di ri-

conoscimento corrispondente. Una volta che i coefficienti di questa relazione lineare sono noti, la metrica generata permette di prevedere le prestazioni dell'algoritmo sotto diverse condizioni ambientali. Studiando la correlazione tra le *features* e l'errore nella misura delle prestazioni è inoltre possibile ridurre l'insieme delle caratteristiche al fine di selezionare quelle più distintive per la mappa di disparità e quindi più determinanti per il risultato.

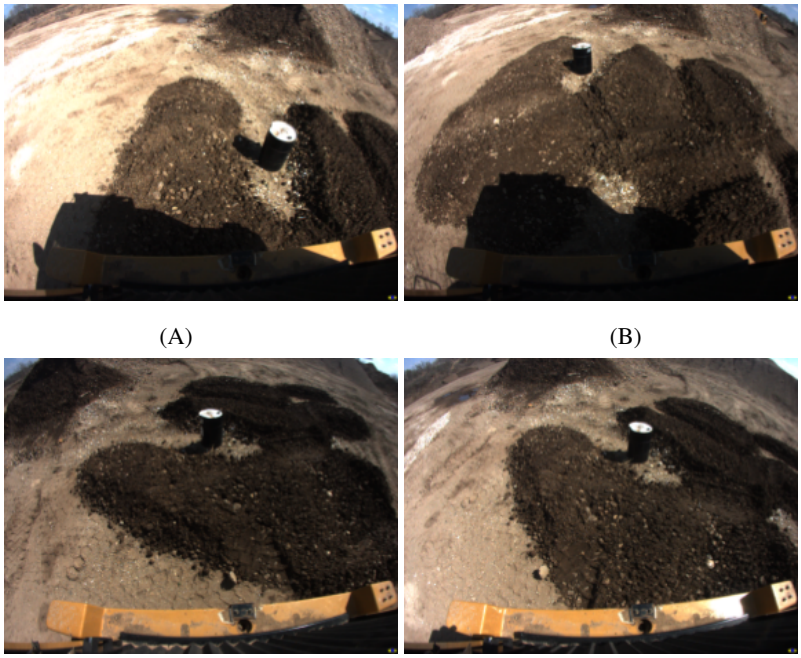
L'attività di ricerca svolta si è quindi concentrata sulla definizione dei parametri di interesse nella mappa di disparità, studiando le performance di riconoscimento al degradarsi delle immagini in ingresso: si è resa quindi rumorosa l'informazione tridimensionale, essenziale per l'individuazione degli ostacoli e del profilo del terreno, valutando l'errore di riconoscimento in condizioni di bassa luminosità, nebbia e sfocamento delle immagini. Utilizzando un insieme di immagini di addestramento è stato quindi parametrizzata una funzione lineare di predizione, di cui sono state valutate le prestazioni su un insieme di *testing* costituito da immagini degradate.

2.4 Elaborazione immagini degradate

Per variare la densità e la rumorosità della mappa di disparità, al fine di studiarne l'incidenza sulle prestazioni, sono state utilizzate alcune immagini di addestramento (figura. 2.9 (A) (B) e (C)) su cui sono stati applicati diversi livelli di degradazione. La degradazione è stata introdotta artificialmente: a partire da immagini di alta qualità, acquisite in condizioni di illuminazione ottime, sono stati aggiunti disturbi come effetti atmosferici, sfocamento, nebbia, oscurità e rumore del sensore.

- **Rumore gaussiano artificiale**: per simulare all'interno delle immagini gli effetti introdotti da una telecamera non a fuoco o gli artefatti dovuti al movimento degli oggetti si sfocano le immagini introducendo del rumore gaussiano; in particolare si calcola la convoluzione dell'immagine originale con la seguente funzione:

$$G(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}} \tag{2.1}$$



(A) (B)

(C) (D)

Figura 2.9: Immagini degradate; (A), (B) e (C) sono parte dell'insieme di addestramento, mentre (D) fa parte dell'insieme di *testing*.

dove σ varia tra 1 e 50 con incremento di 1. La dimensione della matrice di convoluzione è invece compresa tra 1 e 40.

- **Nebbia artificiale**: la presenza di nebbia riduce il contrasto delle immagini in proporzione alla distanza tra l'osservatore e il punto della scena. Questo effetto viene introdotto all'interno dell'immagine originale utilizzando le seguenti equazioni (secondo il modello OpenGL [92])

$$fogged\_pixel(u,v) = pixel(u,v) \cdot f + fogcolor \cdot (1-f) \tag{2.2}$$

dove f è il *fattore di blending*, calcolato usando la funzione esponenziale:

$$f = e^{-d \cdot z} \tag{2.3}$$

dove d è la *densità della nebbia* e z è la distanza tra il punto di osservazione e il centro di massa. La nebbia ha densità che varia tra 0 e 1, ed il suo valore *RGB* è $(127,127,127)$.

- **Oscurità e rumore artificiale**: in condizioni di bassa illuminazione si osservano due fenomeni all'interno immagini, cioè la riduzione della luminosità e l'aumento del rumore introdotto dal sensore. L'oscurità viene quindi simulata utilizzando un fattore di attenuazione e del rumore gaussiano additivo indipendente:

$$att\_pixel(u,v) = (pixel(u,v) \cdot a + noise(u,v)) \cdot gain \tag{2.4}$$

dove a è il fattore di attenuazione, $noise(u,v)$ è il rumore bianco gaussiano aggiunto sul segnale dal sensore e *gain* rappresenta l'amplificazione del segnale applicata dalla camera. Il sistema prevede l'aggiustamento automatico di *shutter* e *gain* per mantenere la luminosità media dell'immagine all'interno di un *range* accettabile. Di conseguenza, all'aumento del fattore di attenuazione il *gain* aumenta, amplificando il rumore sul valore del segnale. Il *range* di attenuazione per a va da 1 a 0; il *gain* varia da 0 a 9.32, mentre il rumore è gaussiano con σ da 1 a 5. Questi *range* sono stati definiti secondo le specifiche della camera utilizzata [7].

Ogni immagine viene quindi acquisita in condizioni di illuminazione ottime, con valori di *shutter* e *gain* noti, e successivamente si applicano iterativamente diversi livelli di degradazione. In figura 2.10 sono mostrati alcuni esempi di disturbi introdotti con rispettiva mappa di disparità.

2.5 Estrazione delle features

Per l'elaborazione delle *features* relative alla mappa di disparità ci si concentra sull'individuazione di caratteristiche specifiche per questo sistema, non applicabili in generale su altre applicazioni similari. L'intento è appunto quello di sviluppare un algoritmo *ad-hoc* per questo progetto; qualsiasi insieme personalizzato di *features* può comunque essere applicato al metodo descritto in seguito per renderlo più versatile.

Per l'estrazione delle caratteristiche distintive alcuni metodi esistenti prevedono l'uso di tecniche di valutazione globali: una limitazione per questa tipologia di metriche è dovuta al fatto che le prestazioni in alcune regioni posso differire sostanzialmente dalla misura fornita globalmente. Perciò, sebbene sia possibile disporre di valutazioni globali sull'immagine e sulla disparità, si ritiene opportuno considerare le immagini suddivise in regioni più piccole per poter ottenere di una misura locale della loro qualità. L'uso di metriche locali, a differenza di quelle globali, permette di individuare eventuali prestazioni non accettabili in specifiche aree.

Nell'analisi locale si dimostrano particolarmente cruciali i passi per la segmentazione dell'immagine in regioni più piccole e la scelta della loro dimensione, geometria e sovrapposizione. Anche con un ridotto numero di parametri e di regioni la quantità di *features* potenziali da estrarre dalla mappa di disparità tende a crescere velocemente verso centinaia di elementi. La scelta del criterio di suddivisione dell'immagine di disparità è ulteriormente complicata dal costante cambiamento delle caratteristiche della scena a dalla non uniformità dei disturbi introdotti dalle degradazioni sulle diverse regioni dell'immagine.

Una prima suddivisione naturale dell'immagine prevede quindi di considerare aree non sovrapposte (quadrati) di dimensione predefinita al fine studiare la distribuzione di certe caratteristiche sull'intera immagine, mantenendo fissa la metrica di

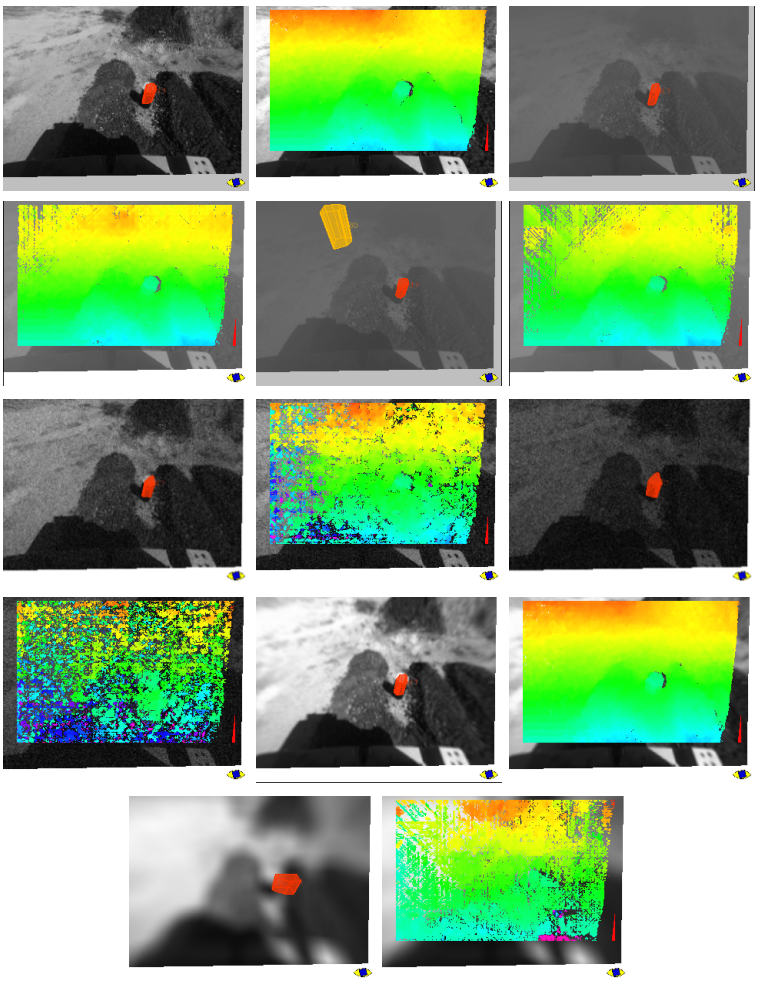


Figura 2.10: Esempi di degradazioni introdotte artificialmente e relativa mappa di disparità. Dall'alto: immagine in condizioni di illuminazione ottime; nebbia con densità $d = 0.30$; nebbia con densità $d = 0.37$, primo falso positivo; oscurità artificiale e rumore con $attenuazione = 93.5\%(a = 0.065)$, $gain = 18.062dB$; oscurità artificiale e rumore con $attenuazione = 96.5\%(a = 0.035)$, $gain = 22.499dB$; Sfocamento artificiale, con $\sigma = 6$, dimensione matrice 6; Sfocamento artificiale, $\sigma = 50$, dimensione matrice 50.

valutazione. Poiché alcune *features*, come ad esempio le ombre, non sono note a priori, una segmentazione in regioni non sovrapposte può mascherare i primi indicatori della riduzione delle performance: per questa ragione ci si concentra inoltre sull'analisi di aree tra loro sovrapposte.

Elaborando mappe di disparità 320×240 pixel si considerano le seguenti suddivisioni:

- Bande (*DSIRange*): 18 bande orizzontali non sovrapposte corrispondenti ad aree del terreno larghe $0.5m$ e consistenti con la risoluzione dell'algoritmo utilizzato per segmentare lo spazio di interesse in terreno attraversabile ed oggetti.

- Quadrati (*DSIFixed*): 103 quadrati non sovrapposti di dimensione 20×20 pixel.

- Finestre (*DSIMobile*): 186 quadrati sovrapposti costituiti da finestre di dimensione 150×150 pixel che vengono fatte scorrere sull'intera immagine di 9 pixel per volta.

- Immagine (*DSIGlobal*): intera area di elaborazione della mappa di disparità; quest'area può essere più piccola dell'intera immagine.

Si introducono inoltre i seguenti simboli utilizzati per l'estrazione delle caratteristiche:

- N numero dei pixel dell'immagine destra per i quali si cerca un *match* (omologo) con i pixel dell'immagine sinistra. Una mappa di disparità completamente densa contiene N punti. N è sempre minore di $largezza \times altezza$ dell'immagine.

- $N_{stripe-i}$ numero teorico di punti di disparità, sull'$i-esima$ banda dell'immagine, nel caso di mappa completamente densa (corrispondente a un'area del terreno larga $0.5m$),

- $N_{fixed-i}$ numero teorico di punti di disparità in ciascun quadrato di 20×20 pixel in cui è suddivisa l'immagine; considerando solo le regioni di interesse per il rilevamento degli ostacoli si ottengono 103 aree di dimensione fissa.

- $N_{mobile-i}$ numero teorico di punti di disparità in ciascuna finestra di 150×150 pixel in cui è suddivisa l'immagine; ogni area si sovrappone con quella vicina orizzontalmente e verticalmente di 141 pixel; si ottengono in totale 186 finestre mobili.

Per l'elaborazione delle *features* relative alla mappa di disparità si valutano, per ciascuna regione, le seguenti caratteristiche:

- *Filling*: densità della mappa di disparità $\frac{n}{N_e}$, dove n è il numero effettivo di punti di disparità ottenute nell'area e N_e è il corrispondente numero teorico di punti atteso.

- *Mean*: disparità media dell'area.

- *Variance*: varianza della disparità nell'area.

- *Diff*: media delle differenze tra i valori di disparità nell'area e i rispettivi valori attesi nell'ipotesi di terreno piatto.

- *GapsNumber*: numero di regioni, con area maggiore di 15 pixel, che racchiudono valori mancanti di disparità (buchi).

- *GapsMeanArea*: area media $\frac{\sum a_{gaps}}{N_{gaps}}$ dei buchi di disparità individuati.

- *GapsStandardDeviation*: deviazione standard sull'area dei buchi individuati.

- *GapsFilling*: rapporto tra l'area dei buchi e il numero totale di punti di disparità mancanti sull'intera immagine $\frac{\sum a_{gaps}}{N-n}$; questa *feature* misura l'importanza dei buchi rispetto la densità generale della disparità.

- *GapsSumDistances*: distanza media tra ogni buco e i suoi vicini, normalizzata rispetto il raggio dell'area occupata dal buco.

Il numero totale di *features* è circa 840. Poiché i valori appartengono a *range* diversi i dati sono normalizzati: per ogni caratteristica viene calcolata la sua deviazione standard su frames differenti ed il valore ottenuto viene utilizzato come fattore di normalizzazione. La normalizzazione viene eseguita *offline* durante la fase di addestramento.

2.6 Analisi delle features

Per rimuovere la ridondanza e la rumorosità dei dati, diversi metodi di riduzione dello spazio delle *features* sono stati studiati in letteratura [65] [38] ed alcuni studi comparativi [104] [86] [85] sono stati condotti per evidenziarne potenzialità e difetti. Modelli basati su filtri [55] e *wrapper* [59] sono le due principali classi di metodi utilizzate.

Come anticipato precedentemente in questa fase dello sviluppo l'obiettivo è stato quello di individuare una relazione tra i valori delle *features* ed il corrispondente errore di rilevamento in uscita. L'analisi condotta prevede l'utilizzo di un insieme di immagini contenenti un ostacolo rilevato dall'algoritmo in condizioni in cui la mappa di disparità appare ottima. Per ciascuna immagine si eseguono i seguenti passi:

1. l'ostacolo è rilevato salvando la sua posizione (x_r, y_r, z_r);
2. si degrada l'immagine;
3. l'ostacolo è nuovamente rilevato sull'$i-esima$ immagine degradata alla posizione (x_i, y_i, z_i);
4. si calcola l'errore di rilevazione come distanza tra i baricentri, normalizzata rispetto la distanza dell'ostacolo:

$$e_i = \frac{\sqrt{(x_i - x_r)^2 + (y_i - y_r)^2 + (z_i - z_r)^2}}{\sqrt{x_r^2 + y_r^2 + z_r^2}} \tag{2.5}$$

5. si ritorna al secondo passo, incrementando il livello di degradazione.

Al termine del processo si ottiene un vettore di errori E e una matrice di *features* A. La matrice A ha un numero di colonne uguale al numero di *features* calcolate ed un numero di righe equivalente al numero di immagini analizzate. Dall'analisi di un numero sufficientemente grande di immagini degradate si risolve il problema:

$$Aw = E \tag{2.6}$$

dove w è il *vettore dei pesi delle features*. Il vettore contiene 840 pesi, uno per ogni caratteristica calcolata.

Poiché il sistema lineare è sovradimensionato si utilizza la tecnica di regressione ai minimi quadrati per determinare la soluzione ottima, che meglio rappresenta i dati in ingresso. In assenza di ostacoli non viene memorizzato alcun valore: quando l'algoritmo non è in grado di rilevare un ostacolo si evita infatti di introdurre un valore di errore arbitrario che potrebbe incidere negativamente sulla risoluzione della trasformazione lineare.

2.6.1 Analisi preliminare

Da una prima fase di analisi sull'intero insieme di *features* F descritto è emerso che:

- le *features Filling* sono predominanti;
- le *features* più importanti sono localizzate attorno l'ostacolo o in regioni strettamente dipendenti dalla scena inquadrata, dove la mappa di disparità tende a degradarsi in modo molto più rapido e significativo rispetto al resto dell'immagine (ad esempio in presenza di ombre).
- le *features Gaps*,*Variance* and *Diff* hanno un peso molto basso, praticamente ininfluente sulle prestazioni finali.

