UNIVERSITÀ DELLA CALABRIA

Dipartimento di Elettronica,
Informatica e Sistemistica

Dottorato di Ricerca in
Ingegneria dei Sistemi e Informatica
XXIVciclo

Tesi di Dottorato

Metodi di Fusione dei Dati per Sistemi di Assistenza alla Guida

Lupia Marco

Settore Scientifico Disciplinare: ING-INF/04

UNIVERSITÀ DELLA CALABRIA

Dipartimento di Elettronica,
Informatica e Sistemistica

Dottorato di Ricerca in
Ingegneria dei Sistemi e Informatica
XXIV ciclo

Tesi di Dottorato

Metodi di Fusione dei Dati per Sistemi di Assistenza alla Guida

Lupia Marco

Coordinatore
Prof. Luigi Palopoli

Supervisore
Prof. Alessandro Casavola

DEIS

Settore Scientifico Disciplinare: ING-INF/04

DEIS- DIPARTIMENTO DI ELETTRONICA, INFORMATICA E SISTEMISTICA
Novembre

Settore Scientifico Disciplinare: ING-INF/04

Ai miei genitori

Ringraziamenti

Desidero innanzitutto ringraziare il mio supervisore prof. Alessandro Casavola per l'aiuto e la disponibilità dimostrata durante tutto il corso di studi.

Un ringraziamento particolare va a Gianni, con il quale ho condiviso parte delle ricerche di dottorato e senza il quale non avrei raggiunto i risultati ottenuti e a Gianfranco da sempre compagno di studi nel mio percorso universitario.

Ringrazio gli amici del "Borgo", Giuseppe, Ketty, Flavia, Gianluca, Saverio, Rino, Anna, Maurizio e Antonio amici e compagni di "viaggio e campeggio" con i quali durante questi anni ho trascorso indimenticabili serate insieme.

Non ultima la mia famiglia, i miei genitori, Valeria e Riccardo il miglior fratello che si possa immaginare e Claudia che con il suo arrivo un anno fa ha portato una ventata di allegria nella vita di tutti noi.

Sommario

Questo documento riporta tutta l'attività svolta durante il corso di dottorato riguardo lo studio, l'applicazione e la sperimentazione di metodi di Data Fusion nell'ambito dei sistemi avanzati di assistenza alla guida per il miglioramento delle prestazioni e dell'affidabilità. In linea di principio, sembra ragionevole affermare che combinando in modo ottimale le informazioni provenienti da più sensori è possibile progettare e realizzare un sistema meno sensibile alle variazioni ambientali e a possibili errori di misura dovuti alla presenza di outliers. Tuttavia, dal punto di vista pratico i costi aggiuntivi che si devono sostenere, sia in termini di hardware necessario alla raccolta dei dati dai sensori addizionali che per la necessità di disporre di sistemi di calcolo più potenti, potrebbero non essere giustificati se i miglioramenti ottenuti nelle situazioni più probabili e più realistiche di utilizzo sono modesti.

Il presente documento offre, innanzitutto, una panoramica su vari algoritmi di visione artificiale utilizzati per il riconoscimento della segnaletica orizzontale e per la stima del tempo di invasione (t_{TLC}). Quest'ultimo parametro gioca un ruolo determinante nell'avvertimento tempestivo del conducente in caso di superamento dei limiti della carreggiata. I vari algoritmi sono stati testati in varie condizioni di guida, valutandone le prestazioni conseguibili e il carico computazionale richiesto.

Segue un'analisi dello stato dell'arte dei metodi e delle tecniche di Data Fusion più promettenti e che che meglio si prestano a migliorare l'accuratezza del calcolo della stima del tempo di invasione t_{tlc} grazie alla disponibilità di altri sensori oltre alla telecamera. Specificatamente, si sono confrontati i vari metodi e algoritmi di Data Fusion, particolarizzati rispetto a vari modelli matematici della vettura e ai sensori disponibili, valutando le loro prestazioni in situazioni tipiche di guida e soprattutto rispetto all'errore percentuale di stima del tempo di invasione t_{tlc} ottenuto, valutando anche il carico computazionale corrispondente.

Gli algoritmi più promettenti sono stati implementati su piattaforma embedded.

Indice

1

Introduzione

Grazie al costante progresso dell'elettronica e della relativa miniaturizzazione dei componenti è in corso da vari anni nell'industria automobilistica una vera e propria corsa ad integrare nuovi sensori, attuatori e micro-controllori per il controllo e la gestione del motore, della driveline, delle sospensioni e dei freni. Un primo passo in questa direzione è stata l'adozione di leggi di controllo digitale per il controllo motore, esigenza soprattutto nata e legata al rispetto delle prime normative anti-inquinamento degli anni 70, che da allora hanno imposto ai costruttori vincoli sempre più stretti sulle emissioni e l'adozione di sistemi di diagnosi a bordo per la verifica del corretto funzionamento della combustione. In parallelo, legati invece ad aspetti di sicurezza, sono stati introdotti i primi sistemi attivi di ausilio alla guida **(advanced driver assistance systems - ADAS)**, dei quali i più famosi sono sicuramente il controllo elettronico della frenata **(anti-lock braking - ABS)** e dell'assetto **(electronic stability control - ESC)**, che offrono al pilota aiuto in situazioni limite di guida.

Nell'ambito dell'infortunistica stradale il superamento involontario della corsia è una delle principali cause di incidenti automobilistici. Le cause di tale evento sono per lo più imputabili alla distrazione del conducente (utilizzo del cellulare, sonnolenza, guida distratta, uso autoradio ed altri sistemi di bordo). I sistemi ADAS che tentano di mitigare queste evenienze accidentali sono genericamente denominati **(lane keeping systems - LKS)** e si distinguono in sistemi che semplicemente avvisano con un certo anticipo il guidatore dell'invasione di corsia **(lane departure warning systems - LDWS)** e quelli che invece correggono autonomamente la guida per riportare il veicolo al centro della corsia quando necessario. Questi sistemi utilizzano una telecamera per identificare le strisce della segnaletica orizzontale, calcolare la posizione e la direzione di marcia rispetto alla direzione delle strisce e stimare il tempo probabile di invasione corsia attigua **(Time to lane crossing - t_{TLC})**, utilizzando anche altri dati provenienti dai sensori dell'autoveicolo come l'angolo di sterzata, la velocità longitudinale e le velocità angolari delle ruote. Quando il valore di t_{TLC} stimato è sotto una certa soglia il sistema avvisa il guidatore del pericolo (avviso sonoro o tramite vibrazioni sullo sterzo o sul sedile). I

sistemi attivi hanno la capacità addizionale, quando necessario, di agire sui freni o sullo sterzo in maniera autonoma per riportare l'autoveicolo al centro della carreggiata.

L'obiettivo primario della ricerca effettuata è stato lo studio e la sperimentazione di metodi e algoritmi di fusione dei dati particolarmente adatti e facilmente applicabili ai sistemi di avviso di superamento della carreggiata (LDWS). In questo documento finale vengono riportati in sintesi tutti i risultati ottenuti. L'attività è consistita, per una ristretta classe di questi algoritmi, nella definizione di obbiettivi chiari e indici di confronto consistenti, nella personalizzazione di questi al problema oggetto di indagine, allo studio rigoroso delle loro proprietà e nello svolgimento di una campagna di sperimentazione approfondita atta a valutare, per diverse condizioni di guida e ambientali, le prestazioni e i miglioramenti ottenibili in termini di aumento della robustezza e accuratezza dei risultati rispetto a sistemi LDWS sprovvisti di funzionalità di Data Fusion. Segue un breve sommario sull'organizzazione del documento.

La tesi è organizzata come segue.
Nel **Capitolo 2** sono introdotte le caratteristiche dei sistemi avanzati di assistenza alla guida e con una panoramica sul sistema di avviso di superamento di corsia attigua. Vengono descritte la struttura e le principali funzioni di un sistema LDWS. Particolare attenzione è rivolta agli algoritmi di visione per l'identificazione e il tracking delle linee.

Il **Capitolo 3** è dedicato alla definizione del tempo di invasione di corsia attigua (t_{tlc}), il cui calcolo rappresenta il risultato finale del sistema LWDS e la cui accuratezza consente di giudicare la bontà del sistema e di confrontare le prestazioni di sistemi LDWS basati su metodi e sensori diversi. Il t_{tlc} è l'intervallo di tempo entro cui è prevista l'invasione di una corsia attigua se il pilota non intervenisse modificando le velocità e l'orientazione del veicolo rispetto alla carreggiata. Ai fini del confronto, si presenta un metodo di calcolo del t_{tlc} basato soltanto su dati inferibili dal sistema di visione e sulla conoscenza della sola velocità longitudinale che non richiede l'utilizzo di metodi di data fusion. Questa tecnica, di per se utilizzabile nei sistemi LWDS commerciali, è qui utilizzata come benchmark per valutare i miglioramenti conseguibili integrando il sistema con ulteriori sensori disponibili nell'autovettura, fondendo i corrispondenti dati a quelli provenienti dalla telecamera.

Gli algoritmi di fusione dei dati che si sono utilizzati richiedono un modello matematico del moto dell'autovettura adeguato. A tal fine nel **Capitolo 4** sono presentati diversi modelli cinematici e un modello dinamico. Questi modelli differiscono per il set di sensori supposti disponibili. I modelli cinematici hanno un numero ridotto di parametri da tarare e rappresentano bene il moto dell'autovettura durante le fasi stazionarie a velocità costanti. Il modello dinamico è più complesso e richiede uno sforzo maggiore per la taratura. Se ben tarato riesce a descrivere il comportamento dell'autovettura anche durante le

fasi di transitorio nelle accelerazioni e consente di migliorare l'errore di stima degli algoritmi in queste situazioni, cosa invece impossibile per i modelli cinematici.

Nel **Capitolo 5** vengono presentati gli algoritmi Extended Kalman Filter (EKF) e Unscented Kalman Filter (UKF), adattati ai quattro modelli di autovettura considerati, dandone una corretta formulazione matematica direttamente implementabile. Solo questi due metodi saranno analizzati nel resto del documento perché considerati i più promettenti.

Il **Capitolo 6** è dedicato alla descrizione dei percorsi di test simulati. I dati necessari a testare gli algoritmi di fusione dei dati sono ottenuti dal software Carsim, che permette di simulare il comportamento dinamico di un'autovettura su percorsi arbitrari con profili di velocità e accelerazione prestabiliti. In particolare, si sono considerati quattro test su un tratto di strada rettilineo, percorso con profili diversi di velocità e accelerazioni, sia costanti che variabili. I dati relativi a ogni percorso sono presentati e descritti tramite opportuni grafici.

I risultati delle simulazioni su tutti i percorsi di test sono inserite nel **Capitolo 7**. Le prestazioni degli algoritmi di fusione dei dati sono state calcolate su tutti i percorsi test. Un'analisi approfondita è stata eseguita sulla precisione delle stime ottenute, relative in particolare ai valori di posizione entro la carreggiata, di velocità laterale e longitudinale e del valore di t_{tlc}, allegando ad ogni simulazione il grafico degli errori percentuali commessi lungo il profilo del test.

Nel **Capitolo 8** l'architettura di Data Fusion più promettente rivelatosi nello studio precedente è testata anche in condizioni di occlusione della telecamera. Questo test è particolarmente importante perché permette di valutare appieno il vantaggio di utilizzare metodi di fusione dei dati in questo tipo di applicazione. Le occlusioni della telecamera rappresentano situazioni limite che in pratica si realizzano a causa di brusche variazioni della luminosità o altre condizioni ambientali sfavorevoli. Un sistema LDWS basato sulla fusione dei dati deve dimostrare di essere robusto rispetto a temporanee assenza dei dati provenienti dal sistema di visione o da altri sensori e continuare a funzionare anche se con funzionalità ridotte.

Il **Capitolo 9** descrive l'implementazione hardware dell'architettura software ritenuta più promettente su piattaforma embedded e i test eseguiti in laboratorio e su strada. In particolare è descritta l'ottimizzazione del software di visone su DSP e l'integrazione con interfaccia utente che è eseguita su processore ARM. Il sistema LDWS ottenuto è stato testato in laboratorio utilizzando i dati forniti dal simulatore Carsim e su strada a bordo di un'autovettura reale.

Il **Capitolo 10** riporta una serie di conclusioni sui risultati ottenuti.

2

Lane Departure Warning System

2.1 Introduzione

La sicurezza stradale è un problema fondamentale nel settore dell'industria automobilistica, con rilevanti impatti economici e sociali. L'introduzione sui nuovi veicoli in commercio di avanzati sistemi di assistenza alla guida comporta lo studio e lo sviluppo di algoritmi e metodologie che possano garantire l'interazione tra sistemi diversi e l'utilizzo ottimale delle informazioni provenienti da più sensori.

In questo contesto si inseriscono gli algoritmi di fusione dei dati che in letteratura [1] sono generalmente definiti come l'uso di tecniche che combinano dati da sorgenti multiple, talvolta non omogenee, e raccolgono informazioni al fine di ottenere inferenze. Inferenze che saranno più efficienti e più accurate rispetto a quelle ottenute tramite l'uso di una singola sorgente.

L'area di ricerca sulla fusione di dati è in rapida crescita e richiede conoscenze interdisciplinari che coinvolgono l'elaborazione di segnali, la teoria dei processi stocastici, la teoria del controllo, l'intelligenza artificiale, etc.

La fusione di dati si riferisce ad una combinazione sinergica di dati correlati e/o ridondanti provenienti da più sensori in modo da:

- compensare temporanei malfunzionamenti, mancanza o elevata imprecisione di misura di un sensore
- rendere più accurate e affidabili le elaborazioni dei dati rispetto a quanto si potrebbe fare utilizzando un singolo sensore
- fondere informazioni ottenute da sensori eterogenei, ad esempio video e laser, e ottenere una maggiore completezza e complementarietà di informazione sulla realtà osservata.

Originariamente l'integrazione dei dati da più sensori era essenzialmente legata ad applicazioni militari di sorveglianza, difesa e tattica sul campo. Più recentemente l'utilizzo di tali tecniche ha visto una crescente diffusione in sistemi di diagnostica medica, nella robotica, nell'analisi ambientale e nei sistemi

automotive [5]. Infatti, le informazioni complementari da sensori multipli permettono la percezione di caratteristiche dell'ambiente circostante che prima era impossibile ottenere utilizzando informazioni elaborate singolarmente [6]. I risultati ottenuti con la fusione dei dati aiutano l'utente a prendere decisioni in scenari complessi.

Per le caratteristiche appena descritte l'applicazione di tali tecniche in ambito automotive risulta essere particolarmente vantaggiosa, in particolare per applicazioni riguardanti sistemi di assistenza alla guida. I sistemi elettronici di assistenza alla guida dovrebbero essere in grado di gestire informazioni eterogenee quali:

- comandi del guidatore (angolo di sterzo, cambiamento di corsia, sorpasso di un veicolo, etc.);
- velocità del veicolo;
- flusso del traffico (basso o denso);
- comportamento del conducente (sonnolenza, guida aggressiva, uso di un telefono cellulare, etc.).

I sensori utilizzati per acquisire queste informazioni presentano vantaggi e svantaggi che dipendono dalle loro caratteristiche fisiche. Le telecamere hanno alcuni inconvenienti, come la bassa capacità di rilevamento della profondità, e il vantaggio di una maggiore capacità di discriminazione degli oggetti e dell'ambiente circostante. Il radar ha un campo visivo ridotto o una risoluzione ridotta a grandi distanze. Il laser, sebbene abbia un largo campo visivo ha una bassa capacità di discriminazione.

Ogni tecnologia ha quindi i suoi limiti in un determinato campo di azione. La fusione dei dati si propone come un metodo di integrazione di tecnologie diverse al fine di colmare tali lacune e ottenere a costi minori informazioni di qualità superiore.

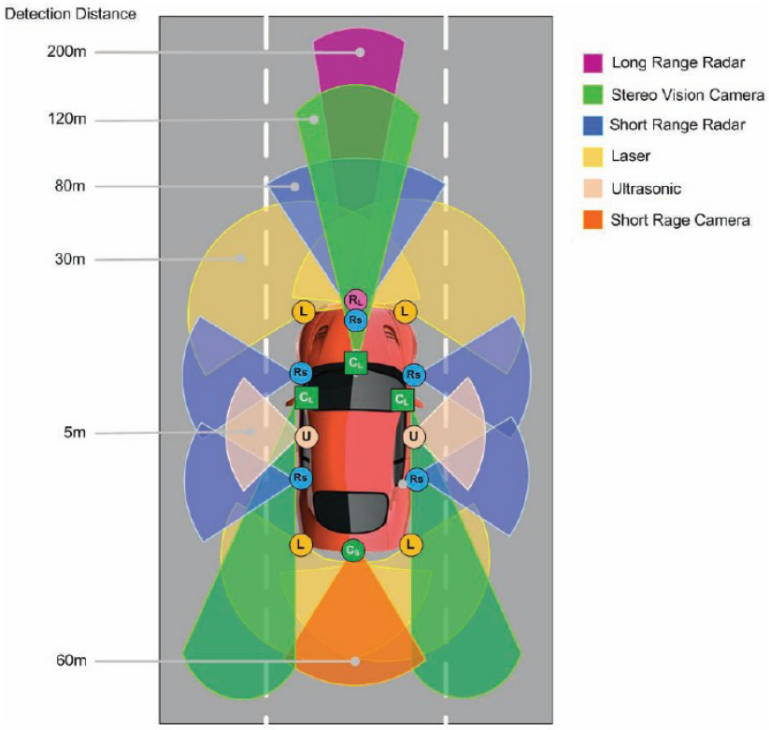

Figura 2.1. Rete di sensori per sistemi di assistenza alla guida

Per questo motivo la ricerca automobilistica dedica alle tecniche di fusione dei dati un crescente interesse [7].

I sistemi LDWS hanno il compito di avvisare il conducente in caso di imminente uscita dalla corsia di marcia in tempo utile ad effettuare le manovre correttive del caso. La configurazione classica di un sistema di questo tipo prevede l'uso di una telecamera che legge la posizione delle linee di demarcazione della corsia rispetto alla vettura. L'utilizzo di un solo sensore di visione rende questi sistemi vulnerabili ad errori di misura e fortemente condizionati da temporanee occlusioni. In queste situazioni LDWS si disattiva e viene sospeso il supporto al guidatore.

Sfruttando le proprietà degli algoritmi di fusione dei dati si vuole progettare un sistema che utilizzi oltre alla telecamera anche i dati forniti da altri sensori a bordo vettura. In questo modo si vuole rendere LDWS robusto a temporanee occlusioni o malfunzionamenti della telecamera e naturalmente più preciso e affidabile nell'avviso di eventuali situazioni di pericolo. Il sistema LDWS basato su fusione dei dati, inoltre, dovrà ridurre i falsi allarmi che provocano distrazione del conducente e a lungo andare riducono la fiducia nel sistema. Per arrivare a conclusioni oggettive, sono state valutate strategie diverse al fine di determinare l'algoritmo di fusione dei dati migliore per

questa specifica applicazione. Gli algoritmi sono stati progettati in ambiente Matlab/Simulink e testati tramite software di simulazione Carsim.

2.2 Lane Departure Warning Systems

La guida intesa come attività giornaliera è un'operazione complessa che necessita di un elevato grado di interazione tra guidatore, veicolo e ambiente circostante. I guidatori generalmente dividono la loro attenzione tra la guida del veicolo, il monitoraggio del traffico e degli oggetti circostanti e l'esecuzione di operazioni secondarie come l'utilizzo di sistemi di bordo (climatizzatore, gps, radio), conversazione, etc. La complessità delle operazioni e l'incertezza dell'ambiente circostante rende la guida un compito molto rischioso. Il rapporto annuale 2008 dell'Osservatorio Europeo sulla Sicurezza Stradale (ERSO) [12] riporta oltre 1.000.000 incidenti, con circa 40.000 morti solo nel corso del 2005. In Italia, più del 15% degli incidenti è dovuto a stanchezza e negligenza da parte del guidatore [13].

Negli ultimi anni la ricerca in campo automobilistico ha focalizzato la sua attenzione sullo sviluppo di nuovi sistemi di assistenza alla guida che, oltre ad aumentare il comfort di guida, possano avere un impatto rilevante sulla riduzione del numero di incidenti. Le più importanti case automobilistiche offrono un numero sempre maggiore di sistemi di sicurezza attivi innovativi:

- Attiva: agiscono autonomamente sui dispositivi elettro-meccanici del veicolo (sterzo, freni, etc.) in caso di pericolo cercando di evitare l'impatto;
- Preventiva: hanno il compito di avvisare il guidatore in situazioni particolari delegando a quest'ultimo azioni correttive sulla guida;
- Passiva: agiscono pochi istanti prima dello scontro ormai inevitabile mitigando i suoi effetti.

Tra i vari dispositivi di sicurezza attiva (sistemi per la frenata di emergenza, sistemi per il riconoscimento dei pedoni, sistemi anti-collisione), i dispositivi per l'avviso di superamento involontario della corsia di marcia (Lane Departure Warning System - LDWS) risultano di particolare interesse vista la facilità di installazione anche su vetture già in circolazione e la possibilità di realizzazione a costi contenuti.

I sistemi LDWS sono dispositivi di visione artificiale real-time che attraverso l'uso di una telecamera (figura (2.2))

- identificano le strisce orizzontali della carreggiata e la corretta posizione e direzione di marcia dell'autovettura
- stimano l'intervallo di tempo entro il quale il veicolo raggiungerà i limiti della corsia di marcia proseguendo con l'attuale velocità e direzione di marcia
- avvisano il conducente in caso di superamento di tali limiti in tempo utile ad effettuare una manovra correttiva

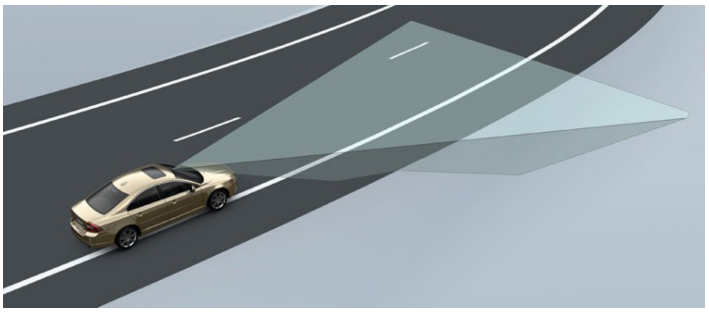

Figura 2.2. Lane Departure Warning System

Da recenti studi europei [14] risulta che l'introduzione di sistemi LDWS a bordo vettura comporterebbe:

- una potenziale riduzione del 25% del numero di collisioni frontali e uscite di strada
- una riduzione sulla gravità delle ferite del 25% per urti frontali e del 15% per uscite di strada
- una riduzione del 60% del numero di urti laterali fra autovetture con LDWS integrato ad un sistema di assistenza al cambio di corsia

La funzione principale del sistema LDWS è quella di aiutare il conducente a rimanere nella corsia di marcia. Una telecamera è montata sul cruscotto anteriore ed è usata per identificare la posizione e l'orientamento del veicolo rispetto alla corsia di marcia (attraverso il riconoscimento delle strisce di demarcazione orizzontali).

Questi dati sono integrati con altre informazioni ottenute dai sistemi di controllo elettronici esistenti a bordo (velocità delle ruote, angolo di sterzo, etc.) oppure ottenute con sensori aggiuntivi. La fusione di tutte queste informazioni consente di stimare il Tempo di Invasione Corsia Attigua (Time to Lane Crossing t_{tlc}) [15],[16],[17],[18], calcolato in modo approssimato dividendo la posizione laterale entro la corsia di marcia (y), misurata rispetto ai limiti delle linee di demarcazione, con la velocità laterale (V_y) (figura (2.3))

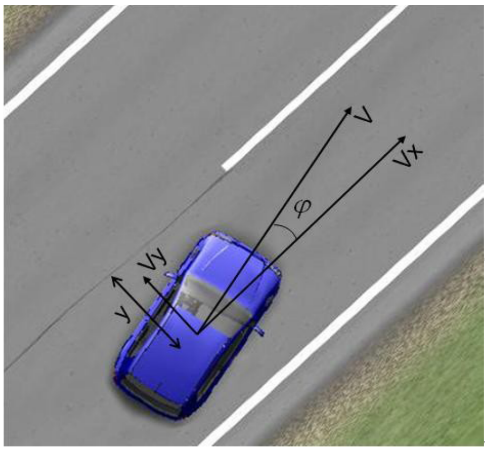

Figura 2.3. Calcolo t_{tlc}

Quando il tempo di t_{tlc} è più piccolo di una soglia prefissata, il sistema avverte il conducente che il veicolo sta avvicinandosi pericolosamente alle strisce orizzontali che delimitano la corsia di marcia e potrebbe invadere la corsia attigua o finire sul guard-rail.

Molti sistemi generano un avviso acustico o ottico (utilizzando ad esempio il display del navigatore se le due funzioni sono integrate), mentre altri utilizzano vibrazioni sul sedile o sul volante o forniscono una piccola coppia al volante nella direzione di correzione, non sufficiente a correggere del tutto l'azione del conducente. Infatti, la filosofia di base di questi sistemi è che sia sempre il conducente a decidere le azioni correttrici. Il sistema esegue le seguenti operazioni:

1. acquisisce le immagini dalle telecamere e dai sensori installati;
2. processa a ogni istante di campionamento l'immagine della strada identificando la segnaletica orizzontale e i limiti della corsia di marcia [9]. Esegue inoltre test di consistenza sulle immagini processate mantenendo la continuità dei risultati ottenuti nei vari frame;
3. Utilizza un algoritmo di fusione dei dati per integrare i dati forniti dai sensori installati sulla vettura con quelli forniti dal sistema di visione e determina la posizione e l'orientamento all'interno della carreggiata;
4. Calcola il tempo di invasione di corsia attigua e avverte il conducente in caso di imminente pericolo se questo è sotto una determinata soglia.

La sua operatività deve essere assicurata anche di notte quando sono accesi i proiettori. Il sistema è particolarmente utile per aiutare il guidatore ad evitare sbandamenti dovuti a sonnolenza o distrazione su strade che richiedono un basso livello di attenzione quali autostrade o lunghi rettilinei. In tutti questi casi è necessario una predizione affidabile del t_{tlc} nelle diverse condizioni di

guida e ambientali per evitare la generazione di falsi allarmi che disturbano la guida, piuttosto che aiutarla, se generati con troppo frequenza.

Al fine di aumentare la robustezza e l'accuratezza nel calcolo del t_{tlc}, sono state approfondite le metodologie di fusione dei dati che utilizzano stimatori non-lineari quali il l'Extended Kalman Filter e l'Unscendent Kalman Filter, basati su modelli matematici non-lineari del moto del veicolo (modelli cinematici e/o dinamici). Per tutte le combinazioni sono stati valutati:

- i vantaggi che si ottengono utilizzando modelli del veicolo sempre più complessi e un maggior numero di sensori;
- le prestazioni di ogni architettura di data fusion;
- il carico computazionale e la complessità del sistema embedded risultante per ogni architettura considerata.

2.2.1 Struttura algoritmo per LDWS

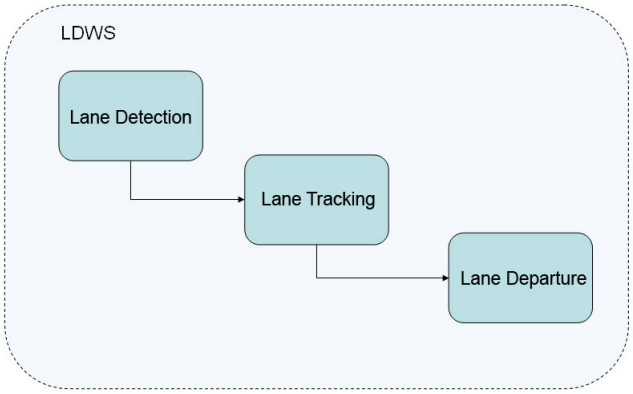

Figura 2.4. Struttura algoritmo LDWS

Un tipico algoritmo per un sistema LDWS può essere suddiviso essenzialmente in tre macro-blocchi funzionali (figura (2.4)):

- Lane Detection: riconoscimento delle linee che delimitano la carreggiata in un singolo frame d'immagine
- Lane Tracking: riconoscimento e associazione delle linee riconosciute in più frame consecutivi d'immagine
- Lane Departure: segnalazione di superamento dei limiti della corsia

2.3 Algoritmi di identificazione delle linee su singolo frame

Le fasi necessarie all'identificazione delle linee su singolo frame (Lane Detection) sono elencate in dettaglio in figura (2.3).

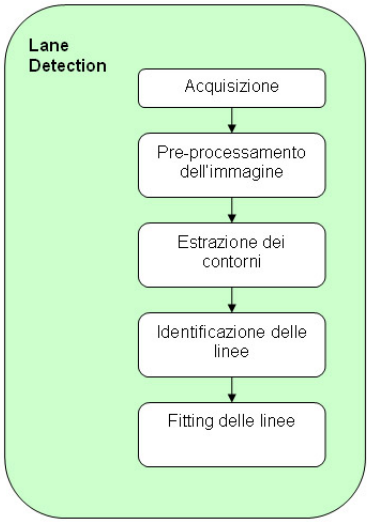

Figura 2.5. Lane Detection

Dopo aver acquisito un frame d'immagine da telecamera avviene una prima fase di pre-processamento per migliorarne la qualità ed eseguire eventualmente alcune correzioni prospettiche. L'immagine pre-processata subisce un filtraggio necessario ad estrarre i contorni e identificare i pixel appartenenti alle linee di demarcazione della carreggiata. L'ultima fase è il fitting dei pixel delle linee con cui si ottiene una rappresentazione matematica della segnaletica orizzontale all'interno dell'immagine.

Nelle prossime sotto-sezioni si riporta una descrizione approfondita degli algoritmi utilizzabili in ogni fase.

2.3.1 Pre-processamento dell'immagine

Il pre-processing dell'immagine è un'elaborazione iniziale eseguita allo scopo di semplificare e rendere più efficace la successiva estrazione delle sue caratteristiche. Si utilizzano tecniche di filtraggio per:

- Migliorare il contrasto dell'immagine
- Ridurre il rumore

- Separare meglio gli oggetti dallo sfondo

In particolare sono state utilizzate: miglioramento livelli di grigio, media locale variabile, equalizzazione dell'istogramma, smoothing dell'immagine.

Miglioramento livelli di grigio

Il più semplice miglioramento dell'immagine è la variazione dei livelli di grigio intervenendo sul valore di intensità dei pixel. Via software è possibile cambiare l'intensità di un pixel in ingresso in un nuovo valore in uscita. Il rimappare i livelli di grigio è spesso chiamato lo **stretching** di un immagine in gergo tecnico, poichè viene effettuato uno stiramento dei livelli di grigio di un'immagine all'interno dell'intero set di valori disponibili. La variazione dei livelli di grigio dipende dalla funzione di rimappatura utilizzata, ad esempio:

- $f(x) = x^{0.5}$ aumenta tutte le intensità ma, a causa della sua non linearità, aumenta di più quelle basse rispetto alle alte
- $f(x) = x^{1/\gamma}$, nota anche come correzione gamma, permette di scegliere se aumentare o diminuire l'intensità dei grigi al variare di gamma

Media locale variabile

La media locale variabile appiattisce un'immagine riducendo la variazione delle intensità localmente. Il valore di intensità di un pixel è ottenuto effettuando la media delle intensità dei pixel in un suo intorno. É, inoltre, possibile variare la dimensione dell'intorno. Ciò è desiderabile per immagini che presentano molto rumore, sopratutto in corrispondenza dei contorni dell'immagine (edge).

La rappresentazione matematica di questo metodo è la seguente. Supponendo $a \in \Re^x$ l'immagine originale e $N : X \to 2^x$ una funzione dell'intorno $x \in X$. Se n_y indica il numero di punti in $N(y)$, l'immagine b ottenuta è:

$$b(y) = \frac{1}{n_y} \sum_{x \in N(y)} a(x) \tag{2.1}$$

Equalizzazione dell'istogramma

L'equalizzazione dell'istogramma è una tecnica che riscala l'intensità dei pixel di un'immagine per produrre un'immagine in cui i valori dei pixel sono distribuiti più uniformemente. L'immagine ottenuta tende ad avere un maggiore contrasto.

La formalizzazione matematica è la seguente. Consideriamo $a \in \Re_i^x$ l'immagine sorgente, con $\Re_i = \{0, 1, \cdots, i-1\}$, $n = card(X)$ e n_j il numero di volte che il livello di grigio j è presente nell'immagine a. Allora, l'immagine b ottenuta è data da

$$b(x) = i \sum_{j=0}^{a(x)} \frac{n_j}{n} \tag{2.2}$$

Smoothing dell'immagine

I filtri di smoothing sono usati per ridurre il rumore e per sfuocare le immagini. Solitamente le immagini vengono sfuocate durante il pre-processamento se si vogliono eliminare piccoli dettagli prima di estrarre grandi oggetti o eliminare piccoli salti nelle linee e nelle curve. La riduzione del rumore può essere effettuata sfuocando l'immagine con un filtro lineare o non lineare.

La risposta di uno smoothing con filtro lineare è semplicemente la media dei pixel contenuti nei limiti della maschera del filtro. Questi filtri solitamente sono chiamati filtri mediani o filtri passa-basso. Sostituendo il valore di ogni pixel con la media dei livelli di grigio all'interno della maschera del pixel, il risultato del processo è un'immagine con transizioni di grigio più dolci (smussate). Poichè il rumore casuale tipicamente consiste in transizioni nette nei livelli di grigio, l'applicazione più comune di tali filtri è la riduzione del rumore.

Comunque, considerando che anche i contorni sono caratterizzati da brusche transizioni nei livelli di grigio il filtraggio potrebbe avere effetti indesiderati nell'identificazione dei contorni. Risulta quindi necessario trovare il giusto compromesso tra riduzione del rumore ed eccessivo sfuocamento.

2.3.2 Inversione prospettica

Un sistema di visone per l'identificazione della carreggiata richiede informazioni quali la segnaletica orizzontale, la curvatura della strada e la posizione del veicolo a ogni istante di campionamento. Montando una telecamera sul veicolo le immagini acquisite forniscono una visione prospettica della strada che distorce la forma reale della strada in larghezza, altezza e profondità. Questi parametri rappresentano le componenti x, y e z.

Per eliminare tali distorsioni è necessaria una fase di pre-elaborazione nota come inversione prospettica (IPM) che permette di avere una visione "dall'alto" della strada. Questa trasformazione permette di rimuovere l'effetto prospettico dell'immagine acquisita, rimappandola in un nuovo dominio a due dimensioni nel quale il contenuto informativo è distribuito omogeneamente su tutti i pixel. Ovviamente l'applicazione della trasformazione IPM richiede conoscenze sulle specifiche condizioni di acquisizione (posizione della camera, orientazione, ottica, etc.) e alcune assunzioni sulla scena ripresa nell'immagine.

In seguito la discussione sarà limitata al caso di una superficie planare: in questo caso l'uso di IPM permette di ottenere una vista dall'alto della scena.

IPM è una trasformazione geometrica che rimappa una immagine prospettica 2D di un oggetto 3D in una nuova immagine planare a 2D. Le equazioni geometriche che regolano tale trasformazione sono universalmente accettate e largamente presenti in letteratura anche in articoli espressamente rivolti ad applicazioni automotive.

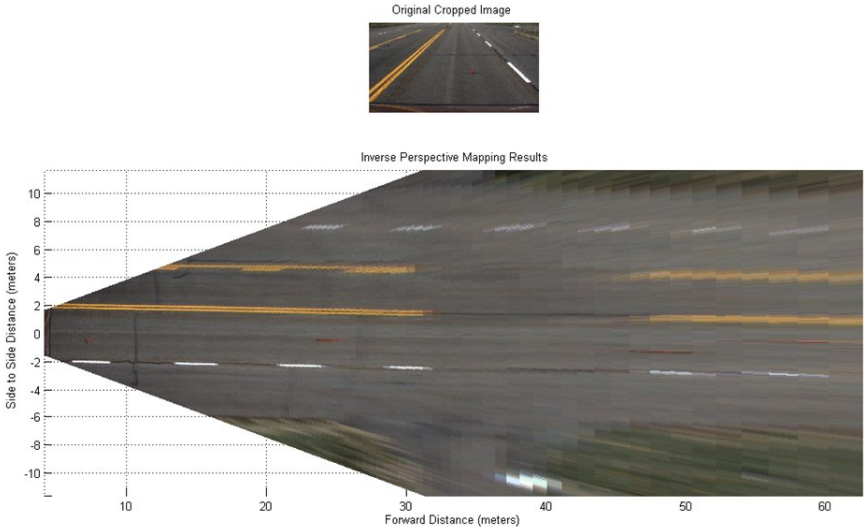

Figura 2.6. Immagine originale e corrispondente inversione prospettica

Matematicamente può essere descritta come una proiezione da uno spazio euclideo 3D, $W = (x, y, z)$ (Spazio Reale) a uno spazio planare a 2D, $I = (u, v)$(Spazio Immagine).

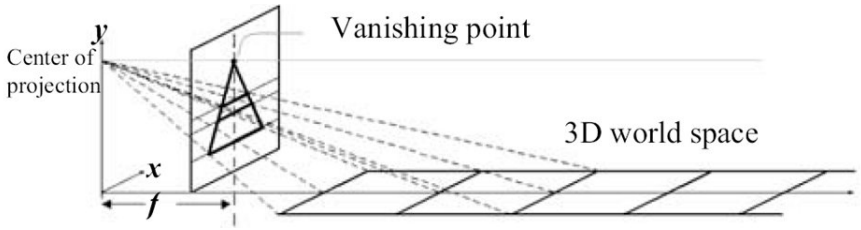

Figura 2.7. Corrispondenza piano immagine mondo reale, f:lunghezza focale videocamera

Le equazioni derivate per il modello IPM sono:

- Conversione dallo spazio immagine allo spazio reale

$$\begin{aligned} x(r) &= h \cdot \left(\frac{1+\left[1-2\left(\frac{r-1}{m-1}\right)\right] tan\alpha_v tan\theta_0}{tan\theta_0 - \left[1-2\left(\frac{r-1}{m-1}\right)\right] tan\alpha_v} \right) \\ y(r,c) &= h \cdot \left(\frac{1+\left[1-2\left(\frac{c-1}{r-1}\right)\right] tan\alpha_u}{sin\theta_0 - \left[1-2\left(\frac{r-1}{m-1}\right)\right] tan\alpha_v cos\theta_0} \right) \end{aligned} \tag{2.3}$$

- Conversione dallo spazio reale allo spazio immagine:

$$\begin{aligned} r(x) &= \frac{m-1}{2} \left(1 + \frac{h - x tan\theta_0}{h tan\theta_0 + x} cotan\alpha_v \right) + 1 \\ c(x,y) &= \frac{n-1}{2} \left(1 - \frac{y}{h sin\theta_0 + x cos\theta_0} cotan\alpha_u \right) + 1 \end{aligned} \tag{2.4}$$

Con

- m,n numero di righe e colonne di pixel dell'immagine
- r,c righe e colonne pixel immagine
- α_v e α_u angoli di visione verticale e orizzontale
- θ_0 Angolo tra l'asse della telecamera e l'orizzontale

La vista laterale e la vista dall'alto del modello geometrico dell'inversione prospettica è visibile in figura (2.8).

L'inversione prospettica permette di rimappare il frame acquisito da telecamera in una nuova immagine in cui a ogni pixel è assegnata una coordinata metrica nel mondo reale (x,y), nel caso in esame l'immagine ottenuta è una rappresentazione del piano stradale con dimensioni $20x32m$ come illustrato in figura (2.9). L'asse x dell'immagine invertita prospetticamente coincide con l'asse ottico della telecamera e con l'asse longitudinale della vettura come mostrato in figura (2.8).

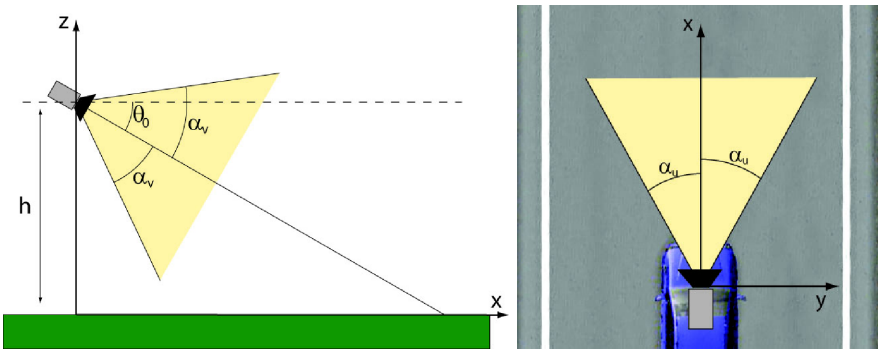

Figura 2.8. Inversione prospettica: vista laterale (sx), vista dall'alto (dx)

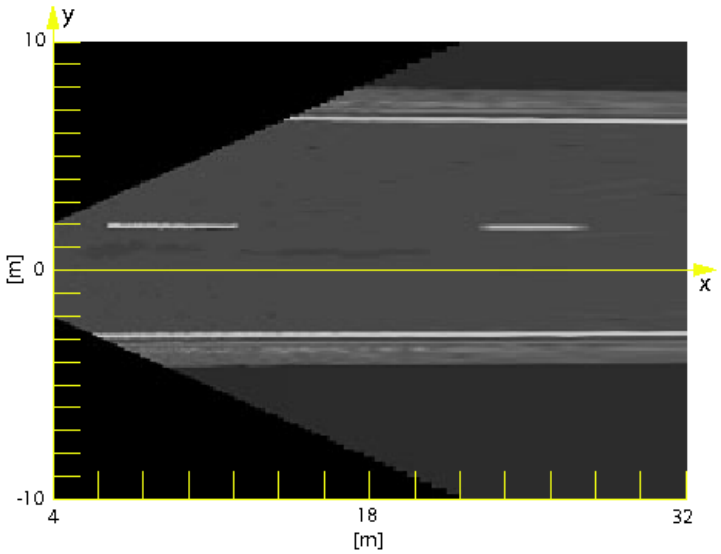

Figura 2.9. Inversione prospettica: riferimenti metrici

2.3.3 Estrazione dei contorni

Il contorno (contour, edge, border) di un oggetto rappresenta la linea di separazione tra l'oggetto e lo sfondo o tra l'oggetto ed altri oggetti.

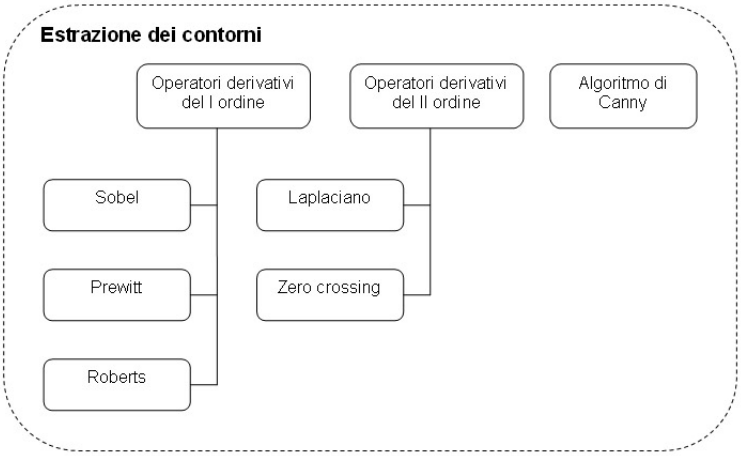

Figura 2.10. Estrazione dei contorni

Gli algoritmi di segmentazione edge-based sono basati sulla discontinuità di proprietà dei pixel, quali per esempio il livello di grigio, il colore, la regolarità spaziale, la velocità.

La maggior parte degli operatori per la rivelazione delle discontinuità fa uso di operatori derivativi. Tali operatori vengono applicati direttamente all'immagine (graylevel o RGB) o ad una sua qualche grandezza derivata, ottenuta applicando una opportuna trasformazione.

I punti di discontinuità dell'immagine vengono rivelati e concatenati in modo da ottenere dei contorni chiusi che delimitano aree distinte. La derivata prima e seconda sono diverse da zero solo in corrispondenza delle transizioni scuro-chiaro, chiaro-scuro.

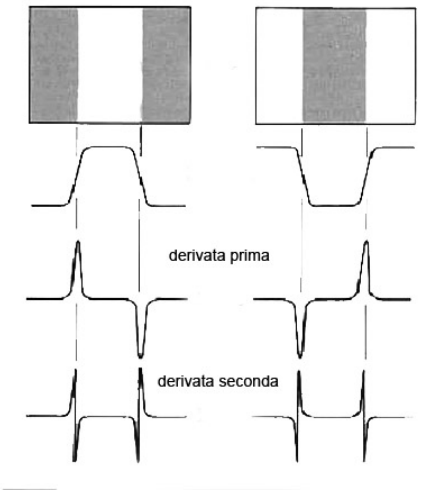

Figura 2.11. Derivata prima e seconda

Nella figura si osserva che:

- la derivata prima del profilo è positiva in corrispondenza di una transizione scuro-chiaro, negativa in corrispondenza di una transizione chiaro-scuro, nulla nelle zone a livello di grigio costante.
- La derivata seconda positiva in prossimità del lato scuro di un contorno, negativa dal lato chiaro, nulla dove il livello di grigio costante, e passa per zero (zero crossing) in corrispondenza delle transizioni.

In generale quindi, per un immagine in scala di grigi $I(x, y)$, un contorno è definito dal gradiente della funzione d'intensità:

$$\nabla I(x,y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)^T \approx (D_x, D_y)^T \tag{2.5}$$

con

- magnitudine: $|\nabla I(x,y)| \approx |D_x| + |D_y|$
- orientazione: $\theta(x,y) = tan^{-}1\,(D_y/D_x)$

Le derivate parziali,D_x e D_y, non sono solitamente calcolate direttamente ma vengono approssimate utilizzando opportuni operatori derivativi (coppia di filtri digitali).

Operatori derivativi del primo ordine

Tali operatori utilizzano due kernels, uno per i cambiamenti verticali e uno per quelli orizzontali. La convoluzione tra i kernels e l'immagine originaria permette di calcolare le derivate parziali approssimate.

Figura 2.12. Immagine originale

- **Sobel**

$$D_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, D_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Figura 2.13. Operatore di Sobel

- **Prewitt**

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, D_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

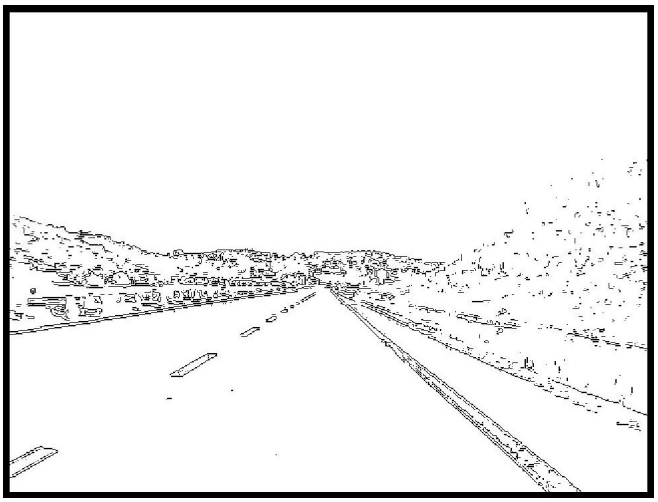

Figura 2.14. Operatore di Prewitt

- **Roberts**

$$D_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, D_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

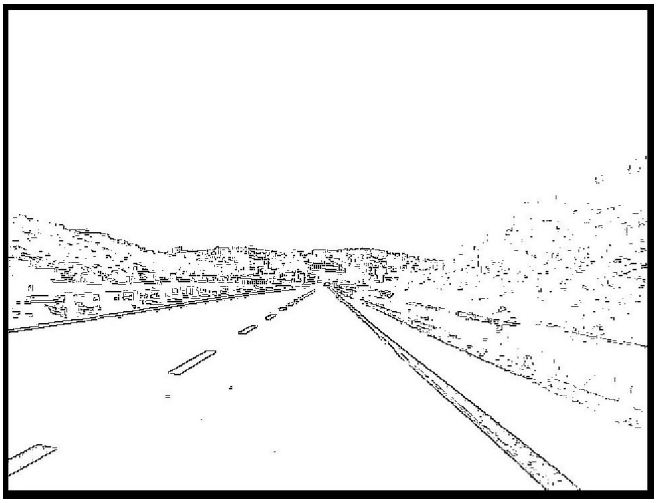

Figura 2.15. Operatore di Roberts

Operatori derivativi del secondo ordine

- **Laplaciano di Gaussiana**
  Un modo molto comune di effettuare le operazioni di derivata seconda di una funzione $f(x, y)$ in un punto è quello di calcolare il laplaciano in quel punto. Ricordiamo che data $f(x, y)$, il laplaciano è definito come:

$$L(x, y) = \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.6}$$

  Approssimando il Laplaciano nel caso discreto con le differenze si ha

$$L(x, y) = 4f(x, y) - f(x-1, y) - f(x+1, y) - f(x, y-1) - f(x, y+1) \tag{2.7}$$

  ottenendo il filtro con la seguente risposta impulsiva:

$$L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & -4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

  Il laplaciano è eccessivamente sensibile al rumore (in quanto operatore di derivata seconda) ed incapace a rilevare la direzione del contorno (in quanto entità scalare). Per tali motivi, il laplaciano raramente è usato da solo per l'edge-detection. Una soluzione proposta da Marr e Hildreth utilizza il laplaciano in connessione con un filtro di smoothing, ancora una volta una gaussiana, realizzando un operatore detto Laplaciano della Gaussiana o LoG.

$$\nabla^2 h = -\frac{r^2 - \sigma^2}{2\sigma^2} e^{\left(-\frac{r^2}{2\sigma^2}\right)} \tag{2.8}$$

  La convoluzione dell'immagine con tale operatore equivalente a filtrare l'immagine con un filtro di smoothing gaussiano, per ridurre gli effetti del rumore, e successivamente calcolare il laplaciano del risultato. La localizzazione dei contorni quindi consiste solo nella ricerca dei punti di attraversamento a zero (zero crossing) della derivata seconda dell'immagine.

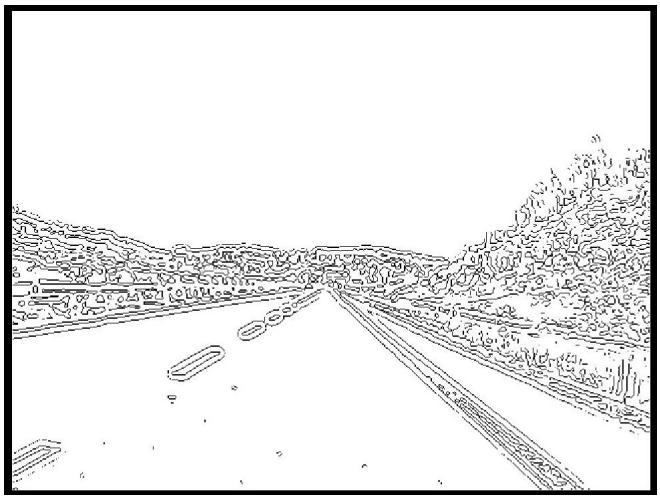

Figura 2.16. Operatore Laplaciano di una Gaussiana

- **Zero Crossing**
  Esistono vari altri metodi definiti come zero crossing che si basano sullo stesso concetto del Laplaciano di una Gaussiana ma che utilizzano altri tipi di filtri per effettuare lo smoothing dell'immagine invece che quello di Gauss.

Algoritmo di Canny

Tutti gli operatori derivativi finora presentati hanno un'origine euristica. Un approccio analitico è stato invece seguito da J. Canny, che ha studiato in dettaglio il comportamento dell'operatore gradiente applicato ad un'immagine rumorosa.

Canny ha fissato tre criteri di qualità per un edge detector ed ha determinato analiticamente il filtro ottimo nei confronti di questi tre criteri. L'edge-detection è effettuata per convoluzione del contorno rumoroso $f(x)$ con una $h(x)$, scelta in modo da soddisfare tre criteri:

1. Capacità di individuazione (detection) dell'edge
2. Capacità di localizzazione dell'edge
3. Unicità della risposta

Un edge è individuato in corrispondenza ad un massimo locale della convoluzione $f(x) * h(x)$.

Figura 2.17. Operatore di Canny

Confronto tra i vari algoritmi di edge-detection

Le varie tecniche di estrazione dei contorni sopra citate sono state confrontate sia dal punto di vista computazionale sia dal punto di vista dell'affidabilità del risultato. In particolare, per questo secondo aspetto, è stato utilizzato l'indice FOM (figure of merit) di Pratt:

$$FOM = \frac{1}{I_N} \sum_{i=1}^{I_A} \frac{1}{1 + ad^2(i)} \tag{2.9}$$

dove:

- $I_N = max(I_I, I_A)$, con I_I e I_A rispettivamente numero di punti sul contorno ideali ed individuati
- a fattore di scala, solitamente pari a $1/9$
- $d(i)$ distanza di separazione tra il pixel i individuato e la sua corretta posizione nell'immagine ideale.

L'immagine ideale è ottenuta manualmente indicando la posizione dei pixel ideali in un'area di interesse dell'immagine originaria.

Figura 2.18. Immagine originaria

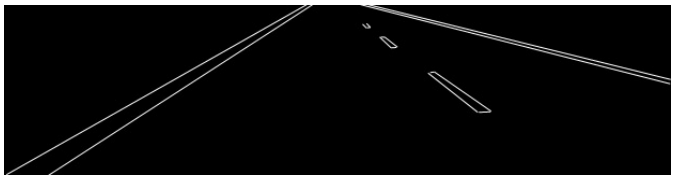

Figura 2.19. Immagine ideale

Edge Detector	Time (s)	FOM
Sobel	0,015	0,315
Prewitt	0,015	0,311
Roberts	0,018	0,275
Canny	0,193	0,217
LoG	0,045	0,252

Tabella 2.1. Confronto Edge Detector

Dai risultati in Tabella 1, risulta chiaro che per i nostri scopi è opportuno utilizzare l'operatore di Sobel o quello di Prewitt. Tali operatori infatti risultano più performanti sia rispetto al tempo di esecuzione che rispetto al FOM.

2.3.4 Identificazione delle Linee

Funzione di distribuzione dei contorni EDF

La funzione di distribuzione dei contorni $F(d)$ definita come l'istogramma della magnitudine del gradiente rispetto alla sua orientazione.

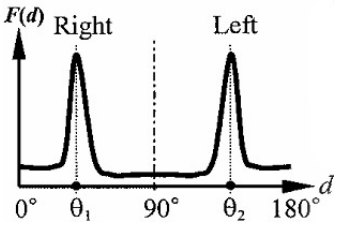

Figura 2.20. Funzione di distribuzione dei contorni

Su uno sfondo stradale, i massimi locali di $F(d)$ corrispondono alle linee della carreggiata.

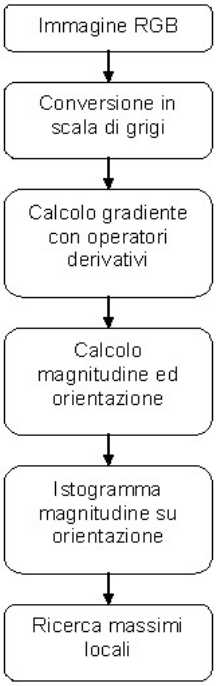

Figura 2.21. Identificazione linee con EDF

Trasformata di Hough

È una tecnica che permette di riconoscere particolari configurazioni di punti presenti nell'immagine, come segmenti, curve o altre forme prefissate. Il principio fondamentale è che la forma cercata può essere espressa tramite una funzione nota che fa uso di un insieme di parametri. Una particolare istanza

della forma cercata è quindi completamente precisata dal valore assunto dall'insieme di parametri.

Per esempio assumendo come rappresentazione della retta la forma $y = ax+b$, qualunque retta completamente specificata dal valore dei parametri (a, b).

Se si assume un tipo di rappresentazione diversa,$\rho = xcos\theta + ysin\theta$, l'insieme di parametri varia di conseguenza; in questo caso la retta completamente specificata dalla coppia (ρ, θ).
Quindi, fissata la forma di interesse e la sua rappresentazione, è possibile considerare una trasformazione dal piano dell'immagine allo spazio dei parametri.

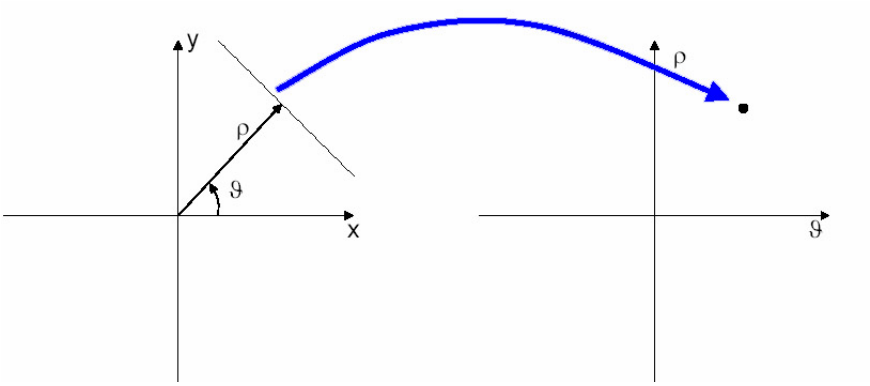

Figura 2.22. Trasformazione piano immagine - spazio parametri

Nel piano dell'immagine un punto rappresentato dall'intersezione di più rette. Ad ogni punto P corrisponderà, nel piano dei parametri, la curva formata dai punti immagine delle rette passanti per P.

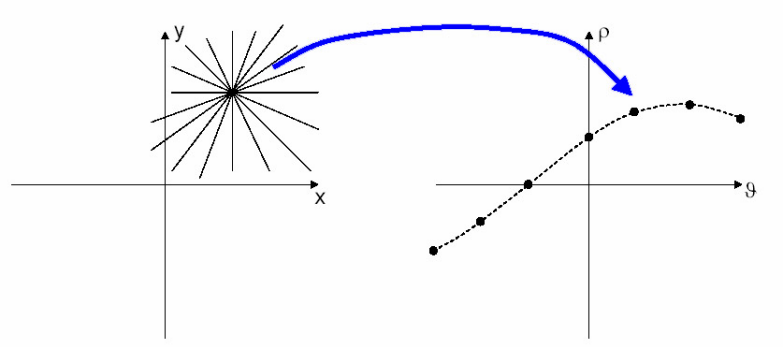

Figura 2.23. Trasformata di un punto

Le curve, che corrispondono alle trasformazioni dei vari punti, si intersecano in un punto del piano trasformato che è l'immagine della retta su cui giacciono i punti.

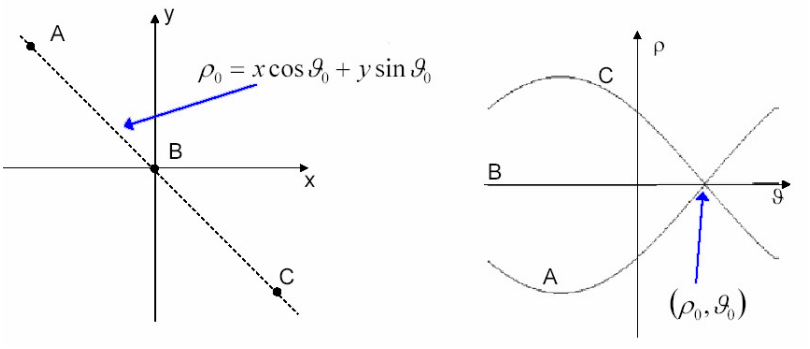

Figura 2.24. Trasformata di tre punti

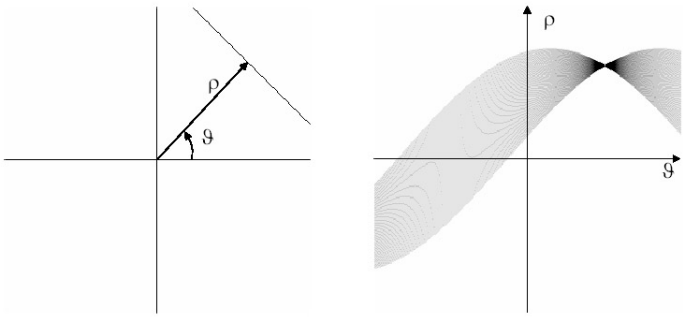

Figura 2.25. Trasformata di una retta

Per la determinazione dei punti di intersezione, e quindi delle rette corrispondenti, si consideri una discretizzazione del piano dei parametri (ρ, θ). Ciò permette di rappresentare tale piano su una matrice $H(m, n)$ di accumulazione i cui indici di riga e di colonna corrispondono ai valori quantizzati di ρ e θ. Per ogni punto P(x,y) nell'immagine

- per $\theta_n \in [-\pi/2, \pi/2]$ con passo $d\theta$
  - si valuti $\rho(n) = xcos(\theta_n) + ysin(\theta_n)$
  - si ricavi l'indice m corrispondente a $\rho(n)$
  - si incrementi $H(m, n)$

si individuano i massimi locali su H corrispondenti ai parametri dei segmenti individuati.

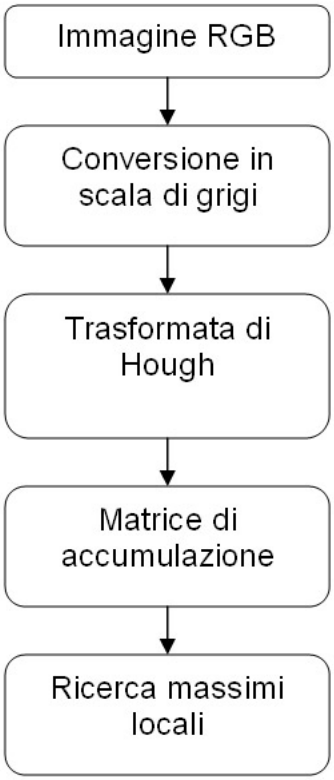

Figura 2.26. Identificazione delle linee con Hough

Zero-crossing del gradiente su immagine filtrata con filtri Gaussiani

E' un metodo che utilizza il Laplaciano di un'immagine individuando i punti in cui esiste un attraversamento a zero. Questa condizione si presenta solitamente in corrispondenza dei contorni di un'immagine dove l'intesità cambia rapidamente. Il punto di partenza del metodo è sempre un'immagine precedentemente filtrata da un operatore di smoothing di tipo Gaussiano. Il risultato è quindi fortemente influenzato dalla dimensione del kernel del filtro.

Nei sistemi di visione per l'analisi delle immagini, l'estrazione dei contorni e delle texture sono spesso usati filtri orientati. In queste applicazioni è necessario applicare i filtri in diverse configurazioni ed esaminarne l'uscita che è funzione sia della fase che della direzione. L'approccio classico per trovare la risposta di un filtro in varie direzioni consiste nell'applicare diverse versioni dello stesso filtro che differiscono solo per piccoli angoli di rotazione. Questa soluzione risulta essere computazionalmente onerosa soprattutto nel caso

di implementazione in sistemi embedded. Un approccio più efficiente, facente uso di "filtri ruotabili" (steerable filters), è calcolare i filtri in poche fissate direzioni e ottenere la risposta nelle altre orientazioni come interpolazione lineare. Con determinati filtri e con la corretta funzione di interpolazione è possibile applicare un filtro a una arbitraria orientazione senza calcolarlo esplicitamente.