Si conclude quindi che:

- le caratteristiche locali sono preferibili rispetto quelle globali, come precedentemente descritto nella sezione 2.5: ci sono regioni dell'immagine che sono molto più sensibili alle degradazione, la loro posizione non è nota a priori e le loro caratteristiche distintive sono indefinite ed imprevedibili;

- le *featuresGaps* sono ininfluenti. La presenza di valori mancanti nella mappa di disparità genera sicuramente errori di rilevazione, ma evidentemente, nell'ambito di questa analisi, il loro numero e la loro dimensione non sono in relazione lineare con l'errore di riconoscimento.

- al contrario, gli errori di rilevamento risultano essere particolarmente influenzati dai valori delle *features Filling*. Tre tipi di densità sono valutati nell'ambito di questa analisi: fissa, mobile e per *range*. Le caratteristiche mobili sono attualmente troppe da gestire e particolarmente costose dal punto di vista computazionale; l'analisi per intervalli (bande) non si dimostra adatta per il rilevamento di degradazioni verticali; le finestre a dimensione fissa appaiono quindi quelle in grado di offrire il migliore *trade-off* tra precisione e complessità di calcolo.

In conclusione, l'insieme ridotto di caratteristiche F_{red} è costituito solo da *features* $DSIFixedFillig$.

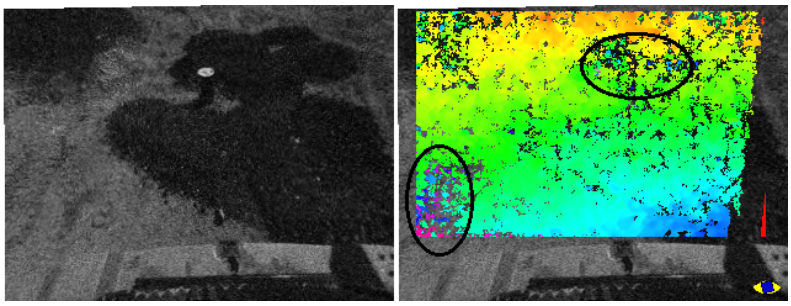
2.6.2 Generalizzazione

Come descritto nel paragrafo precedente, le caratteristiche locali sono usate per stimare l'errore di riconoscimento. Tuttavia, se un'immagine è caratterizzata da valori di disparità errati in una specifica area, la corrispondente *feature* di densità mantiene un peso alto, segnalando l'indebolimento delle performance; la stessa regione, su di un'immagine differente, può non essere più importante, come mostrato in figura 2.11.

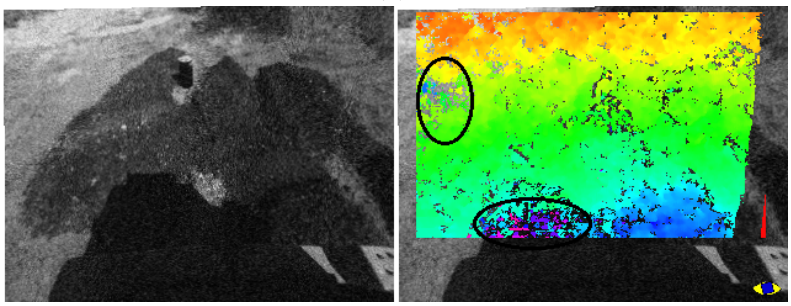
Per ovviare a questo problema sono state definite due funzioni sulle caratteristiche locali:

- *DSI Filling Mean*: media delle densità delle celle 20×20 pixel analizzate $\frac{\sum DSIFixedFillig_i}{N}$;

- *DSI Filling Min*: media delle tre *features* $DSIFixedFillig$ che presentano i valori minori. Questa caratteristica è volta ad evidenziare il caso peggiore ovvero il valore di densità relativo alla regione meno densa.



(a)



(b)

Figura 2.11: Esempio di come le regioni più degradate delle immagine possono cambiare da un frame all'altro; *(a)* immagine di test A; *(b)* immagine di test C (con riferimento alla figura 2.9) .

Dall'analisi e testing delle nuove caratteristiche introdotte è stato riscontrato che i loro pesi *w* si mantengono pressoché costanti: significa quindi che la relazione tra l'errore di riconoscimento e questi due valori è indipendente dalla scena inquadrata.

In figura 2.12 è mostrato l'andamento dell'errore rispetto le *features DSI Filling Mean*, *DSI Filling Min* e rispetto la migliore densità. Sia per il caso *(a)* che *(b)* si nota che:

1. esiste una *feature* che segue l'andamento dell'errore meglio di tutte le altre: questa caratteristica è stata quindi selezionata e riportata sul grafico. Sfortunatamente la *feature* in questione non è la stessa per le due scene inquadrate, ma rappresenta le densità di disparità in due regioni diverse delle immagini.

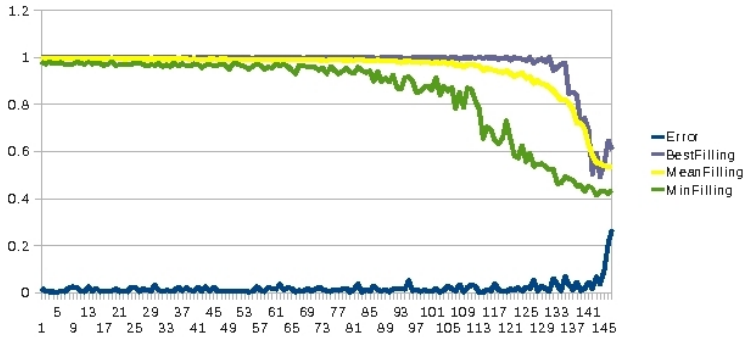
2. le *features DSI Filling Mean* e *DSI Filling Min* sono sempre presenti e nelle due immagini seguono la funzione d'errore allo stesso modo;

3. il modo in cui *DSI Filling Mean* and *DSI Filling Min* seguono l'andamento dell'errore è tale da permettere l'individuazione di una relazione matematica tra errore e valore delle caratteristiche;

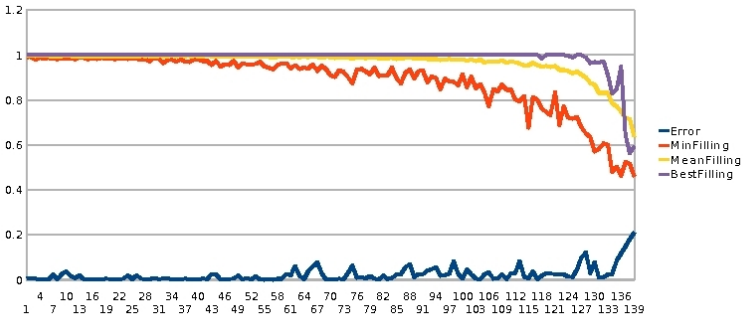
4. le *features DSI Filling Mean* e *DSI Filling Min* tendono ad anticipare leggermente l'andamento della funzione di errore: si nota particolarmente nei tratti crescenti, soprattutto per la *feature* che rappresenta la densità minima. Questo comportamento è giustificabile dal fatto che questa caratteristica rappresenta il caso peggiore.

2.6.3 Calcolo dei pesi finali

Al termine di questa analisi, sulla base delle considerazioni 2) 3) e 4) del precedente paragrafo si conclude che le *features DSI Filling Mean* e *DSI Filling Min* sono le caratteristiche più discriminanti per la rappresentazione della qualità della mappa di disparità: si utilizzano quindi queste due *features* per prevedere il comportamento del sistema. Eseguendo l'intero algoritmo sull'intero *training set* sono stati ottenuti i seguenti i pesi:



(a)



(b)

Figura 2.12: Andamento dell'errore rispetto alla densità media, densità migliore (della scena inquadrata) e peggiore, nelle immagini di test definite in figura 2.9. (*a)* Immagine di test A; (*b)* immagine di test C. Tutti i valori sono definiti in percentuale: errore di stima del baricentro rispetto la distanza dell'ostacolo; percentuale di densità. Tutti i valori sono riportati rispetto un livello di degradazione pari a 150: oscurità e rumore, con rumore gaussiano di 5 sigma. L'attenuazione di luminosità varia nell'intervallo da 0% a 100%, con passo di 1% fino a 50% e passo di 0.5 da 50% a 100%.

Feature	*Peso*
DSI Filling Mean	-0.307369
DSI Filling Min	-0.174026
(Const)	0.481814

dove *Const* rappresenta una *feature* fittizia utilizzata per rendere risolvibile linearmente il sistema di equazione 2.6.

2.7 Risultati

A questo punto si misurano le capacità di predizione delle *features* selezionate: dall'addestramento eseguito sui casi A, B e C (figura 2.9), si valutano le prestazioni di predizione sul caso D per verificare che le *features* selezionate rappresentino effettivamente una metrica di valutazione generale, che non funziona solamente per le immagini utilizzate in fase di addestramento.

In figura 2.13 si mostra l'andamento dell'errore rilevato sul caso D, simulando le condizioni di bassa illuminazione, rispetto alla predizione dell'errore basata sulle *features DSI Filling Mean* e *DSI Filling Min* utilizzando i pesi riportati in tabella 2.6.3. L'errore predetto tende a seguire l'andamento di quello rilevato, divergendo in corrispondenza della fine di ogni sequenza. Questo è principalmente dovuto a due ragioni: la prima è che i pesi finali per la predizione sono ottenuti come *trade-off* di tutte le relazioni tra errori e features analizzate nelle numerose sequenze del *training set*; queste relazioni devono essere lineari quindi in caso di bassa densità i valori tendono a crescere linearmente.

Inoltre quando la degradazione è tale da impedire il rilevamento di un ostacolo, non si memorizza alcun valore di errore (nel grafico si possono infatti osservare dei punti mancanti) per evitare di introdurre valori di *default* che rappresenterebbero degli *outlier* per l'analisi lineare. In questo caso i valori alti sull'errore di predizione sono conseguenza di valori di densità molto bassi, tanto bassi da non permettere la rilevazione dell'ostacolo. In altre parole il malfunzionamento non consiste nell'errore di stima della posizione dell'ostacolo, ma nella sua mancata rilevazione. Si può in-

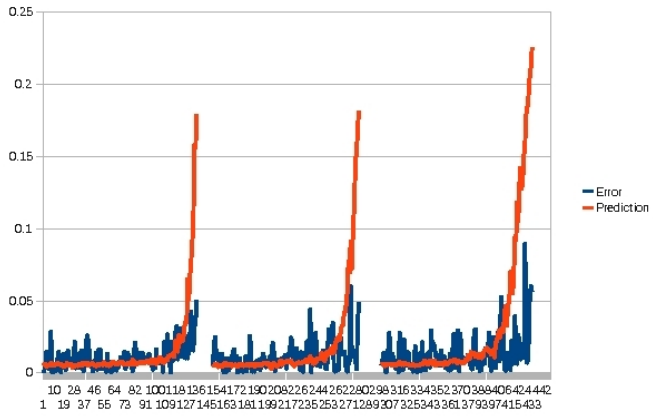


Figura 2.13: Errore misurato sul caso D, comparato con la predizione dell'errore basata sulle *features* significative estratte dalla mappa di disparità, entrambe riportate con 450 livelli di degradazione relativa ad oscurità e rumore (1-150 $\sigma = 3$, 151-300 $\sigma = 4$ e 301-450 $\sigma = 5$).

terpretare questa sovrastima come sintomo di un significativo malfunzionamento che include, oltre che agli errori, le mancate rilevazioni. In figura 2.14 si mostra l'andamento dell'errore di previsione basato sulle stesse *features* rispetto l'errore ottenuto applicando sfocamento gaussiano artificiale sull'immagine di test D. Anche in questo caso l'andamento generale dell'errore viene perfettamente seguito dalla predizione, con una sovrastima dopo il frame 120. Dopo questo punto l'errore stimato cresce rapidamente non appena lo sfocamento α diventa significativo. Questo può essere spiegato guardando cosa accade nell'ultimo esempio in figura 2.10: qui lo sfocamento causa bassi livelli di densità soprattutto in corrispondenza dell'angolo più in alto a sinistra. Questo è un tipico caso di degradazione localizzata. Di conseguenza, poiché il predittore dell'errore non conosce dove si trova l'ostacolo, fornisce una stima per il caso di scenario peggiore, come evidenziato nella sezione 2.6.2, con la descrizione della *feature DSI Filling Min*

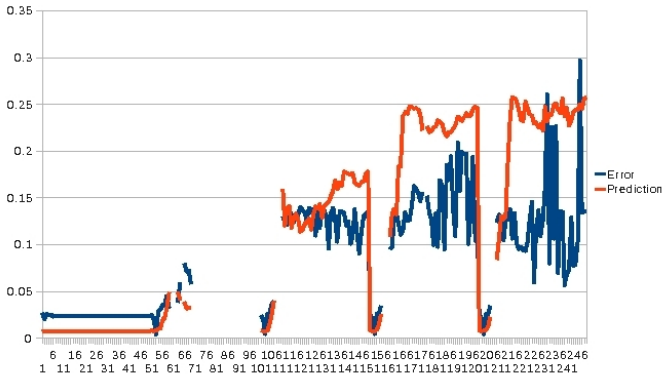
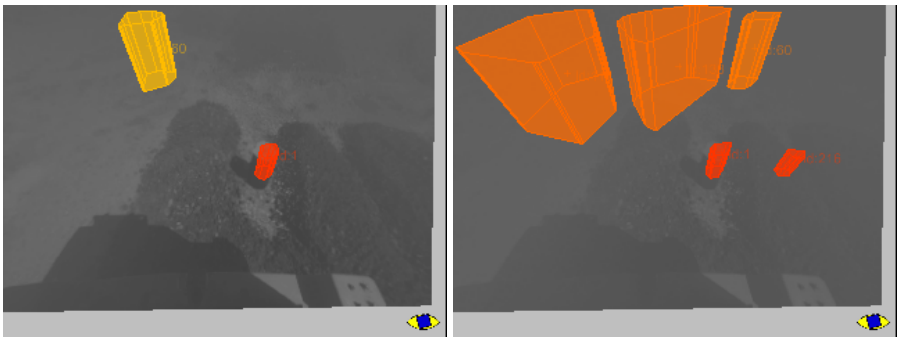


Figura 2.14: Errore misurato sul caso D, comparato con la predizione dell'errore basata sulle *features* significative estratte dalla mappa di disparità, entrambe riportate con 250 livelli di degradazione relativa a sfocamento introdotto tramite rumore gaussiano (1-50 $\sigma = 1-50$, radius=1; 51-100 $\sigma = 1-50$, raggio=10; 101-150 $\sigma = 1-50$, raggio=20; 151-200 $\sigma = 1-50$, raggio=30; 201-250 $\sigma = 1-50$, raggio=40).

2.7.1 Analisi delle immagini con nebbia

Un caso speciale sono invece le sequenze con nebbia: questa tipologia di degradazione non genera infatti valori mancanti nella mappa di disparità, riducendone la densità. Il principale effetto della nebbia è quello di modificare i valori di disparità rendendoli errati. La nebbia tende quindi a rendere l'immagine omogenea, mascherando i bordi e riducendo la nitidezza.



(a) (b)

Figura 2.15: Rilevazioni errate in caso di nebbia; si nota come i falsi positivi iniziano ad apparire nell'area più lontana, dove la nebbia è più densa.

In figura 2.16 si mostra l'andamento dell'errore in caso di nebbia rispetto la previsione basata sulle *features* di densità (Prediction N). Appare evidente che la relazione tra errore e *features* è meno evidente e significativa. Intorno al frame 30 la densità di disparità decresce leggermente e rimane costante fino al termine della simulazione.

In figura 2.16 (Prediction F) si riporta la predizione dell'errore utilizzando, come caratteristiche distintive, le *features DSIRangeDiff* e *DSIRangeMean* dell'ultima banda (vedi sezione 2.5). Si nota come la predizione sia decisamente migliore rispetto al caso precedente. Questo conferma l'analisi precedente: progressivamente, l'aggiunta della nebbia non causa cambiamenti significativi nella densità della mappa di disparità, ma modifica il valore medio di disparità. Sfortunatamente la relazione lineare matematica tra l'andamento dei valori assoluti di disparità e l'andamento dell'errore

dipende in modo significativo sia dai *range* di disparità (che dipendono dai parametri di calibrazione) che dalla scena specifica, quindi dall'andamento del terreno e dagli ostacoli. Ripetendo la stessa analisi su immagini differenti e sistemi stereo diversi porta inevitabilmente a diverse configurazioni di pesi e a diverse relazioni lineari tra valori di disparità e errori di rilevazione

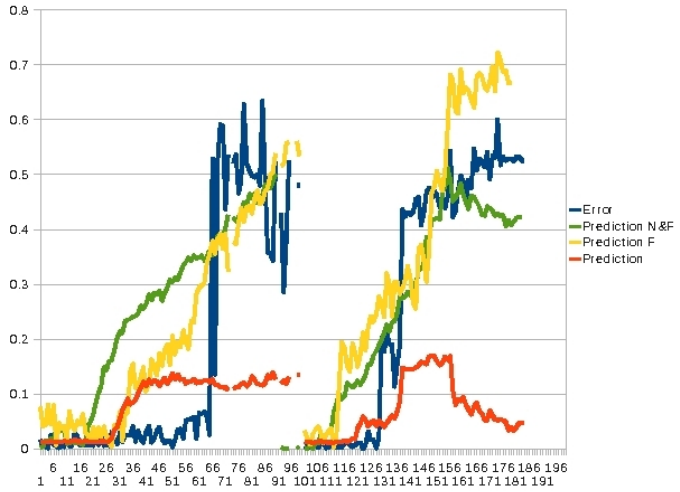


Figura 2.16: Predizione dell'errore con nebbia; errori misurati sulle immagini A (0-100) e B (101-200), degradate con nebbia, comparati con la predizione dell'errore basata sulle *features* di disparità. La predizione N considera i pesi individuati nella sezione 2.6.3; la predizione F si basa sulle *features* significative in caso di nebbia; le predizione N&F combina le *features* di densità con quelle individuate in caso di nebbia.