Consideriamo a scopo di esempio una funzione gaussiana simmetrica a 2 dimensioni scritta in coordinate cartesiane x,y:

$$G(x, y) = e^{-(x^2+y^2)} \tag{2.10}$$

l'operatore derivata direzionale è ruotabile, indichiamo con G_n la derivata $n - esima$ di G e con G^θ l'operazione di rotazione della funzione G di un angolo θ. La derivata prima di G lungo l'asse x sarà data da

$$G_1^0 = \frac{\partial}{\partial x} e^{-(x^2+y^2)} = -2xe^{-(x^2+y^2)} \tag{2.11}$$

La stessa funzione, ruotata di 90 gradi,

$$G_1^{90} = \frac{\partial}{\partial y} e^{-(x^2+y^2)} = -2ye^{-(x^2+y^2)} \tag{2.12}$$

Utilizzando le proprietà dei filtri ruotabili è possibile calcolare la derivata prima di G per un'orientazione arbitraria θ come combinazione lineare di G_1^0 e G_1^{90}

$$G_1^\theta = cos(\theta)G_1^0 + sin(\theta)G_1^{90} \tag{2.13}$$

dove G_1^0 e G_1^{90} sono le *basi* del filtro per G_1^θ e $Cos(\theta)$ e $sin(\theta)$ sono le corrispondenti funzioni di interpolazione per queste basi. Poiché l'operatore di convoluzione è un operatore lineare, è possibile calcolare un'immagine filtrata per una qualsiasi direzione effettuando l'interpolazione dell'immagine filtrata con G_1^0 e G_1^{90}. Indicando con $*$ l'operatore convoluzione,

$$\begin{aligned} R_1^0 &= G_1^0 * I \\ R_1^{90} &= G_1^{90} * I \end{aligned} \tag{2.14}$$

quindi

$$R_1^\theta = cos(\theta)R_1^0 + sin(\theta)R_1^{90} \tag{2.15}$$

R_1^θ rappresenta la derivata prima dell'immagine I nella direzione θ.

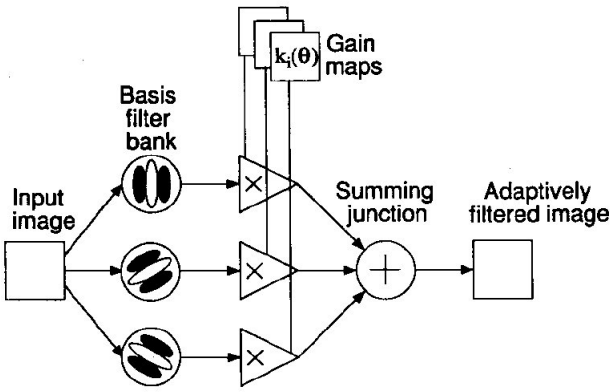

Figura 2.27. Diagramma a blocchi

La seguente implementazione reinterpreta l'algoritmo di Canny per l'estrazione dei contorni utilizzando i filtri ruotabili, migliorandone la velocità computazionale e la risposta. L'identificazione delle strisce avviene analizzando la derivata prima e la derivata seconda dell'immagine in scala di grigi. Come è possibile notare dalla figura (2.28), se si considera un'immagine in intensità di grigi dove il bianco corrisponde al valore uno e il nero al valore zero, ipotizzando di muoversi trasversalmente a una linea bianca in prossimità del centro linea, la derivata prima attraverserà lo zero, mentre la derivata seconda avrà un minimo. Naturalmente è necessario applicare un filtro di smoothing gaussiano che attenui le transizioni ed eviti discontinuità nella risposta.

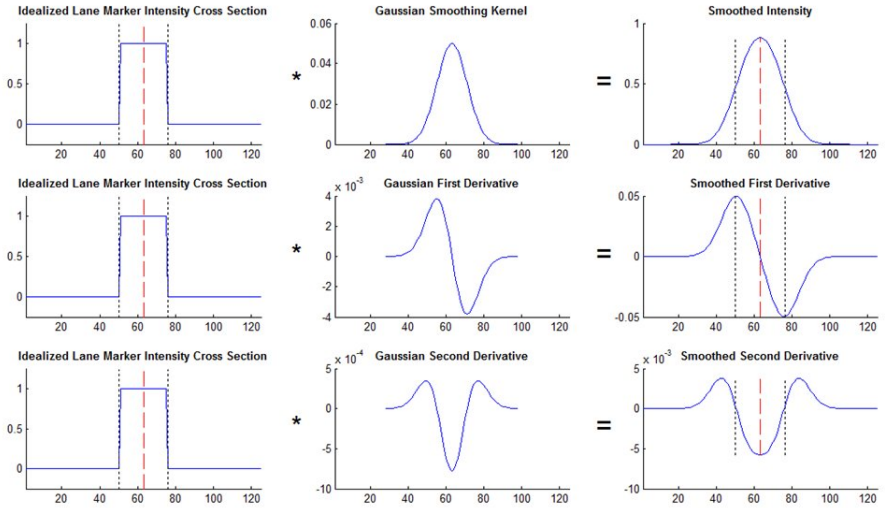

Figura 2.28. Filtraggio Gaussiano

Il metodo sviluppato esegue i seguenti passi:

- Costruzione delle basi dei filtri gaussiani di primo e secondo ordine G_x, G_y, G_{xx}, G_{yy} e G_{xy} in fase di inizializzazione(Offline).
- Acquisizione dell'immagine dalla telecamera
- Preprocessamento tramite inversione prospettica
- Convoluzione dell'immagine IPM con le basi dei filtri per ottenere le derivate dell'immagine di primo e secondo ordine.
- Binarizzazione dell'immagine tramite laplaciano con soglia secondo la formula:

$$\begin{aligned} L(x,y) &= \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \\ I_b &= L(x,y) < treshold * min(L(x,y)) \end{aligned} \tag{2.16}$$

- Utilizzando le proprietà dei filtri ruotabili si determina l'angolo di massima risposta della derivata seconda per ogni pixel secondo la seguente formula

$$\begin{aligned} G_2^\theta(x,y) &= G_{xx}cos(\theta)^2 + G_{yy}sin(\theta)^2 - 2G_{xy}cos(\theta)sin(\theta) \\ \theta_{max} &= tan^{-1}\left(\frac{G_{xx}-G_{yy}+A}{2G_{xy}}\right) \\ A &= \sqrt{G_{xx}^2 - 2G_{xx}G_{yy} + G_{yy}^2 + 4G_{xy}^2} \end{aligned} \tag{2.17}$$

- Si interpola lungo la direzione del gradiente e si individuano i pixel di zerocrossing della derivata seconda(centro linea).

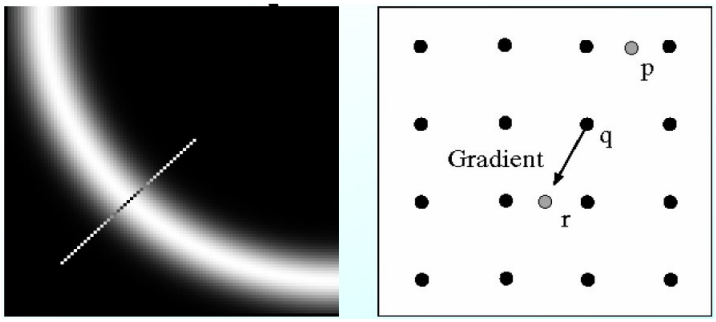

Figura 2.29. Direzione del gradiente

Per ogni punto bisogna quindi individuare la direzione del gradiente e confrontare il modulo del gradiente del punto in esame con quello dei vicini giacenti sulla stessa direzione. Nel caso generale, la direzione del gradiente può individuare dei vicini che non appartengono alla griglia dell'immagine.

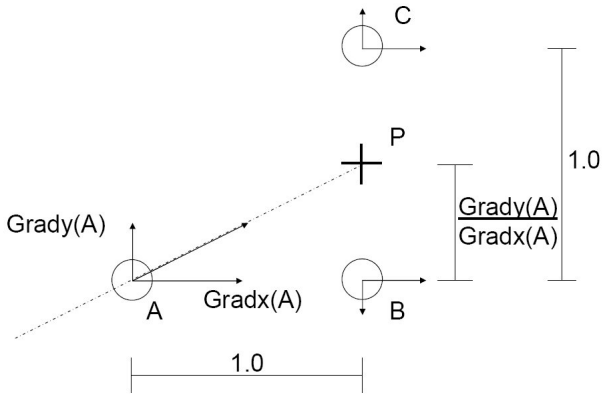

Figura 2.30. Diagramma interpolazione del gradiente

Per stimare il gradiente in P si suppone che l'andamento del gradiente tra pixel adiacenti possa essere interpolato linearmente: $Grad(P)d{*}Grad(B)+(1-d)*Grad(C)$ dove $d = Grady(A)/Gradx(A)$.

- Isolati i pixel appartenenti alla linea si interpolano per determinarne l'equazione matematica.

2.4 Line Fitting

Tipicamente si ha un'immagine suddivisa in un set di tokens - che possono essere pixels, punti isolati, insiemi di punti di contorno, etc. - e si vuole determinare se essi appartengo o no a una semplice famiglia. Per esempio, si voglio aggregare segmenti diversi poiché insieme formano un cerchio. La ricerca di tali gruppi è nota in letteratura come *fitting*.

Il fitting, in genere, utilizza un modello per ottenere una rappresentazione che evidenzi strutture dell'immagine di interesse. Esistono una grande varietà di possibili strategie di fitting. Essenzialmente il problema di fitting delle linee è un problema di determinazione di un buon modello. Per esempio, se si ha un insieme di punti si vuole determinare quale linea li interpola al meglio. Determinati i pixel dell'immagine che appartengono alla segnaletica orizzontale si determina la rappresentazione matematica delle linee della carreggiata tramite tali tecniche.

Il primo passo nel fitting delle linee è stabilire un modello probabilistico che indica come i nostri dati sono correlati a ognuna delle suddette linee; selezionato tale modello, è possibile stabilire quali punti possono essere assegnati a una particolare linea e quante linee bisogna considerare. In particolare è possibile utilizzare diversi modelli via via più complessi:

- lineare
- polinomiale 2,3 ordine
- lineare-parabolico
- clotoide
- Spline

2.4.1 Fitting Lineare

Il fitting lineare può essere eseguito direttamente utilizzando la trasformata di Hough che restituisce i parametri delle rette presenti nell'immagine come precedentemente descritto oppure tramite interpolazione ai minimi quadrati. L'utilizzo dei minimi quadrati assume che tutti i punti che appartengono a una particolare linea sono noti, e devono solamente essere individuati i parametri della linea. Possiamo rappresentare una linea come una collezione di punti nel piano cartesiano(x,y) dove $ax+by+c = 0$. Ogni linea può essere rappresentata in questo modo, ed è possibile pensare una linea come una tripletta di valori (a,b,c).

Il criterio di fitting ai minimi quadrati quindi determina la tripletta (a,b,c) in modo tale che, dati un insieme di punti, è possibile determinare la linea che minimizza la norma della distanza tra i punti e la linea.

2.4.2 Fitting Lineare-parabolico

L'idea di base di effettuare un fitting lineare nel campo vicino dell'immagine e un fitting quadratico nel campo lontano. Fissata quindi una soglia x_m che

separa i due campi, la curva che rappresenta la linea della carreggiata :

$$f(x) = \begin{cases} a + b(x - x_m) & x > x_m \\ a + b(x - x_m)x + c(x - x_m)^2 & x \leq x_m \end{cases} \tag{2.18}$$

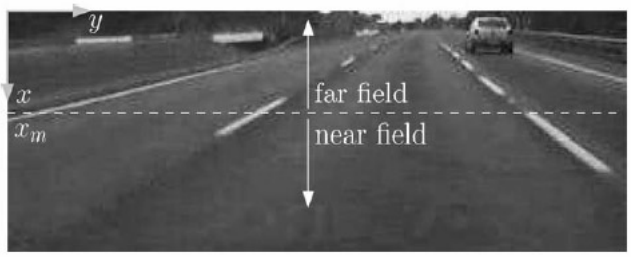

Figura 2.31. Separazione dell'immagine

Il primo passo è l'identificazione di due aree di interesse (LBROI) all'interno del frame corrente, ottenute solitamente dal frame precedente o da una fase di calibrazione iniziale eseguita all'avvio del sistema.

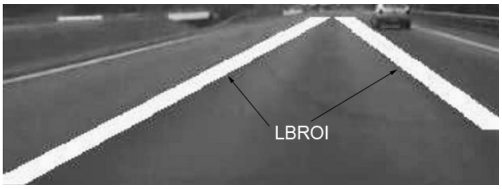

Figura 2.32. LBROI

I parametri a, b, c di (2.18) sono ottenuti tramite fitting nella regione di interesse minimizzando l'errore quadratico E pesato sulla magnitudine dei pixel:

$$E = \sum_{i=1}^{m} M_{n_i} \left[y_{n_i} - f(x_{n_i}) \right]^2 + \sum_{j=1}^{n} M_{f_j} \left[y_{f_j} - f(x_{f_j}) \right]^2 \tag{2.19}$$

dove (x_{n_i}, y_{n_i}), con $i = 1, \ldots, m$, indicano le m coordinate dei pixels diversi da zero nel campo vicino, ed M_{n_i} le rispettive magnitudini. Analogamente, (x_{f_j}, y_{f_j}) e M_{f_j}, con $j = 1, \ldots, n$, rappresentano le stesse caratteristiche per gli n pixels diversi da zero nel campo lontano.

L'errore E viene minimizzato risolvendo il seguente sistema lineare:

$$A^T W A c = A^T W b \tag{2.20}$$

dove

$$A = \begin{bmatrix} 1 & x_{n_1} - x_m & 0 \\ \vdots & \vdots & \vdots \\ 1 & x_{n_m} - x_m & 0 \\ 1 & x_{f_1} - x_m & (x_{f_1} - x_m)^2 \\ \vdots & \vdots & \vdots \\ 1 & x_{f_n} - x_m & (x_{f_n} - x_m)^2 \end{bmatrix} \ , \ W = \begin{bmatrix} M_{n_1} & & & & & \\ & \ddots & & & & \\ & & M_{n_m} & & & \\ & & & M_{f_1} & & \\ & & & & \ddots & \\ & & & & & M_{f_n} \end{bmatrix} \tag{2.21}$$

$$c = \begin{bmatrix} a, b, c^T \end{bmatrix} \ , \ b = [y_{n_1}, \ldots, y_{n_m}, y_{f_1}, \ldots, y_{f_n}] \tag{2.22}$$

La procedura è applicata indipendentemente per la linea destra e la linea sinistra della carreggiata.

2.4.3 Algoritmo RANSAC/K-means

RANSAC è una abbreviazione per "RANdom SAmple Consensus". Si tratta di un metodo iterativo per stimare i parametri di un modello matematico da una serie di dati osservati che contengono outliers. Il presupposto di base è che i dati sono costituiti da inliers, vale a dire punti che appartengono a un modello, e outliers, che sono i punti che non corrispondono al modello.

Gli outliers possono venire, ad esempio, da valori estremi di rumore, da misurazioni erronee o da ipotesi errate circa l'interpretazione dei dati. RANSAC presume inoltre che esiste comunque un un sottoinsieme di punti a partire dai quali è possibile stimare i parametri del modello che li ha generati. RANSAC raggiunge il suo obiettivo selezionando iterativamente un sottoinsieme dei dati originali, questi sono ipotetici inliers. Si costruisce un modello a partire dai punti ipotetici. Tutti gli altri dati sono confrontati con il modello ottenuto calcolandone la distanza. Utilizzando una soglia si classificano i punti come inliers se risultano abbastanza vicini al modello. La quantità di inliers classificati permette di stabilire la bontà del modello che, tuttavia, è stato stimato solo a partire dal set iniziale. Si procede quindi a ristimare un nuovo modello che comprenda tutti i nuovi inliers.

Questa procedura è ripetuta un determinato numero di volte: a ogni iterazione si scartano i modelli con i quali sono stati classificati troppo pochi punti come inliers o si crea un modello più raffinato. In quest'ultimo caso, si mantiene il modello solo se il suo errore è inferiore all'ultimo salvato. Nel nostro caso il modello considerato è una linea con equazione polinomiale del secondo ordine del tipo $y = ax^2 + bx + c$.

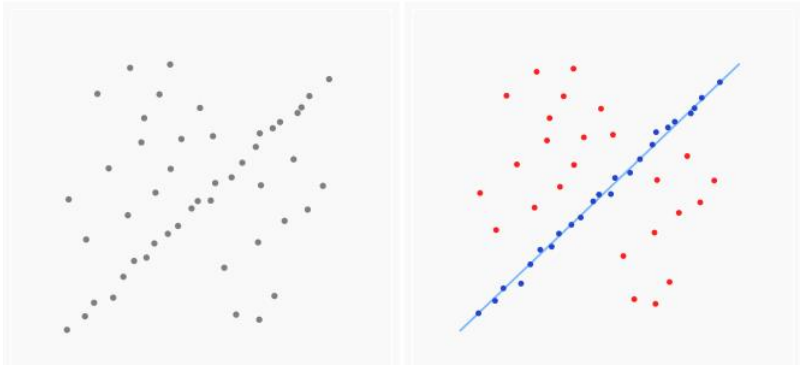

Figura 2.33. Dataset con outliers in cui interpolare una linea, in rosso gli outliers che non influenzano il risultato.

2.5 Algoritmi di tracking delle linee

Il tracking è definito come il problema di stabilire corrispondenze tra oggetti identificati in frame video successivi. Queste corrispondenze possono essere usate per riconoscere l'ambiente circostante o particolari comportamenti. Ci sono vari metodi di tracking che sono basati su coerenza di moto, apparenza e modelli di forma. Il tracking è un processo abbastanza complesso. Se un oggetto si muove all'interno della scena, cambia la sua posizione e l'orientazione rispetto alla camera, in frame multipli può sembrare differente. Lo stesso oggetto può non essere visibile in ogni frame, e ciò rende difficoltoso stabilire le corrispondenze.

Un altro problema che riguarda il tracking, in particolare in ambiente stradale, è dovuto alle occlusioni. Le occlusioni sono dovute a copertura parziale dell'oggetto da parte di altri oggetti, ad es. una autovettura sulle strisce, segnaletica verticale temporanea o fissa, o da caratteristiche ambientali come ombre, neve o foglie. Per

mantenere la corrispondenza è quindi necessario che almeno una piccola parte dell'oggetto sia visibile in frame successivi. Questo è possibile se la segmentazione è accurata e la risoluzione dell'immagine sufficiente. Se ciò non è verificato può accadere che l'inseguimento venga perso periodicamente; è quindi importante che l'algoritmo sia capace di resettarsi e riacquisire l'oggetto.

Gli algoritmi di tracking successivamente presentati tengono conto di tutte queste considerazioni e hanno l'obiettivo di inseguire la posizione delle linee di demarcazione della carreggiata tra frame successivi. Inoltre la conoscenza della posizione delle linee nel frame precedente permette di migliorare l'identificazione e il fitting della linea nel frame successivo. Gli algoritmi di tracking più utilizzati in letteratura e implementati per LDWS sono:

- Filtri ricorsivi su N frames
- Filtro di Kalman
- Selezione della regione di interesse dal frame precedente

2.5.1 Filtro ricorsivo su N frames

Il filtro più semplice che viene utilizzato nel tracking delle linee all'interno di un'immagine è un filtro ricorsivo su N frame precedenti. In pratica esegue una media della posizione della linea su una finestra scorrevole di N frame precedenti.

La posizione della linea nel frame successivo è ipotizzata non discostarsi molto da tale valore. In particolare se definiamo $\theta(k)$ vettore dei parametri della linea identificata all'istante k abbiamo:

$$z(k+1) = z(k) + \theta(k) - \theta(k-N),\ k \geq N+1 \tag{2.23}$$

con N calcolato sperimentalmente. La posizione utilizzata dal sistema LDWS sarà quindi quella descritta dal vettore di accumulazione filtrato z anzichè su θ

2.5.2 Filtro di Kalman

Quando si vogliono inseguire oggetti in movimento, è desiderabile la capacità di predire dove saranno nei frame futuri. Per fare

ciò è necessario utilizzare al massimo le informazioni preesistenti in modo da ridurre le ricerche nei frame successivi. Inoltre è fondamentale risolvere problemi di occlusione temporanea.

Il filtro di Kalman è l'algoritmo più utilizzato per il tracking di oggetti e forme in movimento all'interno di immagini. L'utilizzo di questo filtro per il tracking di oggetti presume che il loro moto sia abbastanza costante o lineare tra frame successivi. Se consideriamo quindi un modello dinamico lineare affetto da rumore di misura e di processo il filtro di Kalman è lo stimatore ottimo se il rumore è bianco, a media nulla e Gaussiano. Il filtro di Kalman, comunque, fornisce spesso una buona stima anche se il rumore non è Gaussiano. Man mano che vengono effettuate nuove misure la stima della posizione e dell'errore viene migliorata ad ogni iterazione. Una caratteristica importante è che il rumore è modellato come la posizione stessa. Questo permette di poter "dimenticare" tutte le posizioni precedenti.

Infine, bisogna notare che il filtro di Kalman lavora mediando i processi che danno risultati erronei in presenza di outliers. Una situazione comune in molte applicazioni di moto. Quindi, è necessario testare ogni predizione per determinare quando queste sono troppo lontane dalla realtà. Se dovesse accadere, l'oggetto in questione probabilmente è parzialmente o completamente nascosto. Una semplice soluzione è assumere che l'oggetto continui nello stesso moto e aspettare che esso riemerga da dietro un'altro oggetto.

Risulta inoltre prudente mantenere memoria di un diverso numero di possibilità per qualche istante in modo tale da poter facilmente riagganciare l'oggetto nel caso si ripresenti in una posizione inaspettata. Consideriamo il seguente sistema dinamico lineare tempo invariante

$$\begin{aligned} x_k &= Ax_{k-1} + w_{k-1} \\ z_k &= Hx_k + v_k \end{aligned} \tag{2.24}$$

con x_k lo stato del sistema w e v rumori a media nulla e con matrice di covarianza Q and R rispettivamente, A la matrice di transizione dello stato e H la matrice di uscita. Ad ogni istante di campionamento k la stima dello stato è

$$\hat{x}_k = \bar{x}_k + K_k(z_k - H\bar{x}_k) \tag{2.25}$$

con

$$\bar{x} = A\hat{x}_{k-1} \tag{2.26}$$

$$\bar{P}_k = A\hat{P}_{k-1}A^T + Q \tag{2.27}$$

$$\hat{P}_k = (I - K_k H)\,\bar{P}_k \tag{2.28}$$

$$K_k = \bar{P}_k H^T \left(H\bar{P}_k H^T + R\right)^{-1} \tag{2.29}$$

la precedente iterazione è inizializzata con $\hat{x}_0 = 0$ e la matrice di covarianza $\hat{P}_0$ è opportunamente scelta.

2.5.3 Selezione Regione di interesse

Un semplice algoritmo di tracking è l'utilizzo di regioni di interesse (ROI) selezionate dal frame precedente. Identificate le linee all'interno del frame corrente, si mantiene memoria della loro posizione per il frame successivo.

Nel frame successivo la ricerca delle linee avverrà esclusivamente in un'area nell'intorno della posizione memorizzata invece che sull'intero frame. In particolare, la linea ottenuta con il fitting è allargata di un certo numero di pixels a destra e a sinistra, diventando il nuovo spazio di ricerca per il frame successivo. In caso di occlusione si mantiene la memoria della ROI anche per più frame finché non viene nuovamente riagganciata la linea, altrimenti si effettua una nuova ricerca sull'intero frame. L'algoritmo risulta molto semplice ma ha il vantaggio di migliorare le performance dell'identificazione delle linee riducendo il campo di ricerca.

3

Time to Lane Crossing

Il tempo di invasione di corsia attigua t_{tlc} (Time to lane crossing) è definito come l'intervallo di tempo a disposizione del guidatore prima dell'attraversamento della linea di carreggiata. Il calcolo real-time del t_{tlc} non è semplice a causa di limitazioni sulla disponibilità dei dati del veicolo, sulla predizione della sua traiettoria e sulla geometria della strada.

3.1 Calcolo trigonometrico del t_{tlc}

Il calcolo del t_{tlc} descritto di seguito permette di determinare il tempo preciso di superamento della linea da utilizzare come riferimento per gli altri metodi di calcolo. In generale abbiamo:

$$t_{tlc_{tri}} = DLC/V \quad per \;\; V > 0 \tag{3.1}$$

con DLC [m] la distanza dal punto di superamento della linea lungo la traiettoria del veicolo e V [m/s] la velocità della vettura lungo il suo asse longitudinale (vedi figura (2.3)). Il valore DLC è calcolato trigonometricamente utilizzando la regola del coseno. Normalmente, la macchina è guidata lungo percorsi curvi alternando svolte a destra e a sinistra. Il raggio di curvatura della traiettoria R_v calcolato come:

$$R_v = V/r \tag{3.2}$$

con r [rad/s] lo yaw rate. In caso di tratto rettilineo, come mostrato in figura (3.1), abbiamo:

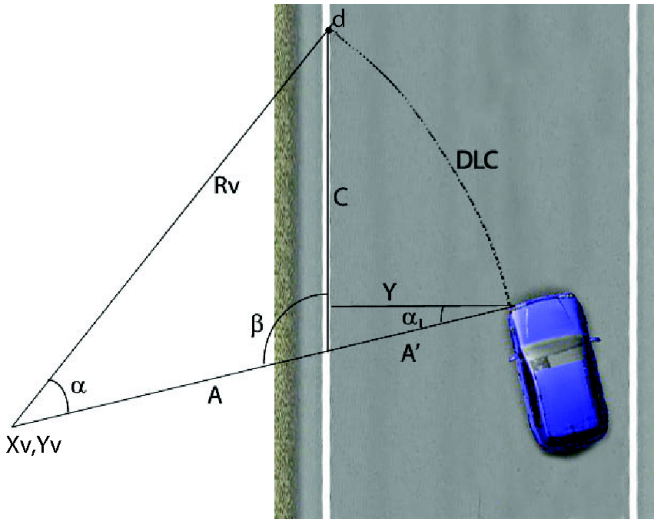

Figura 3.1. Determinazione della lunghezza dell'arco DLC su strade dritte

$$DLC = \alpha \cdot R_v \tag{3.3}$$

in cui α calcolato con la regola del coseno:

$$\alpha = arccos\left(\frac{A^2 + R_v^2 - C^2}{2 \cdot A \cdot C}\right) \tag{3.4}$$

$$A = R_v - A' \tag{3.5}$$

$$A' = y/cos(\alpha_1) \tag{3.6}$$

con y la distanza normale tra la ruota frontale e il bordo della linea e α_1 l'angolo tra la linea perpendicolare alla strada e la linea dalla ruota frontale al punto centrale della curva del veicolo(X_v, Y_v).

$$C = \left(2 \cdot A \cdot cos(\beta) + \sqrt{(2 \cdot A \cdot cos(\beta))^2 - 4 \cdot (A^2 - R_v^2)}\right)/2 \tag{3.7}$$

La figura (3.2) mostra come calcolare il t_{tlc} se la vettura sta percorrendo un tratto di strada curva.

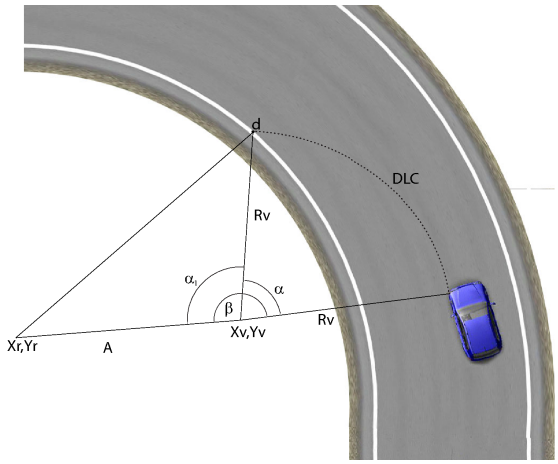

Figura 3.2. Determinazione della lunghezza dell'arco DLC su strada curva

In questo caso il DLC è dato da

$$DLC = \alpha \cdot R_v \tag{3.8}$$

con α

$$\alpha = \beta - \alpha_1 \tag{3.9}$$

e β l'angolo differenza tra la linea dal punto centrale della curva del veicolo(X_v, Y_v) al centro della curva della strada(X_r, Y_r) e la linea dal centro della curva del veicolo(X_v,Y_v) alla ruota frontale sinistra (se il veicolo gira verso l'interno della curva). Infine,

$$\alpha_1 = arccos\left(\frac{A^2 + R_v^2 - R_r^2}{2 \cdot A \cdot R_v}\right) \tag{3.10}$$

con A la distanza tra il centro della curva della strada e il centro della curva del veicolo e R_r il raggio di curvatura del segmento di strada.

3.1.1 Approssimazione del t_{tlc} utilizzando la derivata prima della distanza laterale

Assumendo una velocità laterale costante di avvicinamento alla linea t_{tlc} può essere approssimato con:

$$t_{tlc} = \frac{y}{V_y} \tag{3.11}$$

dove y è la distanza del centro di massa dal bordo della linea e V_y la velocità laterale. Se $V_y = 0$, il t_{tlc_1} è infinito. In particolare, tale velocità è calcolata con

$$V_y = V \sin(\phi) \tag{3.12}$$

dove ϕ è l'angolo di yaw della vettura rispetto alla carreggiata e V la velocità longitudinale della vettura.

Per maggior correttezza nel calcolo del t_{tlc} è necessario riferire la posizione laterale y a una delle ruote anteriori anziché rispetto al centro di massa. La correzione è effettuata aggiungendo un offset al valore di y:

$$y_o = y + offset \tag{3.13}$$

dove l'offset è calcolato con

$$offset = l_f \sin\phi + \frac{D_f}{2} \cos\phi \tag{3.14}$$

l_f è la distanza dell'asse anteriore dal centro di massa e D_f la distanza tra le due ruote anteriori (si veda figura (3.3)).

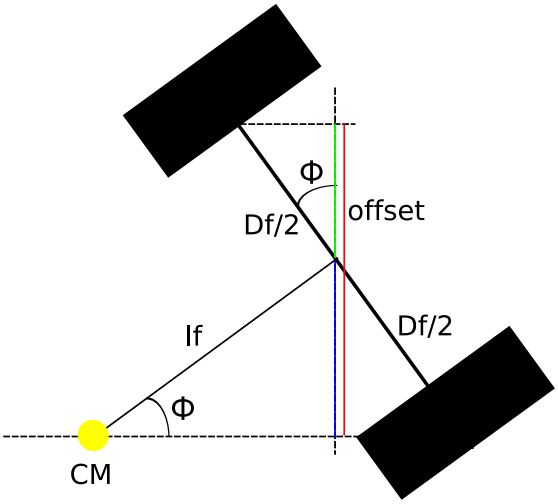

Figura 3.3. Offset posizione laterale

3.2 Posizione e angolo di yaw del veicolo

Il t_{tlc} approssimato con la derivata prima della distanza laterale è calcolato con la posizione e l'angolo di yaw del veicolo oltre che con la velocità longitudinale della vettura. L'angolo di yaw e la posizione possono essere determinati direttamente tramite il sistema di visione o stimati utilizzando gli algoritmi di data fusion. Gli algoritmi di data fusion integrano i dati dei sensori a bordo vettura con quelli del sistema di visione. Le due sottosezioni seguenti descrivono i due metodi di calcolo.

3.2.1 Posizione e angolo di yaw del veicolo da solo sensore di Visione

Un sistema di visone per l'identificazione della carreggiata richiede informazioni quali la segnaletica orizzontale, la curvatura della strada e la posizione del veicolo a ogni istante di campionamento. Montando una telecamera sul veicolo le immagini acquisite forniscono una visione prospettica della strada. La vista prospettica catturata distorce la forma reale della strada in larghezza, altezza e profondità. Questi parametri rappresentano le componenti x, y e z. Per eliminare tali distorsioni è necessaria una fase di preelaborazione nota come inversione prospettica (IPM) che permette di avere una visione "dall'alto" della strada. IPM è una trasformazione geometrica che rimappa una immagine prospettica 2D di un oggetto 3D in una nuova immagine planare a 2D.

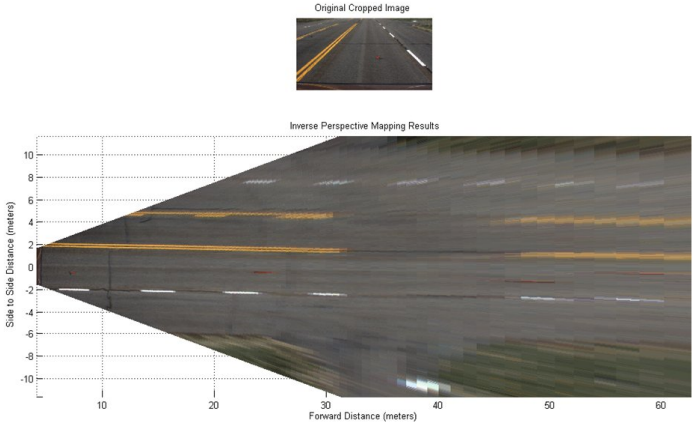

Figura 3.4. Immagine originale e corrispondente inversione prospettica

Matematicamente essa può essere descritta come una proiezione da uno spazio euclideo 3D, $W = (x, y, z)$ (Spazio Reale) a uno spazio planare a 2D, $I = (u, v)$ (Spazio Immagine).