Valutando la combinazione tra le *features* di densità e le caratteristiche distintive rilevate in caso di nebbia (figura 2.16, Prediction N&F) si ottengono risultati abbastanza buoni poiché per queste immagini i valori delle *features* di densità sono molto bassi e la predizione è quindi dominata dalle caratteristiche estratte nel caso di nebbia. Un risultato simile si otterrebbe valutando la predizione dell'errore utilizzando entrambe le tipologie di caratteristiche sulle immagini utilizzate per addestrare

le *features* di densità: in questo caso le caratteristiche dominanti sarebbero quelle di densità, portando comunque buone performance di predizione.

Questo conferma la potenzialità di questa analisi nella selezione delle *features* discriminative e nell'assegnazione dei pesi corretti. Tuttavia, con le *features DSIRangeDiff* e *DSIRangeMean* rimarrebbe il problema delle dipendenze dei pesi dalle specifiche immagini e sistemi stereo: per queste ragioni si escludono queste due caratteristiche dall'insieme delle *features* distintive.

In conclusione non è facile rilevare malfunzionamenti come quelli generati in caso di nebbia, senza fare specifiche assunzioni sulla scena.

2.8 Conclusioni e sviluppi futuri

L'obiettivo del lavoro è stato quello di individuare una relazione tra alcune proprietà della mappa di disparità e l'errore di rilevamento ostacoli del sistema di percezione sviluppato. Al termine di questa analisi si può concludere che esiste effettivamente una relazione lineare tra la densità globale dell'immagine di disparità e l'errore di riconoscimento, caratterizzata dai coefficienti riportati in tabella 2.6.3. Utilizzando un insieme di caratteristiche globali e locali la stima dell'errore si riferisce al caso di scenario peggiore (quello in cui l'ostacolo si trova esattamente nella regione più degradata dell'immagine). Alcuni tipi di disturbo non incidono sulla densità della mappa di disparità, come ad esempio la nebbia: in questo caso cambiano soltanto i valori di disparità quindi è difficile fare considerazioni e previsioni a riguardo senza assunzioni di terreno piatto o dimensione e posizione degli oggetti. Nelle immagini con la nebbia esiste una relazione tra errore di riconoscimento e valori di disparità, tuttavia questo legame dipende strettamente dalle immagini e/o dal sistema stereo utilizzato. Dalla valutazione dei tempi di esecuzione è stato riscontrato che l'elaborazione dell'errore di predizione richiede circa 670 μs su di un Intel Core 2 a 1.86GHz con 1.96 di RAM. Per estendere la versatilità del sistema, l'attività futura riguarderà lo studio di metodologie alternative di analisi numerica al fine di garantire l'affidabilità di funzionamento in tutte le condizioni di degradazione possibili.

Capitolo 3

Sistema di assistenza per la manovra di sorpasso

3.1 Descrizione del problema

Il terzo sistema studiato durante il periodo del dottorato ha come obiettivo il monitoraggio dell'area relativa al punto cieco dell'autista (*blind spot*), cioè la regione che non può essere vista dal guidatore né tramite lo specchietto retrovisore né attraverso quello laterale (figura 3.1).

L'attività di ricerca si è concentrata principalmente sullo studio delle varie metodologie necessarie per la realizzazione di questa applicazione. E' stato successivamente sviluppato un algoritmo per la risoluzione di questa problematica.

3.2 Sensori impiegati

Negli ultimi anni i sistemi di assistenza alla guida sono diventati di fondamentale importanza per i produttori di automobili. Al giorno d'oggi c'è un forte interesse riguardo l'equipaggiamento dei veicoli con telecamere (anche stereo [67, 62]) e piattaforme di elaborazione di immagini per applicare algoritmi in grado di aiutare il guidatore a rimanere entro i limiti della carreggiata, rilevare ed evitare collisioni con

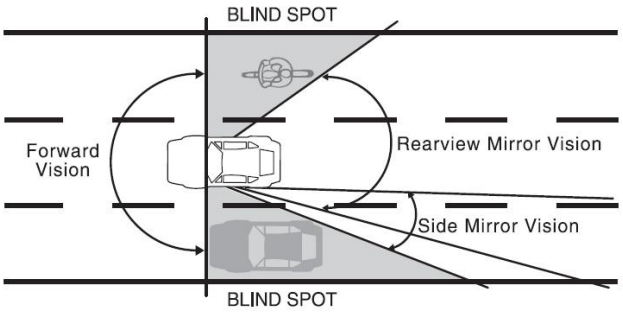


Figura 3.1: Area relativa al Blind Spot.

veicoli che sopraggiungono nella corsia di sorpasso o che cercano di cambiare corsia [3, 5]. Anche se i soli radar costituiscono un'alternativa molto valida, come mostrato in [63], altri autori propongono comunque la fusione di sensori, come radar e visione stereo [57] o laser scanner e visione stereo [64], quali metodi efficaci per evitare collisioni. Il progetto sviluppato nell'ambito di questa tesi prevede l'utilizzo di una telecamera, posta all'altezza dello specchietto laterale, in modo da inquadrare l'area corrispondente al *blind spot* per rilevare la presenza di veicoli (figura 3.2).

3.3 Acquisizione immagini

Diversi fattori, tipici delle applicazioni *automotive*, devono essere presi in considerazione nella realizzazione di un sistema di monitoraggio del *blind spot* basato sull'acquisizione di immagini provenienti da una singola telecamera: le oscillazioni e le vibrazioni dovute alla dinamicità dello scenario, la rotazione della telecamera, la deformazione prospettica introdotta. Inoltre dovendo discriminare tra veicoli e sfondo, entrambi in movimento, l'applicazione da realizzare si dimostra alquanto complessa.

Una prima problematica affrontata riguarda appunto la rotazione delle immagini acquisite (figura 3.3): questo effetto è dovuto all'angolo di roll non nullo della telecamera.

Figura 3.2: Telecamera posta al di sotto dello specchietto laterale sulla macchina VisLab.

Figura 3.3: Immagini ruotate acquisite con angolo di roll non nullo.

Per ovviare a questo problema l'immagine in ingresso è stata trasformata: inizialmente si pensava di effettuare una rotazione di un angolo tale da annullare questo effetto, ma questa scelta avrebbe comportato un'incongruenza tra i parametri della telecamera e le immagini acquisite. Per questo motivo si è scelto di agire direttamente sui parametri della telecamera, rettificando le immagini: i parametri sono quindi acquisti e modificati al fine di rendere nullo l'angolo di rollio; successivamente, attraverso la trasformazione omografica, i nuovi parametri sono assegnati alla telecamera. Il risultato della rettificazione è mostrato in figura 3.4.

Figura 3.4: Esempi di immagini rettificate per rendere nullo l'angolo di rollio della telecamera.

Per rimuovere dall'immagine la regione occupata dal veicolo su cui è montata la telecamera e le parti prive di informazione introdotte dalla rettificazione si utilizza una maschera binaria, mostrata in figura 3.5: solamente i pixel delimitati dall'area nera sono considerati informazione utile.

3.4 Riconoscimento veicoli

Il passo successivo per la realizzazione del sistema di assistenza al cambio di corsia riguarda il riconoscimento dei veicoli in sorpasso. Le principali classi di metodi impiegate prevedono l'analisi del flusso ottico [12, 5] e il riconoscimento di pattern [2]

Figura 3.5: Esempio di maschera utilizzata per filtrare l'area occupata dal veicolo su cui è montata la telecamera.

3.4.1 Stima del flusso ottico

Una parte degli approcci presenti in letteratura si concentra sullo studio del moto relativo tra gli oggetti inseguiti e il sensore della telecamera (che rappresenta il punto di vista dell'immagine). Una stima del campo di moto è data dal flusso ottico, che può essere utilizzato per rappresentare la distribuzione di velocità apparenti date dal movimento degli oggetti nell'immagine. Nel rilevamento del moto degli oggetti l'analisi del flusso ottico è molto utilizzata e, se combinata con un algoritmo di *tracking*, fornisce una buona stima del movimento, separandolo in raggruppamenti simili. La stima del moto può essere basata solo sul flusso ottico o combinata, ad esempio, con informazioni sui bordi [101], sui colori [8] o con l'utilizzo degli *Snake di Kalman* [84]. La rilevazione dei veicoli in movimento attraverso il campo di moto può essere complessa poiché, in presenza di uno scenario dinamico, il flusso ottico dello sfondo e degli oggetti in movimento risulta molto simile (figura 3.6).

Per riconoscere un veicolo in sorpasso tutte le direzioni degli elementi nell'immagine sono elaborate: se un oggetto in movimento si trova vicino alla telecamera viene considerato come veicolo in sorpasso, altrimenti è un elemento dello sfondo.

Per risolvere questo problema, in [6], viene proposto un metodo computazionalmente semplice ed efficiente che combina la stima del flusso ottico al concetto di

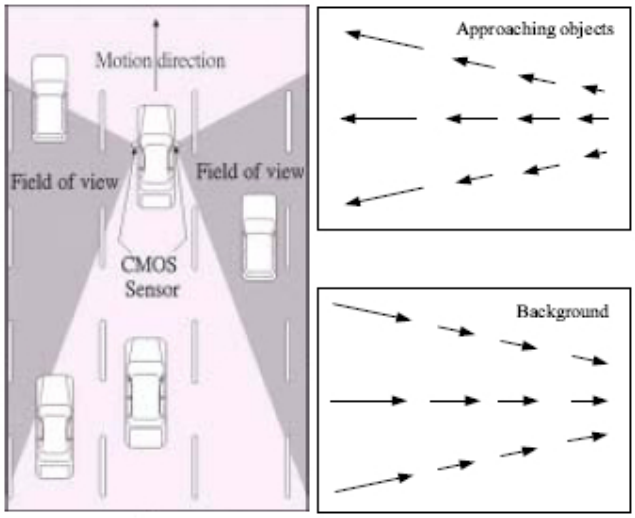


Figura 3.6: Descrizione dello scenario e del movimento degli oggetti nell'immagine.

FOE (Focus Of Expansion); invece di ricercare i veicoli sull'intera immagine viene stimato un piano stradale per limitare la ricerca ai veicoli appartenenti a quest'area. L'analisi del flusso ottico può essere di tipo denso, se il calcolo del campo di moto avviene per ogni pixel dell'immagine, oppure di tipo sparso: in questo caso il flusso ottico è calcolato solo per alcune *features* con caratteristiche particolari [90, 51] allo scopo di individuare specifiche regioni di interesse dove investigare la presenza di un veicolo.

In [99] viene proposta una metodologia per la rilevazione di veicoli in fase di sorpasso basata su flusso ottico sparso e sull'elaborazione degli autovalori. Si separano nell'immagine (figura 3.7) le regioni dinamiche da quelle statiche e successivamente si utilizza il flusso ottico sparso e omogeneo per rendere la rilevazione più robusta ai movimenti improvvisi e alle vibrazioni della camera.

Un aspetto importante dell'analisi del flusso ottico è la sua versatilità: permette infatti di discriminare tra oggetti statici, elementi che si muovono nella stessa direzione del veicolo (dove la telecamera è montata) e veicoli che si muovono in direzione opposta rispetto la telecamera.

Figura 3.7: Sottrazione delle aree statiche e dinamiche dallo sfondo.

Il primo passo per la stima del flusso ottico riguarda l'estrazione dall'immagine di un insieme di caratteristiche distintive: queste *features* sono poi ricercate tra frame differenti per determinarne il loro movimento rispetto la telecamera. Successivamente le *features* sono classificate in modo da selezionare e raggruppare insieme tutti i punti che identificano uno stesso veicolo.

3.4.2 Pattern Recognition

Nell'ambito del riconoscimento dei veicoli, le tecniche di *pattern recognition* prevedono di individuare, all'interno delle immagini, alcune caratteristiche specifiche, come ad esempio la targa, i fanali, il paraurti, le ruote o la parte frontale del veicolo; la ricerca è generalmente supportata dalla presenza di un classificatore [76]. Spesso si utilizzano anche le informazioni relative al gradiente: in [58] viene illustrato un sistema *real time* basato sulla generazione di ipotesi costruite sul gradiente e sul controllo delle stesse a seconda dell'aspetto. Per verificare le ipotesi sul veicolo viene utilizzato un classificatore di tipo *Adaboost*, mentre per irrobustire i risultati l'approccio prevede l'utilizzo di una procedura di tracking simile a SVT (Support Vector Tracking).

Quando un veicolo in sorpasso si trova vicino alla telecamera e la sua parte frontale è occlusa dalla presenza del veicolo su cui è montata la telecamera, si ricorre all'utilizzo di approcci basati su *templare matching* [76] (figura 3.8).

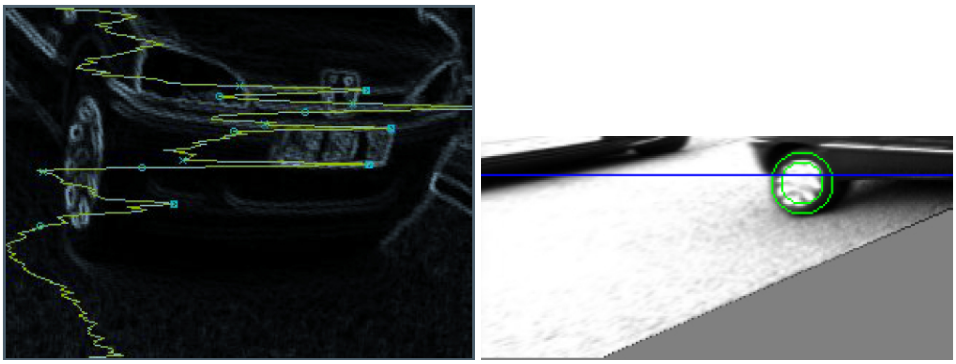


Figura 3.8: Rilevazione delle ruote e della parte frontale di un veicolo tramite *templare matching*.

Lo svantaggio per questa tipologia di approcci riguarda la complessità nella generalizzazione della rilevazione per tutti i tipi di veicoli: poiché macchine, moto, camion hanno caratteristiche molto differenti, il loro riconoscimento può richiedere elaborazioni aggiuntive che rischiano di incrementare significativamente il costo computazionale.

Un'alternativa per le tecniche di *pattern analysis* è la ricerca delle ombre [98] sotto ai veicoli: queste possono essere utilizzate per delimitare la regione di interesse dove verificare l'eventuale presenza di veicoli, utilizzando informazioni distintive come simmetrie[94], bordi [93, 60] o forme (figura 3.9).

Figura 3.9: Utilizzo delle ombre per il riconoscimento dei veicoli.

In questo caso l'elaborazione comporta bassi costi computazionali a fronte, però, di una limitazione delle generale performance: l'estrazione, da una singola immagine, di informazioni sulla posizione degli oggetti non è sempre accurata.

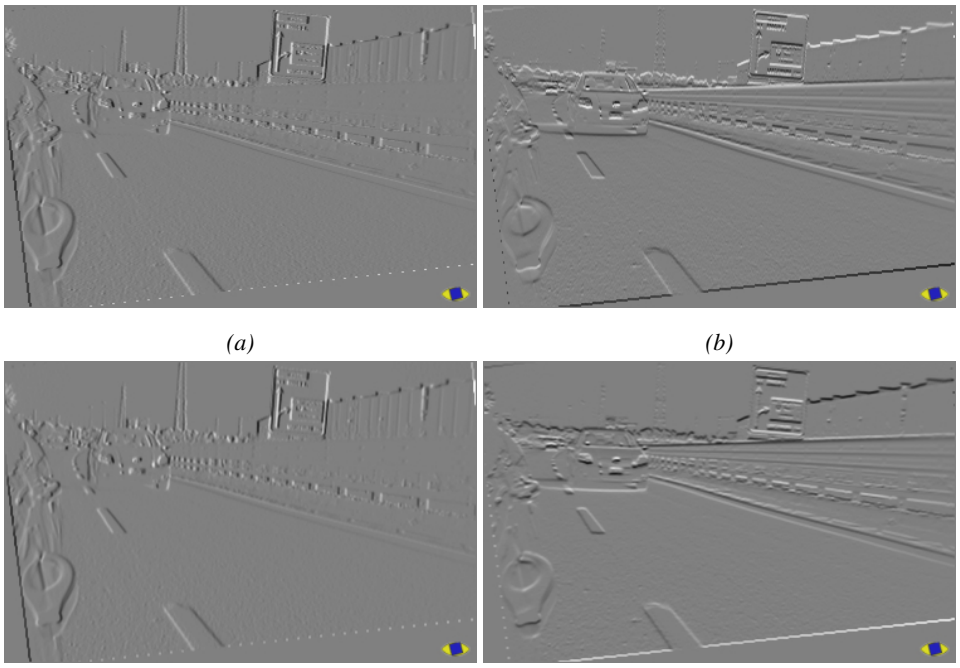
3.5 Estrazione punti caratteristici

Ogni sistema basato sul tracciamento di punti caratteristici dipende strettamente dall'algoritmo di estrazione delle *features* utilizzato: se una caratteristica rilevata in un frame non appare in quello successivo i dati di ingresso per la procedura di *tracking* rischiano di essere inaffidabili.

Un aspetto importante da considerare nell'estrazione delle *features* è che alcuni punti dell'immagine non possono essere inseguiti correttamente: nelle zone morfologicamente e cromaticamente omogenee, prive di tessitura, è impossibile determinare esattamente dove si trova, nel frame corrente, un punto precedentemente rilevato; per questa motivazione per alcuni gruppi di pixel è impossibile tracciarne il movimento. Il bordo di un oggetto permette invece di determinare lo spostamento solo nella direzione perpendicolare al bordo stesso; in presenza di un angolo (*corner*) c'è invece un cambiamento significativo di luminosità che permette di determinare il movimento lungo entrambi gli assi. L'elaborazione delle caratteristiche può essere eseguita elaborando direttamente i toni di grigio delle immagini oppure utilizzando algoritmi basati sull'estrazione dei bordi; in questo caso si considerano tutti i punti con la curvatura massima o dove i bordi si incontrano.