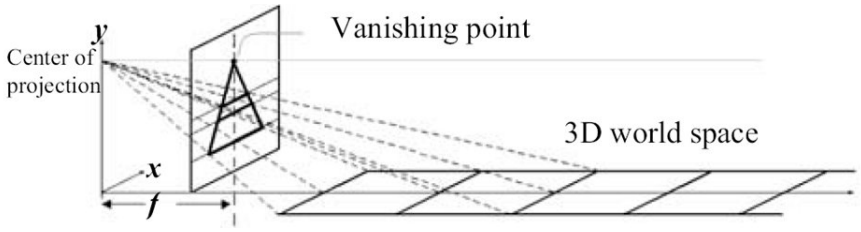

Figura 3.5. Corrispondenza piano immagine mondo reale, f:lunghezza focale videocamera

La nuova vista dall'alto permette di ottenere informazioni su posizione (y) e inclinazione delle linee di delimitazione della carreggiata (ϕ - yaw angle) rispetto al punto di osservazione solidale con il veicolo. L'asse ottico della telecamera coincide con l'asse longitudinale del veicolo e tutte le misure laterali derivate dal sistema di visione sono riferite rispetto a tale asse. Nell'inversione prospettica ogni pixel della telecamera è ri-mappato su una griglia di dimensione 20×32 m con una risoluzione di 6 cm. In assenza di informazioni per determinati punti della griglia i pixel vengono interpolati. A prescindere dall'interpolazione effettuata durante l'inversione prospettica, la precisione delle misure alla massima distanza (32 m) dipende dalla risoluzione della telecamera in accordo con la tabella seguente.

Risoluzione immagine [pixel]	**Precisione IPM [mm]**
320×240	110.3
640×480	52.6
1024×768	35.7

- Considerando il fatto che le strisce della segnaletica orizzontale sulle strade europee hanno larghezze che variano da 150 a 200 mm non è conveniente scendere al disotto dei $320x240$ pixel di risoluzione. A questa risoluzione ogni linea ha una larghezza nel caso peggiore di 1 pixel, un limite che può generare notevoli errori nell'algoritmo di identificazione.

- Il sistema di visone integra un algoritmo di tracking delle linee, basato sul filtro di Kalman, che permette di inseguirne la posizione della segnaletica orizzontale tra frame successivi. Nonostante questo, le misure di y e ϕ presentano disturbi e rumore che si ripercuotono sul calcolo del t_{tlc}.
- Inoltre il sistema **non è robusto per oscuramenti superiori a 30 frame consecutivi** (1 *sec*). Si veda a tal proposito il Cap. 8.

Rumore additivo sull'immagine

Un sistema di visione per sistemi automotive può essere soggetto a disturbi sulla qualità dell'immagine acquisita, questo è il caso di interferenze elettromagnetiche o rumore di acquisizione. Per simulare questa eventualità sono stati effettuati alcuni test degradando la qualità dell'immagine in uscita dalla telecamera.

- Un primo tipo di test è stato effettuato aggiungendo del rumore Salt and Pepper. Questo tipo di rumore genera in modo random sull'intera immagine pixel bianchi e neri. L'algoritmo è **risultato robusto fino al limite del** 3% **di pixel corrotti**.
- Bisogna evidenziare il fatto che il maggior disturbo è provocato dalla presenza di pixel bianchi nell'immagine originale che in seguito alla trasformazione prospettica vengono distorti in tratti di linea che disturbano l'algoritmo di identificazione (figura (3.6)).

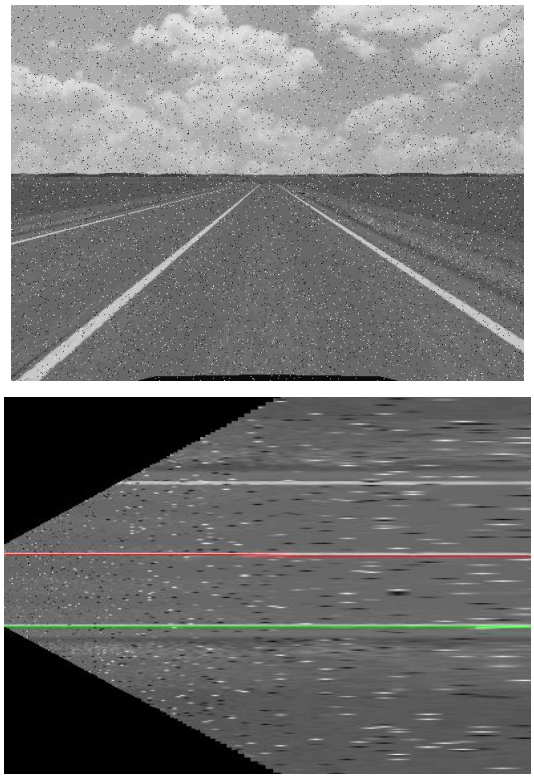

Figura 3.6. Rumore Salt and Pepper 3% dei pixel

- Il secondo test di rumore sull'immagine è stato effettuato sommando del rumore bianco Gaussiano a media nulla e varianza prefissata. L'algoritmo **è robusto se la varianza del rumore bianco è inferiore a** 0.03 (figura (3.7)).

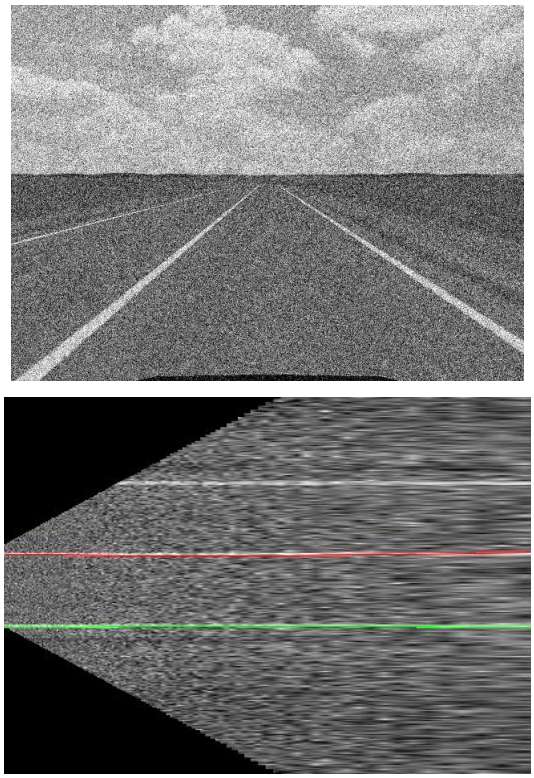

Figura 3.7. Rumore Gaussiano media nulla varianza 0.03

3.2.2 Calcolo del t_{tlc} da solo sensore di Visione

Se si ipotizza di non utilizzare algoritmi di data-fusion, un semplice metodo per il calcolo del t_{tlc} è il seguente. Innanzi tutto si richiede la conoscenza della velocità longitudinale V, che si può ottenere tramite la presa di diagnosi OBD della centralina motore o interfacciandosi alla rete CAN della vettura. La posizione laterale y e l'angolo ϕ della vettura sono ottenute dal sistema di visione ad ogni frame acquisito come illustrato nel paragrafo precedente.

Il calcolo del tempo di invasione, quindi, si effettua direttamente utilizzando le formule (3.11) e (3.12) da cui si ottiene

$$t_{tlc} = \frac{y}{V \sin(\phi)} \tag{3.15}$$

3.2.3 Posizione e angolo di yaw del veicolo con Data Fusion

L'utilizzo di algoritmi di fusione dei dati permette di superare le limitazioni illustrate per il solo sistema di visione. Integrando i dati provenienti dal modulo di visione con i sensori a bordo vettura è possibile ottenere una misura robusta della posizione e dello yaw.

I sistemi basati esclusivamente su sensore video, non permettono di ottenere un valore corretto di t_{tlc} in caso di assenza dei dati provenienti dalla telecamera. L'ambiente stradale in condizioni reali mette a dura prova l'acquisizione di immagini da telecamera. Sono possibili brevi accecamenti dovuti a bagliori o controluce, fino ad arrivare a qualche secondo di completa occlusione dovuta a grossi automezzi che precedono il veicolo o a brevi tratti di strada priva di segnaletica orizzontale.

In ognuno di questi casi affidarsi alla sola telecamera può essere una scelta penalizzante: il sistema di avviso di superamento della carreggiata non avendo a disposizione dati affidabili su cui eseguire il calcolo del t_{tlc} si disattiva interrompendo il supporto di assistenza al guidatore.

Le tecniche di data fusion mirano ad evitare questi momenti di black-out integrando i dati video con i dati provenienti da altri sensori a bordo vettura secondo lo schema di figura (3.8).

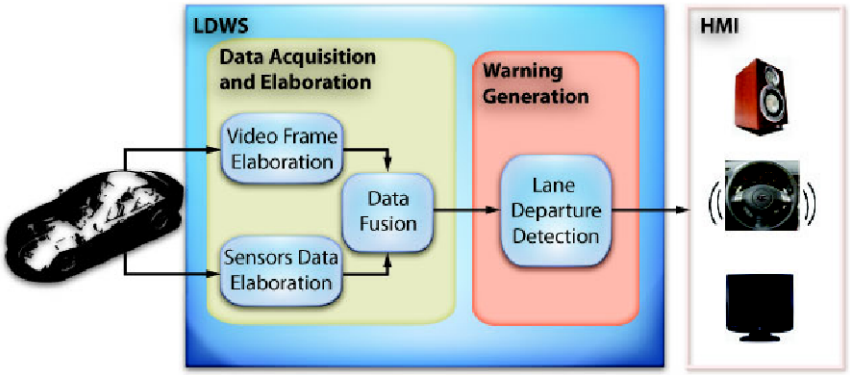

Figura 3.8. Schema algoritmo fusione dei dati

Gli algoritmi di fusione dei dati utilizzano una descrizione matematica dello stato della vettura e stimano la posizione del

veicolo e il suo comportamento cinematico/dinamico utilizzando i dati provenienti sia dal sensore di visione che dai sensori a bordo vettura.

Nel caso in cui i dati provenienti dal sistema di visione non fossero momentaneamente disponibili, la stima della traiettoria e della posizione del veicolo viene effettuata utilizzando i dati dei sensori rimasti sulla base del modello di moto considerato.

Sensori a bordo vettura

In tutte le vetture moderne i dispositivi elettronici di bordo sono collegati tra di loro tramite una rete dati digitale che permette il controllo real-time distribuito e la condivisione dei dati fra tutti i sottosistemi. Il protocollo universalmente adattato, riconosciuto come il più affidabile e sicuro, è il protocollo CAN (controller area network) sviluppato e introdotto da Bosch. Alla rete CAN della vettura oltre a tutte le centraline elettroniche sono connessi anche tutti i sensori. La connessione al bus CAN permette quindi di leggere tutti i dati inviati in broadcast dai sensori e utilizzarli per la propria applicazione.

Gli algoritmi di fusione dei dati oggetto di studio utilizzano sensori già disponibili a bordo vettura. Il numero e il tipo di sensori installati dipende dai dispositivi e dalle funzionalità di cui è dotata la vettura. Gli algoritmi in esame utilizzano i sensori di velocità delle ruote, l'angolo di sterzo, e lo yaw rate. Se si considerano solo i sensori di velocità delle quattro ruote è sufficiente un'autovettura con sistema ABS, mentre se si rendono necessari i dati di angolo di sterzo e yaw-rate è necessario che sia installato anche l'ESP. Riassumendo, i sensori utilizzabili e alcune loro caratteristiche sono:

- **Sensore angolo di sterzo**. Misura l'esatta posizione del volante e solitamente comunica all'ESP la traiettoria desiderata dal conducente. In genere hanno un'accuratezza pari a 2 $grad$ con un fondo scala di ± 720 $grad$
- **Sensori velocità delle ruote**. Misurano la velocità istantanea di tutte e quatto le ruote e permettono alla centralina

ABS di stabilirne l'eventuale blocco. Hanno un'accuratezza di 0.0625 Km/h e un fondo scala superiore a 500 Km/h

- **Sensore di velocità di imbardata**. I sensori d'imbardata misurano velocità di rotazione assolute intorno ai rispettivi assi. Il Programma Elettronico di Stabilità ESP necessita di imbardate intorno all'asse superiore, per poter rilevare differenze di movimento dell'autoveicolo rispetto alla direzione di guida desiderata. L'accuratezza è solitamente di 0.08 $grad/s$ con fondo scala di ± 180 $grad/s$
- **Sensori di accelerazione**. Misurano le accelerazioni lungo i tre assi del veicolo e sono solitamente utilizzati da tutti i sistemi di controllo della stabilità. Possono rilevare accelerazioni dell'ordine dei 0.02 m/s^2 con valori massimi di ± 40 m/s^2 quasi 4 volte l'accelerazione di gravità.

La scelta dei vari sensori da utilizzare dipende dal modello di auto con cui è progettato l'algoritmo di data-fusion.

3.3 Obiettivi

Il presente documento vuole investigare i vantaggi nell'uso di algoritmi di fusione dei dati applicati ai sistemi LDWS. In particolare si studieranno strategie diverse al fine di ottenere un sistema più preciso, affidabile e robusto.

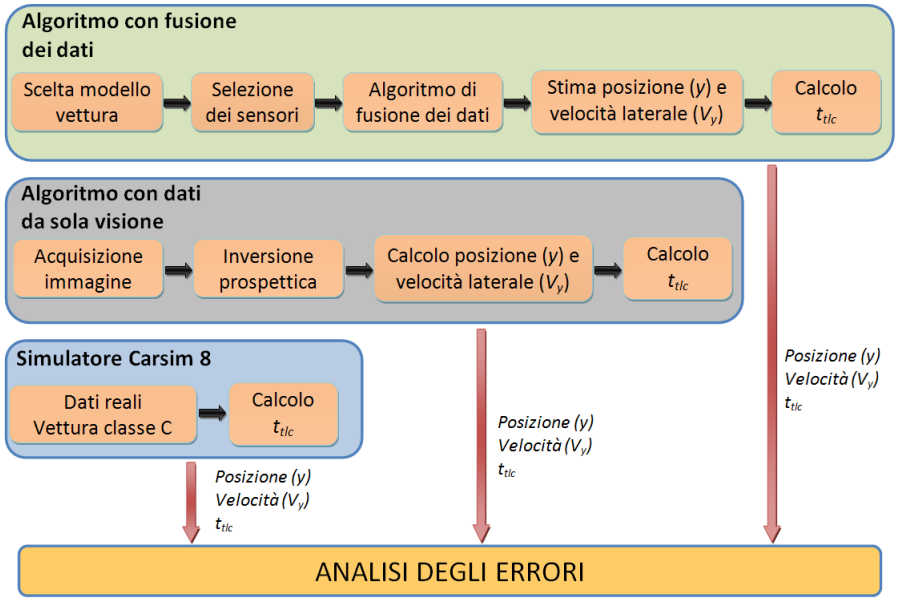

Figura 3.9. Schema analisi algoritmi

Come descritto schematicamente in figura (3.9) l'analisi riguarderà, in primo luogo, la valutazione dei vari modelli matematici necessari per descrive il moto del veicolo. In base a tale scelta saranno selezionati il tipo e il numero di sensori da utilizzare. La stima dello stato del veicolo necessaria per il calcolo del t_{tlc} sarà ottenuta utilizzando i dati dei sensori a bordo vettura e gli algoritmi di fusione considerati.

Le differenti soluzioni saranno progettate in ambiente Matlab/Simulink e testate con il simulatore Carsim. Un'analisi approfondita sarà eseguita sull'accuratezza delle stime ottenibili per la posizione entro la corsia, per le velocità laterale e longitudinale e

per il tempo t_{tlc}, allegando a ogni simulazione gli errori percentuali commessi sul calcolo di queste variabili.

In parallelo verrà analizzato anche un algoritmo per il calcolo di t_{tlc} basato su dati provenienti dal solo sistema di visione, senza richiedere l'integrazione di dati provenienti da altri sensori. I due differenti approcci permetteranno un confronto delle prestazioni e della robustezza, mettendo in evidenza gli eventuali vantaggi ottenibili utilizzando l'approccio basato sulla fusione dei dati. I valori veri per posizione, velocità laterale e longitudinale e t_{tlc} lungo i percorsi test considerati e necessari al calcolo degli errori percentuali sono ottenuti direttamente dal simulatore Carsim 8.

Per capire come l'errore percentuale commesso nel calcolo del t_{tlc} possa essere un utile cifra di merito per valutare le varie architetture di data-fusion si consideri la sua definizione

$$\frac{\Delta t_{tlc}}{t_{tlc}} := \frac{t_{tlc} - t^*_{tlc}}{t^*_{tlc}} \tag{3.16}$$

dove t_{tlc} è la stima calcolata mentre t^*_{tlc} il valore vero. Mettendo in relazione la precisione relativa di t_{tlc} con le precisioni relative di y e V_y si ha

$$\frac{\Delta t_{tlc}}{t^*_{tlc}} = \frac{1}{t^*_{tlc}} \frac{\partial t_{tlc}}{\partial y} \Delta y + \frac{1}{t^*_{tlc}} \frac{\partial t_{tlc}}{\partial V_y} \Delta V_y \tag{3.17}$$

$$= \frac{V^*_y}{y^*} \frac{1}{V^*_y} \Delta y - \frac{V^*_y}{y^*} \frac{y^*}{(V^*_y)^2} \Delta V_y \tag{3.18}$$

Infine, si arriva

$$\frac{\Delta t_{tlc}}{t^*_{tlc}} = \frac{\Delta y}{y^*} - \frac{\Delta V_y}{V^*_y} \tag{3.19}$$

dove ovviamente

$$\frac{\Delta y}{y^*} := \frac{y - y^*}{y^*}, \quad \frac{\Delta V_y}{V^*_y} := \frac{V_y - V^*_y}{V^*_y} \tag{3.20}$$

Risulta utile allora porre in evidenza che:

- l'errore relativo sul t_{tlc} è la somma (a parte il segno) degli errore relativi sulla misura/stima della posizione y e della velocità laterale V_y.

- se le misure/stime di y e V_y fossero entrambe sovrastimate o sottostimate, l'errore relativo sul t_{tlc} sarebbe minore dei singoli errori relativi su y e V_y
- Se gli errori relativi su y e V_y avessero ordini di grandezza diversi l'errore relativo su t_{tlc} avrebbe un ordine di grandezza pari al più grande dei due. Lo sforzo di miglioramento potrebbe allora concentrarsi solo sul miglioramento della stima di quella variabile con il più alto errore relativo
- L'errore su y è essenzialmente legato alla precisione e accuratezza del sistema di visione mentre il calcolo di V_y dipende sia dal sistema di visione che dai sensori di velocità (almendo quello di velocità longitudinale) presenti a bordo.

Segue una breve descrizione degli algoritmi utilizzati e dei risultati delle simulazioni.

4

Modelli matematici del moto dell'autovettura

Gli algoritmi di fusione dei dati necessitano di una rappresentazione matematica del moto dell'autovettura per poter correttamente stimarne lo stato istante per istante. Nel caso in esame, lo stato del veicolo deve contenere almeno la posizione all'interno della carreggiata (coordinata x e y), l'orientazione del veicolo rispetto ai bordi della strada (angolo di Yaw). Si assume che la velocità longitudinale sia facilmente disponibile e che non occorra stimarla. Utilizzando i valori dello stato stimato è possibile quindi calcolare, come precedentemente descritto, il valore di t_{tlc} istante per istante. I modelli del veicolo di seguito considerati sono:

- Cinematico sterzante
- Cinematico Differenziale
- Cinematico yaw rate
- Dinamico

4.1 Modello 1: Cinematico sterzante

Il modello è basato su una descrizione a quattro stati, che comprendono le coordinate Cartesiane (x, y) del centro di massa del veicolo e l'angolo di orientazione rispetto alla strada ϕ come in figura (4.1)

L'angolo di sterzo è indicato con δ e B è l'interasse. Lo stato aggiuntivo $R(k)$ rappresenta il raggio delle ruote ed è utilizzato per compensare gli errori dovuti a scivolamenti e/o compressioni degli pneumatici sull'asfalto.

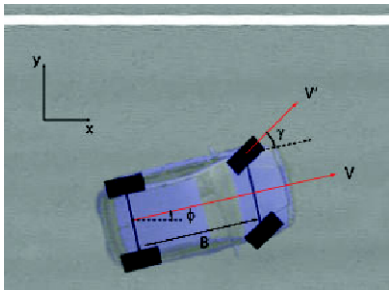

Figura 4.1. Modello Cinematico Sterzante

$$\begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \\ R(k) \end{bmatrix} = \begin{bmatrix} x(k-1) + \omega(k)R(k-1)\Delta T\cos(\phi(k-1)) \\ y(k-1) + \omega(k)R(k-1)\Delta T\sin(\phi(k-1)) \\ \phi(k-1) + \frac{\omega(k)R(k-1)\Delta T}{B}\tan(\delta(k)) \\ R(k-1) \end{bmatrix} + \begin{bmatrix} \epsilon_x(k) \\ \epsilon_y(k) \\ \epsilon_\phi(k) \\ \epsilon_R(k) \end{bmatrix} \tag{4.1}$$

Il vettore additivo $[\epsilon_x(k),\, \epsilon_y(k),\, \epsilon_\phi(k),\, \epsilon_R(k)]^T$, le cui componenti sono processi stocastici a media nulla e varianza fissata, riflette le incertezze del modello di stato e gli errori quando il veicolo è in condizioni stazionarie. Il modello ha come ingressi:

- $\omega(k)$ la media della velocità di rotazione delle ruote destra e sinistra del veicolo misurate sull'asse posteriore. La velocità longitudinale è facilmente calcolabile come $V(k) = R(k)\omega(k)$ e quella laterale $V_y(k) = V(k)\sin(\phi(k))$.
- $\delta(k)$ l'angolo di sterzo.

4.2 Modello 2: Cinematico Differenziale

Il modello cinematico differenziale utilizza le velocità di rotazione di entrambe le ruote destra e sinistra per calcolare lo yaw rate della vettura. In questo caso non e' necessario utilizzare l'angolo di sterzo.

L'angolo di yaw è determinato dividendo la somma delle velocità delle ruote posteriori (non-sterzanti) per l'interasse B. Anche in questo caso è stato aggiunto il raggio delle ruote R come il quarto stato per compensare gli errori dovuti a scivolamenti e/o compressioni degli pneumatici sull'asfalto.

$$\begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \\ R(k) \end{bmatrix} = \begin{bmatrix} x(k-1) + (\frac{\omega_R(k)+\omega_L(k)}{2})R(k-1)\Delta T\cos(\phi(k-1)) \\ y(k-1) + (\frac{\omega_R(k)+\omega_L(k)}{2})R(k-1)\Delta T\sin(\phi(k-1)) \\ \phi(k-1) + (\frac{\omega_R(k)+\omega_L(k)}{B})R(k-1)\Delta T \\ R(k-1) \end{bmatrix} + \begin{bmatrix} \epsilon_x(k) \\ \epsilon_y(k) \\ \epsilon_\phi(k) \\ \epsilon_R(k) \end{bmatrix} \tag{4.2}$$

Il modello ha come ingressi:

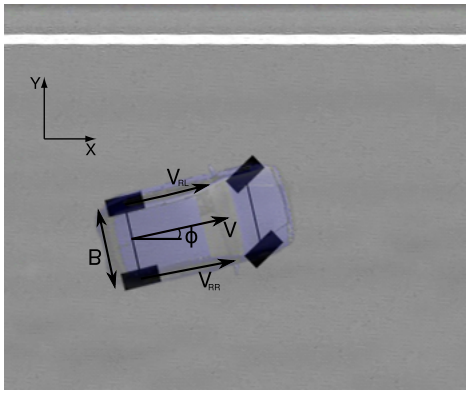

Figura 4.2. Modello Cinematico Differenziale

- $\omega_L(k)$ velocità di rotazione della ruota sinistra del veicolo misurata sull'asse posteriore.
- $\omega_R(k)$ velocità di rotazione della ruota destra del veicolo misurata sull'asse posteriore.
- La velocita' longitudinale $V(k) = R(k)\frac{\omega_L(k)+\omega_R(k)}{2}$ e' calcolata come velocita' media di rotazione delle ruote destra e sinistra per il raggio delle ruote e $V_y(k) = V(k)\sin(\phi(k))$.

4.3 Modello 3: Cinematico yaw rate

Il modello cinematico yaw rate è stato derivato dal modello cinematico differenziale presentato nel paragrafo precedente ipotizzando di conoscere direttamente la velocità longitudinale e lo yaw rate del veicolo. La descrizione di stato del modello è la seguente:

$$\begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \\ V(k) \end{bmatrix} = \begin{bmatrix} x(k-1) + V(k-1)\cos(\phi(k-1))\Delta T \\ y(k-1) + V(k-1)\sin(\phi(k-1))\Delta T \\ \phi(k-1) + \dot{\phi}(k-1)\Delta T \\ V(k-1) \end{bmatrix} + \begin{bmatrix} \epsilon_x(k) \\ \epsilon_y(k) \\ \epsilon_\phi(k) \\ \epsilon_V(k) \end{bmatrix} \tag{4.3}$$

Nel vettore di stato del sistema è stata inserita la velocità del veicolo V. Il modello ha come ingresso $\dot{\phi}$ yaw rate del veicolo.

4.4 Modello 4: Dinamico

Il modello dinamico è il modello più complesso preso in considerazione. E' un modello a sei stati con:

- $x(k)$ posizione longitudinale del veicolo
- $V(k)$ velocità longitudinale
- $y(k)$ posizione laterale del veicolo rispetto ai limiti della carreggiata
- $v_y(k)$ velocità lungo l'asse trasversale della vettura
- $\phi(k)$ angolo di yaw del veicolo rispetto alla carreggiata
- $\dot{\phi}(k)$ yaw rate del veicolo

ed è descritto dalle seguenti equazioni:

$$\begin{bmatrix} x(k) \\ V(k) \\ y(k) \\ v_y(k) \\ \phi(k) \\ \dot{\phi}(k) \end{bmatrix} = \begin{bmatrix} x(k-1) + V(k)\Delta T \cos(\phi(k-1)) \\ V(k-1) + \left(\frac{F_x}{m} + \dot{\phi}(k-1)v_y(k-1)\right)\Delta T \\ y(k-1) + V(k)\Delta T \sin(\phi(k-1)) \\ v_y(k-1) + \left(\frac{1}{mV}(l_r C_{ar} - l_f C_{af})\,\dot{\phi} - \frac{1}{mV}(C_{ar}+C_{af})\,v_y + \frac{1}{m}C_{af}\delta - \dot{\phi}V\right)\Delta T \\ \phi(k-1) + \dot{\phi}\Delta T \\ \dot{\phi}(k-1) + \frac{1}{I_z V}\left(-l_f^2 C_{af} - l_r^2 C_{ar}\right) - \frac{1}{I_z V}\left(l_f C_{af} - l_r C_{ar}\right)v_y + \frac{1}{I_z}l_f C_{af}\delta \end{bmatrix} + \begin{bmatrix} \epsilon_x(k) \\ \epsilon_V(k) \\ \epsilon_y(k) \\ \epsilon_{v_y}(k) \\ \epsilon_\phi(k) \\ \epsilon_{\dot{\phi}}(k) \end{bmatrix} \tag{4.4}$$

Il modello ha come ingressi:

- δ l'angolo di sterzo
- F_x la forza longitudinale applicata alle ruote

Come è possibile notare il modello dinamico è caratterizzato da un gran numero di parametri della vettura:

- m massa del veicolo
- l_r e l_f la distanza del centro di massa dall'asse posteriore e anteriore
- C_{ar} e C_{af} i cornering stiffness delle ruote posteriori e anteriori.
- I_z il momento di inerzia del veicolo

Il sistema è fortemente non lineare, i precedenti parametri variano da vettura a vettura e alcuni di essi variano durante la marcia in dipendenza dello stato corrente. Per esempio, i cornering stiffness variano in dipendenza della velocità, del tipo di strada, del carico su ogni singola ruota. Anche la semplice la massa non è mai costante, basti pensare al numero di passeggeri o ai bagagli che influenzano anche il momento di inerzia del veicolo.

Un modello di questo tipo necessita di un'identificazione specifica per ogni veicolo al fine di generare adeguate lookup-table di parametri. Tale modello non è proponibile nel caso si stia progettando un sistema il più possibile generale e facile da installare. Nelle simulazioni, per l'utilizzo di tale modello si sono scelti dei valori costanti per i parametri molto vicini al valore reale in condizioni stazionarie, con tutte le limitazioni del caso.

4.5 Dati Vettura

Le simulazioni di guida sono state effettuate con il software di simulazione Carsim® utilizzando un modello di autovettura appartenete alla classe C (compatte i.e. Golf, Punto, Fiesta etc.) con le seguenti caratteristiche fisiche:

- $m = 1274\ [Kg]$ (Massa del veicolo)
- $I_z = 1523\ [Kg \cdot m^2]$ (Inerzia del veicolo intorno all'asse z)
- $l_f = 1.016\ [m]$ (Distanza dell'asse anteriore dal centro di massa)
- $l_r = 1.562\ [m]$ (Distanza dell'asse posteriore dal centro di massa)
- $B = 2.578\ [m]$ (Interasse)
- $R = 0.316\ [m]$ (Raggio delle ruote)
- $D_f = 1.539\ [m]$ (Carreggiata anteriore)
- $D_r = 1.539\ [m]$ (Carreggiata posteriore)

5

Algoritmi di Data Fusion

In questo capitolo sarà presentata una descrizione dettagliata degli algoritmi di data fusion utilizzati. Gli algoritmi presentati sono stati poi sviluppati in Matlab/Simulink e adattati alla specifica applicazione.

I metodi più utilizzati in questo settore si basano sul filtro di Kalman, modificato per l'utilizzo con i sistemi non-lineari. In particolare, la scelta è ricaduta sul filtro di Kalman Esteso e sul filtro di Kalman Unscented che risultano i più promettenti per questa applicazione.

I filtro di Kalman, nella sua versione tempo-invariante, è un osservatore asintotico che permette di stimare lo stato del sistema in modo ottimale rispetto alle proprietà statistiche (stazionarie) dei rumori presenti nel sistema, soprattutto relativo agli errori di misura dei sensori, riducendo al minimo la varianza dell'errore di stima.

5.1 Filtro di Kalman

Il più semplice algoritmo utilizzato per il data-fusion è il filtro di Kalman. Consideriamo il seguente sistema dinamico lineare tempo-invariante

$$\begin{aligned} x_k &= Ax_{k-1} + w_{k-1} \\ z_k &= Hx_k + v_k \end{aligned} \tag{5.1}$$

con x_k lo stato del sistema w e v rumori a media nulla e con matrice di covarianza Q and R rispettivamente, A la matrice di transizione dello stato e H la matrice di uscita. Ad ogni istante di campionamento k la stima dello stato è data da

$$\hat{x}_k = \bar{x}_k + K_k(z_k - H\bar{x}_k) \tag{5.2}$$

con

$$\bar{x} = A\hat{x}_{k-1} \tag{5.3}$$
$$\bar{P}_k = A\hat{P}_{k-1}A^T + Q \tag{5.4}$$
$$\hat{P}_k = (I - K_k H)\,\bar{P}_k \tag{5.5}$$
$$K_k = \bar{P}_k H^T \left(H\bar{P}_k H^T + R\right)^{-1} \tag{5.6}$$

La precedente iterazione è inizializzata con $\hat{x}_0 = 0$ e con una matrice di covarianza $\hat{P}_0$ opportunamente scelta.