In figura 3.10 *a-b* e in figura 3.10*c-d* sono mostrate le immagini risultanti dall'applicazione di due metodi tipicamente utilizzati per l'esaltazione dei bordi: rispettivamente, l'operatore di gradiente ed il filtro di *Sobel*. Nelle applicazioni *automotive* la scelta del metodo di estrazione delle *feature* si dimostra quindi cruciale per garantire l'affidabilità dei risultati: in particolare la stabilità temporale, la precisione nell'individuazione delle caratteristiche e il costo computazione sono aspetti importanti da considerare nella scelta.

In figura 3.11 e in figura 3.12 sono mostrati i risultati ottenuti dall'applicazione dei metodi di estrazione delle *features* basati, rispettivamente, su Shi-Tomasi[90] e



(a) *(b)*

(c) *(d)*

Figura 3.10: Estrazione dei bordi *(a)* orizzontali e *(b)* verticali tramite l'operatore gradiente. Estrazione dei bordi *(c)* orizzontali e *(d)* verticali tramite il filtro di *Sobel*.

Harris [51].

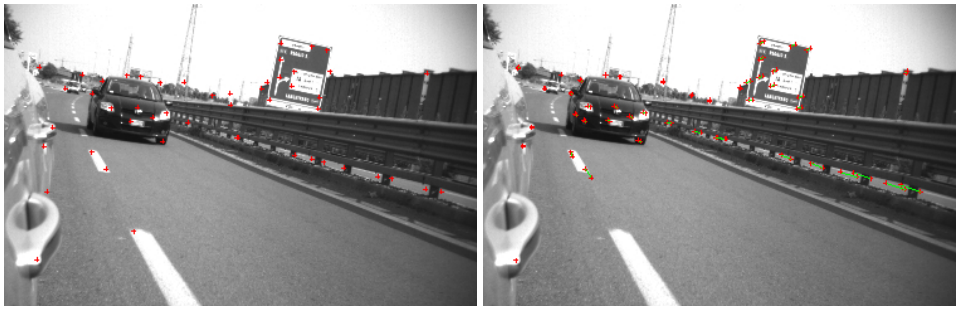


Figura 3.11: Metodo di estrazione delle caratteristiche basato su Shi-Tomasi.

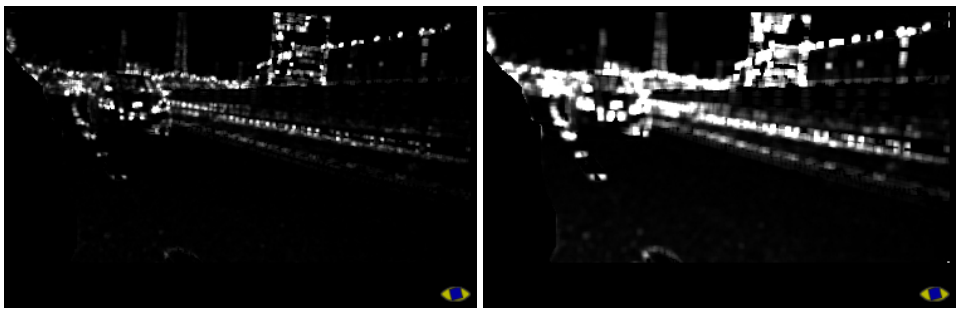


Figura 3.12: Rilevazione dei *corner* di Harris utilizzando, rispettivamente, finestre a 3×3 e 5×5 pixel. A partire dai bordi dell'immagine tutti i *corner* significativi sono rilevati. Aumentando la dimensione della finestra la rilevazione delle caratteristiche diventa più precisa, ma nel contempo cresce il costo computazione.

In questo progetto di ricerca si è preferito utilizzare il metodo di Harris-Stephens poiché presenta tempi di elaborazione inferiori ma prestazioni comunque paragonabili.

3.5.1 Tracking delle features

Per l'inseguimento dei punti caratteristici si ricerca una correlazione coerente tra oggetti in fotogrammi differenti: l'obiettivo è appunto quello di determinare dove si

trova nel *frame* corrente una *feature* precedentemente rilevata, in modo da calcolarne il movimento e stimarne la posizione futura.

Per decidere come strutturare un algoritmo di *tracking* è quindi importante conoscere quali dati si hanno a disposizione: alcuni sistemi sfruttano, ad esempio, le caratteristiche di distribuzione del colore altri invece i bordi; l'uso di immagini a colori è comunque limitato dal fatto che il colore si dimostra particolarmente sensibile alle condizioni di illuminazione e alle condizioni climatiche.

Nella scelta della tecnica di inseguimento adatta bisogna inoltre considerare altri aspetti, come ad esempio il modello di *tracking* da utilizzare (statistico, volumetrici ecc), la dimensione dello spazio delle *features* (in genere bidimensionale o tridimensionale), il tipo di sistema di acquisizione (monoculare, stereo, trinoculare), la mobilità della telecamera (statica o dinamica).

Alcune classi di algoritmi di *tracking* sfruttano l'informazione conosciuta a priori sugli oggetti della scena, definendone un modello da far corrispondere poi alla situazione reale: le informazioni ottenute sono poi utilizzate per il successivo riconoscimento delle azioni e dei comportamenti da intraprendere, giungendo così ad un maggiore livello di astrazione nella comprensione del movimento.

Alternativamente il *tracking* può essere eseguito dopo la fase di rilevazione, utilizzando come ingresso un insieme di regioni di interesse, senza alcuna conoscenza di alto livello: in questo caso il sistema sfrutta la rilevazione del movimento e l'inseguimento delle aree individuate in modo da determinare le traiettorie degli oggetti in movimento. Questa scelta permette di raggiungere una certa versatilità nello scenario di applicazione.

Un esempio di approcci non basati sul modello sono le Hidden Markov Model (HMM), tecniche probabilistiche basate sull'analisi di serie tempo discrete: la struttura base di un HMM è mostrata in figura 3.13. Uno stato generico è connesso con una certa probabilità agli altri stati; i parametri di interconnessione possono essere stimati utilizzando tecniche *feed forward* come l'algoritmo Baum-Welch. Le *features* da inseguire possono essere punti, rette o regioni bidimensionali.

I metodi di inseguimento delle *features* possono inoltre essere basati sullo studio del flusso ottico (figura 3.14): in questo caso l'informazione relativa allo spostamento

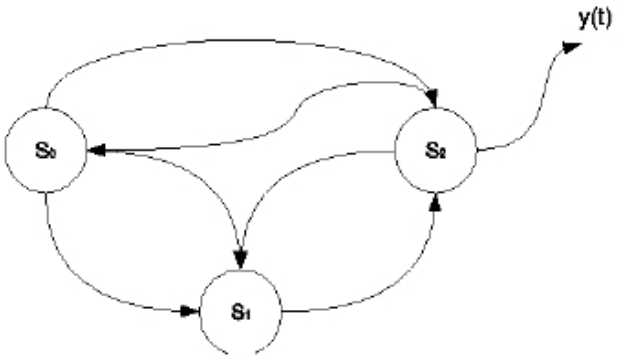


Figura 3.13: Esempio di HMM a tre stati; y(t) è l'osservazione derivante da ogni stato.

degli oggetti è importante per rilevare le dinamiche della scena e correlare le informazioni spaziali con la variazione temporale. A partire dei punti dell'immagine, la stima del il fuso ottico fornisce un'approssimazione bidimensionale del movimento tridimensionale del movimento dei punti. Per realizzare il *tracking* tutte le *features* della scena con movimento uniforme sono raggruppate insieme. Il processo è ripetuto su fotogrammi differenti per determinare nelle immagini tutti gli oggetti con movimento e velocità simili. Infine le traiettorie degli elementi sono determinante estraendo le informazioni relative ai bordi e al baricentro delle regioni di interesse. La conoscenza a priori sugli oggetti permette inoltre di migliorare le prestazioni.

Le difficoltà nella stima del movimento, tuttavia, sono molteplici e spesso riguardano le differenze tra il flusso ottico e il reale campo di movimento: poiché solo il movimento apparente può essere estratto da una sequenza, sono necessarie ulteriori assunzioni ad esempio sui cambi di luminosità, sulle proprietà degli oggetti, e sulle relazioni tra il movimento della scena tridimensionale e la relativa proiezione bidimensionale sul sensore per un'analisi quantitativa della scena.

Un'altra possibile tecnica per l'inseguimento delle *features* è il *templare matching*: si basa sulla localizzazione di regioni specifiche, inseguite tramite tecniche di correlazione. La locazione con il miglior risultato di correlazione è il miglior confronto tra il *template* e l'immagine.

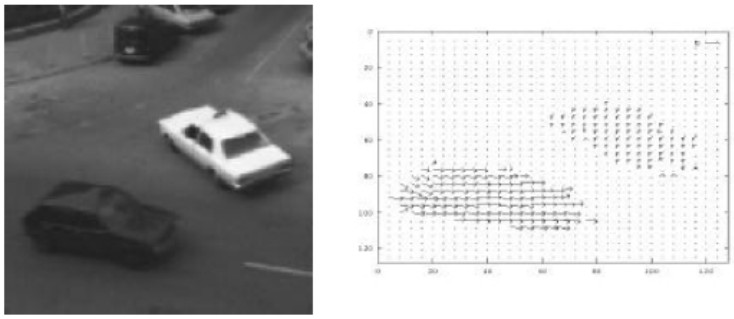


Figura 3.14: Stima del flusso ottico per il *tracking* delle *features*.

Generalmente l'area di interesse è selezionata in modo da rendere la propria rilevazione semplice: questo comporta l'elaborazione di specifici punti dell'immagine (*corner*, in corrispondenza dei cambi di luminosità) e i loro vicini. Il vantaggio per queste tipologie di tecniche riguarda la loro flessibilità. Il *template* è frequentemente aggiornato in modo da rendere il *tracking* robusto alla variazione della scena: se il *template* nel frame corrente è significativamente diverso da quello precedentemente rilevato, un nuovo *template* è rilevato a seconda della nuova regione di interesse; in presenza di piccoli cambiamenti, una versione mediata tra il vecchio e il nuovo *template* è calcolata ed usata come nuovo *template*. In questo modo, piccole deviazioni dovute al rumore vengono tollerate migliorando la stabilità del sistema in presenza di disturbi.

A seconda del tipo di informazione da inseguire, tecniche di correlazione basate sul confronto tra regioni o tra *features* possono essere impiegate per il *template matching*.

Nell'ambito del progetto analizzato per questa tesi di dottorato, la tecnica di inseguimento adottata prevede appunto l'uso del *templare matching* basato su *features tracking*: il risultato ottenuto è mostrato in figura 3.15; l'informazione inseguita corrisponde al livello di grigio in un intorno di ogni *feature* rilevata; si considera inoltre la posizione rispetto al *vanishing point* allo scopo di verificare che il movimento della *feature* sia compatibile con quello di un veicolo in sorpasso. In particolare in un tipico scenario dinamico, la direzione di un veicolo in sorpasso è opposta al movimento

del *vanishing point*.

Figura 3.15: Esempio di *template matching* per il *features tracking*.

Per determinare il movimento delle *features* rispetto al *vanishing point* si calcola l'angolo tra la retta che passante per la *feature* precedentemente rilevata ed il *vanishing point* e la retta definita dalla *feature* corrente e quella precedentemente rilevata.

L'angolo compreso risulta:

$$cos\alpha = \frac{\vec{v} \cdot \vec{w}}{||\vec{v}|| \cdot ||\vec{w}||} \tag{3.1}$$

In figura 3.16 è mostrato l'approccio adottato nell'ambito di questo progetto per la determinazione delle *features* appartenenti allo sfondo e ai veicoli in sorpasso, a seconda della loro direzione: V_p corrisponde al *vanishing point*, *prec* è il punto precedentemente rilevato, *curr* è la *feature* corrente e α è l'angolo compreso tra le rette definite dai vettori $\vec{v}$ e $\vec{w}$.

I *corner* inseguiti sono classificati, in base al loro spostamento, come:

- *Corner Sorpasso*: rappresentano i veicoli in fase di sorpasso, con una velocità relativa positiva maggiore rispetto a quella del veicolo sul quale è montato il sistema. La coppia di *feature* inseguite ha $cos\alpha$ positivo e superiore ad una certa soglia;

- *Corner Sfondo*: sono associati a tutto ciò che appartiene allo sfondo o che ha una velocità relativa negativa. La coppia di *feature* inseguite ha $cos\alpha$ negativo e inferiore ad una certa soglia;

- *Altri Corner*: corrispondono a tutti gli altri elementi dell'immagine caratterizzati da un movimento non compatibile con quelli precedentemente descritti.

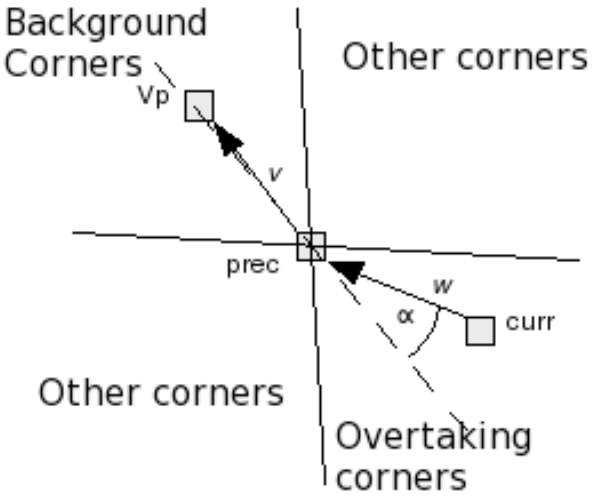


Figura 3.16: Elaborazione degli angoli tra *features* inseguite.

In figura 3.17 si mostrano le *features* rappresentate con colori diversi, a seconda dell'angolo compreso.

Per inseguire coppie di caratteristiche i migliori *match* tra punti correnti e passati sono calcolati, associando un voto specifico ad ogni coppia considerata. Inoltre quando il voto è minore di una certa soglia, la coppia di *features* relativa è scartata per rimuovere contributi di rumore. I risultati ottenuti sono mostrati in figura 3.18.

Per migliorare l'affidabilità dei risultati si memorizza l'informazione sulla storia passata delle *features* inseguite: si considerano la posizione della prima istanza e le coordinate delle *features* correnti e passate per calcolare il numero di frame in cui le coppie di *features* sono state correttamente rilevate. In questo modo è possibile calco-

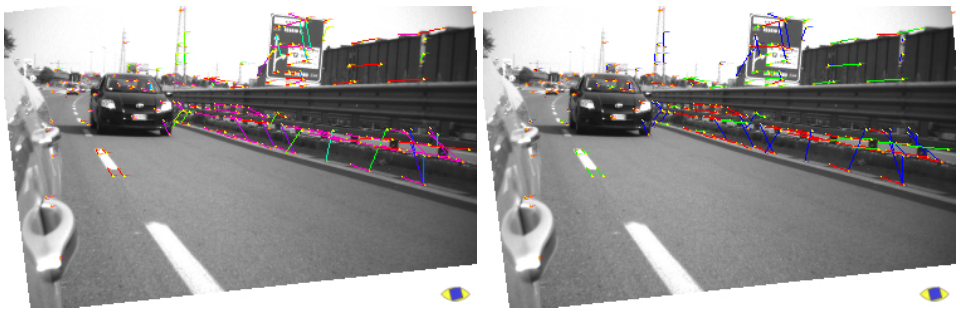


Figura 3.17: Classificazione delle *features* a seconda dell'angolo compreso. In particolare l'immagine destra mostra i risultati di classificazione: i veicoli in sorpasso sono rappresentati tramite segmenti rossi, gli elementi dello sfondo sono evidenziati con rette verdi mentre tutti gli altri oggetti sono rappresentati in blu.

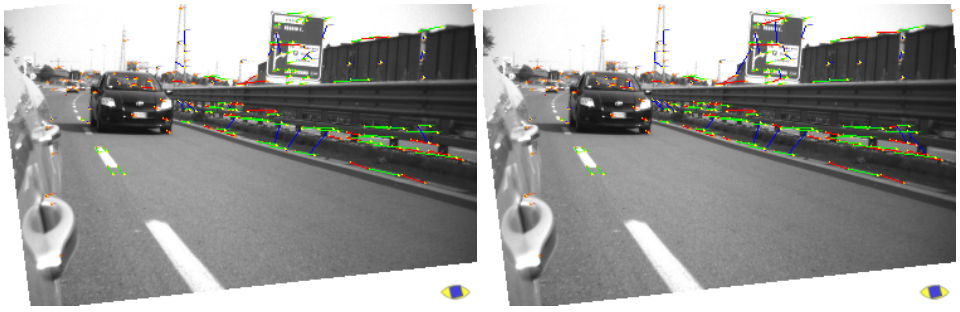


Figura 3.18: *Tracking* delle *features* affidabile e filtraggio del rumore.

lare l'età di ciascuna coppia e sfruttare questo valore per filtrare ulteriormente i risultati. In figura 3.19 sono mostrati i risultati ottenuti dal filtraggio delle caratteristiche in base alla loro storia.

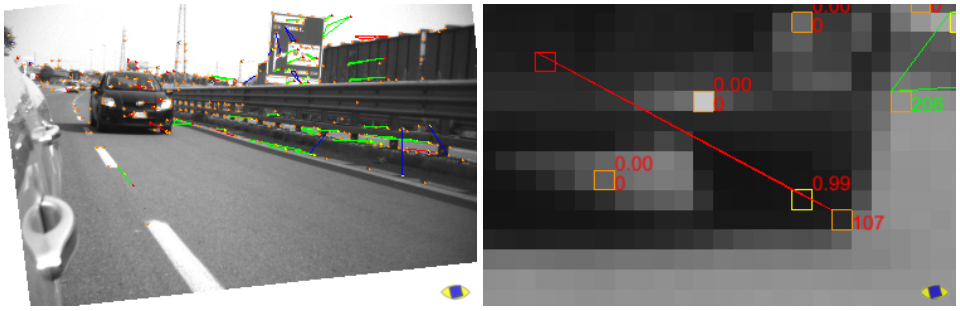


Figura 3.19: Elaborazione della storia per il *tracking* delle features.