5.2 Filtro Esteso di Kalman

Il filtro di Kalman grazie alla sua semplicità, robustezza, tracciabilità e ottimalità è uno dei metodi più diffusi per l'inseguimento e la stima. La sua applicazione ai sistemi non lineari, però, risulta notevolmente difficoltosa. In questi casi l'approccio più comune è l'uso del Filtro Esteso di Kalman (EKF) che linearizza il modello non lineare lungo una traiettoria in modo tale da poter applicare localmente il filtro di Kalman tradizionale a ogni istante di campionamento. Consideriamo il seguente sistema non-lineare a tempo-discreto:

$$x_k = f_{k-1}(x_{k-1}) + w_{k-1}, \tag{5.7}$$
$$z_k = h_k(x_k) + \upsilon_k \tag{5.8}$$

dovex_k rappresenta il vettore di stato del sistema, z_k il vettore delle misure, w_k il rumore di processo dovuto ai disturbi e agli errori di modello e υ_k il rumore di misura. Si assume che i vettori di rumore w_k e υ_k sono a media nulla, incorrelati con matrici di covarianza $Q_k = Q_k^T > 0$ e $R_k = R_k^T > 0$ rispettivamente, cioè

$$w_k \sim \aleph(0, Q_k), \;\; \upsilon_k \sim \aleph(0, R_k)$$

I rumori di misura e di processo sono assunti inoltre incorrelati con lo stato iniziale x_0. Quindi il problema di stima può essere così descritto: date le osservazioni $Z_k := \{z_0, z_1, \ldots, z_k\}$ si calcoli una stima $\hat{x}_k$ di x_k che minimizzi un appropriato indice. Considerando uno stimatore ad errore quadratico medio, il valore stimato è il valore del vettore stocastico che minimizza la seguente media condizionata (funzione di costo):

$$J[\hat{x}_k] = E[(x_k - \hat{x}_k)^2 | Z_k] \tag{5.9}$$

A ogni istante di tempo k, il filtro EKF esegue due operazioni: predizione e correzione. Nella prima, data la stima corrente dello stato del processo $\hat{x}_{k-1}$ e della matrice di covarianza P_{k-1} sulla base delle equazioni linearizzate dello stato (5.7)

$$\Phi_k = \left.\frac{\partial f_k}{\partial x}\right|_{x=\hat{x}_{k-1}} \tag{5.10}$$

L'aggiornamento della matrice di covarianza e la predizione dello stato $\hat{x}_{k|k-1}$ è effettuata come segue:

$$\begin{aligned} P_{k|k-1} &= \Phi_k P_{k-1} \Phi_k^T + Q_{k-1}, \\ \hat{x}_{k|k-1} &= f_k(\hat{x}_{k-1}) \end{aligned} \tag{5.11}$$

Quindi, date le misure correnti z_k e linearizzando le equazioni di uscita (5.8) secondo:

$$H_k = \left. \frac{\partial h_k}{\partial x} \right|_{x=\hat{x}_{k|k-1}} \tag{5.12}$$

il guadagno dell'osservatore di kalman è calcolato come

$$K_k = P_{k|k-1} H_k^T (R_k + H_k P_{k|k-1} H_k^T)^{-1} \tag{5.13}$$

Infine, sono aggiornati le predizioni dello stato e della matrice di covarianza con

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k(z_k - h_k(\hat{x}_{k|k-1})), \tag{5.14}$$

$$P_k = (I - K_k H_k) P_{k|k-1} \tag{5.15}$$

e la procedura è ripetuta.

5.3 Filtro di Kalman Unscented

E' ben noto che il filtro di Kalman esteso può dar luogo a basse prestazioni di stima quando il modello dell'impianto è fortemente non-lineare, poiché il suo funzionamento è basato su un modello linearizzato della descrizione dello spazio di stato. Tale inconveniente può essere evitato utilizzando un ulteriore ampliamento del filtro di Kalman: il filtro di Kalman Unscented (UKF).

L'UKF è basato su un'idea differente: la media e la matrice di covarianza sono aggiornate utilizzando una tecnica deterministica di campionamento, la *trasformata unscented*, con la quale vengono selezionati un appropriato sottoinsieme minimo di punti (*punti sigma*). Quindi tutti i punti sigma sono propagati utilizzando la funzione di transizione di stato $f_k(\cdot)$ e la funzione di osservazione $h_k(\cdot)$, con le quali vengono ottenute la media e la matrice di covarianza della stima dello stato. In particolare, i punti sigma sono selezionati come segue:

$$\begin{aligned} (X_{k-1})^{(0)} &= \hat{x}_{k-1} \\ (X_{k-1})^{(m)} &= \hat{x}_{k-1} + col_m \left(\sqrt{(n+\lambda) P_{k-1}} \right), \\ &\qquad m = 1, \ldots, n \\ (X_{k-1})^{(m)} &= \hat{x}_{k-1} + col_{m-n} \left(\sqrt{(n+\lambda) P_{k-1}} \right), \\ &\qquad m = n+1, \ldots, 2n \end{aligned} \tag{5.16}$$

dove $n = dim(x_k)$, e $\lambda = \alpha^2(n + \chi) - n$, con α e χ fattori di scala. Essenzialmente l'algoritmo UKF consiste in due fasi. Nella prima fase vengono elaborate le predizioni dei punti sigma dello stato e della matrice di covarianza dell'errore con

$$\begin{aligned} \hat{x}_{k|k-1} &= \sum_{m=0}^{2M} \omega^{(m)} X_k^{(m)} \\ P_{k|k-1} &= \sum_{m=0}^{2M} \Omega^{(m)} \Big(X_k^{(m)} - \hat{x}_{k|k-1} \Big) \Big(X_k^{(m)} - \hat{x}_{k|k-1} \Big)^T + Q_{k-1} \end{aligned} \tag{5.17}$$

dove $X_k^{(m)} = f_k \left(X_{k-1}^{(m)} \right)$. Quindi viene eseguita una fase di correzione usando le seguenti equazioni:

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k \left(Z_k - \hat{z}_k^{(m)} \right) \tag{5.18}$$

$$P_k = P_{k|k-1} + K_k S_k K_k^T \tag{5.19}$$

dove

$$Z_k^{(m)} = h_{k-1}(X_{k-1}^{(m)}) \tag{5.20}$$

$$\hat{z}_k = \sum_{m=0}^{2M} \omega^{(m)} Z_k^{(m)} \tag{5.21}$$

$$S_k = \sum_{m=0}^{2M} \Omega^{(m)} (Z_k^{(m)} - \hat{z}_k)(Z_k^{(m)} - \hat{z}_k)^T + R_k \tag{5.22}$$

e

$$K_k = \sum_{m=0}^{2M} \Omega^{(m)} (X_k^{(m)} - \hat{x}_{k|k-1})(Z_k^{(m)} - \hat{z}_k)^T S_k^{-1} \tag{5.23}$$

con $\omega^{(m)}$ and $\Omega^{(m)}$, pesi fissi scelti come segue

$$\begin{aligned} \omega^{(0)} &= \tfrac{\lambda}{(n+\lambda)}, \\ \omega^{(m)} &= \tfrac{1}{2(L+\lambda)}, \ m = 1, \ldots, 2n, \\ \Omega^{(0)} &= \tfrac{\lambda}{(n+\lambda)} + (1 - \alpha^2 + \beta), \\ \Omega^{(m)} &= \tfrac{1}{2(n+\lambda)}, \ m = 1, \ldots, 2n, \end{aligned}$$

dove β è un parametro usato per incorporare ogni conoscenza a priori riguardo la distribuzione dello stato x_k.

5.4 Carico computazionale Algoritmi di Data Fusion

I due algoritmi implementati sono stati progettati in ambiente Matlab/Simulink. Il software di simulazione mette a disposizione uno strumento di valutazione del carico computazionale (*Profiler*). Il *Profiler* permette di valutare il tempo di CPU speso durante l'intera simulazione all'interno di ogni blocco in cui è stato diviso l'algoritmo restituendo un report dettagliato. Si è scelto di esaminare solamente le performance dei blocchi di Data Fusion senza considerare il carico computazionale degli algoritmi di visione, ottenendo i seguenti risultati:

Total recorded time	40.01 s
Number of Block Methods	435
Number of Internal Methods	7
Number of Nonvirtual Subsystem Methods	35
Clock precision	0.01 s
Clock Speed	2133 Mhz

Tabella 5.1. Parametri di Simulazione

Algoritmo	Tempo totale [s]	Tempo Totale [%]	Numero Chiamate	Tempo singola esecuzione [s]
EKF	2,55	6,40	2383	0,001070
UKF	7,61	19,00	2383	0,003193

Tabella 5.2. Tempo di CPU

6

Manovre di test

Per testare gli algoritmi in condizioni reali è necessario conoscere istante per istante la posizione della vettura all'interno della carreggiata. Il simulatore software utilizzato è Carsim 7 prodotto da Mechanical Simulation. Carsim è un pacchetto software per la simulazione dinamica di veicoli. Permette l'animazione e l'analisi del comportamento di un veicolo sottoposto a determinati ingressi di controllo. I modelli matematici sono completamente non lineari; ogni vettura è caratterizzata da più di centocinquanta equazioni differenziali che permettono di ottenere dati molto simili alla realtà.

Le manovre di seguito descritte sono state effettuate su tracciato stradale rettilineo di $1200m$. Ogni carreggiata è larga 4 m, la linea centrale ha coordinata $y = 0$, la carreggiata destra avrà quindi valori di ordinata negativi,quella di sinistra positivi.

6.1 Primo Percorso Test: Percorso su rettilineo a velocità costante

Il veicolo esegue un percorso di guida ripetuto secondo la figura (6.1). Prima viene eseguito un cambio di corsia, in seguito due movimenti di avvicinamento: uno alla linea centrale e l'altro alla linea destra della carreggiata. La velocità di percorrenza è costante e vale 50 Km/h.

Le accelerazioni subite dal veicolo lungo l'asse x e y sono visibili di seguito (figura (6.3)).

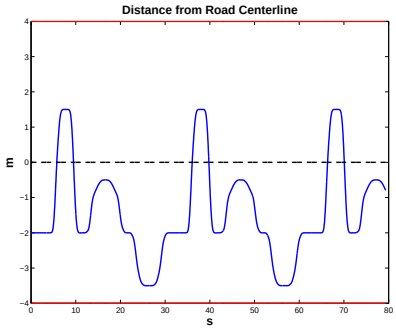

Figura 6.1. Posizione vettura Carsim

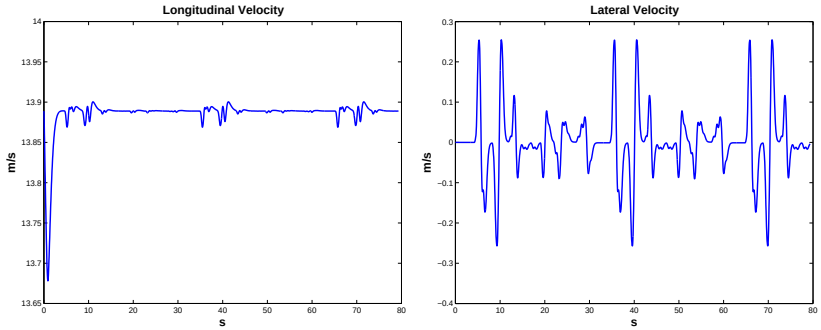

Figura 6.2. Velocità vettura Carsim

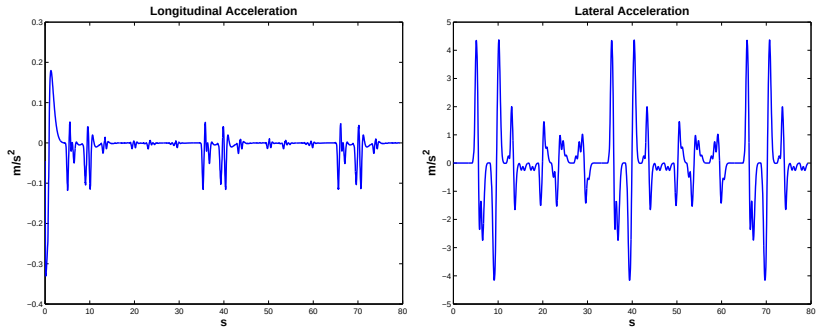

Figura 6.3. Accelerazioni vettura Carsim

6.2 Secondo Percorso Test: Percorso su rettilineo a velocità variabile

La vettura esegue un percorso di guida secondo la figura (6.4). Il profilo di velocità simula un tragitto autostradale: la vettura percorre i primi 30 sec di simulazione e i due primi cambi di corsia a 100 Km/h, in seguito accelera fino a 120 Km/h eseguendo una manovra di avvicinamento alla linea centrale per poi muoversi verso la linea destra della carreggiata. Segue poi una decelerazione fino a 90 Km/h durante un ulteriore cambio di corsia, per poi riportarsi alla velocità finale di 110 Km/h (6.5).

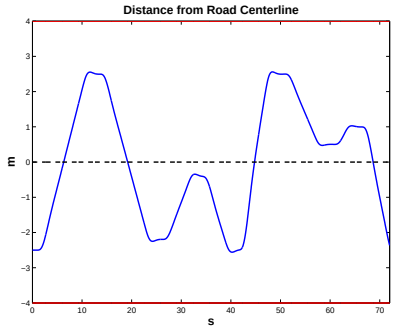

Figura 6.4. Posizione vettura Carsim

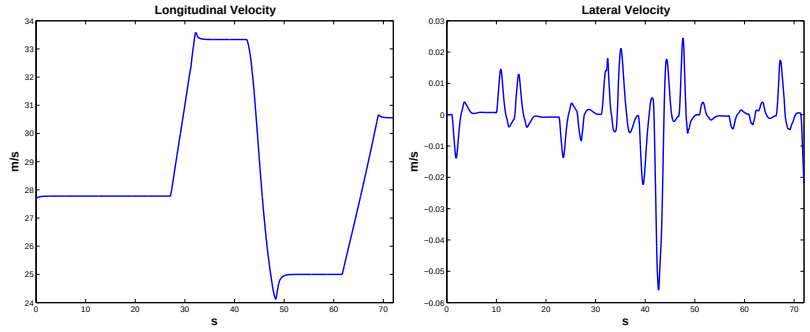

Figura 6.5. Velocità vettura Carsim

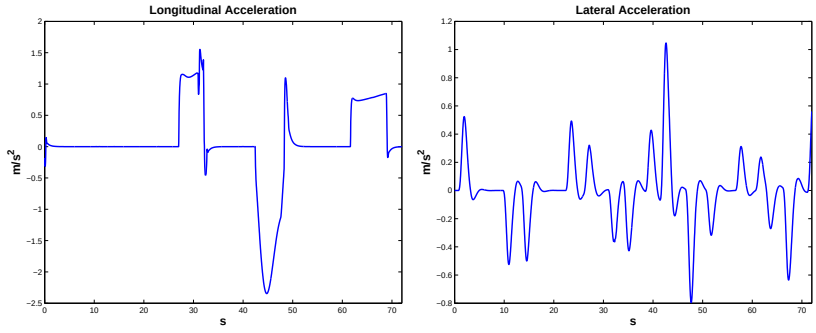

Figura 6.6. Accelerazioni vettura Carsim

7

Risultati

Al fine di analizzare il comportamento degli algoritmi di data fusion in condizioni simulative il più possibile vicine alla realtà ogni combinazione filtro/modello-vettura è stata testata utilizzando su tutti e quattro i profili test descritti nella sezione 7.

I dati reali utilizzati per i confronti e il calcolo degli errori sono stati ottenuti tramite il software di simulazione Carsim che permette di generare l'animazione 3D della vista della carreggiata acquisita dalla telecamera virtuale a bordo della macchina simulata in Carsim. Tale vista è elaborata dal sistema di visione implementato in Matlab/Simulink per determinare la posizione laterale e l'angolo di Yaw (ϕ) del veicolo rispetto alla strada.

Di seguito verranno illustrati tutti i risultati ottenuti. Il capitolo è diviso in quattro sezioni, una per ogni modello di vettura considerato, numerate in ordine crescente (modello 1, modello 2, etc.). Ogni sezione è a sua volta suddivisa in altrettante due parti una per ogni percorso di test effettuato sul simulatore. Sui due percorsi descritti è stato testato in primo luogo il Filtro di Kalman Esteso e in seguito il Filtro di Kalman Unscented, effettuando analisi su velocità, posizione, e t_{tlc}.

In particolare, per ognuna delle tre variabili verrà analizzato il grafico della sequenza degli errori relativi percentuali lungo i passi di simulazione, definito come:

$$X_{err_i}\% = \frac{X_i - X_i^*}{X_i^*} \cdot 100, \quad \forall i = 1 \dots n \tag{7.1}$$

dove X_i^* rappresenta il valore reale della variabile in esame (Posizione Laterale, Velocità Laterale o t_{tlc}), X_i la sua stima ed n il numero totale di campioni della simulazione.

7.1 Modello 1: Cinematico sterzante

7.1.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

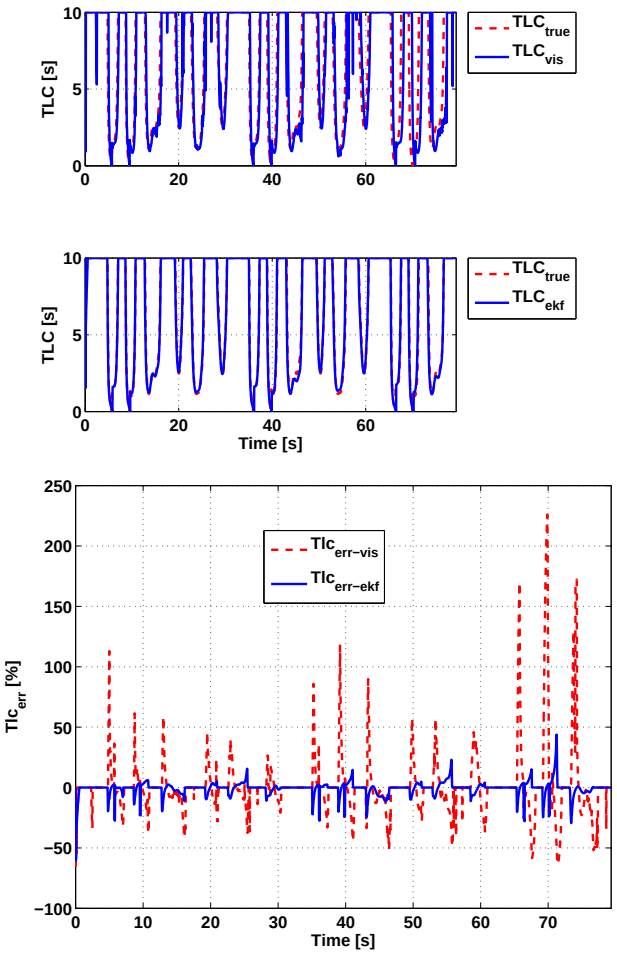

Figura 7.1. Tempo di invasione t_{tlc} EKF

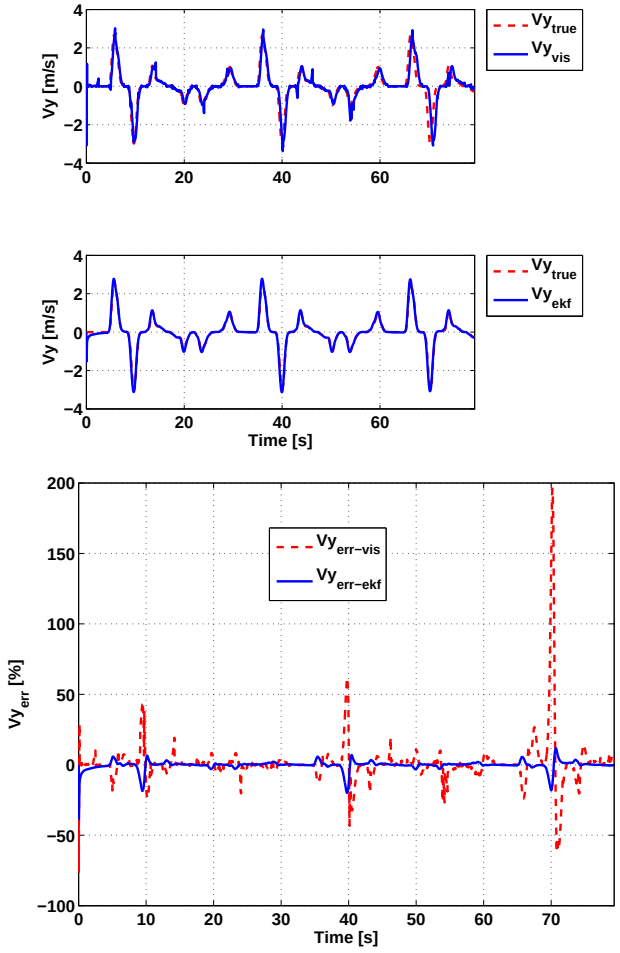

Figura 7.2. Velocità laterale EKF

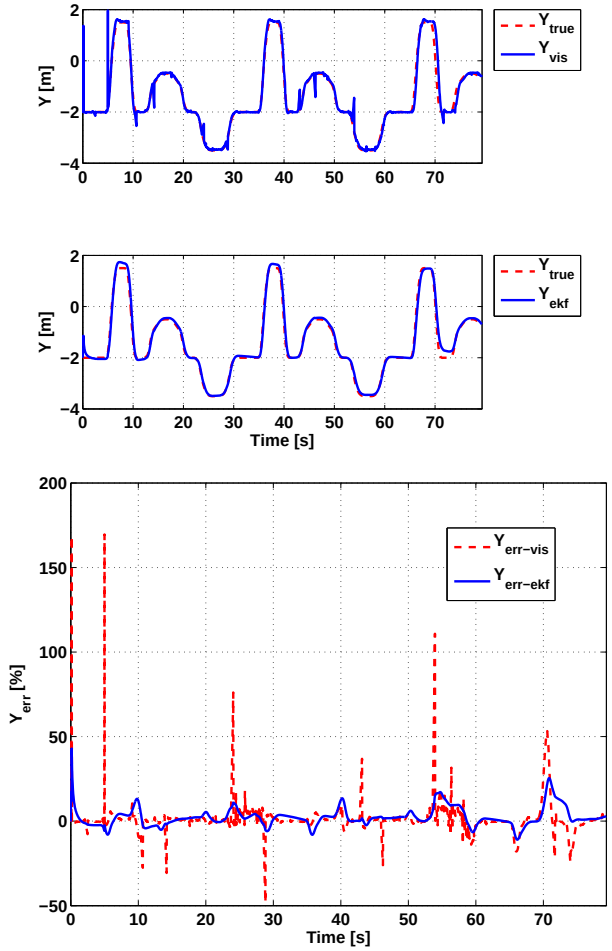

Figura 7.3. Posizione laterale EKF

Il t_{tlc} calcolato con filtro di Kalman Esteso e modello cinematico sterzante è visibile in figura (7.1). La simulazione effettuata a velocità costante di $50Km/h$ ha una durata di 80 secondi.

Il picco iniziale è da imputare al reset del filtro. Una volta a regime l'errore sul t_{tlc} si mantiene al di sotto del 25%, valore ottenuto in corrispondenza dei picchi causati dai cambi di corsia. Durante la marcia all'interno della carreggiata il t_{tlc} calcolato con il filtro di Kalman Esteso ha errore nullo. Il t_{tlc} calcolato da visione invece presenta picchi anche in corrispondenza di questi brevi intervalli.

Il miglior contributo alla precisione nel calcolo del t_{tlc} è dato dalla velocità laterale (figura (7.2)), per la quale l'errore di stima negli intervalli in questione

è sempre al di sotto del 3% e comunque non supera mai il 15% in prossimità dei picchi di errore. La stima della posizione, invece, presenta un errore leggermente più alto che si attesta intorno a ±10% come visibile in (figura (7.3).

Simulazioni con filtro UKF

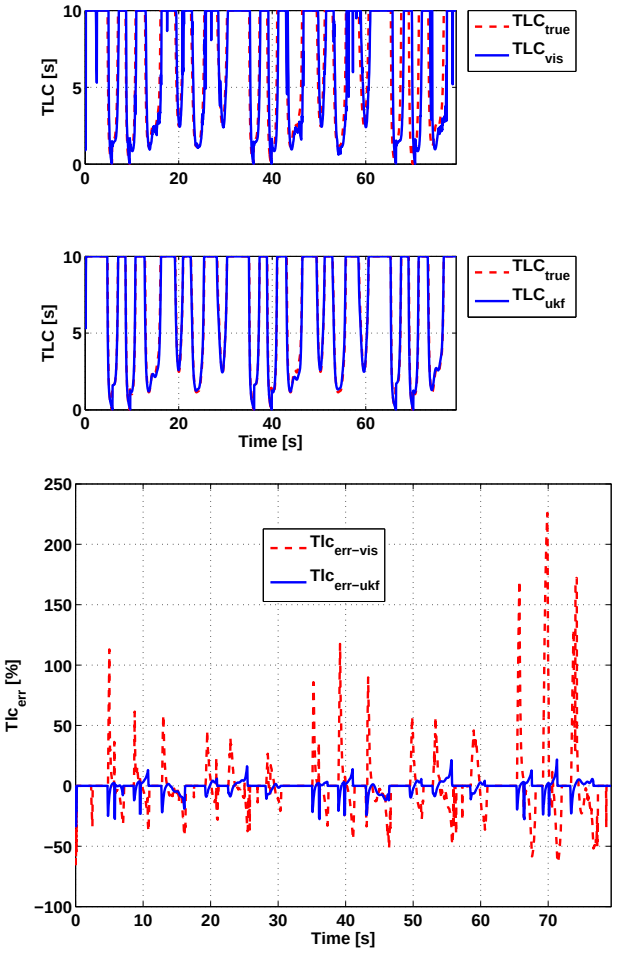

Figura 7.4. Tempo di invasione t_{tlc} UKF

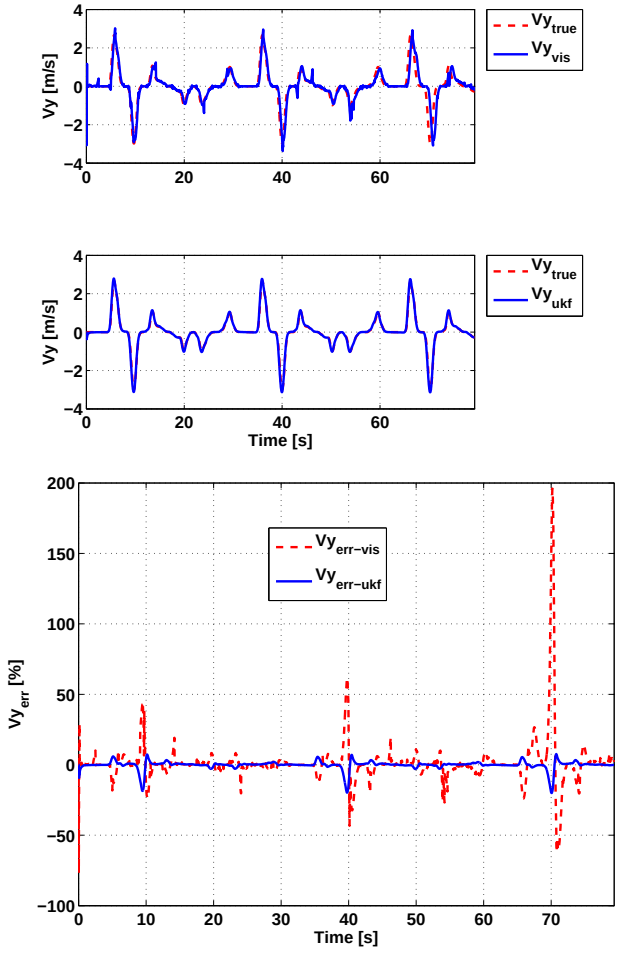

Figura 7.5. Velocità laterale UKF

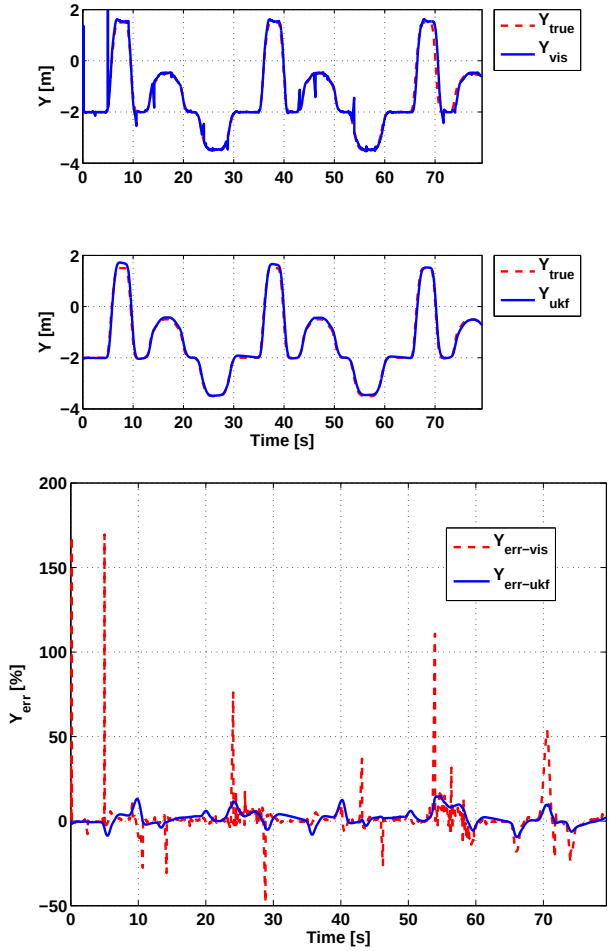

Figura 7.6. Posizione laterale UKF

Utilizzando il filtro di Kalman Unscented, il t_{tlc} stimato ha l'andamento mostrato in figura (7.4). Come avvenuto per la simulazione con cambi di corsia a velocità costante (sezione 7.1.2), anche in questo caso il comportamento del filtro risulta comparabile a quello del filtro di Kalman Esteso. Le osservazioni fatte per le figure (7.1-7.3) valgono anche per le figure (7.4-7.6).

Confronto tra EKF e UKF

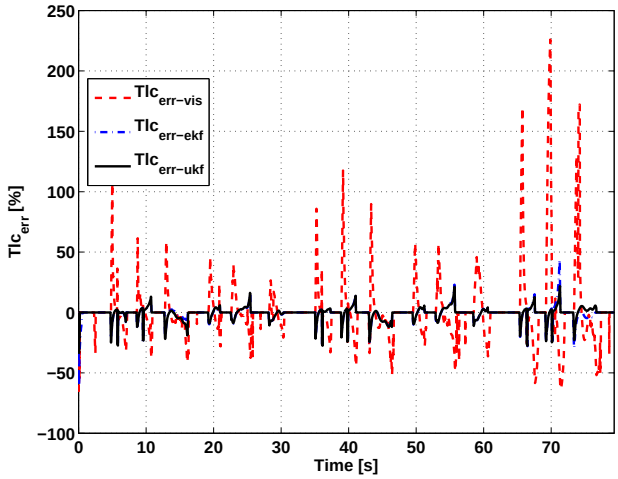

Figura 7.7. Errore sul t_{tlc}

La figura (7.7) mostra il confronto degli errori commessi nel calcolo del t_{tlc}. I filtri di Kalman Unscendent ed Esteso hanno un comportamento quasi del tutto identico. Permane un errore di offset sulla traiettoria XY visibile in figura (7.8) da imputare alla stima non corretta della coordinata X. Questo naturalmente non influenza il calcolo del t_{tlc} che si basa solo sulle dinamiche laterali Y e V_y.

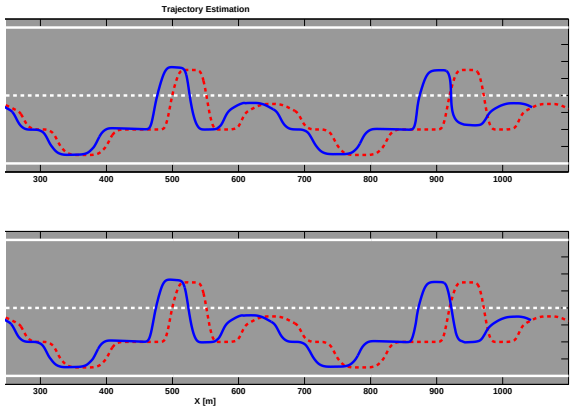

Figura 7.8. Traiettoria su strada

In figura (7.9)) è evidente come l'algoritmo con dati da sola visione generi falsi allarmi in sette diversi punti del tracciato. In particolare, abbiamo false attivazioni ai metri 150, 180, 550, 610, 750, 980 e 1060. Inoltre non viene correttamente attivato l'allarme di attraversamento della carreggiata a 960 metri. Gli allarmi generati invece utilizzando il t_{tlc} ottenuto con i filtri di Kalman è sempre corretto.