3.6 Algoritmo sviluppato

Lo schema generale dell'algoritmo implementato per il monitoraggio del *blind spot* è riportato in figura 3.20: il modulo posto al livello superiore definisce un algoritmo di alto livello che interagisce con tre blocchi indipendenti che implementano le *routine* di basso livello.

Questi blocchi operano contemporaneamente e comunicano le loro uscite al modulo posto al livello superiore. Tutti i risultati sono poi fusi insieme dall'algoritmo di alto livello per determinare il comportamento da intraprendere.

Come detto in precedenza, per ogni immagine acquisita dalla telecamera i *corner* di Harris sono estratti ed inseguiti in modo da rilevare tutti gli oggetti in movimento della scena. I veicoli in sorpasso sono invece rilevati tramite approcci basati su *pattern matching*: si utilizzano tecniche basate sulle ricerca dell'ombra sotto il veicolo (figura 3.9).

Contemporaneamente si esegue la rilevazione delle linee, permettendo all'applicazione di alto livello di avvertire l'autista quando un veicolo in fase di sorpasso si trova in prossimità della linea più vicina al suo veicolo; inoltre la rilevazione di di-

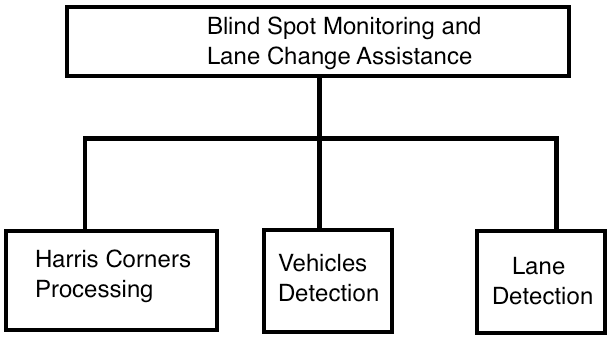


Figura 3.20: Schema dell'algoritmo per il monitoraggio del *Blind Spot*.

verse linee permette di valutare la loro intersezione per migliorare la stima del FOE ed ottenere un valore che non dipende strettamente dai parametri di calibrazione.

3.6.1 Riconoscimento tramite classificatori

Alternativamente all'uso delle ombre per la rilevazione dei veicoli sono state fatte alcune prove con il classificatore *AdaBoost* basato sull'utilizzo delle*feature di Haar*.

Sfruttando i dati di calibrazione della camera, la trasformata prospettica e la dimensione tipica dei veicoli è stato possibile determinare, all'interno delle immagini, specifiche aree dove verificare, tramite il classificatore, la presenza di un veicolo. La calibrazione del sistema di visione consente infatti di calcolare la trasformata prospettica inversa (IPM), che permette di passare dai punti in coordinate immagine ai punti in coordinate mondo. Attraverso l'uso della conoscenza sulla dimensione tipica dei veicoli e della trasformata prospettica, è possibile stabilire la dimensione dell'area occupata nell'immagine da un veicolo posto ad una specifica distanza dalla telecamera.

I vari tipi di autovetture presenti in commercio hanno un rapporto fra larghezza e altezza (*aspect ratio*) riconducibile ad un valore medio comune, pertanto possono

essere considerate come insieme omogeneo. Noto quindi il rapporto tra l'altezza e la larghezza dell'oggetto da identificare (in questo caso un veicolo), si individuano quindi le aree utili sulle quali applicare il classificatore Tramite la trasformata prospettica si determinano, nell'ipotesi di terreno piatto, i punti corrispondenti alle possibili basi delle auto. Per ogni riga dell'immagine si stabilisce quindi l'area che un veicolo può occupare, compatibilmente con la distanza codificata da quella specifica riga: in questo modo si ottiene, per ogni riga dell'immagine una dimensione del *bounding box* su cui testare il classificatore. Si campiona poi punto per punto tutta l'area utile dell'immagine processandola con il classificatore opportuno alla ricerca di un autoveicolo. Si applica quindi il classificatore *Ada Boost* a specifiche posizioni di immagine per verificare se quella regione contiene effettivamente un veicolo. L'uscita del classificatore corrisponde alla probabilità che quell'area contenga un veicolo.

Il risultato dell'applicazione di questa tecnica è mostrato in figura 3.21: si garantisce il funzionamento affidabile dell'algoritmo anche nel caso in cui l'ombra al di sotto dei veicoli non è visibile.

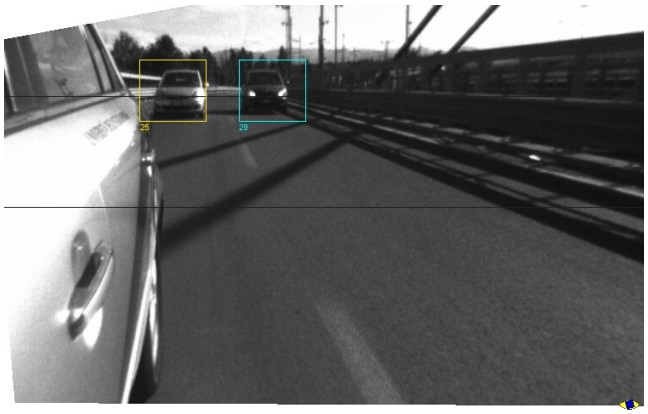


Figura 3.21: Utilizzo del classificatore *Ada Boost* per la rilevazione dei veicoli.

Seppur promettente quest'approccio mostra alcune limitazioni, soprattutto in presenza del beccheggio del veicolo che rende rumorosi i dati in ingresso alla trasformata prospettica, introducendo errori nella stima della base dei veicoli: si osserva perciò la poca accuratezza nella determinazione delle regioni contenti il veicolo.

Gli sviluppi futuri saranno sicuramente orientati verso il miglioramento di questa metodologia.

3.6.2 Ricerca delle linee

L'algoritmo di ricerca delle linee si basa sullo studio del modulo e della direzione dell'intensità del gradiente (figura 3.22) definiti dalle equazioni:

$$G_m = \sqrt{I_x^2 + I_y^2} \tag{3.2}$$

$$G_\theta = \tan^{-1}(\frac{I_y}{I_x}) \tag{3.3}$$

dove G_m rappresenta il modulo del gradiente, G_θ il rispettivo angolo, mentre I_x e I_y sono, rispettivamente, le componenti del gradiente orizzontale e verticale.

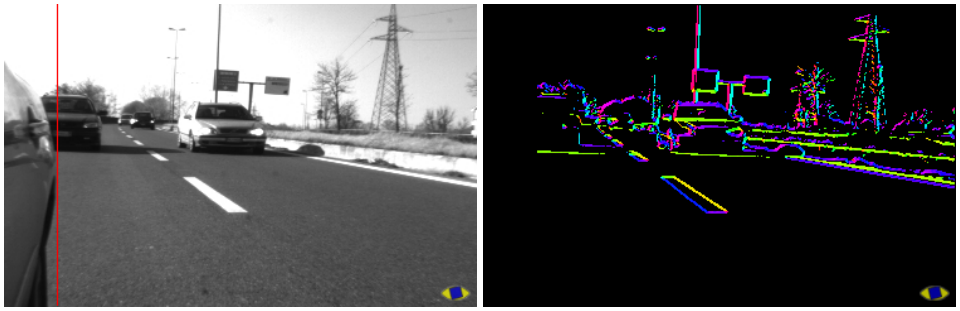


Figura 3.22: Colorazione dei punti dell'immagine in base all'angolo del gradiente.

L'immagine viene analizzata dall'alto verso il basso e da sinistra verso destra: per ogni pixel con un gradiente significativo viene controllato se la direzione del gradiente è compresa fra $0 e \pi$, memorizzando il punto corrispondente come un possibile bordo sinistro della linea. Se alla sua destra è presente un pixel con modulo simile e angolo opposto, esso rappresenta il bordo destro della linea. Nel caso in cui siano presenti entrambi i bordi della linea viene memorizzato il punto centrale come punto appartenente ad una linea della strada come rappresentato in figura 3.23. Partendo

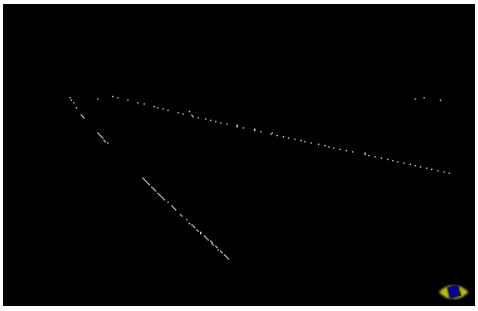


Figura 3.23: Rilevamento dei punti delle linee; la linea rossa si riferisce alla zona esclusa dalla ricerca perché contenente il profilo del veicolo.

dai punti appartenenti alle linee della strada raffigurati in figura 3.23 sono stati accorpati quelli appartenenti alla stessa linea: analizzando l'immagine dal basso verso l'alto e da sinistra verso destra e memorizzando il primo punto di una ipotetica linea si ricerca se nella riga superiore dell'immagine, in direzione del punto di fuga, è presente un altro punto appartenente ad una linea della strada, in una finestra di larghezza impostata. Per migliorare ulteriormente i risultati ottenuti sono state eliminate le discontinuità di primo ordine andando ad unire in tratti continui, non solo per i punti nella linea superiore dell'immagine, ma anche per quelli nella riga successiva. I tratti di retta ottenuti da questa operazione vengono classificati come appartenenti alla stessa linea della strada se la distanza tra loro non supera un determinato valore. Il risultato di questa classificazione è riportato in figura 3.24, dove vengono disegnati in colore diverso i tratti appartenenti a rette diverse. Sui tratti di retta classificati come appartenenti alla stessa linea viene applicata una regressione lineare ai minimi quadrati, considerata valida solo se l'errore di approssimazione non supera una soglia impostata. Le rette ottenute vengono ordinate prima in ordine decrescente in base al numero di punti e poi in ordine crescente in base alla loro posizione, da sinistra verso destra. Questa operazione colloca le rette più rilevanti in una lista ordinata in base alla posizione, consentendo di estrarre le linee della strada collocate nelle prime posizioni della lista. In figura 3.24 viene mostrato il risultato dell'operazione, dove vengono colorate in modo diverso le due linee della strada riconosciute

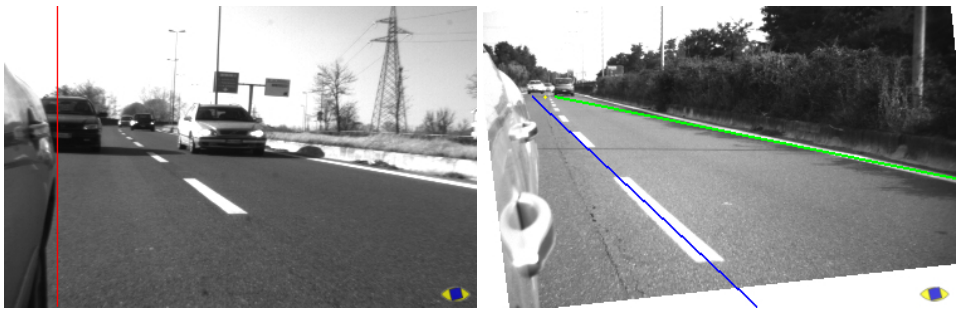


Figura 3.24: Classificazione dei tratti appartenenti alla stessa linea della strada ed estrazione delle linee.

3.6.3 Eliminazione punto cieco

Il modulo di alto livello combina i risultati dei blocchi sottostanti: per poter inseguire, tra fotogrammi differenti, gli elementi selezionati è necessario estrarre un numero significativo di *corner* di Harris. Al fine di identificare, all'interno della zona di pericolo, tutti i veicoli in sorpasso si selezionano, tra tutti i possibili candidati, solo se le auto individuate si trovano ad una certa distanza dalla seconda linea (se questa è rilevata). Se la distanza dalla telecamera di un veicolo in sorpasso è minore di una certa soglia, un segnale acustico e visuale è attivato per avvertire l'autista riguardo la situazione di pericolo.

Un esempio dei risultati ottenuti è mostrato in figura 3.25.

Quando si emette il segnale acustico l'area relativa al *blind spot* è monitorata per avvertire l'autista sulla presenza dei veicoli in quella regione: tuttavia il segnale rimane attivo solo se il numero dei *corner* di Harris relativi al veicolo in sorpasso è maggiore di una soglia specifica, altrimenti si disattiva automaticamente.

In figura 3.26 la presenza di un veicolo nell'area relativa al punto cieco (immagine destra) implica l'attivazione del segnale di pericolo per l'immagine destra; quando il veicolo si allontana dall'area pericolosa (immagine sinistra) il segnale si disattiva.

Nel sistema diverse soglie sono definite per garantire l'affidabilità dei risultati:

- *Distanza di pericolo*: definisce il limite spaziale entro cui la manovra di cambio di corsia è considerata pericolosa;

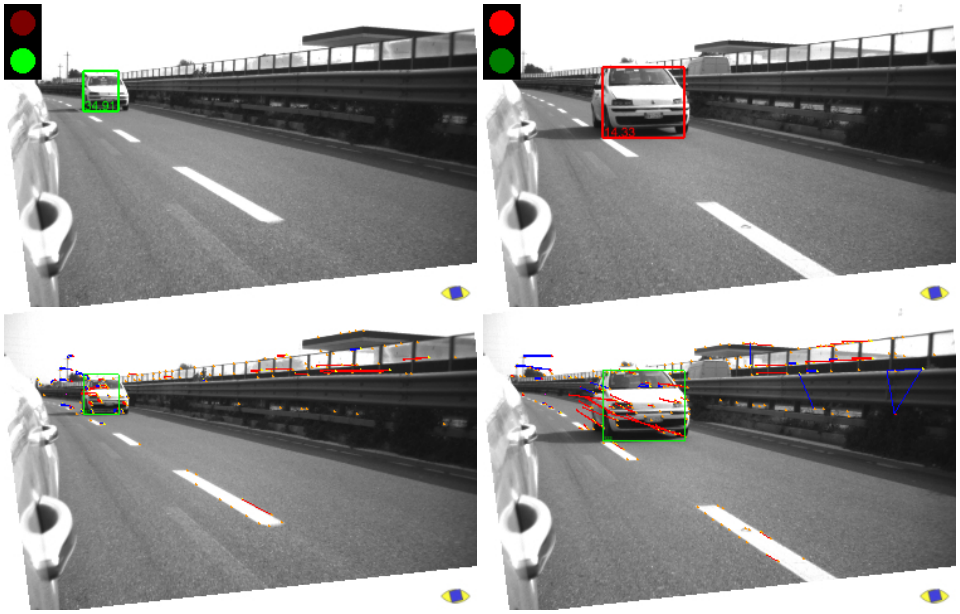


Figura 3.25: Esempio di un sistema per l'assistenza durante il cambio di corsia. L'immagine sinistra non rappresenta una situazione pericolosa perché i veicoli rilevati in prossimità della seconda linea sono lontani dalla telecamera, così il sistema non emette il segnale di avvertimento. Vice-versa lo scenario rappresentato dall'immagine a destra rappresenta una situazione critica: i veicoli rilevati sono vicini alla telecamera, quindi si attiva il segnale di pericolo (luce rossa) e la zona occupata dal veicolo in sorpasso è delimitata da un *bounding box* rosso per evidenziarne la pericolosità.

Figura 3.26: Esempio di rilevazione di un veicolo in corrispondenza del punto cieco.

- *Corner minimi*: stabilisce il numero minimo di *features* per considerare come veicolo un candidato rilevato;
- *Età minima*: è la storia minima delle coppie di *feature* per il *tracking* dei veicoli validi;
- *Corner di pericolo per il blind spot*: definisce il numero minimo di corner nell'area relativa al punto cieco per considerare la regione occupata da un veicolo.

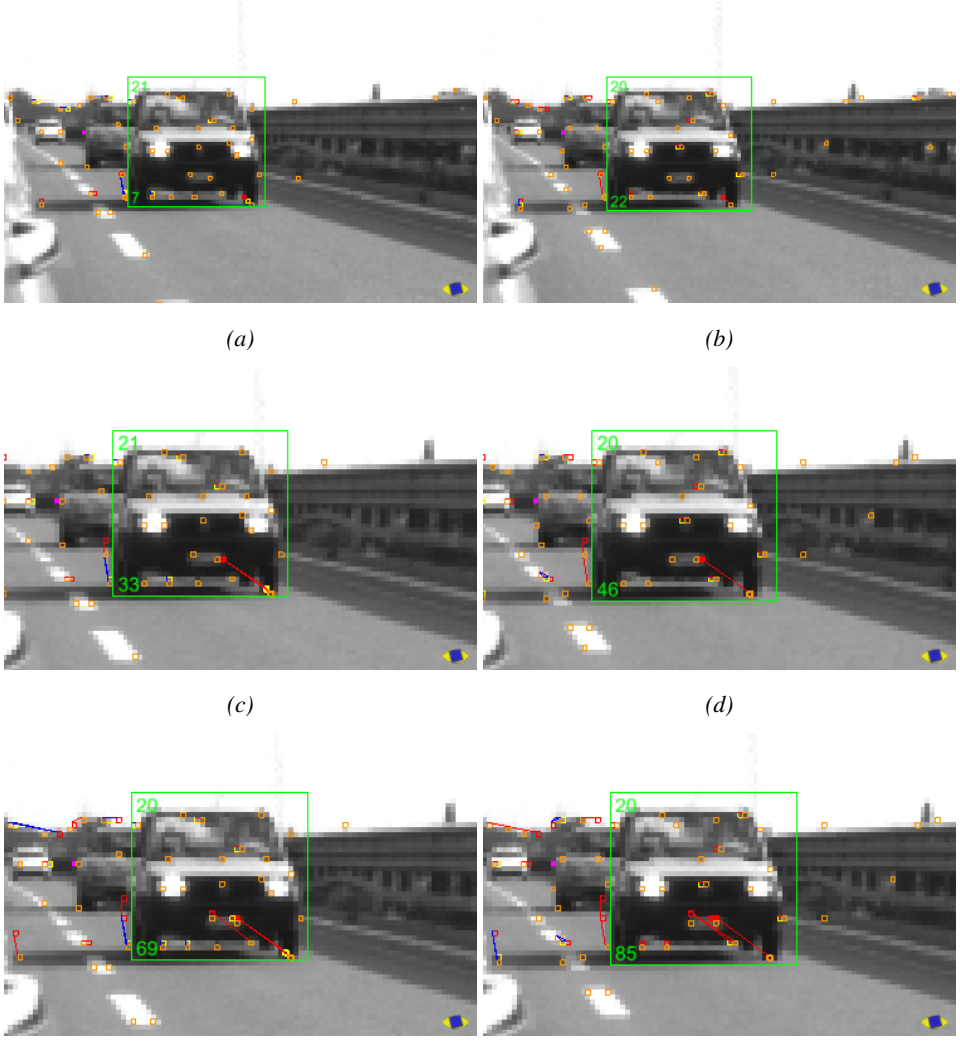
In figura 3.27 sono riportati i risultati relativi al *tracking* dei corner di Harris precedentemente descritti, variando la dimensione della finestra utilizzata per determinare il valori massimi nell'immagine di Harris.

3.7 Conclusioni e sviluppi futuri

Scopo di questo progetto è stato quello di sviluppare un sistema di visione artificiale per l'individuazione di veicoli e assistenza durante la fase di sorpasso, attraverso una telecamera posizionata sotto allo specchietto laterale del veicolo. Particolare attenzione è stata dedicata alla fase di calibrazione della telecamera per conoscere la distanza dei veicoli in sorpasso e alla preelaborazione dell'immagine che ha permesso di eliminare l'angolo di rollio della telecamera per una migliore individuazione dei veicoli. Il sistema utilizza metodi di visione basati sull'estrazione delle *features* e sul loro inseguimento per rilevare i veicoli in fase di sorpasso. Il sistema è stato testato su diverse sequenze di immagini diurne, ed è stata dimostrata l'affidabilità nell'individuazione di tutti i tipi di veicoli tranne i motocicli e gli autocarri. I tempi di esecuzione sono in linea con le specifiche real time del progetto consentendo un'elaborazione ad un frame rate di 15Hz [1].

I possibili sviluppi futuri di questo progetto di tesi possono essere orientati verso il miglioramento della fase di individuazione dei veicoli, integrando, oltre all'inse-

[1] Le prove sono state effettuate su un PC del laboratorio di visione artificiale dotato di processore Intel Pentium 4 2.80GHz e di 2GB di memoria RAM.



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figura 3.27: Tracking dei corner di Harris con finestre di varie dimensioni, *(a)-(b)* 5 pixel, *(c)-(d)* 7 pixel, *(e)-(f)* 9 pixel. Aumentando la dimensione della finestra i *corner* rilevati diventano più stabili migliorando le prestazioni dell'algoritmo di *tracking*.

guimento dei corner di Harris in fase di sorpasso, l'inseguimento delle regioni classificate come veicoli; questo permetterebbe la loro individuazione anche quando non viene rilevata l'ombra sotto al veicolo. Ulteriori studi possono inoltre essere dedicati all'integrazione della rilevazione basata su *AdaBoost* con la tecnica di estrazione delle *features*: questo garantirebbe una maggiore versatilità, permettendo il riconoscimento di motocicli e autocarri. Inoltre sarebbe interessante integrare i dati CAN del veicolo per conoscerne la velocità e poter variare la soglia della distanza di sicurezza in base alla velocità dei veicoli in sorpasso. Infine una fase di test più accurata, in condizioni atmosferiche e di illuminazione diverse, fornirebbe informazioni più precise sulle prestazioni del sistema.

Capitolo 4

Conclusioni sull'attività di ricerca svolta

I problemi di aspetto progettuale e le relative soluzioni che sono state descritte nel capitoli precedenti, consentono di stilare un resoconto critico sul lavoro svolto. Gli approcci utilizzati sono tipicamente empirici basati su considerazioni scaturite da esperienze acquisite o direttamente sul campo oppure dopo avere compiuto analisi in laboratorio di sequenze registrate. Il parametro progettuale che ha limitato maggiormente il presente studio è quello relativo ai limiti hardware del sistema, dovuti alla necessità di rendere i dispositivi accessibili e commercializzabili. I sistemi descritti devono inoltre essere in grado di operare solamente tramite l'analisi dei frame acquisiti, senza dimenticare che un eccessiva complessità computazionale rischierebbe di rallentare e quindi di compromettere l'intero sistema. La realtà è composta da situazioni naturali ed ambientali che non è possibile prevedere in toto, quindi in non si è possibile affermare con sicurezza che gli algoritmi saranno in grado di adattarsi alle numerose situazioni: per questo motivo le proposte risolutive descritte nei capitoli precedenti saranno sicuramente migliorabili alla luce delle nuove esperienze che verranno acquisite. Per quel che concerne i tempi di elaborazione, si osserva che i sistemi non necessitano di particolari calcoli od operazioni di conseguenza i tempi si mantengono relativamente bassi e non comportano problemi al resto del sistema. Dallo

studio effettuato si evince inoltre come i classificatori possano essere impiegati per la realizzazione di molteplici funzionalità, compatibilmente con lo scopo del sistema: consentono infatti di determinare, dato un insieme di possibili candidati individuati tramite il *processing* delle immagini, le classi di appartenenza di ciascun campione, come nel caso del sistema per il riconoscimento di segnaletica stradale verticale. Inoltre i classificatori permettono di prevedere le performance di riconoscimento a partire da un insieme di caratteristiche salienti estratte dall'immagine originale, come per il sistema di individuazione degli ostacoli in ambito *off -road*. Infine permettono di individuare, analizzando all'interno delle immagini regioni specifiche di dimensione fissa, un insieme di possibili candidati, come per il sistema di individuazione dei veicolo nell'area corrispondente al *blind spot*.

Appendice A

Reti Neurali

Le Reti Neurali nascono dall'idea di poter riprodurre alcune delle funzioni e capacità del cervello umano. L'area di applicazione dominante delle Reti Neurali è il riconoscimento di pattern (*pattern recognition*), e l'obiettivo fondamentale di tale caratteristica è la classificazione: dato un input la rete è in grado di analizzarlo e di formulare un output che corrisponda ad una determinata e significativa categorizzazione. Un esempio delle sue potenzialità è la capacità di riconoscere volti, voci, caratteri alfanumerici, etc. La classificazione consiste nel decidere a quale delle categorie indicate alla Rete Neurale, un pattern di input si avvicina maggiormente in termini di una distanza precedentemente definita. Sostanzialmente le reti neurali sono sistemi adattativi che imparano ad eseguire correttamente un determinato compito tramite esempi. L'informazione contenuta in questi ultimi è utilizzata per regolare un insieme di parametri interni alla rete in grado di influenzarne successivamente il funzionamento a regime: è naturale, dunque, che i principali campi di applicazione siano la soluzione di problemi di classificazione, clusterizzazione, ottimizzazione e predizione. Una rete neurale artificiale è un modello semplificato del sistema nervoso centrale umano; è composta da un certo numero di unità di base, detti neuroni, che sono in grado di comunicare tra di loro attraverso delle connessioni dette sinapsi (fig A.1). Queste possono essere organizzate in vari modi, a seconda del tipo di rete che si vuole creare. Partendo da esempi forniti in ingresso, la rete elabora per pas-

si successivi una sua rappresentazione interna dei dati, che produce, infine, l'uscita desiderata.

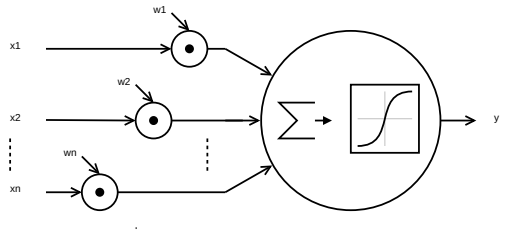


Figura A.1: Struttura di un neurone artificiale

A.1 Struttura della Rete Neurale

Ogni neurone è un'unità indipendente con una serie di connessioni in ingresso e una sola uscita. Ogni connessione in input ha una sua importanza, definita da una grandezza detta *peso*. Il neurone somma tutti gli input tenendo conto dei pesi delle relative connessioni, poi calcola l'uscita per mezzo di una funzione di attivazione, che può avere varie formulazioni. Chiamando:

- $x_1,\ x_2, \ldots,\ x_n$: i vari ingressi del neurone, che possono arrivare sia dall'input esterno che dall'output di altre unità;
- $w_1,\ w_2, \ldots,\ w_n$: i pesi delle relative connessioni;

possiamo definire il valore di uscita y_j associato al neurone j-esimo:

$$y_j \ = \ f\left(\sum_{i=1}^{n} x_i \cdot w_{ij}\right)$$

dove $f(x)$ è la funzione di attivazione del neurone. In fig A.2 sono riportate le funzioni di attivazione più comuni. Ogni rete neurale può essere divisa generalmente in tre strati: uno strato di ingresso, per mezzo del quale vengono forniti alla rete i dati

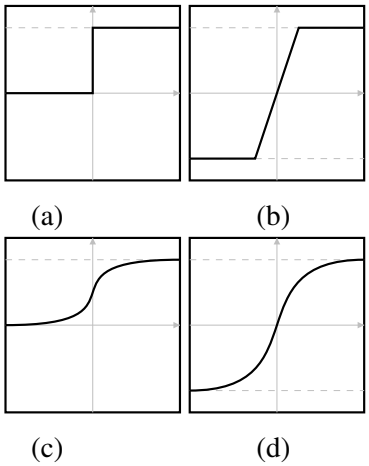


Figura A.2: Funzioni di attivazione tipiche: (a) logistica; (b) rampa; (c) sigmoidale; (d) tangente iperbolica

da elaborare, uno strato nascosto di elaborazione ed uno di uscita. Una tipica rete ha una struttura come quella mostrata in fig A.3. Generalmente i vari livelli sono completamente connessi con il precedente e il successivo, ed in alcuni casi sono anche connesse tra loro le unità che formano un singolo livello.

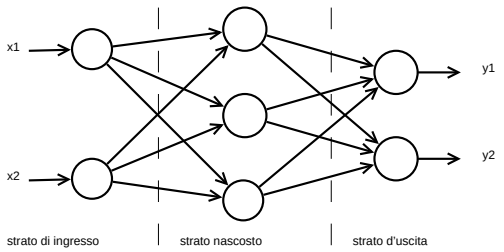


Figura A.3: Struttura di una tipica Rete Neurale Artificiale

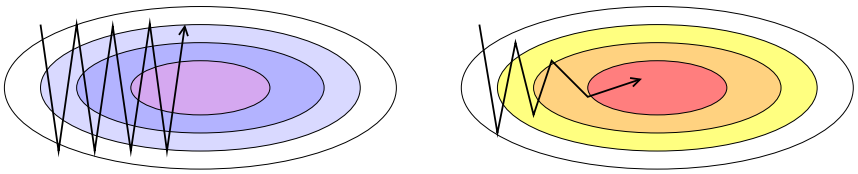
A.2 Apprendimento di una Rete Neurale

In generale, addestrare una rete neurale significa ottimizzazione la funzione dei pesi. Possiamo dividere i vari algoritmi di apprendimento in due grandi categorie: gli algoritmi con supervisore o supervisionati, e gli algoritmi non supervisionati. Il tipo di rete da utilizzare dipende molto dal problema che si deve affrontare. Per problemi basati su risultati ben definiti o conosciuti almeno in parte, è opportuno utilizzare reti supervisionate, che riescono a generalizzare partendo dagli esempi forniti in ingresso. Se invece gli input sono disorganizzati o non ben definiti e non si conosce abbastanza bene quale classificazione fornire come riferimento alla rete, è più indicato lasciare che sia questa a trovarla da sola, utilizzando quindi una rete auto-organizzante. In questa sede si intende approfondire l'analisi delle tecniche utilizzate nell'apprendimento delle reti supervisionate. Per rendere il comportamento della rete neurale conforme alla supervisione occorre minimizzare una funzione di errore che dipende dalla scelta dei pesi e misura l'errore rispetto alle informazioni del supervisore. Il problema di ottimizzare funzioni in grossi spazi è generalmente difficile per la potenziale presenza di minimi locali, che può rendere inefficaci le classiche metodologie di ottimizzazione basate sulla tecnica di massima discesa del gradiente. Il corretto funzionamento di una rete neurale sull'insieme di apprendimento non offre, ovviamente, garanzia di un altrettanto soddisfacente funzionamento su altri dati relativi allo stesso insieme da classificare, ma non utilizzati nella fase di apprendimento[1]. Inoltre, è evidente che anche l'architettura della rete gioca un ruolo fondamentale per l'efficienza della fase di apprendimento. Quando il numero delle unità nascoste cresce, non solo aumenta il potere computazionale, ma si può dimostrare che il problema della presenza dei minimi locali diventa progressivamente meno rilevante. Tuttavia, al crescere della dimensione della rete, la capacità di generalizzare su nuovi esempi tende a diminuire dato che il *fitting*[2] sull'insieme di apprendimento ha luogo in un enorme spazio di parametri vincolati solo da pochi esempi. Questo fatto origina una sorta di principio di indeterminazione dell'apprendimento secondo il quale non

[1] L'insieme dei dati utilizzati come esempi nella fase di addestramento è indicato dal termine *training set*.

[2] Termine che indica la corretta ottimizzazione della funzione dei pesi.

è possibile, al variare dei pesi della rete neurale, ottenere funzioni di errore senza incorrere in minimi locali[3]. Nasce dunque un problema critico, poiché se la rete tende a specializzarsi troppo sugli esempi forniti, perde il suo potere di generalizzazione, che ne costituiva uno tra i tanti pregi. Particolare successo ha avuto lo sviluppo dell'algoritmo di *backpropagation*. Tale metodo agisce sulla correzione dell'errore generato dalla differenza tra l'output prodotto dalla rete e quello desiderato, mediante la modifica dei pesi di tutte le connessioni. Per ogni ciclo di apprendimento viene fornita alla rete una coppia di vettori (input, output), grazie all'implementazione di una regola improntata al calcolo dell'errore, la δ-rule, si calcolano gli scarti a partire dall'ultimo strato procedendo a ritroso fino allo strato di ingresso[4], e infine, si correggono i valori dei pesi di tutte le connessioni, con riferimento ad uno specifico parametro, il *learning rate*. Il processo, ripetuto numerose volte, riduce progressivamente l'errore finché non si è scesi al di sotto di una certa soglia. Il parametro *learning rate* indica la misura con cui si modificano i pesi delle connessioni. In presenza di minimi locali o globali puó succedere che un *learning rate* troppo elevato non consenta alla soluzione di convergere, instaurando così un regime oscillatorio. Tramite l'utilizzo del parametro *momentum* si agisce sulla modifica dei pesi tenendo conto dei cambiamenti di segno del gradiente, smorzando le oscillazioni e agevolando la convergenza (fig A.4). Durante una tipica sessione di addestramento, la maggior parte degli sforzi



(a) (b)

Figura A.4: Convergenza della soluzione: (a) in assenza di momentum si può instaurare un regime oscillatorio; (b) in presenza di momentum si ha convergenza verso la soluzione mediante moto oscillatorio smorzato

[3] A tal proposito si usa il termine *over-fitting*.

[4] da cui il termine *retropropagazione*.

dell'utente sono volti al *tuning* dei parametri interni dell'algoritmo di apprendimento, in particolare il *learning rate* ed il *momentum*.

A.3 Gestione dell'over-fitting

La strategia di continuare l'addestramento finché l'errore quadratico medio sul set di apprendimento scende al di sotto di una certa soglia è, come già introdotto, molto rischiosa, perché la retropropagazione dell'errore è suscettibile a riprodurre in maniera sempre migliore i pattern di addestramento, al costo di diminuire la capacità di generalizzazione a pattern non utilizzati nella fase di apprendimento. Si osserva infatti che l'errore sull'insieme di validazione prima diminuisce, e poi aumenta, anche se l'errore sull'insieme di addestramento continua a diminuire (fig A.5). Ciò è dovuto al fatto che i pesi della rete vengono sintonizzati per riprodurre le peculiarità degli esempi usati nell'addestramento, che non sono rappresentative nella generalità dei pattern. Una delle strategie di maggior successo per evitare l'*overfitting*, descritto

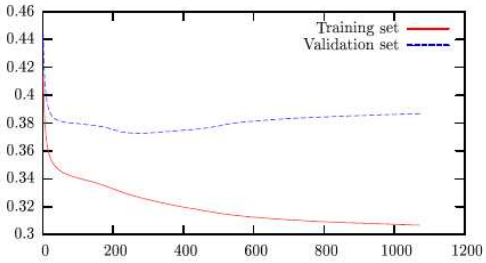


Figura A.5: Esempio di *overfitting*: grafico riportante l'andamento degli errori in addestramento e validazione al trascorrere delle epoche.

nel libro di Mitchell [74], prevede di usare un insieme di validazione unito all'insieme di addestramento. L'algoritmo controlla l'errore sull'insieme di validazione e usa l'insieme di apprendimento per calcolare la discesa del gradiente. Il numero ideale di iterazioni sarà quindi quello che produce l'errore minimo *sull'insieme di validazione*, dato che è quest'ultimo il miglior indicatore del comportamento della rete su esempi mai visti.