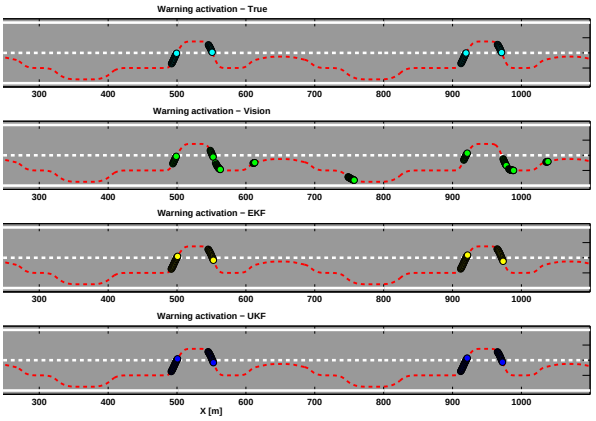

Figura 7.9. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.1.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

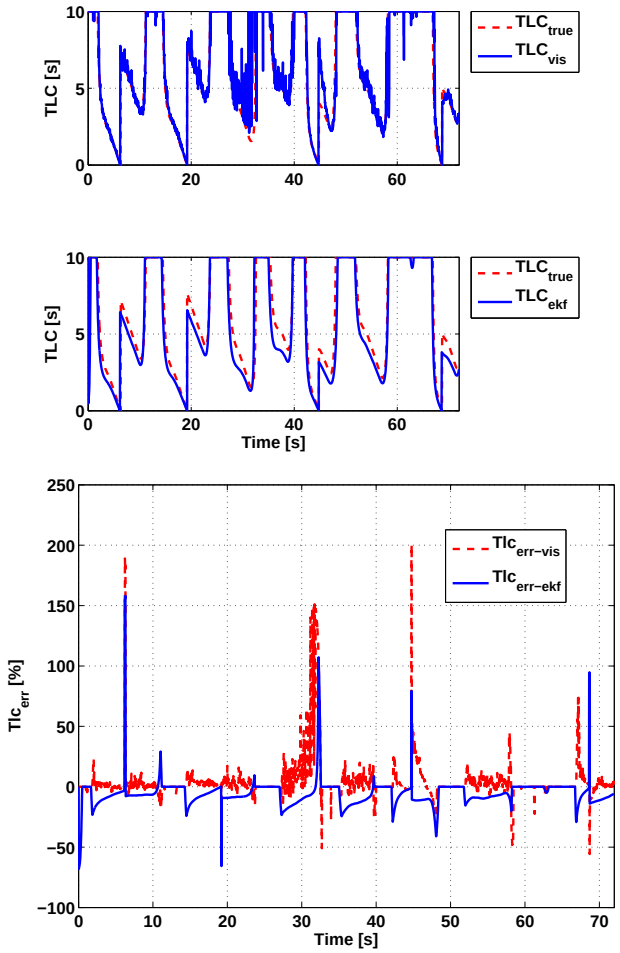

Figura 7.10. Tempo di invasione t_{tlc} EKF

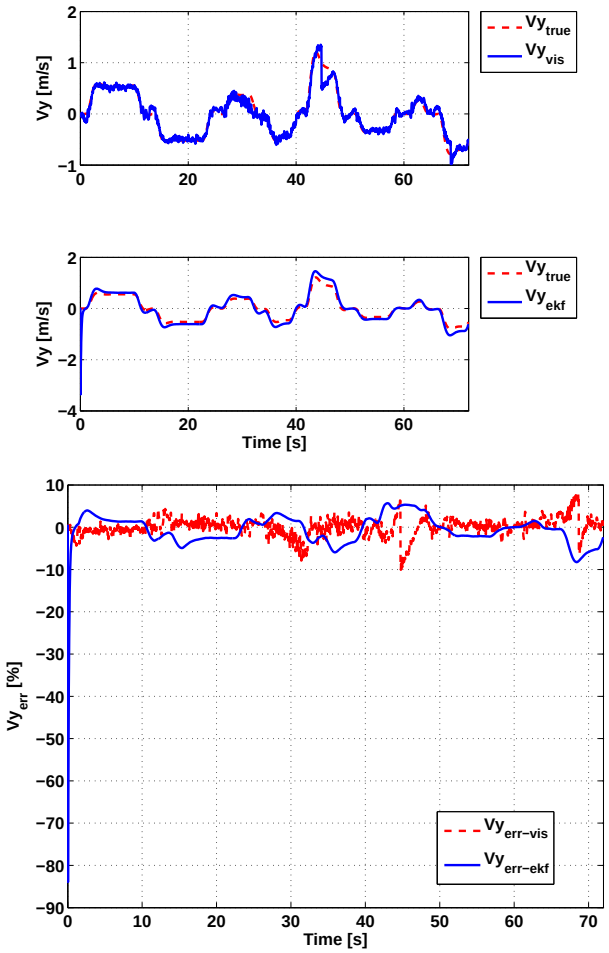

Figura 7.11. Velocità laterale EKF

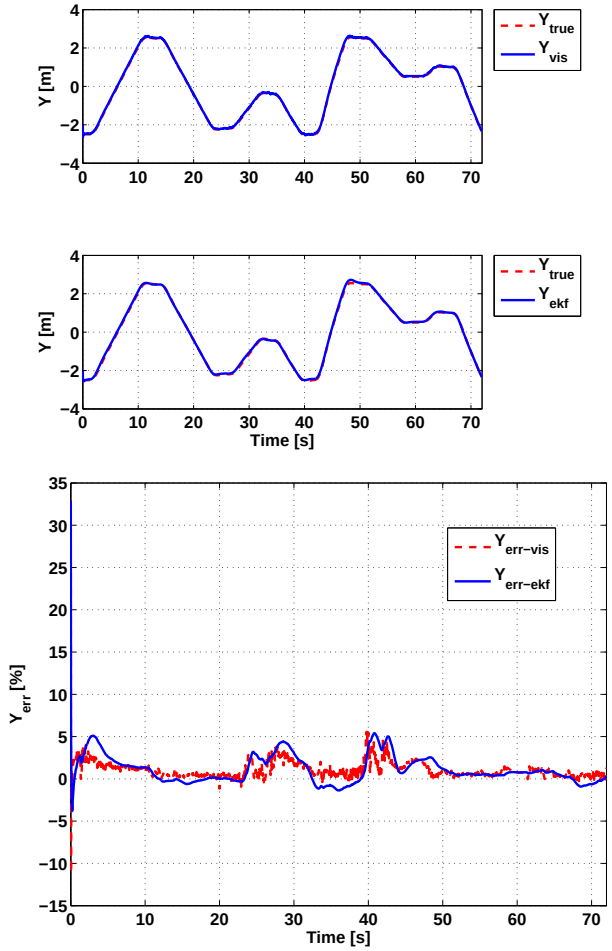

Figura 7.12. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h. La simulazione ha una durata di 72 secondi. Come è visibile in figura (7.10), il filtro di Kalman Esteso permette di filtrare l'elevato rumore di cui è affetto il t_{tlc} calcolato direttamente dai dati da visione.

Nel grafico relativo alla velocità laterale (V_y)(figura (7.11)), si nota come l'effetto del filtraggio, pur riducendo la variabilità del segnale, introduca un errore a volte maggiore di quello ottenuto utilizzando i soli dati da visione. L'errore sulla velocità laterale rimane comunque inferiore al 10% sia per il filtro di Kalman Esteso sia per il metodo da sola visione. Discorso analogo vale per la posizione laterale che, come mostra la figura (7.12), presenta un

errore inferiore al 5%. Anche in questo caso, il maggior contributo all'errore del t_{tlc} è quindi dovuto alla stima della velocità laterale.

Simulazioni con filtro UKF

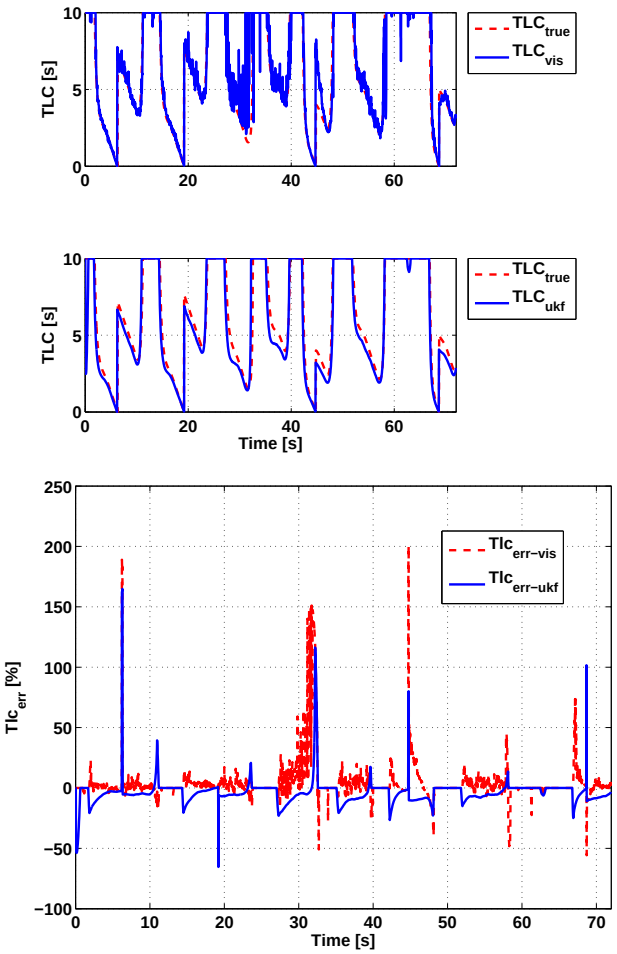

Figura 7.13. Tempo di invasione t_{tlc} UKF

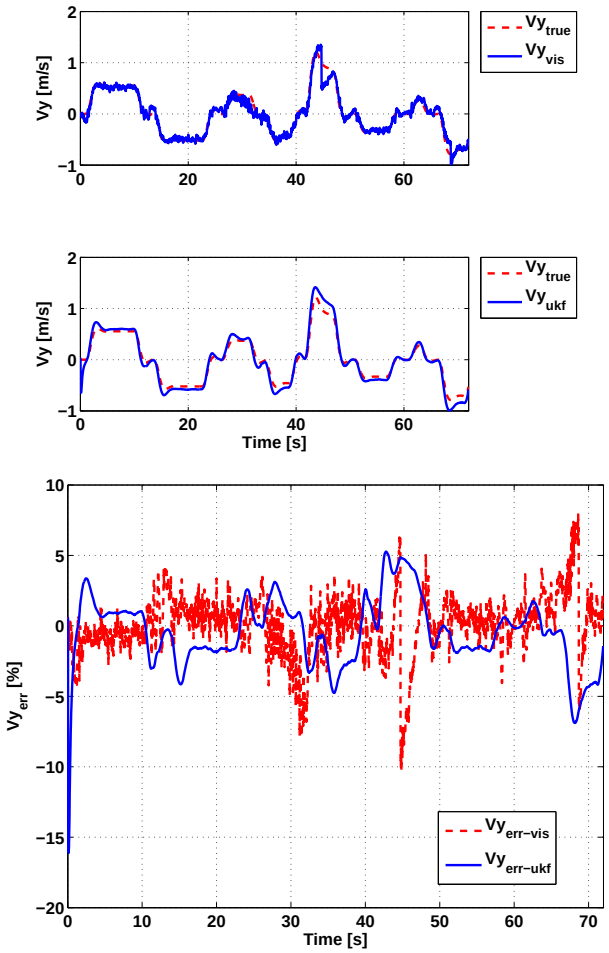

Figura 7.14. Velocità laterale UKF

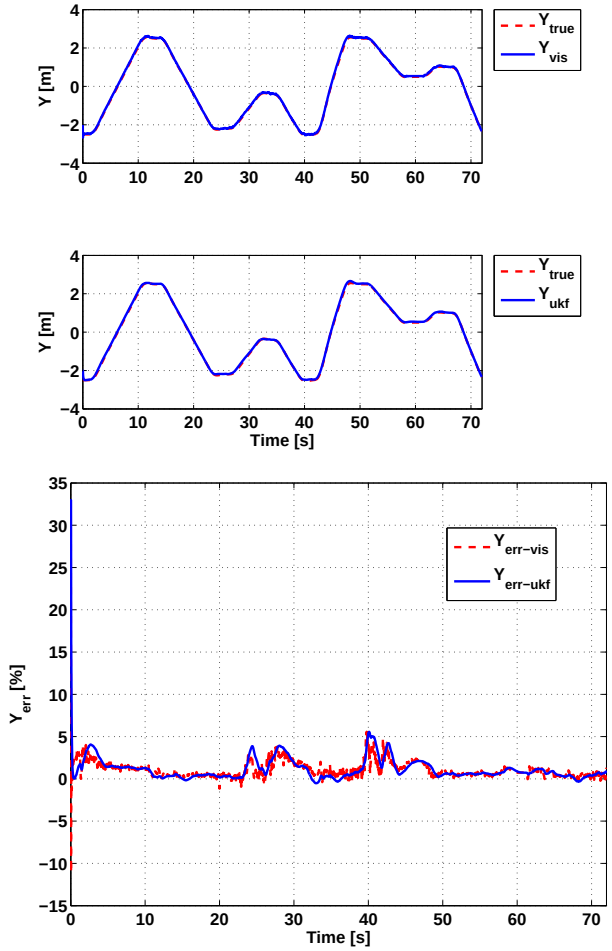

Figura 7.15. Posizione laterale UKF

La figura (7.13) descrive l'andamento del t_{tlc} utilizzando il filtro di Kalman Unscented. Anche in questo caso come avvenuto con l'uso del filtro di Kalman Esteso il rumore di misura visibile nei dati di visione viene attenuato. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso.

La figura (7.15) descrive l'andamento della posizione laterale. Il filtro di Kalman Unscented riesce a inseguire con più precisione la posizione rispetto alla velocità laterale (figura (7.14)).

Confronto tra EKF e UKF

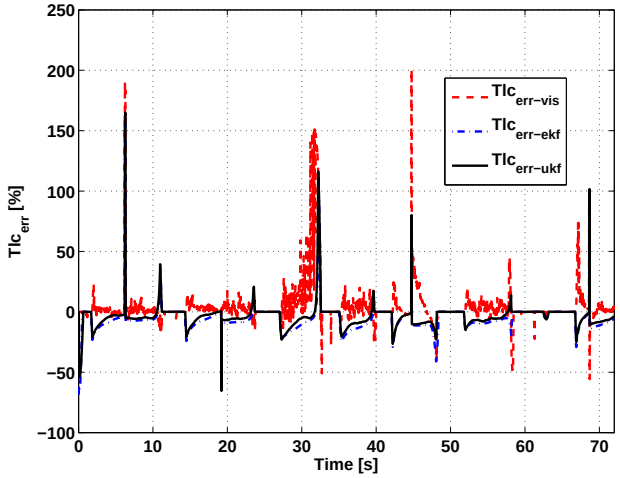

Figura 7.16. Errore sul t_{tlc}

La figura (7.16) riassume gli errori commessi nel calcolo del t_{tlc}. Sia il filtro di Kalman Unscendent che quello Esteso riducono il rumore presente nei dati di sola visione ma, in alcuni istanti, si perde precisione nella stima. Il filtro di Kalman Unscented presenta un leggero vantaggio in termini di errore rispetto al filtro di Kalman Esteso nell'ordine di 1-2 punti percentuali.

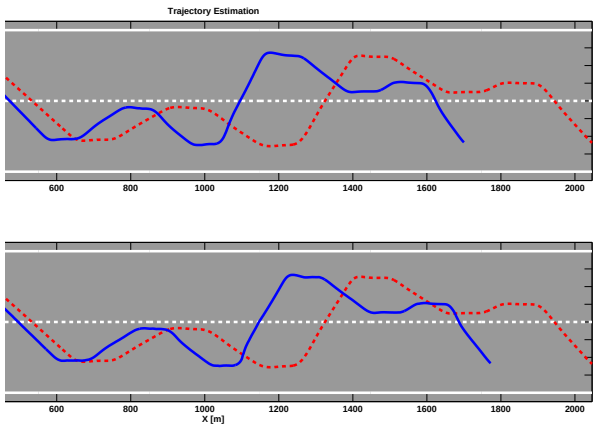

Figura 7.17. Traiettoria su strada

Permane un errore di offset sulla traiettoria XY visibile in figura (7.17), dovuto essenzialmente alla stima non corretta della coordinata X. Questo naturalmente non influenza il calcolo del t_{tlc} che si basa solo sulle dinamiche laterali (Y e V_y). In conclusione, gli allarmi generati, riportati in figura (7.18), non presentano differenze tra i diversi approcci di calcolo e risultano tutti corretti.

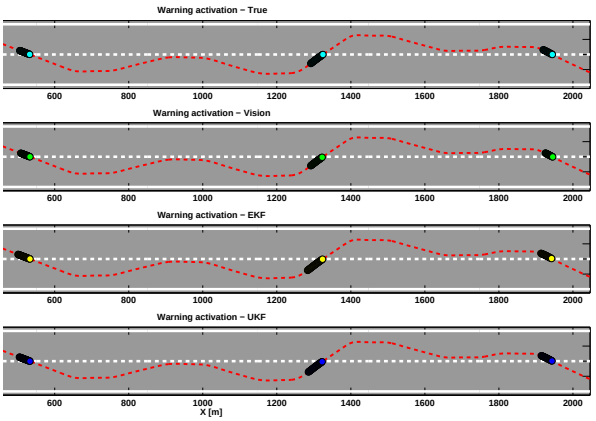

Figura 7.18. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.2 Modello 2: Cinematico Differenziale

7.2.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

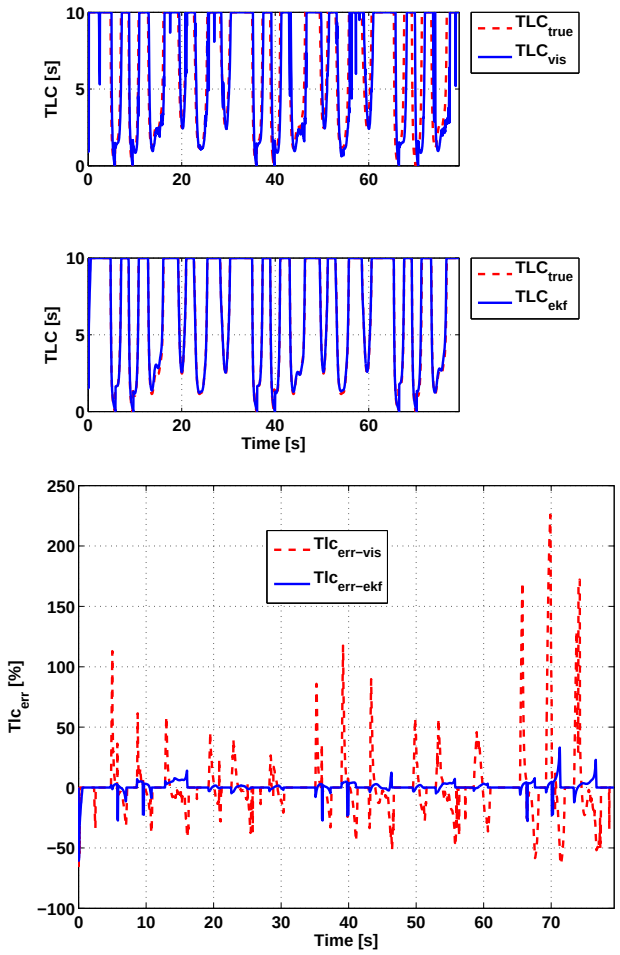

Figura 7.19. Tempo di invasione t_{ttc} EKF

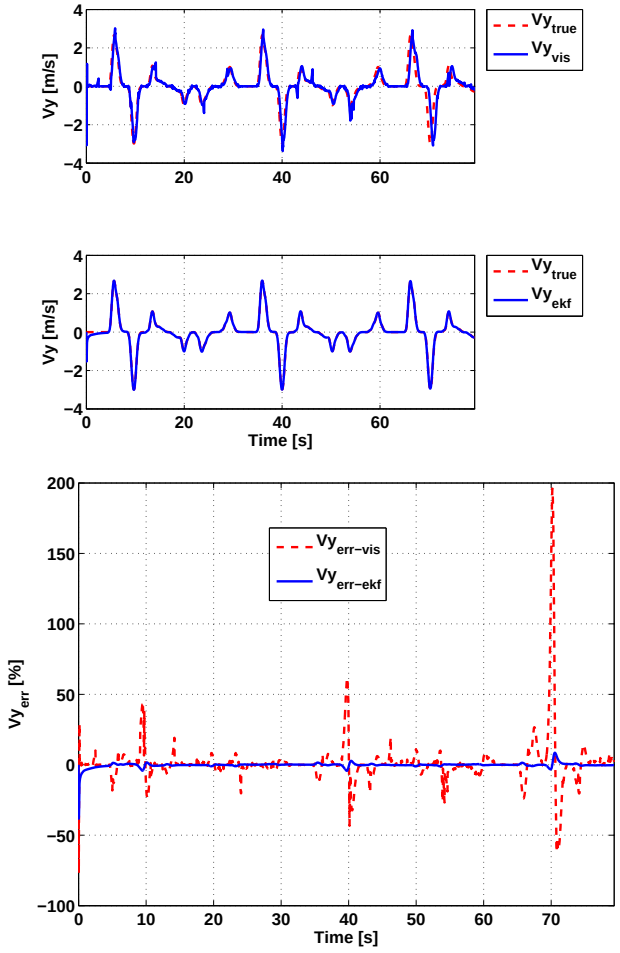

Figura 7.20. Velocità laterale EKF

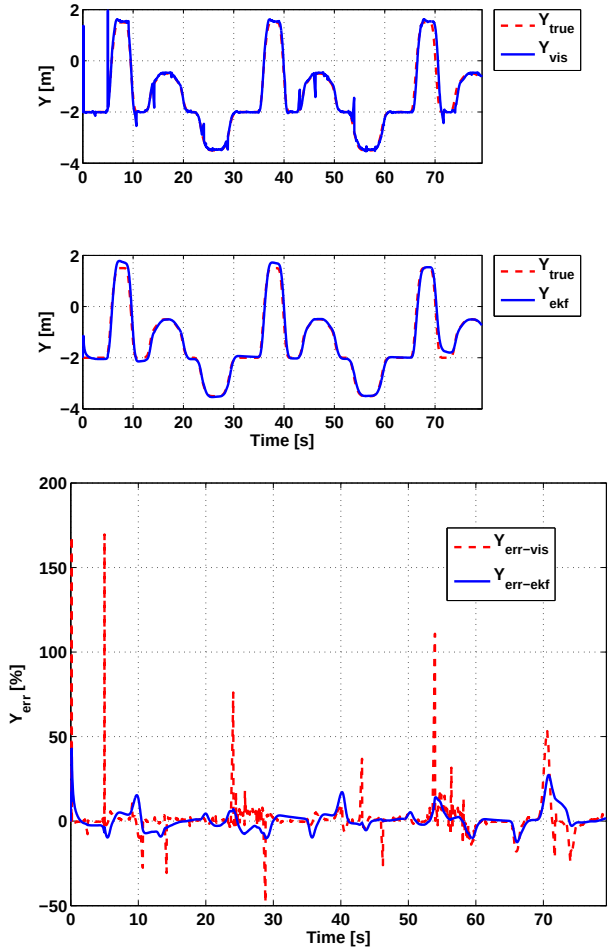

Figura 7.21. Posizione laterale EKF

Il t_{tlc} calcolato con filtro di Kalman Esteso e modello cinematico differenziale è visibile in figura (7.19). La simulazione effettuata a velocità costante di $50Km/h$ ha una durata di 80 secondi. Il t_{tlc} stimato con filtro di Kalman Esteso ha un numero di picchi di errore nettamente minore rispetto al calcolo con i dati da visione. In particolare i picchi non superano mai il 25% mentre negli intervalli privi di improvvise escursioni l'errore si mantiene inferiore al 4%.

Per quanto riguarda la velocità laterale (figura (7.20)), l'errore rimane inferiore all'1% tranne che ai secondi 10, 40 e 70 di simulazione quando raggiunge il 4% e l'8%. Negli stessi istanti di tempo la velocità laterale calcolata con i dati di visione ha errori che raggiungono il 50% e l'200%.

La stima della posizione laterale (figura (7.21)) presenta un errore che si attesta intorno a ±10% e che sale al 15% e al 20% ai secondi 10, 40 e 70.

Simulazioni con filtro UKF

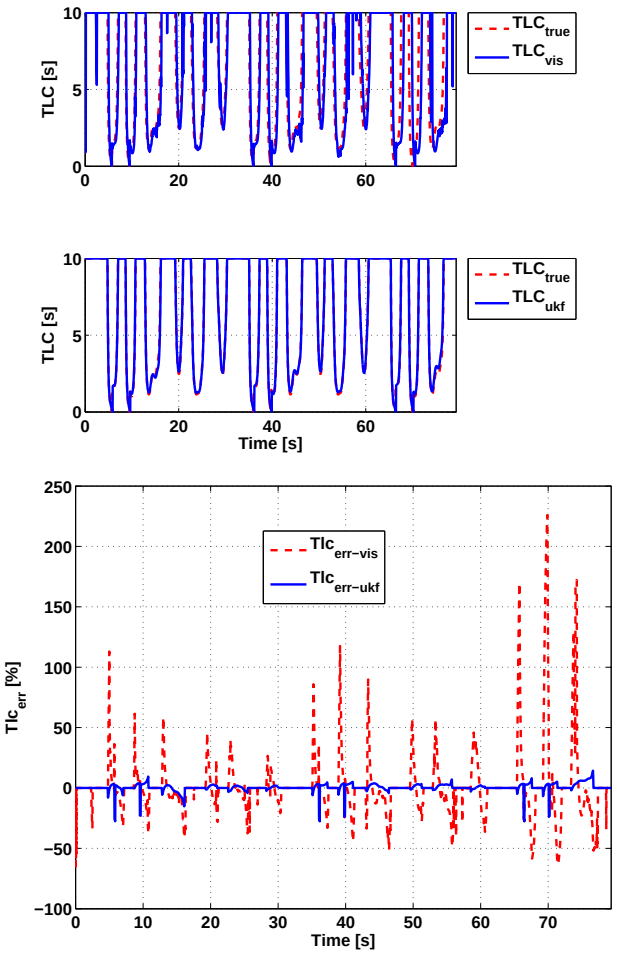

Figura 7.22. Tempo4 di invasione t_{tlc} UKF

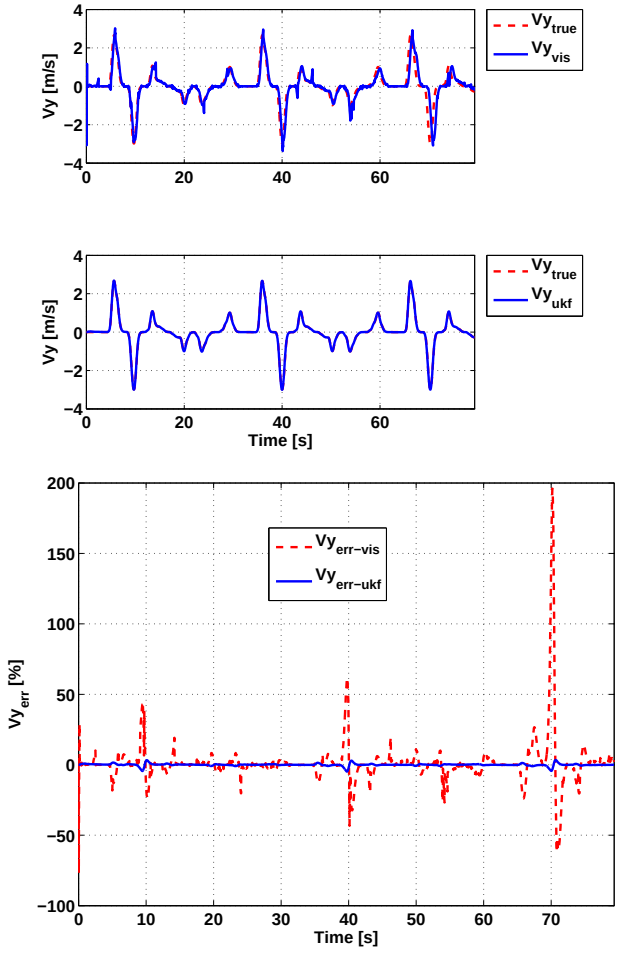

Figura 7.23. Velocità laterale UKF

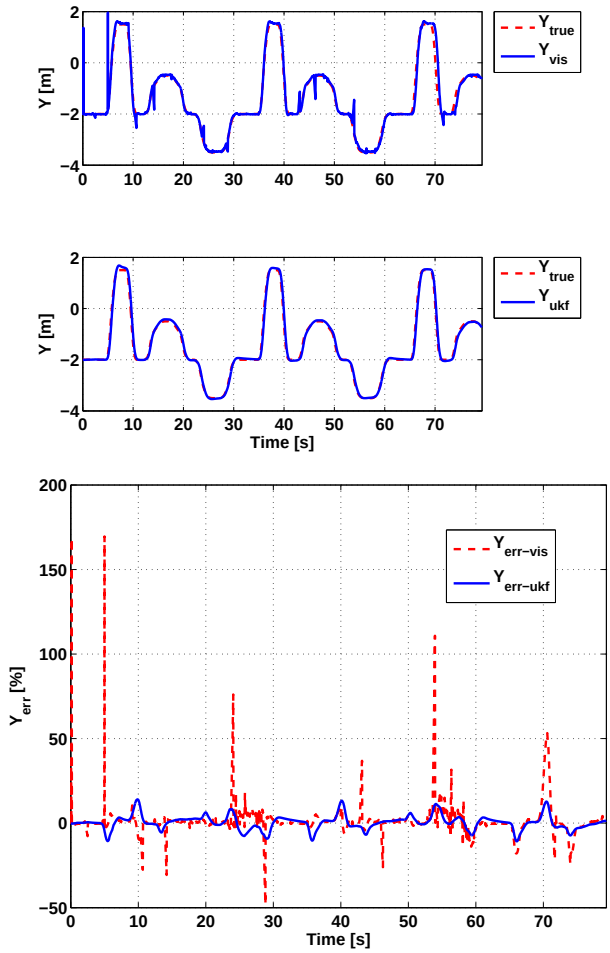

Figura 7.24. Posizione laterale UKF

La figura (7.22) mostra il t_{tlc} ottenuto con il filtro di Kalman Unscented. Il filtro stima il t_{tlc} con un errore in media inferiore al $3,5\%$. I picchi presenti intorno ai secondi 10, 40 e 70 raggiungono il 23% e sono dovuti ai cambi di corsia.

In figura (7.23) è riportato il grafico della velocità laterale. In particolare, l'errore di stima del filtro di Kalman Unscented si mantiene minore dello 0.8% tranne che ai secondi 10, 40 e 70 in cui ha delle escursioni fino al 4%.

Per quel che riguarda la posizione laterale (figura (7.24)), l'errore di stima si attesta intorno al 10%.

Confronto tra EKF e UKF

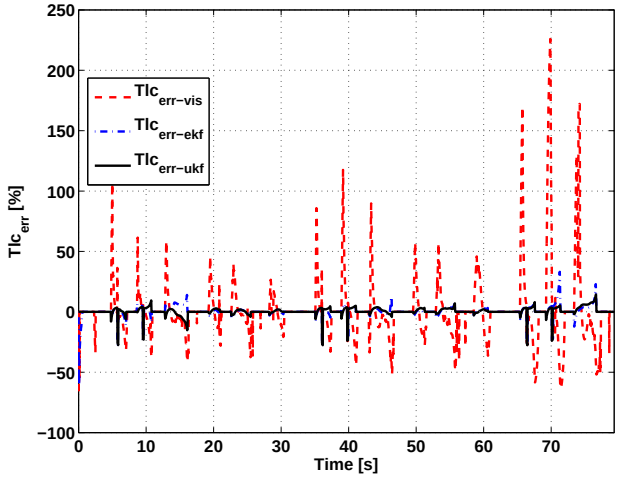

Figura 7.25. Errore sul t_{tlc}

Il confronto sul t_{tlc} di figura (7.25) mette in evidenza il vantaggio dell'uso dei filtri di Kalman rispetto ai soli dati da visione. I due filtri di Kalman Unscented e Esteso hanno un comportamento molto simile e per buona parte della simulazione l'errore commesso coincide. Il filtro di Kalman Unscented però negli intervalli tra 10 e 20 e tra 70 e 80 è più performante nella stima del t_{tlc}.