Appendice B

AdaBoost

Nell'ambito delle metodologie di apprendimento automatico supervisionato il concetto di *boosting* fu introdotto per la prima volta negli anni '80, nell'articolo [95]; i primi algoritmi furono successivamente descritti da Schapire [88] e Freund [43] che nel 1997 formularono insieme il meta-algoritmo *AdaBoost*. Il nome *AdaBoost* deriva dalla contrazione di *adaptive boost*: l'algoritmo di classificazione si basa su tecniche di *boosting* rafforzate da una componente adattativa legata all'errore commesso ad ogni iterazione. A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing) La tecnica del *boosting* consiste nel potenziare iterativamente, secondo una determinato criterio, un insieme di classificatori deboli[1] (*weak learner*) aggiungendoli ad un classificatore forte finale; dopo l'aggiunta di un *weak learner* i dati vengono nuovamente pesati in modo che le istanze classificate erroneamente guadagnino peso, mentre quelle corrette perdano peso. Pertanto, i futuri *weak learners* si concentreranno maggiormente sulle istanze che i precedenti *weak learners* classificavano

[1] Un classificatore è definito debole se è poco correlato con la corretta classificazione dei campioni, ma garantisce sempre prestazioni di predizione migliori rispetto ad un assegnamento casuale. Ad esempio in un problema di classificazione binario (a due classi) un *weak learner* è in grado di classificare correttamente il 50% più uno degli ingressi. Vice-versa uno *strong learner* è arbitrariamente ben correlato alla corretta classificazione degli esempi.

erroneamente. L'algoritmo così come viene presentato in [44], [45] e [46] presenta due versioni chiamate AdaBoost.M1 ed AdaBoost.M2.

B.1 AdaBoost versione M1

La versione M1 dell'algoritmo *AdaBoost* è definita secondo il seguente schema:

Input: una sequenza di m campioni $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$
con etichette $y_i \in Y = \{1, \ldots, k\}$
un algoritmo di apprendimento debole, **WeakLearn**
un intero T, che rappresenta il numero di iterazioni da eseguire.

Inizializza $D_1(i) = 1/m$ per ogni i.

Per ogni iterazione $t = 1, 2, \ldots, T$

1. Esegui l'algoritmo **WeakLearn**, utilizzando la distribuzione D_t.
2. Ritorna un'ipotesi $h_t : X \to Y$.
3. Calcola l'errore di h_t:
$$\varepsilon_t = \sum_{i:h_t(x_i) \neq Y_i} D_t(i)$$
Se $\varepsilon_t > 1/2$ allora setta $T = t - 1$ e interrompi il ciclo.
4. Setta $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$
5. Aggiorna la distribuzione D_t:
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{se } h_t(x_i) = y_i \\ 1 & \text{altrimenti} \end{cases}$$
dove Z_t è una costante di normalizzazione scelta in modo che D_{t+1} sia una distribuzione.

Output: l'ipotesi finale:
$$h_{fin} = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}$$

Si pone quindi in ingresso all'algoritmo un insieme di addestramento formato da m campioni $S = \langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$, in cui ogni generico x_i rappresenta un'istanza sorteggiata da uno spazio X (ad esempio un vettore di attributi), mentre $y_i \in Y$ è l'etichetta associata alla classe di appartenenza di x_i e ha cardinalità finita k. L'algoritmo *AdaBoost* prevede di richiamare per T iterazioni un classificatore debole. Ad ogni ciclo t, si esegue il *weak learner* con una distribuzione D_t sul training set S. Al termine il classificatore debole calcola l'ipotesi di classificazione $h_t : X \to Y$, che dovrebbe classificare una parte non banale dell'insieme di addestramento. L'obiettivo del *weak learner* è di trovare l'ipotesi h_t che minimizza l'errore di addestramento $\varepsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$. L'errore viene misurato rispetto alla distribuzione D_t fornita al *weak learner*. Questo processo viene ripetuto iterativamente per T cicli e alla fine,il *booster* combina le ipotesi deboli $h_1, h_2, \ldots h_T$ in una singola ipotesi finale h_{fin}. I vari schemi di *boosting* si differenziano tra loro per i diversi metodi utilizzati per calcolare la distribuzione D_t e l'ipotesi finale h_{fin}. AdaBoost.M1 risponde con la semplice regola mostrata nell'algoritmo. La distribuzione iniziale è distribuita uniformemente su S, quindi $D_1(i) = 1/m$ per ogni i. Per calcolare la distribuzione D_{t+1} dalla D_t, e l'ultima ipotesi debole h_t, moltiplichiamo il peso del campione i per un numero $\beta_t \in [0, 1)$ se h_t classifica x_i correttamente, altrimenti il peso rimane inalterato. Il peso viene in seguito rinormalizzato dividendolo per una costante di normalizzazione Z_t. Il numero β_t è calcolato, come mostrato nell'algoritmo, come funzione di ε_t. Come anticipato precedentemente, si nota come i campioni semplici, che vengono classificati in modo corretto da molte delle precedenti ipotesi deboli, hanno peso basso, mentre i campioni complicati, che tendono ad essere classificati in modo errato, acquistano peso maggiore. In questo modo *AdaBoost* concentra l'attenzione sui campioni che sembrano più difficilmente classificabili per il *weak learner*. Infine l'ipotesi finale h_{fin} è ottenuta come un voto pesato calcolato a partire dalle ipotesi deboli. Per una data istanza x, h_{fin} restituisce quindi un'etichetta y che massimizza la somma dei pesi delle ipotesi deboli che avevano predetto quell'etichetta. Il peso dell'ipotesi h_t è definito come $\ln(1/\beta_t)$ cosicché i pesi maggiori sono dati dalle ipotesi con gli errori più bassi. Un'importante proprietà di AdaBoost è data dal teorema 2.1: si dimostra che, anche se le ipotesi finali hanno costantemente errore leggermente migliore di $1/2$, l'errore

dell'ipotesi finale h_{fin} va a zero con velocità esponenziale. Per i problemi di classificazione binaria, questo vuol dire che, le ipotesi deboli, devono essere leggermente migliori di un valore di tipo casuale.

Teorema B.1 *Si suppone di avere un algoritmo di apprendimento debole weakLearn, che, chiamato dall'algoritmo AdaBoost.M1, genera delle ipotesi con errore* $\varepsilon_1, \ldots, \varepsilon_T$*, dove* ε_t *è definito come nell'algoritmo illustrato in precedenza.*
Si assuma che ogni $\varepsilon_t \leq 1/2$ *e si prenda* $\gamma_t = 1/2 - \varepsilon_t$*.*
Allora si è possibile dimostrare che il limite superiore sull'errore dell'ipotesi finale h_{fin}*, è:*

$$\frac{1}{m}|\{i : h_{fin}(x_i) \neq y_i\}| \leq \prod_{t=1}^{T} \sqrt{1-4{\gamma_t}^2} \leq \exp\left(-2\sum_{t=1}^{T} {\gamma_t}^2\right)$$

L'algoritmo *AdaBoost.M1* ha lo svantaggio di non riuscire a trattare ipotesi deboli che generano un errore maggiore di $1/2$. In generale l'errore atteso di un'ipotesi che assegni un'etichetta a caso è $1-1/k$, dove k è il numero delle possibili etichette. Così il requisito per *AdaBoost.M1* per $k = 2$ è che la predizione sia leggermente migliore della scelta casuale. Tuttavia, quando $k > 2$, il requisito di *AdaBoost.M1* è molto più restrittivo e difficile da trovare.

B.2 AdaBoost versione M2

L'algoritmo *AdaBoost.M2* cerca di sopperire alle problematiche riscontrare nella versione precedente estendendo la comunicazione tra l'algoritmo di *boosting* ed il *weak learner*; la descrizione schematica dell'algoritmo è la seguente:

Input: sequenza di m campioni $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$
con etichette $y_i \in Y = \{1, \ldots, k\}$
un algoritmo debole di apprendimento, **WeakLearn**
un intero T, che rappresenta il numero di iterazioni.

Considera $B = \{(i, y) : i \in \{1, \ldots, m\}, y \neq y_i\}$

Inizializza $D_1(i, y) = 1/|B|$ per $(i, y) \in B$.

Per ogni iterazione $t = 1, 2, \ldots, T$

1. esegui l'algoritmo **WeakLearn**, utilizzando la distribuzione D_t.
2. Ritorna un'ipotesi $h_t : X \times Y \rightarrow [0,1]$.
3. Calcola la pseudo perdita di h_t:

$$\varepsilon_t = \frac{1}{2} \sum_{(i,y) \in B} D_t(i,y)(1 - h_t(x_i, y_i) + h(x_i, y))$$

.

4. Setta $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$
5. Aggiorna la distribuzione D_t :

$$D_{t+1}(i,y) = \frac{D_t(i,y)}{Z_t} \cdot \beta_t^{(1/2)(1+h_t(x_i,y_i)-h(x_i,y))}$$

dove Z_t è una costante di normalizzazione scelta in modo che D_{t+1} sia una distribuzione.

Output: l'ipotesi finale:

$$h_{fin}(x) = \arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t} \right) h_t(x,y)$$

In questo caso il *weak learner* riesce a generare ipotesi molto più espressive che non si limitano a restituire una singola etichetta in Y, ma un insieme di possibili *label*. Si permette quindi al classificatore debole di indicare un "grado di confidenza": le ipotesi restituiscono un vettore di valori tra $[0,1]$ in cui il valore della y-esima componente rappresenta quanto è plausibile affermare che l'etichetta corretta da associare all'ingresso sia y. Le componenti con valore vicino ad 1 oppure a 0 corrispondono alle etichette considerate rispettivamente plausibili o non plausibili. Al fine di migliorare l'espressività del *weak learner*, si introduce maggiore complessità ai requisiti sulle prestazioni delle ipotesi deboli: in particolare, invece che utilizzare il tipico errore di predizione, si definisce un errore più sofisticato, chiamato pseudo-perdita (*pseudo-loss*). Contrariamente all'errore ordinario, calcolato rispetto alla distribuzione sui campioni di esempio, la pseudo-perdita è calcolata rispetto alla distribuzione

sull'insieme di tutte le coppie di campioni e di etichette errate. Questa distribuzione può essere definita in modo che l'algoritmo di *boosting* concentri i *weak learner* non solo per classificare i campioni di esempio, ma soprattutto per elaborare maggiormente le etichette errate che risultano più difficili da discriminare. Secondo questo criterio, l'algoritmo di *AdaBoost.M2* realizza il *boosting* solo se ogni ipotesi debole ha una pseudo perdita migliore della scelta casuale. Formalmente, un'etichettatura scorretta (*mislabel*) è una coppia (i,y), dove i è l'etichetta del campione di training e y è un' etichetta errata associata al campione i. Prendendo B come l'insieme di tutte le *mislabel* si ottiene che:

$$B = \{(i,y) : i \in \{1,\ldots,m\}, y \neq y_i\}$$

Una *mislabel distribution* è una distribuzione che è definita nell'insieme B di tutte le *mislabel*. Per ogni ciclo t dell'algoritmo, *AdaBoost.M2* fornisce al *weak learner* una *mislabel distribution* D_t. In risposta il *weak learner* calcola un'ipotesi h_t della forma $h_t : X \times Y \in [0,1]$. Non ci sono restrizioni su

$$\sum_y h_t(x,y) \tag{B.1}$$

in particolare, il vettore di predizione non definisce una distribuzione di probabilità. Intuitivamente ogni *mislabel* (i,y) stabilisce se l'etichetta predetta per il campione x_i è y_i (etichetta corretta) o y (una delle etichette errate) Con questa interpretazione, il peso $D_t(i,y)$ assegnato al *mislabel* rappresenta l'importanza di distinguere l'etichetta y errata sul campione x_i. Un'ipotesi debole h_t viene interpretata nel modo seguente. Se $h_t(x_i,y_i) = 1$ e $h_t(x_i,y) = 0$ allora h_t ha predetto correttamente che questa x_i-esima è y_i e non y, poiché da h_t riteniamo che y_i può essere plausibile ed y non plausibile. Similmente se $h_t(x_i,y_i) = 0$ e $h_t(x_i,y) = 1$ allora h_t ha fatto, in modo non corretto, la previsione opposta. Se $h_t(x_i,y_i) = h_t(x_i,y)$, allora la h_t-esima previsione è presa con decisione casuale. I valori per h_t presi nell'intervallo $(0,1)$ sono interpretati in modo probabilistico. Questa interpretazione ci porta a definire la pseudo perdita dell'ipotesi h_t rispetto alla *mislabel distribution* D_t con la formula

$$\varepsilon_t = \frac{1}{2} \sum_{(i,y)\in B} D_t(i,y)(1 - h_t(x_i,y_i) + h(x_i,y)) \tag{B.2}$$

La derivazione completa di questa formula è riportata in dettaglio in [45]. La pseudo perdita è minimizzata quando l'etichetta corretta y_i acquista valori vicini ad 1 e le etichette errate $y \neq y_i$ hanno valore vicino a 0. Si nota inoltre che la pseudo perdita 1/2 è banalmente acquisita da qualunque valore costante dell'ipotesi h_t: se si ha un ipotesi h_t con valore più alto di 1/2 è possibile rimpiazzarla con l'ipotesi $1 - h_t$ in cui la pseudo perdita è minore di 1/2. L'obiettivo dei *weak learner* è di trovare un ipotesi debole h_t con una bassa pseudo perdita. Dopo aver ricevuto h_t la *mislabel distribution* viene aggiornata utilizzando una regola simile a quella utilizzata in *AdaBoost.M1*. L'output dell'ipotesi finale h_{fin} per una data istanza x è l'etichetta y che massimizza una media pesata dei valori delle ipotesi deboli $h_t(x,y)$. Il seguente teorema assegna un limite all'errore di apprendimento (*training error*) dell'ipotesi finale: questo teorema richiede solo che le ipotesi deboli abbiano pseudo perdita minore di 1/2. Inoltre, sebbene le ipotesi deboli siano valutate rispetto la pseudo perdita, si valuta l'ipotesi finale usando la misura dell'errore ordinario.

Teorema B.2 *Si suppone di avere un algoritmo di apprendimento debole* weakLearn *che, chiamato dall'algoritmo AdaBoost.M1, genera delle ipotesi con errore* $\varepsilon_1, \ldots, \varepsilon_T$, *dove* ε_t *è definito come nell'algoritmo illustrato in precedenza. Si prenda* $\gamma_t = 1/2 - \varepsilon_t$. *Allora l'*upper bound *sull'errore dell'ipotesi finale* h_{fin} *è:*

$$\frac{1}{m}|\{i : h_{fin}(x_i) \neq y_i\}| \leq (k-1)\prod_{t=1}^{T}\sqrt{1-4{\gamma_t}^2} \leq (k-1)\exp\left(-2\sum_{t=1}^{T}{\gamma_t}^2\right)$$

dove k è il numero delle classi.

Il principale vantaggio di *AdaBoost* è legato alla sua estrema semplicità di programmazione, non avendo parametri da impostare, ad eccezione del numero di iterazioni. Inoltre non richiede una conoscenza a priori degli algoritmi di classificazione deboli e si può combinare con un qualunque tipo e numero di questi. Infine, nel caso in cui si disponga di una quantità sufficiente di dati e di un buon classificatore debole, *AdaBoost* fornisce delle garanzie teoriche sull'esistenza del limite superiore dell'errore, come visto nei teoremi precedentemente citati. I due svantaggi principali riguardano la sensibilità al rumore e la notevole dipendenza dai dati e dai classifi-

catori deboli scelti: l'algoritmo, infatti, fallisce nel caso in cui i *weak learner* siano effettivamente troppo deboli.

B.3 Features di Haar

Particolari tipologie di classificatori deboli, generalmente utilizzate nell'ambito dell'algoritmo *AdaBoost*, sono le *features di Haar*: esse si presentano come maschere di diverso tipo che vengono sovrapposte all'immagine (figura).

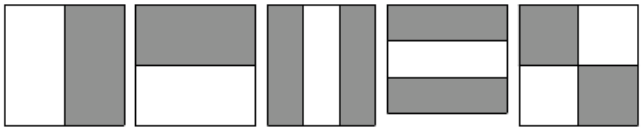


Figura B.1: Esempio di feature di Haar.

Le *features*, oltre che per tipo, si differenziano tra loro anche per dimensione e posizione all'interno dell'area di interesse. Il valore di una *feature*, una volta applicata su una regione dell'immagine, è dato dalla differenza (normalizzata) tra i livelli di grigio dei pixel appartenenti ai rettangoli bianchi rispetto a quelli neri.

Le *features* permettono quindi di estrarre le componenti salienti di un'immagine e per migliorare la fedeltà dei classificatori è possibile definirne di nuove, progettate ad.hoc a seconda dell'applicazione.

Figura B.2: *Features di Haar* specifiche per il riconoscimento delle gambe di un pedone.

Bibliografia

[1] Assistenza al traffico: il nostro impegno per la sicurezza. http://www.autostrade.it.

[2] Blind spot information system (blis) by volvo. http://volvo.comt.

[3] Hitachi: Image processing camera. http://www.hitachi.co.jp.

[4] Incidenti stradali. http://www.istat.it/salastampa/comunicati.

[5] Mobileye n.v. blind spot detection and lane change assist (bsd/l- ca). http://www.mobileye-vision.com.

[6] *Lateral Driving Assistance Using Optical Flow and Scene Analysis*, 2007.

[7] Allied Vision Technologies. Stringray Technical Manual. Avaiable http://www.alliedvisiontec.com/.

[8] Y. Altunbasak, P. E. Eren, and A. M. Tekalp. Region-Based Parametric Motion Segmentation Using Color Information. In *Graphical Models and Image Processing*, pages 13–23, 1998.

[9] V. Andrey and K. H. Jo. Automatic Detection and Recognition of Traffic Signs using Geometric Structure Analysis. *SICE-ICASE, 2006. International Joint Conference*, pages 1451–1456, Oct. 2006.

[10] Y. Aoyagi and T. Asakura. A Study on Traffic Sign Recognition in Scene Image Using Genetic Algorithms and Neural Networks. In *Procs. IEEE*

22nd Intl. Conf. on Industrial Electronics, Control, and Instrumentation, 1996, volume 3, pages 1838–1843, Aug. 1996.

[11] N. Barnes and A. Zelinsky. Real-Time Radial Symmetry for Speed Sign Detection. In *Procs. IEEE Intelligent Vehicles Symposium, 2004*, pages 566–571, June 2004.

[12] P. H. Batavia, D. A. Pomerleau, and C. E. Thorpe. Overtaking Vehicle Detection Using Implicit Optical Flow. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pages 729–734, 1997.

[13] A. Bensrhair, I. Cabani, and G. Toulminet. A Fast and Self-adaptive Color Stereo Vision Matching; a first step for Road Obstacle Detection. In *Procs. IEEE Intelligent Vehicles Symposium, 2006*, June 2006.