Il leggero vantaggio del filtro Unscented è apprezzabile anche dalla figura (**??**) dove la stima della traiettoria è più precisa.

I due filtri riescono entrambi ad eliminare tutti i falsi allarmi, mentre le attivazioni ottenute con il t_{tlc} calcolato dai dati da visione non sono altrettanto precise (figura (7.26)). In particolare con i dati da visione abbiamo falsi allarmi a 130, 190, 550, 610, 760, 990 e 1070 metri, inoltre a 960 metri non viene rivelato un attraversamento di corsia.

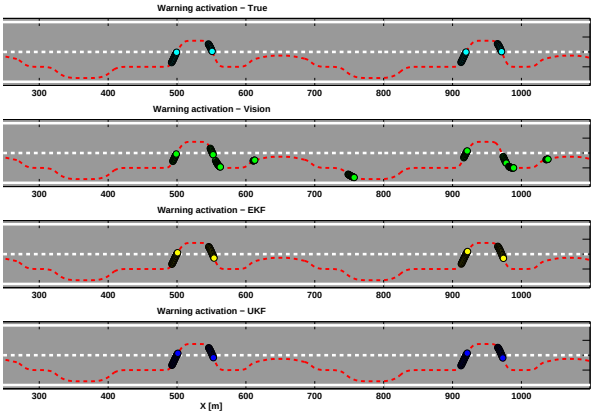

Figura 7.26. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.2.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

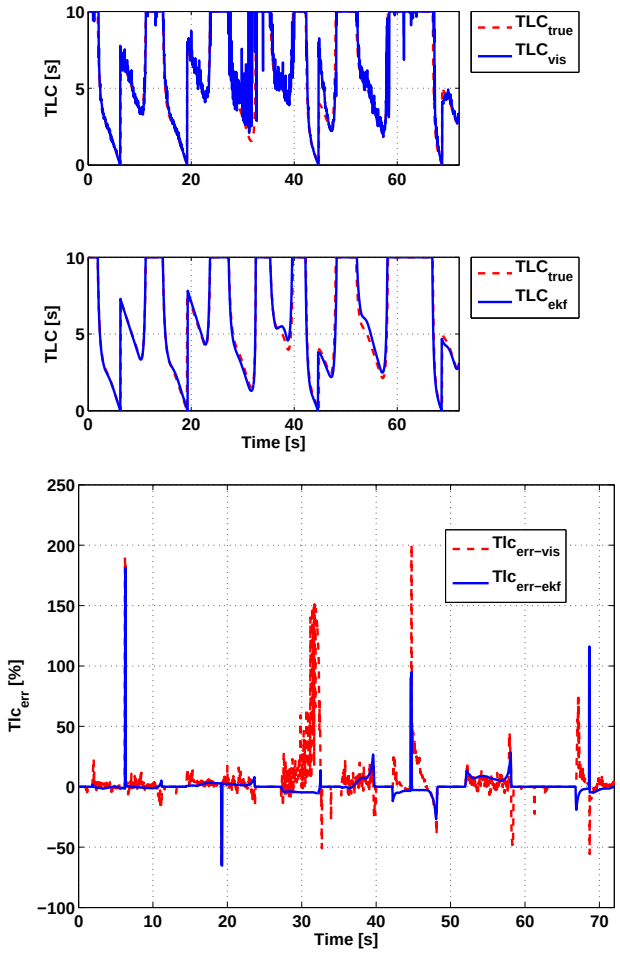

Figura 7.27. Tempo di invasione t_{tlc} EKF

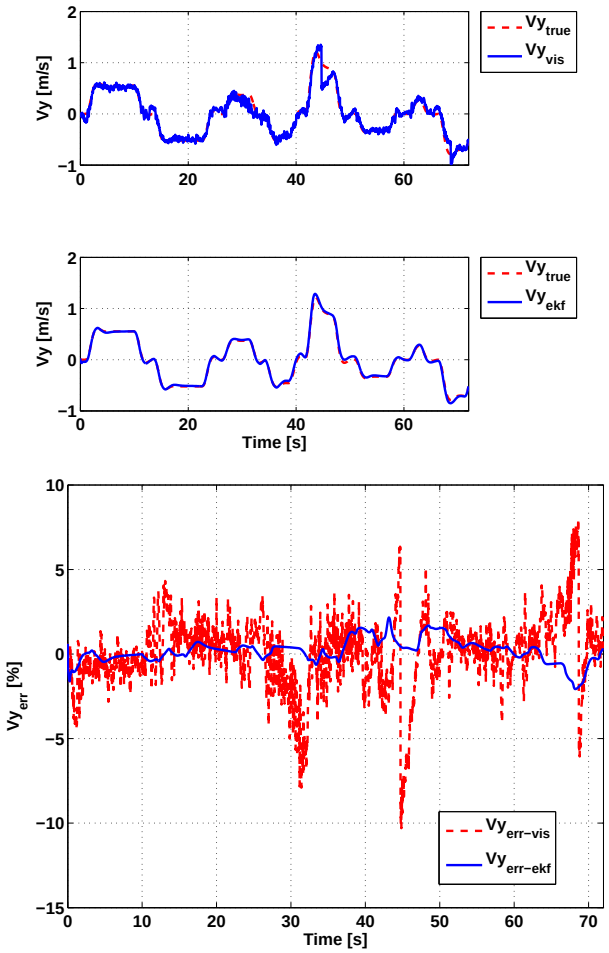

Figura 7.28. Velocità laterale EKF

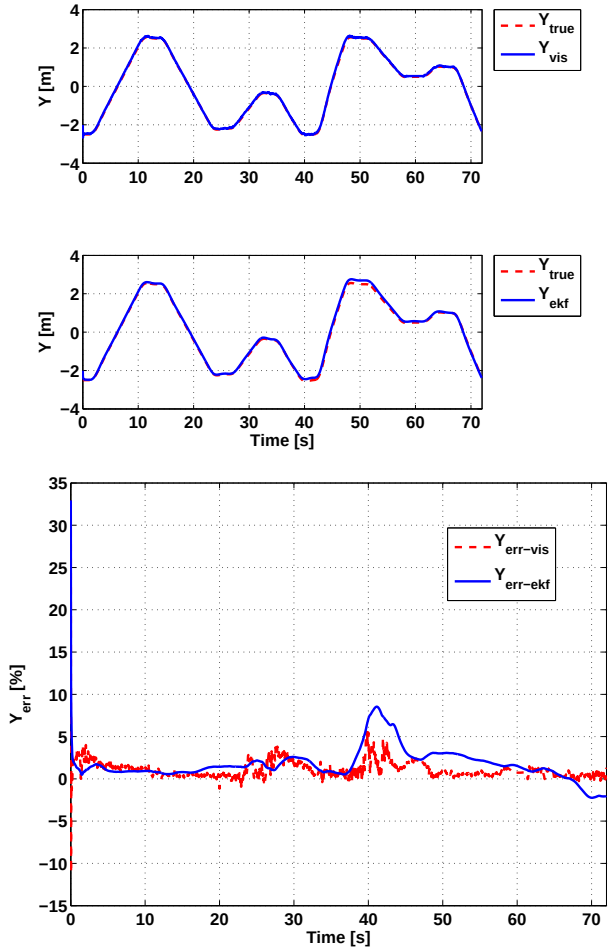

Figura 7.29. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h con brusche frenate e accelerazioni in corrispondenza dei cambi di corsia. La simulazione ha una durata di 72 secondi. Come è visibile in figura (7.27), il t_{tlc} calcolato dai dati da visione è estremamente rumoroso. Filtrando i dati con il filtro di Kalman Esteso e utilizzando il modello cinematico differenziale, il t_{tlc} viene ripulito dal rumore e l'errore scende a valori quasi sempre inferiori al 10% con intervalli di errore nullo durante la marcia all'interno della carreggiata. I 4 picchi di errore in corrispondenza dei cambi di corsia, quando il t_{tlc} tende a zero, non hanno influenza sulla generazione degli avvisi.

Come si evince dal grafico relativo alla velocità laterale (V_y)(figura (7.28)),

il filtro di Kalman Esteso elimina completamente l'effetto del rumore introdotto dai dati di visione con un errore di stima inferiore al 2%.

Maggiori difficoltà di stima si riscontrano nella posizione laterale che, come mostra la figura (7.29), ha un errore a volte maggiore rispetto a quello generato dai dati da visione, mantenendosi comunque, anche nel caso peggiore, inferiore al 10%. Il maggior contributo all'errore sul t_{tlc} è quindi dovuto alla stima della posizione laterale.

Simulazioni con filtro UKF

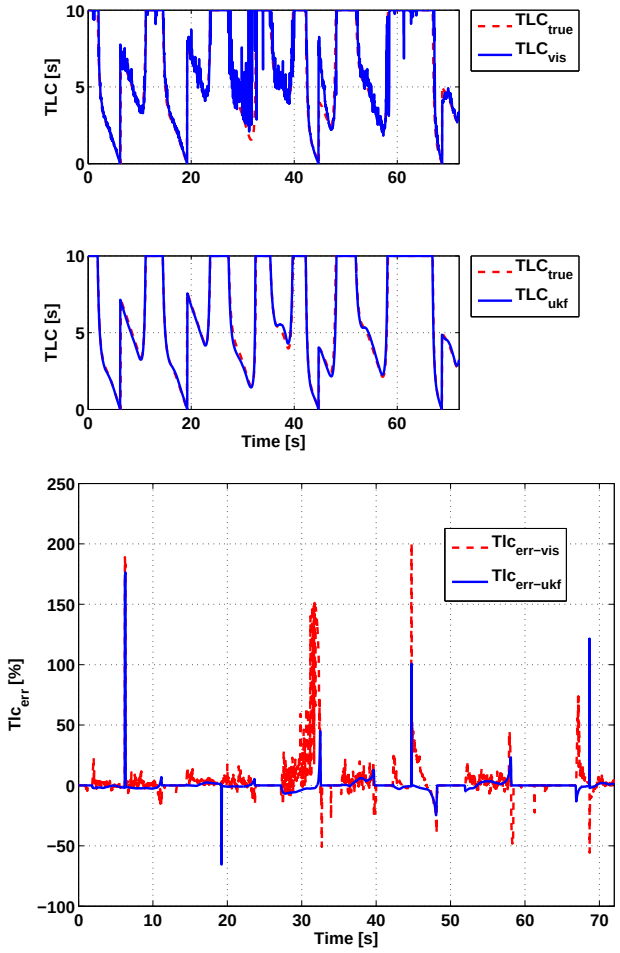

Figura 7.30. Tempo di invasione t_{tlc} UKF

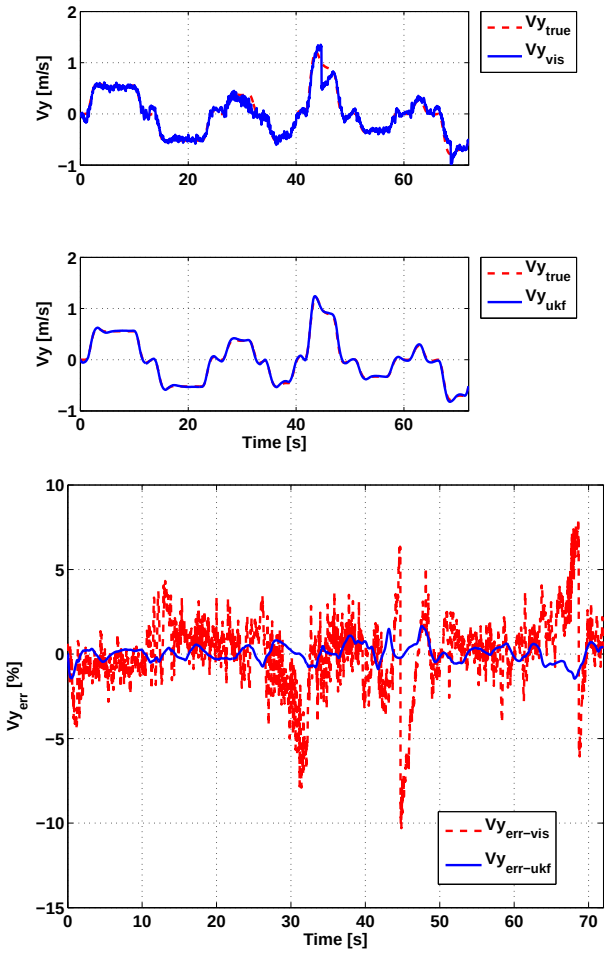

Figura 7.31. Velocità laterale UKF

La figura (7.30) descrive l'andamento del t_{tlc} utilizzando il filtro di Kalman Unscented. Anche in questo caso, come avvenuto con l'uso del filtro di Kalman Esteso, il rumore di misura visibile nei dati da visione viene attenuato. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso ma l'errore, al di fuori dei pochi picchi in prossimità dei cambi di corsia, è minore del 6%.

La figura (7.31) descrive l'andamento della velocità laterale. Il filtro di Kalman Unscented elimina il rumore presente nei dati di visione e riesce a mantenere un errore di stima minore del 1.5% durante tutta la simulazione.

Il filtro riesce a inseguire con precisione anche la posizione laterale (figura (7.31)): tra 0 e 20 e tra 50 e 72 secondi l'errore è minore del 2%, nella parte centrale della simulazione invece non supera mai il 4%.

Confronto tra EKF e UKF

La figura (**??**) riassume gli errori commessi nel calcolo del t_{tlc}. Sia il filtro di Kalman Unscented che quello Esteso riducono il rumore presente nella simulazione con dati da sola visione, ottenendo un netto miglioramento nel calcolo del t_{tlc}.

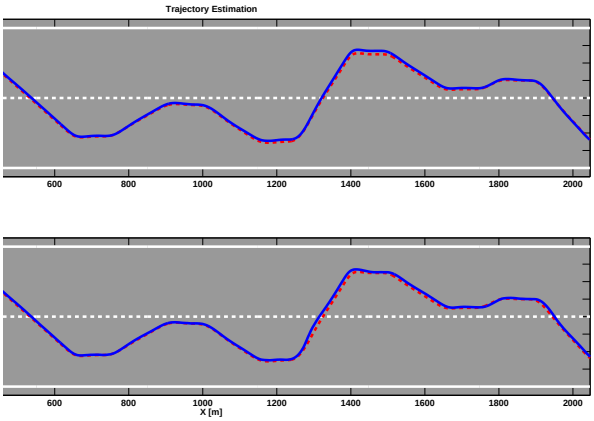

Figura 7.32. Traiettoria su strada

La traiettoria XY visibile in figura (7.32) mostra un buon inseguimento per entrambi i filtri e non risulta evidente una differenza nell'uso dell'uno o dell'altro.

Gli allarmi generati, riportati in figura (7.33), evidenziano la corretta stima dei due filtri di Kalman. Al contrario utilizzando esclusivamente i dati da visione viene generato un falso allarme a 1800 metri e non ne viene rilevato uno a 900 metri.

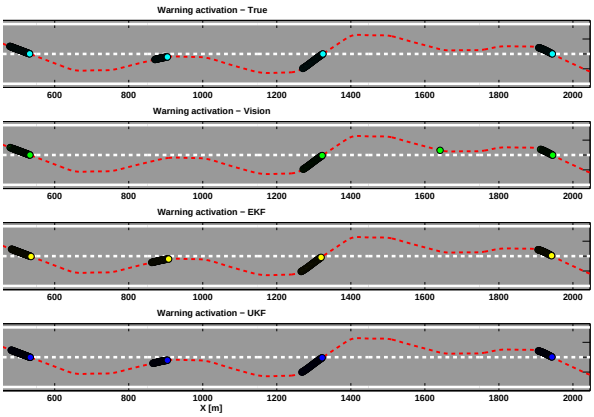

Figura 7.33. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.3 Modello 3: Cinematico yaw rate

7.3.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

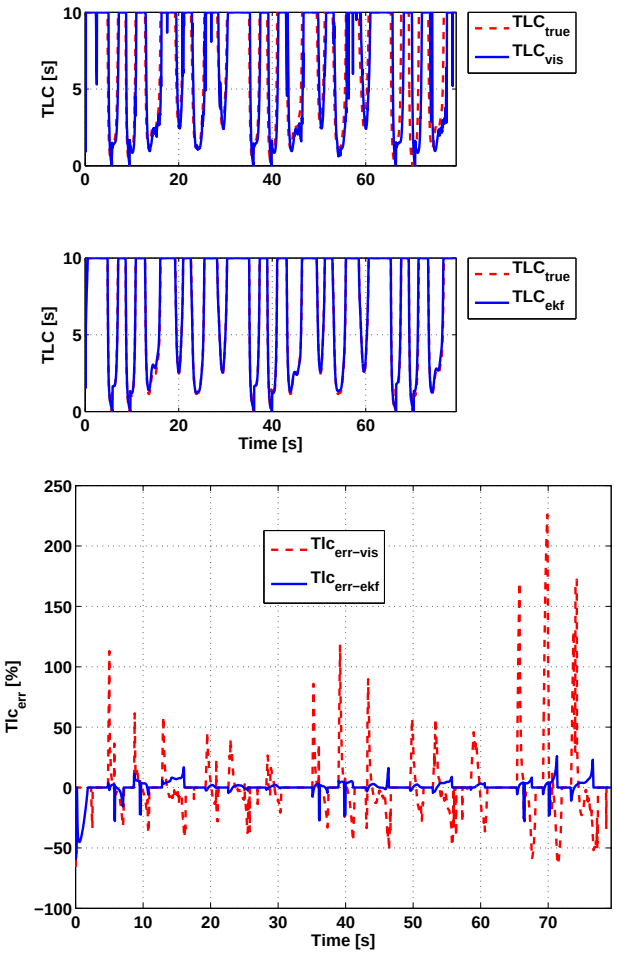

Figura 7.34. Tempo di invasione t_{ttc} EKF

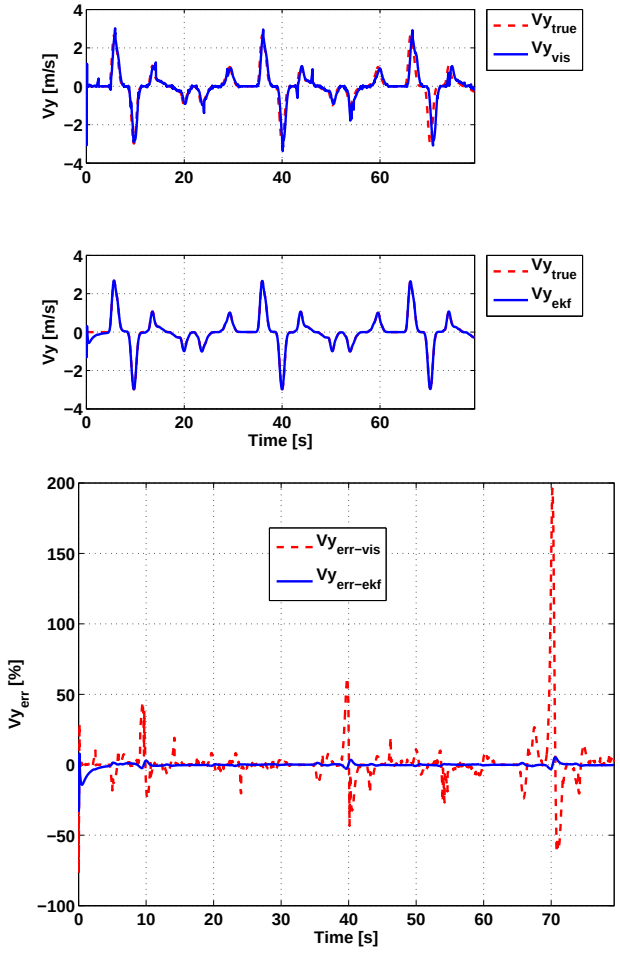

Figura 7.35. Velocità laterale EKF

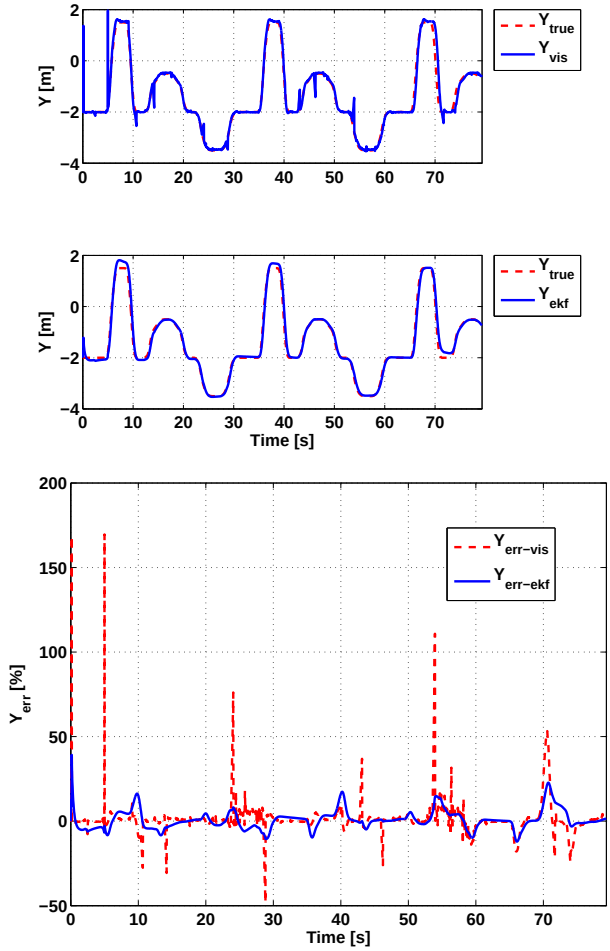

Figura 7.36. Posizione laterale EKF

Il t_{tlc} calcolato con filtro di Kalman Esteso e modello cinematico con yaw rate è visibile in figura (7.34). La simulazione effettuata a velocità costante di $50Km/h$ ha una durata di 80 secondi. Il t_{tlc} stimato con filtro di Kalman Esteso ha un ridotto numero di picchi di errore rispetto al calcolo con i dati da visione e la loro ampiezza non supera mai il 25%. Negli intervalli privi di improvvise escursioni l'errore si mantiene inferiore al 5%.

La velocità laterale (figura (7.35)) ha un errore inferiore all'1% tranne che ai secondi 10, 40 e 70 di simulazione quando raggiunge il $4-5\%$. Negli stessi istanti di tempo la velocità laterale calcolata con i dati di visione ha errori che raggiungono il 50% e il 200%.

La stima della posizione laterale (figura (7.36)) presenta un errore che si

attesta intorno a ±10% e che sale al 15% e al 20% ai secondi 10, 40 e 70.

Simulazioni con filtro UKF

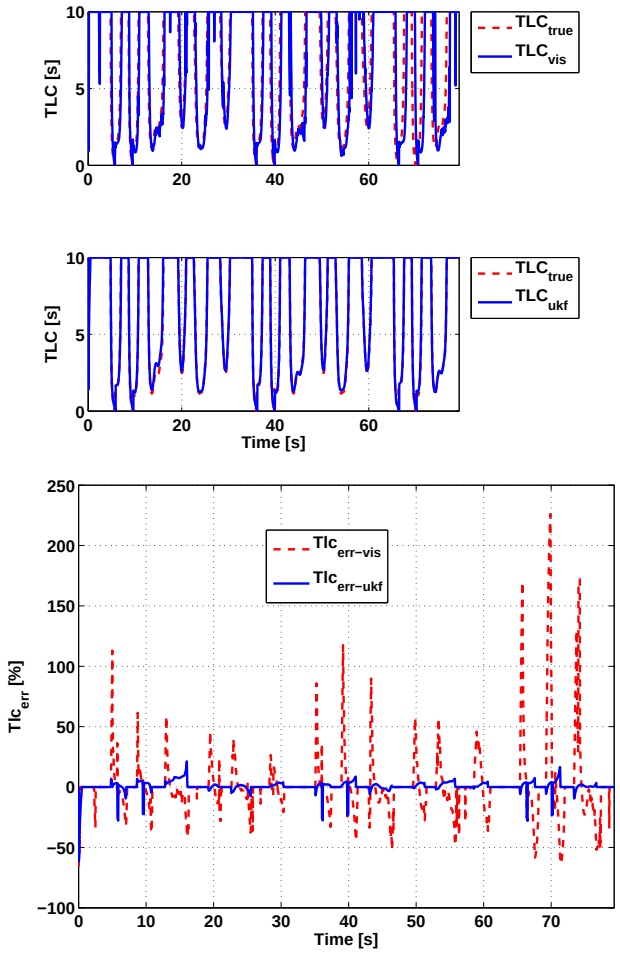

Figura 7.37. Tempo di invasione t_{tlc} UKF

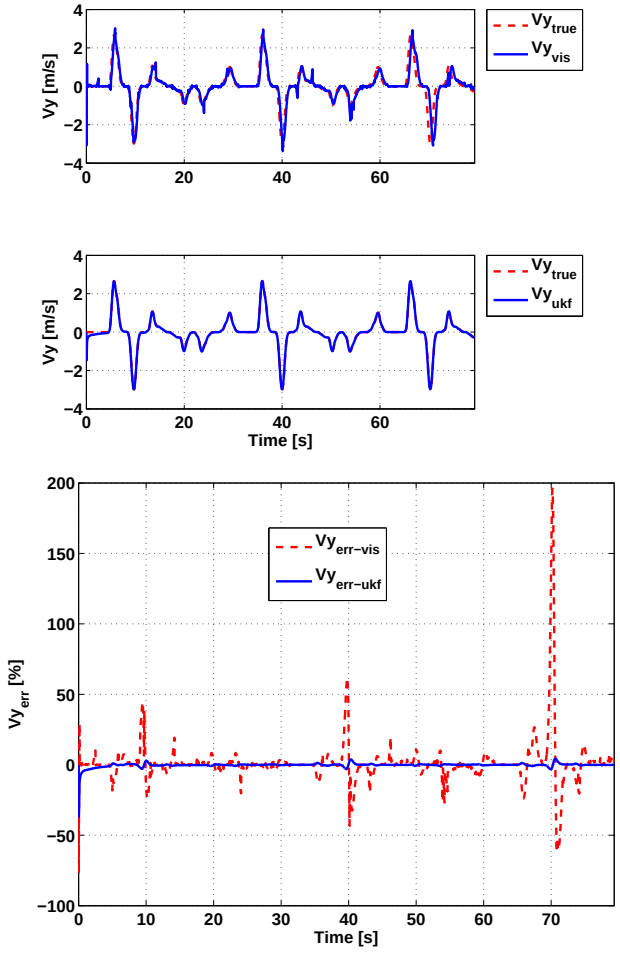

Figura 7.38. Velocità laterale UKF

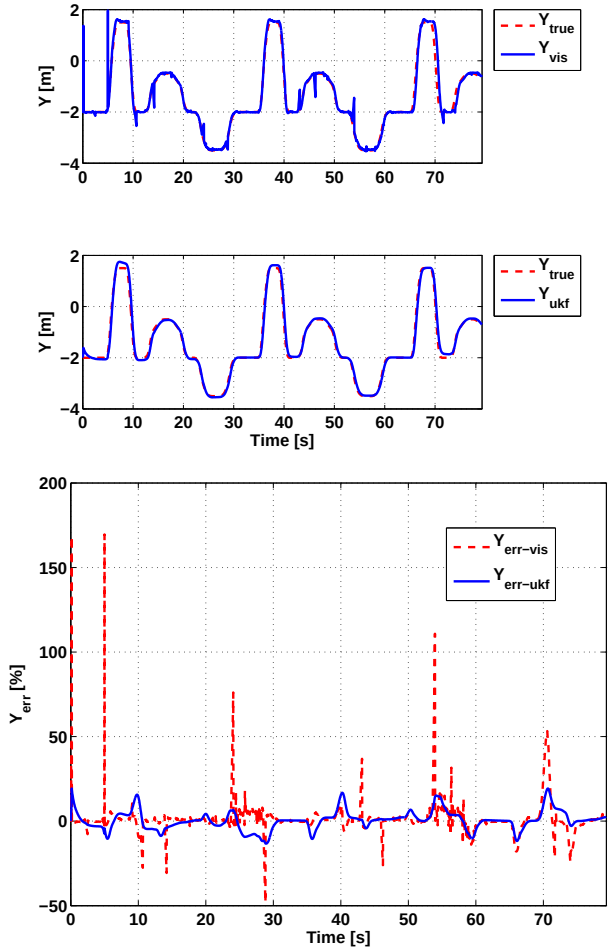

Figura 7.39. Posizione laterale UKF

La figura (7.37) mostra il t_{tlc} ottenuto con il filtro di Kalman Unscented la stima ha un errore in media inferiore al 4% ad esclusione dei secondi 10, 40 e 70 in cui alcuni picchi raggiungono, a causa dei cambi di corsia, il 23%.

In figura (7.38) è riportato il grafico della velocità laterale. In particolare, l'errore di stima del filtro di Kalman Unscented si mantiene minore dello 0.8% tranne che ai secondi 10, 40 e 70 in cui ha delle escursioni fino al 4%.

Per quanto riguarda la posizione laterale (figura (7.39)), l'errore di stima si attesta intorno al $\pm 15\%$.

Confronto tra EKF e UKF

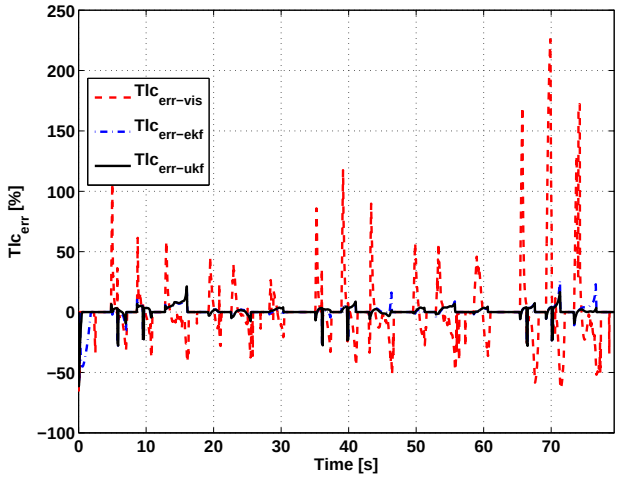

Figura 7.40. Errore sul t_{tlc}

Il confronto sul t_{tlc} di figura (7.40) mette in evidenza il vantaggio dell'uso dei filtri di Kalman rispetto ai soli dati da visione. I due filtri di Kalman Unscented e Esteso hanno un comportamento molto simile e per buona parte della simulazione l'errore commesso coincide. Il filtro di Kalman Unscented però negli intervalli tra 10 e 20 e tra 70 e 80 è più performante nella stima del t_{tlc}.

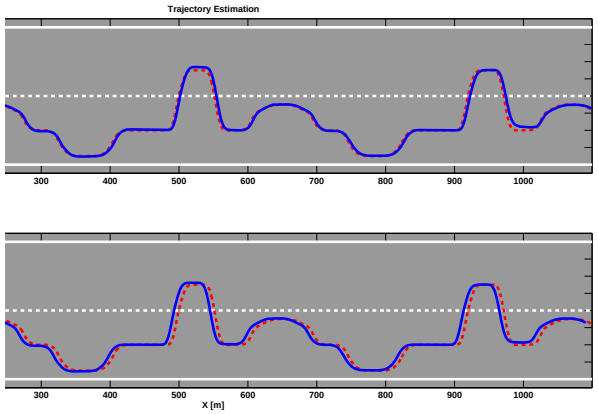

Figura 7.41. Traiettoria su strada

La traiettoria stimata con filtro di Kalman Esteso è leggermente più precisa rispetto a quella ottenuta con il filtro Unscented come evidenziato in figura (7.41).

I due filtri riescono entrambi ad eliminare tutti i falsi allarmi, mentre le attivazioni ottenute con il t_{tlc} calcolato dai dati da visione non sono altrettanto precise (figura (7.42)). In particolare con i dati da visione abbiamo falsi allarmi a 130, 190, 550, 610, 760, 990 e 1070 metri, inoltre a 960 metri non viene rivelato un attraversamento di corsia.