[14] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and ADABOOST for music classification. *Mach. Learn.*, 65:473–484, December 2006.

[15] M. Bertozzi, L. Bombini, A. Broggi, M. Buzzoni, E. Cardarelli, S. Cattani, P. Cerri, A. Coati, S. Debattisti, A. Falzoni, R. I. Fedriga, M. Felisa, L. Gatti, A. Giacomazzo, P. Grisleri, M. C. Laghi, L. Mazzei, P. Medici, M. Panciroli, P. P. Porta, P. Zani, and P. Versari. VIAC: an Out of Ordinary Experiment. In *Procs. IEEE Intelligent Vehicles Symposium 2011*, pages 175–180, Baden Baden, Germany, June 2011. ISSN: 1931-0587.

[16] M. Bertozzi, A. Broggi, E. Cardarelli, R. I. Fedriga, L. Mazzei, and P. P. Porta. VIAC Expedition Toward Autonomous Mobility. *Robotics and Automation Magazine*, 18(3):120–124, Sept. 2011. ISSN: 1070-9932.

[17] M. Bertozzi, A. Broggi, G. Conte, and A. Fascioli. Vision-based Automated Vehicle Guidance: the experience of the ARGO vehicle. In *IAPRVA - Tecniche di Intelligenza Artificiale e Pattern Recognition per la Visione Artificiale*, pages 35–40, Apr. 1998.

[18] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis, and F. Chausse. Pedestrian localization and tracking system with Kalman filtering. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 584 – 589, june 2004.

[19] M. Bertozzi, A. Broggi, P. Grisleri, T. Graf, and M. Meinecke. Pedestrian detection in infrared images. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pages 662 – 667, june 2003.

[20] E. Binelli, A. Broggi, A. Fascioli, S. Ghidoni, P. Grisleri, T. Graf, and M. Meinecke. A modular tracking system for far infrared pedestrian recognition. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 759 – 764, june 2005.

[21] A. Broggi, P. Cerri, P. Medici, P. P. Porta, and G. Ghisio. Real Time Road Signs Recognition. In *Procs. IEEE Intelligent Vehicles Symposium 2007*, pages 981–986, Istanbul, Turkey, June 2007.

[22] A. Broggi, A. Fascioli, M. Carletti, T. Graf, and M. Meinecke. A multiresolution approach for infrared vision-based pedestrian detection. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 7 – 12, june 2004.

[23] A. Broggi, P. Medici, E. Cardarelli, P. Cerri, A. Giacomazzo, and N. Finardi. Development of the control system for the VisLab Intercontinental Autonomous Challenge. In *Procs. IEEE Intelligent Transportation Systems 2010*, pages 635–640, 2010. ISSN: 2153-0009.

[24] S. Buluswar and B. Draper. Color recognition in outdoor images. In *Computer Vision, 1998. Sixth International Conference on*, pages 171 –177, Jan. 1998.

[25] D. Burschka and G. D. Hager. Vision-Based 3D Scene Analysis for Driver Assistance. In *Procs. IEEE Intl. Conf. on Robotics Automation*, pages 812–818, Apr. 2005.

[26] A. Cappalunga, S. Cattani, A. Broggi, M. S. McDaniel, and S. Dutta. Real Time 3D Terrain Elevation Mapping Using Ants Optimization Algorithm and

Stereo Vision. In *Procs. IEEE Intelligent Vehicles Symposium 2010*, pages 902–909, San Diego (CA), USA, June 2010.

[27] C. Caraffi, E. Cardarelli, P. Medici, P. P. Porta, G. Ghisio, and G. Monchiero. An algorithm for Italian de-restriction Signs Detection. In *Procs. IEEE Intelligent Vehicles Symposium 2008*, pages 834–840, Eindhoven, Netherlands, June 2008.

[28] N. Casagrande. MultiBoost: An open source multi-class AdaBoost learner, 2005. http://iro.umontreal.ca/ casagran/multiboost/.

[29] L. Cinti and L. Masetti. Il progetto lightweight neural network ++: un'implementazione c++ di una rete feed forward con backprpagation, apr 2004. http://lwneuralnetplus.sourceforge.net.

[30] A. Colorni, M. Dorigo, and V. Maniezzo. Distributed Optimization by Ant Colonies. In *European Conference on Artificial Life*, pages 134–142, 1991.

[31] A. de la Escalera, J. M. Armignol, and M. Mata. Traffic sign recognition and analysis for intelligent vehicles. *Image and Vision Computing*, 21(3):247–258, Mar. 2003.

[32] A. de la Escalera, J. Armingol, J. Pastor, and F. Rodriguez. Visual sign information extraction and identification by deformable models for intelligent vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 5(2):57–68, June 2004.

[33] A. de la Escalera, J. M. Armingol, and M. Salichs. Traffic Sign Detection for Driver Support Systems. In *3rd Intl. Conf. on Field and Service Robotics, Espoo, Finlandia, 2001*, June 2001.

[34] A. de la Escalera, L. E. Moreno, E. A. Puente, and M. A. Salichs. Neural Traffic Sign Recognition for Autonomous Vehicles. In *Procs. IEEE 20th Intl. Conf. on Industrial Electronics, Control and Instrumentation*, volume 2, pages 841–846, Sept. 1994.

[35] A. de la Escalera, L. E. Moreno, M. A. Salichs, and J. M. Armingol. Road traffic sign detection and classification. *Industrial Electronics, IEEE Transactions on*, 44(6):848–859, 1997.

[36] Defence Advanced Research Projects Agency (DARPA). Urban Challenge 2007. Avaiable http://www.darpa.mil/grandchallenge.

[37] E. Dickmanns, R. Behringer, D. Dickmanns, T. Hildebrandt, M. Maurer, F. Thomanek, and J. Schiehlen. The seeing passenger car 'VaMoRs-P'. In *Intelligent Vehicles '94 Symposium, Proceedings of the*, pages 68 – 73, oct. 1994.

[38] C. Ding and H. Peng. Minimum Redundancy Feature Selection from Microarray Gene Expression Data. *Computational Systems Bioinformatics Conference, International IEEE Computer Society*, 0:523, 2003.

[39] M. Dorigo. *Optimization, Learning and Natural Algorithms*. Ph.D. dissertation, Politecnico di Milano, 1992.

[40] S. Estable, J. Schick, F. Stein, R. Janssen, R. Ott, W. Ritter, and Y. J. Zheng. A Real-Time Traffic Sign Recognition System. In *Procs. of the Intelligent Vehicles '94 Symposium*, pages 213–218, Oct. 1994.

[41] C. Fang, C. Fuh, S. Chen, and P. Yen. A road sign recognition system based on dynamic visual model. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–750 – I–755 vol.1, June 2003.

[42] M. Felisa and P. Zani. Incremental Disparity Space Image computation for automotive applications. In *Procs. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, St.Louis, Missouri, USA, Oct. 2009.

[43] Y. Freund. Boosting a weak learning algorithm by majority. *Inf. Comput.*, 121:256–285, September 1995.

[44] Y. Freund and R. Schapire. A short introduction to boosting. *Japonese Society for Artificial Intelligence*, 14(5):771–780, 1999.

[45] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37. Springer-Verlag, 1995.

[46] Y. Freund and R. E. Schapire. Experiments with a New Boosting Algorithm. In *In Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, 1996.

[47] D. Gavrila. Pedestrian Detection from a Moving Vehicle. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, ECCV '00, pages 37–49, London, UK, 2000. Springer-Verlag.

[48] D. Ghica, S. W. Lu, and X. Yuan. Recognition of Traffic Signs by Artificial Neural Network. In *Procs. IEEE Intl. Conf. on Neural Networks, 1995*, volume 3, pages 1444–1449, Nov.–Dec. 1995.

[49] P. Grisleri and I. Fedriga. The BRAiVE platform. In *Procs. 7th IFAC Symposium on Intelligent Autonomous Vehicles*, Lecce, Italy, Sept. 2010.

[50] D. Guo, T. Fraichard, M. Xie, and C. Laugier. Color modeling by spherical influence field in sensing driving environment. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 249 –254, 2000.

[51] C. Harris and M. Stephens. A Combined Corner and Edge Detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151.

[52] B. Hoferlin and K. Zimmermann. Towards reliable traffic sign recognition. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 324 –329, June 2009.

[53] F. Jeun-Haii and L. Gang. A Vision-Aided Vehicle Driving System: Establishment of a Sign Finder System. In *Procs. Conf. on Vehicle Navigation and Information Systems, 1994*, pages 33–38, Aug.–Sept. 1994.

[54] G.-Y. Jiang and T. Y. Choi. Robust Detection of Landmarks in Color Image Based on Fuzzy Set Theory. In *Procs. IEEE 4th Intl. Conf. on Signal Processing*, volume 2, pages 968–971, Oct. 1998.

[55] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant Features and the Subset Selection Problem. In *MACHINE LEARNING: PROCEEDINGS OF THE ELEVENTH INTERNATIONAL*, pages 121–129. Morgan Kaufmann, 1994.

[56] S. Kantawong. Road traffic signs detection and classification for blind man navigation system. In *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, pages 847 –852, Oct. 2007.

[57] T. Kato, Y. Ninomiya, and I. Masaki. An obstacle detection method by fusion of radar and motion stereo. *IEEE Transactions on Intelligent Transportation Systems*, 3(3):182–188, 2002.

[58] A. Khammari, E. Lacroix, F. Nashashibi, and C. Laurgeau. Vehicle detection combining gradient analysis and adaboost classification. In *IEEE Conf. on Intelligent Transportation Systems*, pages 66 – 71, 2005.

[59] R. Kohavi and G. H. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[60] K.Sasaki and S.Kawamoto. Visconti Image Processing System Specific to Automotive Domain. In *Toshiba Review*.

[61] A. Kuehnle. Symmetry-based recognition of vehicle rears. *Pattern Recogn. Lett.*, 12:249–258, April 1991.

[62] R. Labayrade and D. Aubert. In-vehicle obstacles detection and characterization by stereovision. In *in Proceedings the 1st International Workshop on In-Vehicle Cognitive Computer Vision Systems*, pages 13–19, 2003.

[63] R. Labayrade and D. Aubert. The car-following and lane-changing collision pre- vention system based on the cascaded fuzzy inference system. In *IEEE Transactions on Vehicular Technology*, volume 54, pages 910– 924, 2005.

[64] R. Labayrade, C. Royere, D. Gruyer, and D. Aubert. Cooperative Fusion for Multi-Obstacles Detection With Use of Stereovision and Laser Scanner. *Auton. Robots*, 19:117–140, September 2005.

[65] J. Liu and G. Wang. A hybrid feature selection method for data sets of thousands of variables. In *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, volume 2, pages 288 –291, 2010.

[66] W. Liu and K. Maruya. Detection and recognition of traffic signs in adverse conditions. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 335 –340, June 2009.

[67] X. Liu and K. Fujimura. Pedestrian detection using stereo night vision. *Vehicular Technology, IEEE Transactions on*, 53(6):1657–1665, 2004.

[68] G. Loy and N. Barnes. Fast shape-based road sign detection for a driver assistance system. In *Procs. IEEE 10th Intl. Conf. on Intelligent Robots and Systems, 2004*, volume 1, pages 70–75, Sept.–Oct. 2004.

[69] Q. T. Luong, J. Weber, D. Koller, and J. Malik. An Integrated Stereo-Based Approach To Automatic Vehicle Guidance. In *Procs. IEEE 5th Intl. Conf. on Computer Vision, 1995*, pages 52–57, June 1995.

[70] S. Maldonado Bascón, J. Acevedo Rodríguez, S. Lafuente Arroyo, A. Fernndez Caballero, and F. López-Ferreras. An optimization on pictogram identification for the road-sign recognition task using SVMs. *Comput. Vis. Image Underst.*, 114:373–383, Mar. 2010.

[71] S. Maldonado Bascón, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, , and F. López-Ferreras. Road-Sign Detection and Recognition Based on Support Vector Machines. *Intelligent Transportation Systems, IEEE Transactions on*, 8(2):264–278, June 2007.

[72] J. Markoff. Google cars drive themselves, in traffic. *New York Times*, Oct. 2010.

[73] N. Matthews, P. An, D. Charnley, and C. Harris. Vehicle detection and recognition in greyscale imagery. In *2nd Int. Workshop on Intelligent Autonomous Vehicles*, pages 1—6, 1995.

[74] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[75] J. Miura, T. Kanda, and Y. Shirai. An active vision system for real-time traffic sign recognition. In *Intelligent Transportation Systems, 2000. Proceedings. 2000 IEEE*, pages 52 –57, 2000.

[76] T. H. N. Blanc, B. Steux. LaRA SideCam: a Fast and Robust Vision-Based Blindspot Detection System. In *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium*, pages 480 – 485, 2007.

[77] H. Nanda, C. Benabdelkedar, and L. Davis. Modelling pedestrian shapes for outlier detection: a neural net based approach. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pages 428 – 433, june 2003.

[78] H. Nanda and L. Davis. Probabilistic template based pedestrian detection in infrared videos. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 1, pages 15 – 20, june 2002.

[79] Y.-Y. Nguwi and A. Kouzani. A Study on Automatic Recognition of Road Signs. *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*, pages 1–6, June 2006.

[80] C. Nunn, A. Kummert, and S. Muller-Schneiders. A two stage detection module for traffic signs. In *Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on*, pages 248 –252, Sept. 2008.

[81] H. Ohara, I. Nishikawa, S. Miki, and Yabuki. Detection and Recognition of Road Signs Using Simple Layered Neural Networks. In *Procs. IEEE 9th Intl. Conf. on Neural Information Processing, 2002*, volume 2, pages 626–630, Nov. 2002.

[82] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian Detection Using Wavelet Templates. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 193–, Washington, DC, USA, 1997. IEEE Computer Society.

[83] U. Ozguner, K. A. Redmill, and A. Broggi. Team TerraMax and the DARPA Grand Challenge: A General Overview. In *Procs. IEEE Intelligent Vehicles Symposium 2004*, pages 232–237, Parma, Italy, June 2004.

[84] N. Peterfreund. Robust Tracking of Position and Velocity With Kalman Snakes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21:564–569, June 1999.

[85] M. Pirooznia, J. Yang, M. Q. Yang, and Y. Deng. A comparative study of different machine learning methods on microarray gene expression data. *BMC Genomics*, 9(Suppl 1), 2008.

[86] M. Rogati and Y. Yang. High-Performing Feature Selection for Text Classification. pages 659–661, 2002.

[87] A. Ruta, Y. Li, and X. Liu. Real-time traffic sign recognition from video by class-specific discriminative features. *Pattern Recogn.*, 43:416–430, Jan. 2010.

[88] R. E. Schapire. The Strength of Weak Learnability. *Mach. Learn.*, 5:197–227, July 1990.

[89] W. G. Shadeed, D. I. Abu-Al-Nadi, and M. J. Mismar. Road traffic sign detection in color images. In *Procs. IEEE 10th Intl. Conf. on Electronics, Circuits and Systems, 2003*, volume 2, pages 890–893, Dec. 2003.

[90] J. Shi and C. Tomasi. Good Features to Track, 1994.

[91] A. Soetedjo and K. Yamada. Fast and robust traffic sign detection. *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, 2:1341–1346 Vol. 2, Oct. 2005.

[92] B. Sun, R. Ramamoorthi, S. G. Narasimhan, and S. K. Nayar. A practical analytic single scattering model for real time rendering. In *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[93] Z. Sun, G. Bebis, and R. Miller. On-road Vehicle Detection: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:694–711, 2006.

[94] J. K. S.Y. Kim, S.Y. Oh and Y. Ryu. Front and Rear Vehicle Detection and Tracking in the Day and Night Times Using Vision and Sonar Sensor Fusion. In *International Conference on Intelligent Robots and Systems*, pages 2173 – 2178, 2005.

[95] L. G. Valiant. A theory of the learnable. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, STOC '84, pages 436–445, New York, NY, USA, 1984. ACM.

[96] S. Vitabile, A. Gentile, and F. Sorbello. A Neural Network Based Automatic Road Signs Recognizer. In *Procs. IEEE Intl. Joint Conf. on Neural Networks, 2002*, volume 3, pages 2315–2320, May 2002.

[97] S. Vitabile, G. Pollaccia, G. Pilato, and E. Sorbello. Road signs recognition using a dynamic pixel aggregation technique in the HSV color space. In *Image Analysis and Processing, 2001. Proceedings. 11th International Conference on*, pages 572 –577, Sept. 2001.

[98] B. D. H. Y. N. W. W. Liu, X. Wen. Rear Vehicle Detection and Tracking for Lane Change Assist. In *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium*, pages 252 – 257, 2007.

[99] J. Wang, G. Bebis, and R. Miller. Overtaking Vehicle Detection Using Dynamic and Quasi-Static Background Modeling. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops - Volume 03*, pages 64–, Washington, DC, USA, 2005. IEEE Computer Society.

[100] X. Wang, M. Yu, Y. Yang, and G. Jiang. Research on subjective stereoscopic image quality assessment. In *Proc. SPIE 7255, 725509 (2009)*, 2009.

[101] J. M. Y. Mae, Y. Shirai and Y. Kuno. Object Tracking in Cluttered Background Based on Optical Flow and Edges. In *in Proc. of the 13th Int. Conf. on Pattern Recognition*, pages 196–200, 1996.

[102] H.-M. Yang, C.-L. Liu, K.-H. Liu, and S.-M. Huang. Traffic Sign Recognition in Disturbing Environments. In N. Zhong, Z. Ras, S. Tsumoto, and E. Suzuki, editors, *Foundations of Intelligent Systems*, volume 2871 of *Lecture Notes in Computer Science*, pages 252–261. Springer Berlin Heidelberg, 2003.

[103] J. Yang, C. Hou, Y. Zhou, Z. Zhang, and J. Guo. Objective quality assessment method of stereo images. In *Proc. 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, May 2009.

[104] Y. Yang and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. pages 412–420. Morgan Kaufmann Publishers, 1997.