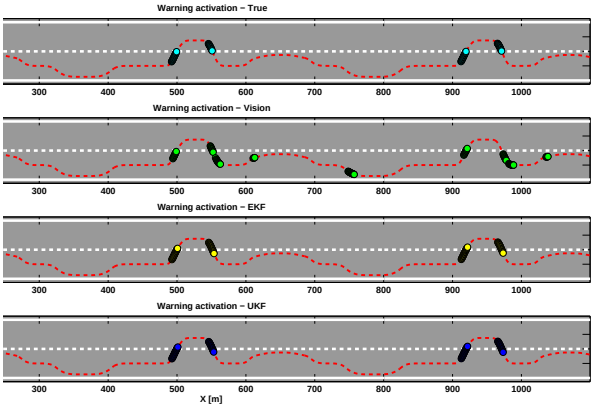

Figura 7.42. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.3.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

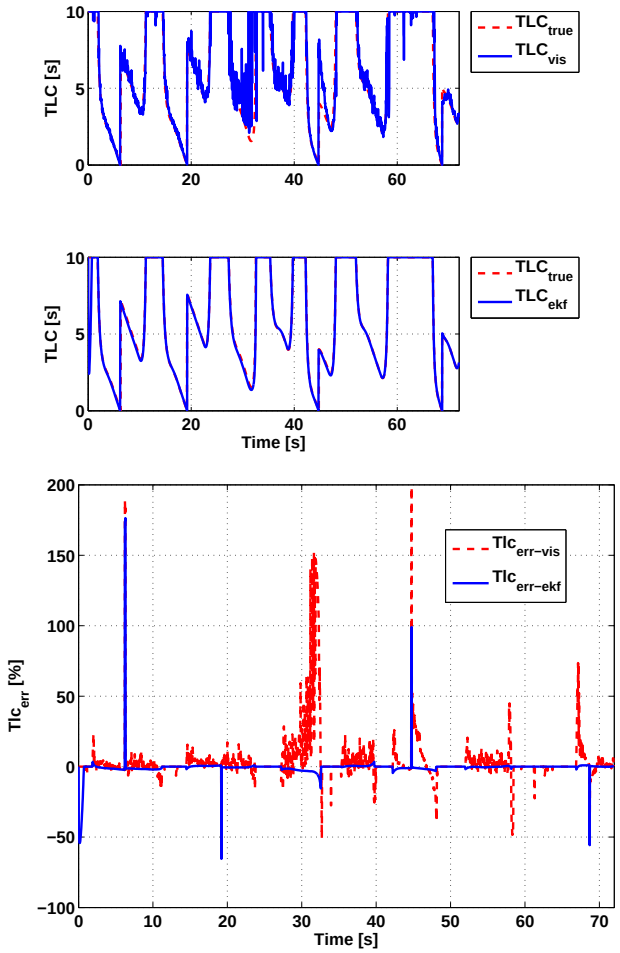

Figura 7.43. Tempo di invasione t_{tlc} EKF

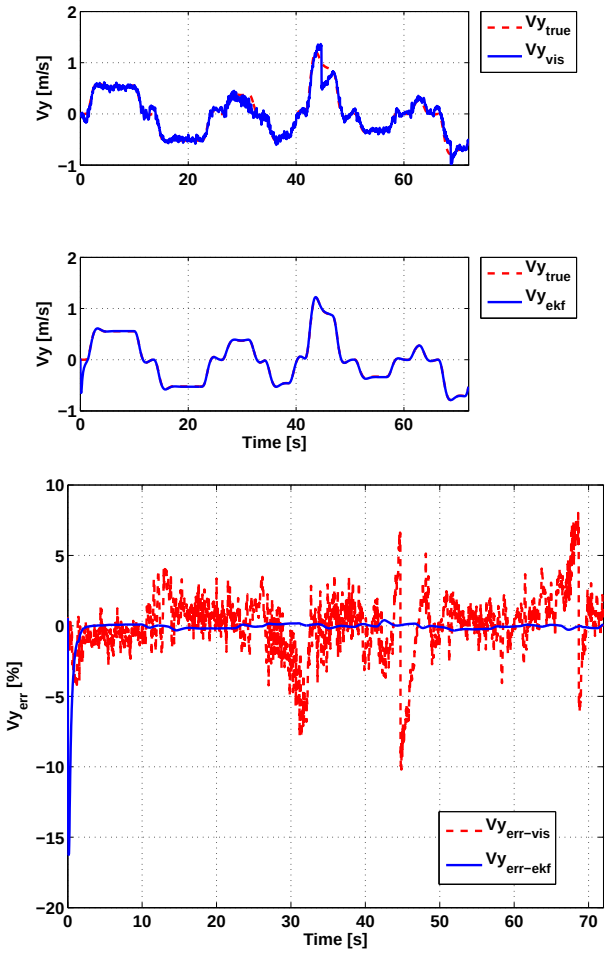

Figura 7.44. Velocità laterale EKF

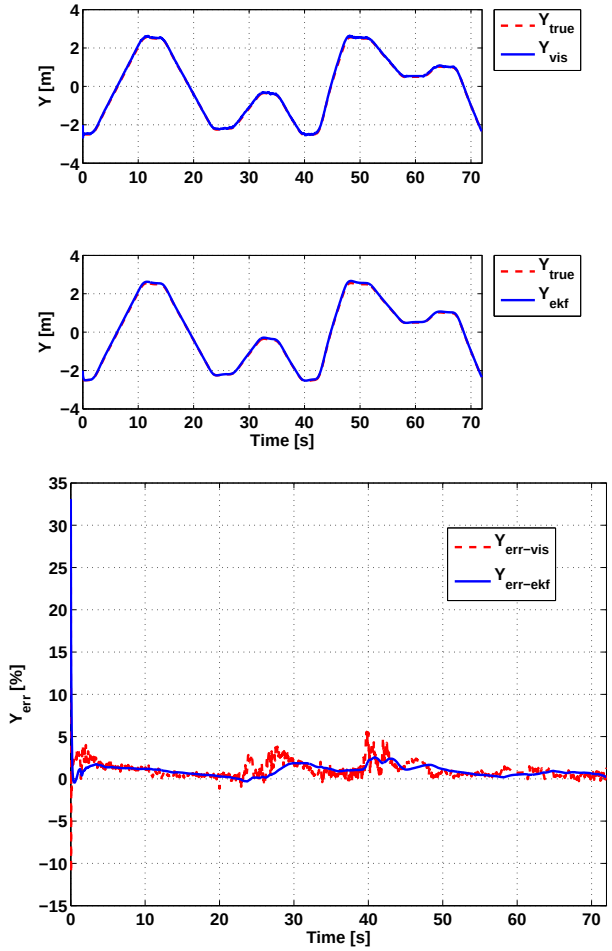

Figura 7.45. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h con brusche frenate e accelerazioni in corrispondenza dei cambi di corsia. La simulazione ha una durata di 72 secondi. Come è visibile in figura (7.43), il t_{tlc} calcolato dai dati da visione è estremamente rumoroso ma filtrando i dati con filtro di Kalman Esteso basato su modello cinematico yaw rate, il t_{tlc} viene ripulito dal rumore e l'errore scende a valori quasi sempre inferiori al 5% con intervalli di errore nullo durante la marcia all'interno della carreggiata. I 4 picchi di errore in corrispondenza dei cambi di corsia, quando il t_{tlc} tende a zero, non hanno influenza sulla generazione degli avvisi.

La velocità laterale (V_y)(figura (7.44)) è stimata con un errore inferiore allo

0.4% grazie al filtro di Kalman Esteso che elimina completamente l'effetto del rumore introdotto dai dati di visione.

Anche l'errore di stima della posizione laterale si mantiene molto basso e, come mostra la figura (7.45), ha un errore sempre minore al 2.5%.

Simulazioni con filtro UKF

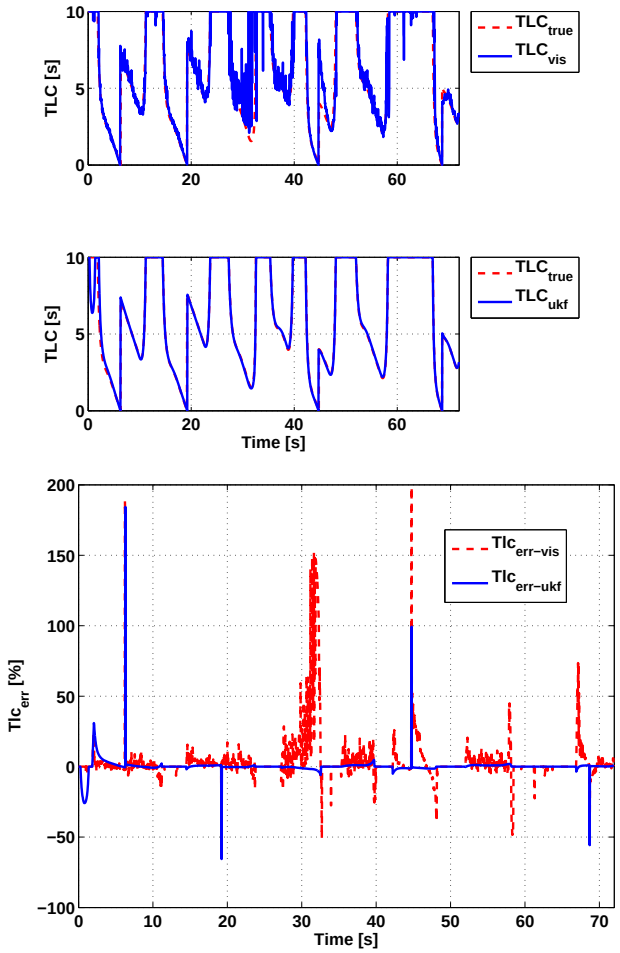

Figura 7.46. Tempo di invasione t_{tlc} UKF

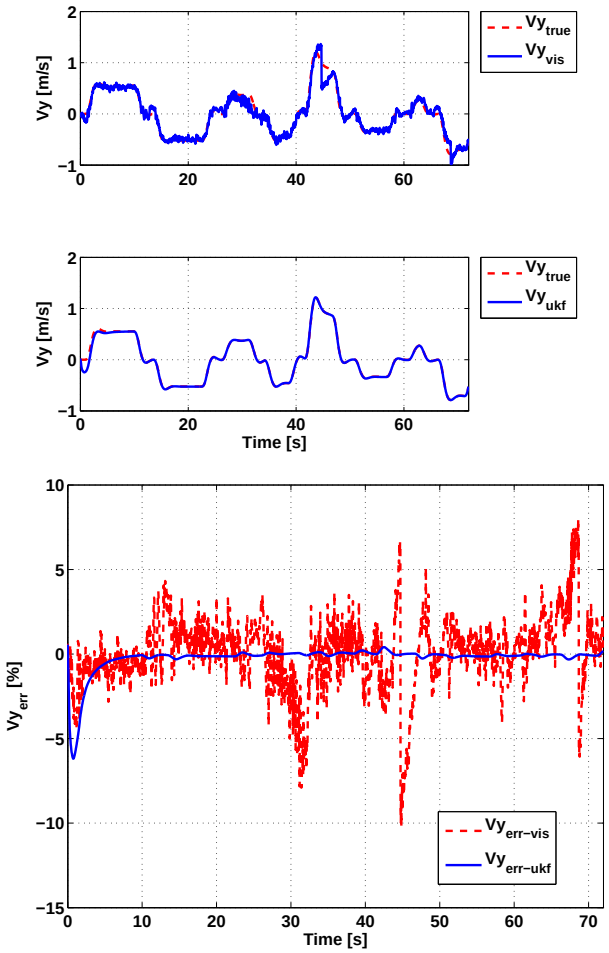

Figura 7.47. Velocità laterale UKF

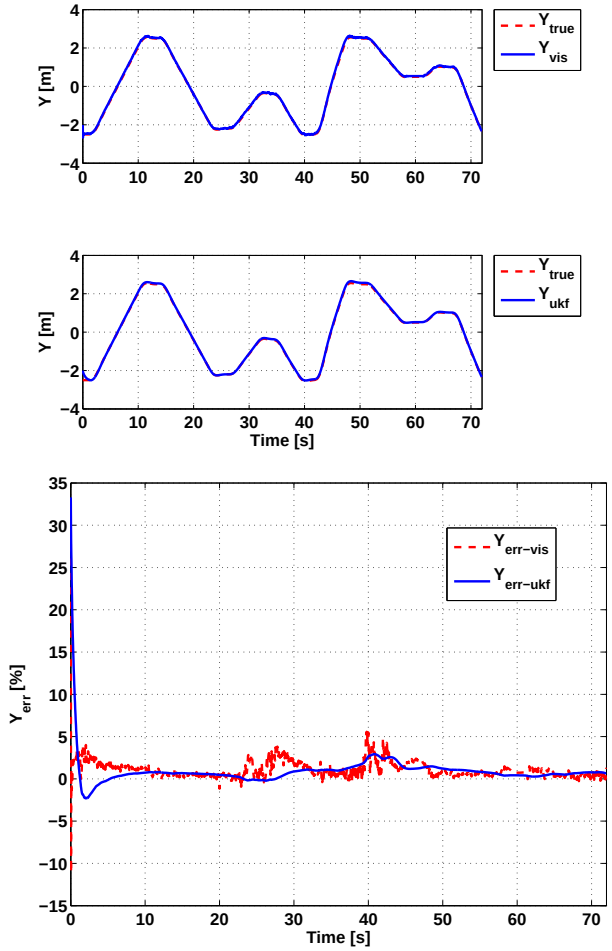

Figura 7.48. Posizione laterale UKF

La figura (7.46) descrive l'andamento del t_{tlc} utilizzando il filtro di Kalman Unscented. Anche in questo caso il rumore di misura, visibile nei dati da visione, viene attenuato. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso ma l'errore, al di fuori dei picchi in prossimità dei cambi di corsia, è minore del 4%.

La figura (7.47) descrive l'andamento della velocità laterale. Il filtro di Kalman Unscented elimina il rumore presente nei dati di visione e riesce a mantenere un errore di stima minore dello 0.3% durante tutta la simulazione.

Il filtro insegue con precisione anche la posizione laterale (figura (7.47)) in cui l'errore non supera mai il 3%.

Confronto tra EKF e UKF

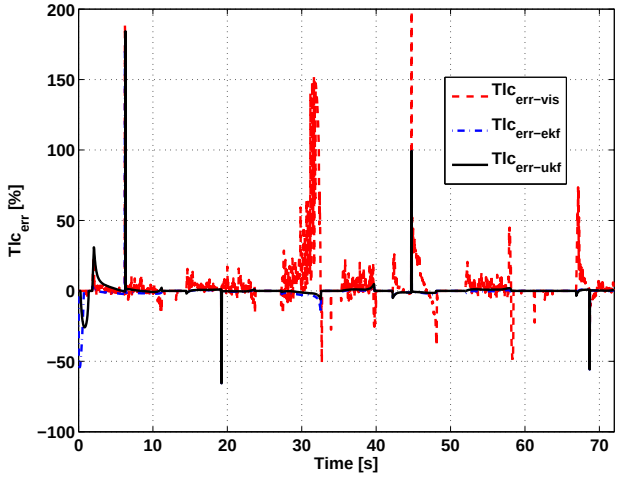

Figura 7.49. Errore sul t_{tlc}

Sia il filtro di Kalman Unscented che quello Esteso riducono il rumore presente nella simulazione con dati da sola visione, ottenendo un netto miglioramento nel calcolo del t_{tlc} (figura (7.49)).

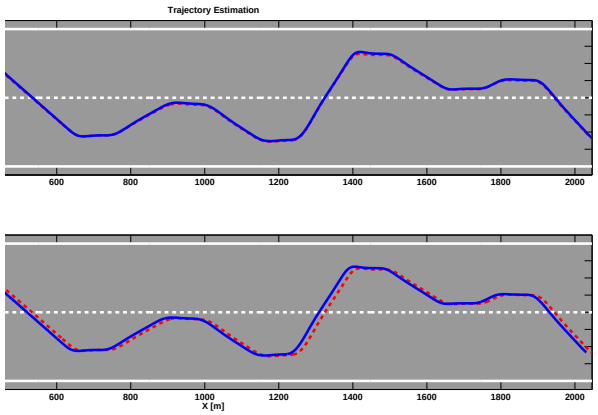

Figura 7.50. Traiettoria su strada

La traiettoria XY visibile in figura (7.50) mostra un buon inseguimento

per entrambi i filtri con un leggero vantaggio da parte del filtro di Kalman Esteso. Gli allarmi generati sono riportati in figura (7.51) ed evidenziano la corretta stima dei due filtri di Kalman.

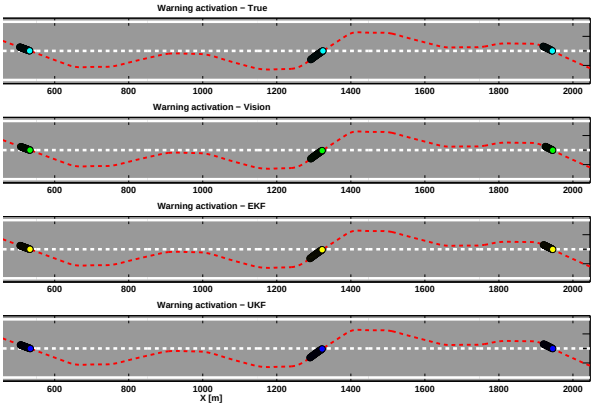

Figura 7.51. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.4 Modello 4: Dinamico

7.4.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

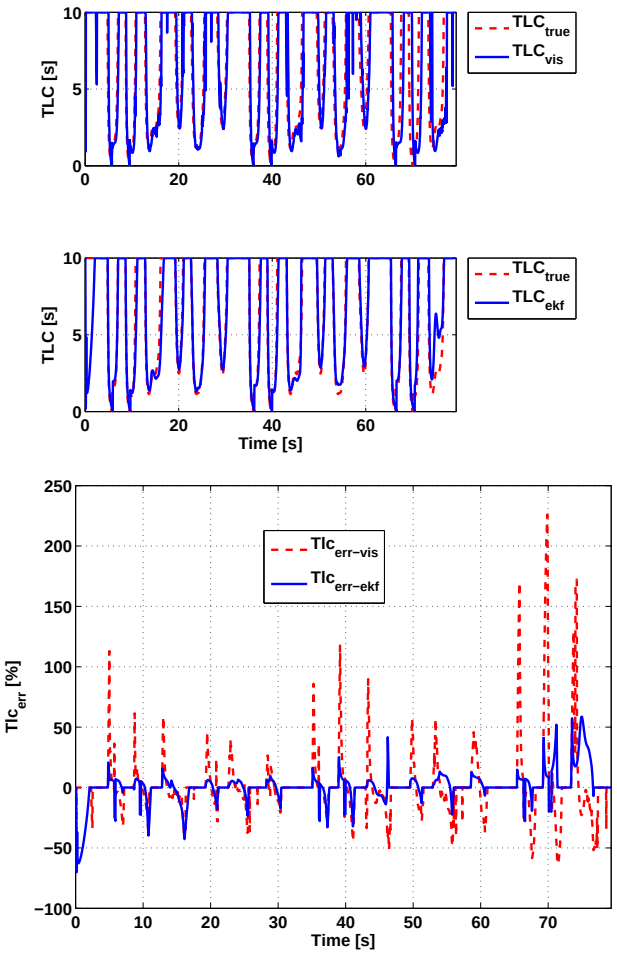

Figura 7.52. Tempo di invasione t_{ttc} EKF

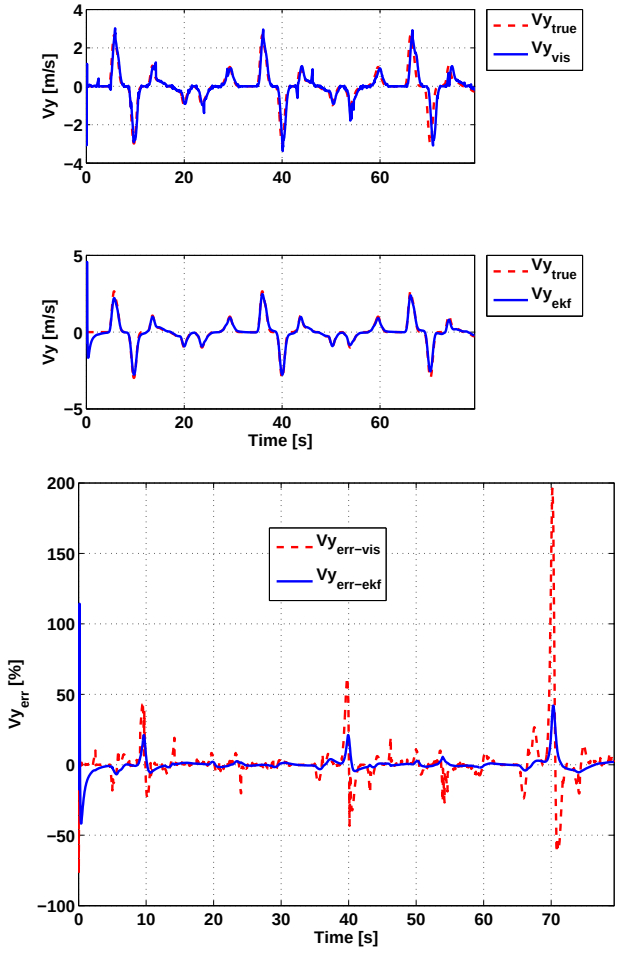

Figura 7.53. Velocità laterale EKF

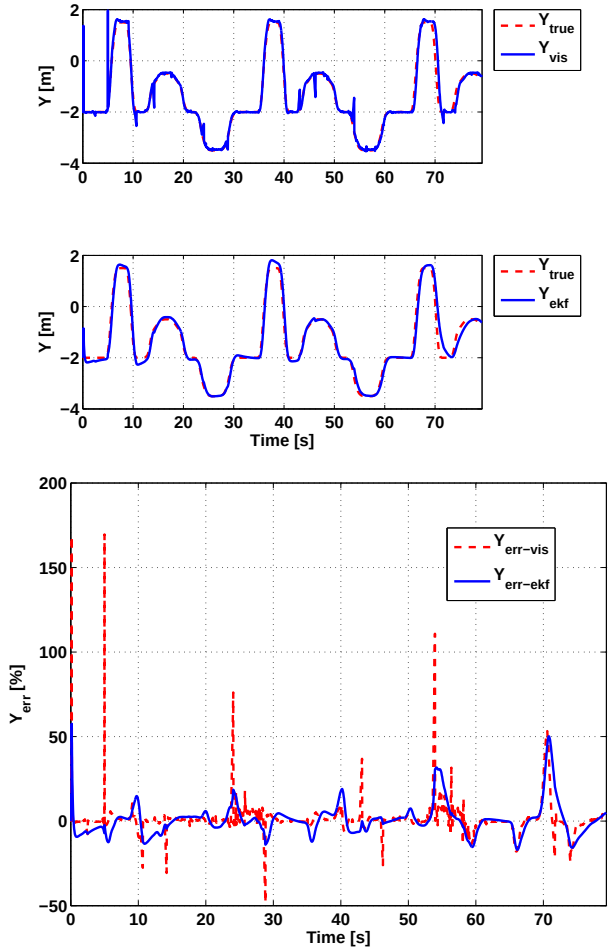

Figura 7.54. Posizione laterale EKF

Il t_{tlc} calcolato con filtro di Kalman Esteso e modello dinamico è visibile in figura (7.52). La simulazione effettuata a velocità costante di $50Km/h$ ha una durata di 80 secondi. Il t_{tlc} ottenuto con il filtro di Kalman Esteso presenta numerosi picchi di errore in corrispondenza dei cambi di corsia anche superiori al 50%. Tali picchi comunque non influenzano la corretta attivazione degli allarmi poichè avvengono quando la vettura ha già attraversato la linea di demarcazione della carreggiata. L'errore sul t_{tlc} tra picchi consecutivi è dell'ordine del $5 - 10\%$ e spesso si annulla del tutto.

Per quanto riguarda la velocità laterale (figura (7.53)), l'errore rimane inferiore al 5% tranne nei secondi 10, 40 e 70 di simulazione quando raggiunge il $20 - 30\%$ trascinato da un elevato errore nei dati provenienti dal sistema di

visione.

La stima della posizione laterale presenta un errore che si attesta intorno a ±20% come visibile in (figura (7.54)),con picchi che raggiungono anche il 50% nell'ultima parte della simulazione.

Simulazioni con filtro UKF

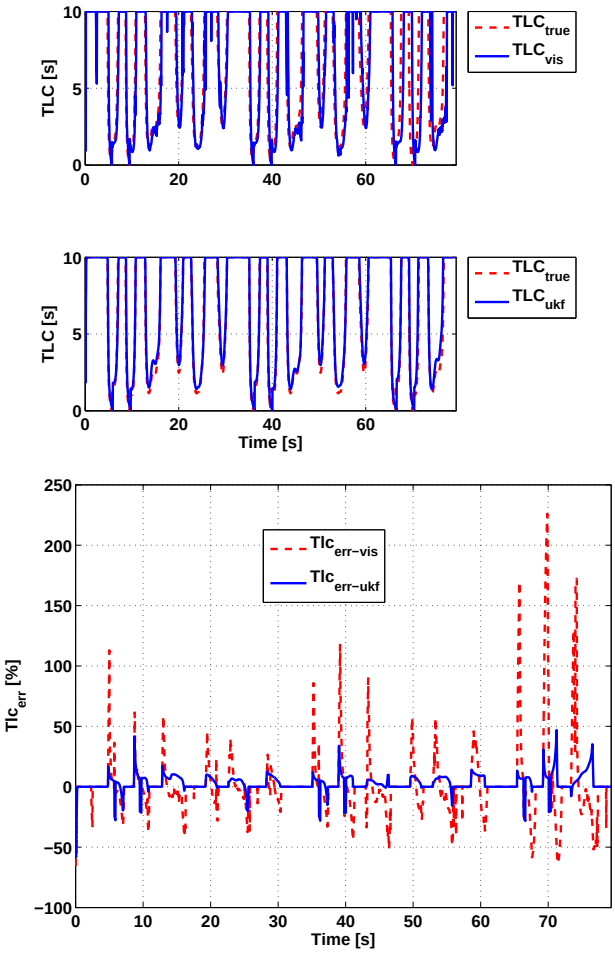

Figura 7.55. Tempo di invasione t_{tlc} UKF

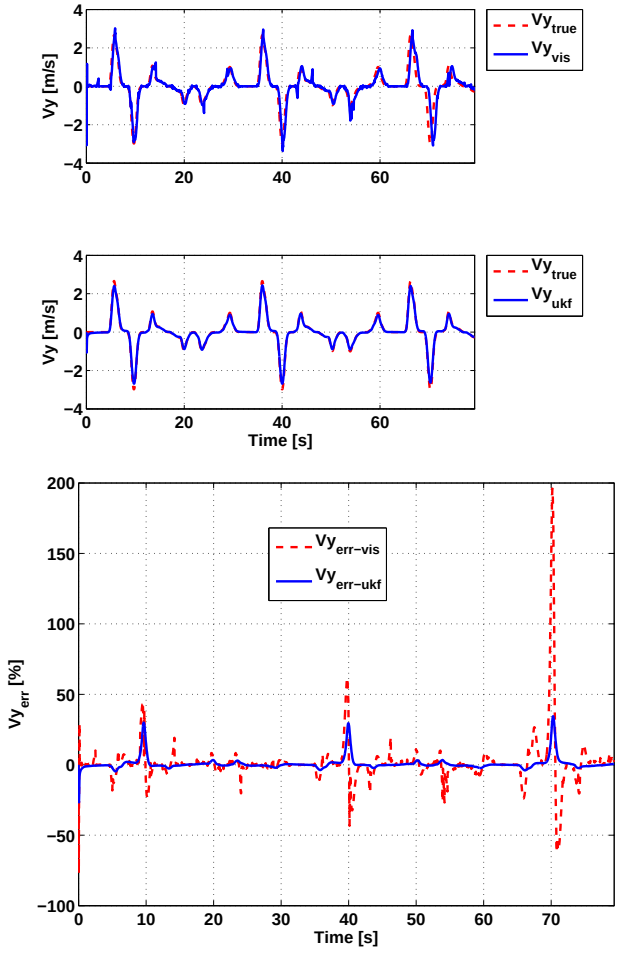

Figura 7.56. Velocità laterale UKF

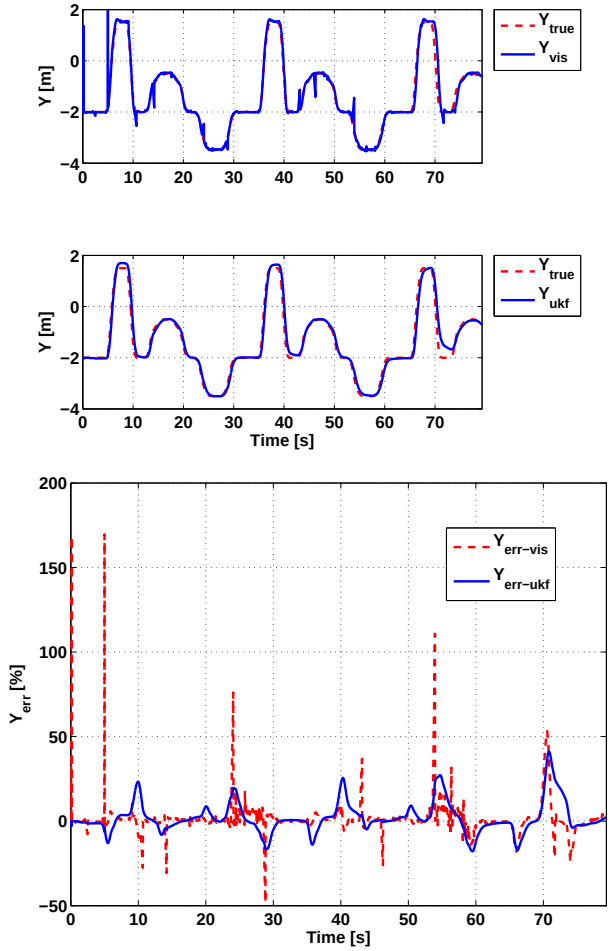

Figura 7.57. Posizione laterale UKF

La figura (7.55) mostra il t_{tlc} ottenuto con il filtro di Kalman Unscented. Come per il calcolo con filtro di Kalman Esteso, il t_{tlc} presenta una serie di picchi di errore durante tutta la simulazione nelle fasi in cui il veicolo effettua dei cambi di corsia. In questo caso però le ampiezze dei picchi sono ridotte e l'errore tra picchi successivi si annulla quasi del tutto.

In figura (7.56) è riportato l'andamento della velocità laterale. In particolare, quella stimata con il filtro di Kalman Unscented presenta un errore di stima minore del 4% per tutta la simulazione tranne nei secondi 10, 40 e 70 in cui raggiunge il 30%.

L'errore sulla posizione laterale (figura (7.57)) è dell'ordine del 20%, con picchi che raggiungono anche il 50% nell'ultima parte della simulazione.

Confronto tra EKF e UKF

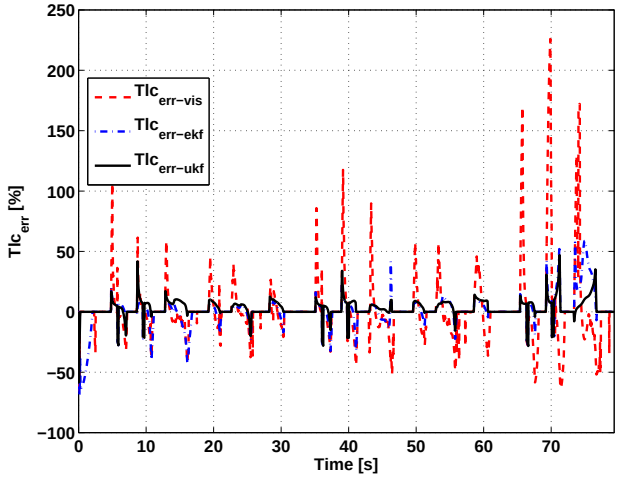

Figura 7.58. Errore sul t_{tlc}

Il confronto del t_{tlc} di figura (7.58) evidenzia il minor numero di picchi di errore nella stima effettuata con filtro di Kalman Unscented. Entrambi i filtri hanno comunque un comportamento migliore rispetto all'utilizzo dei dati da sola visione.

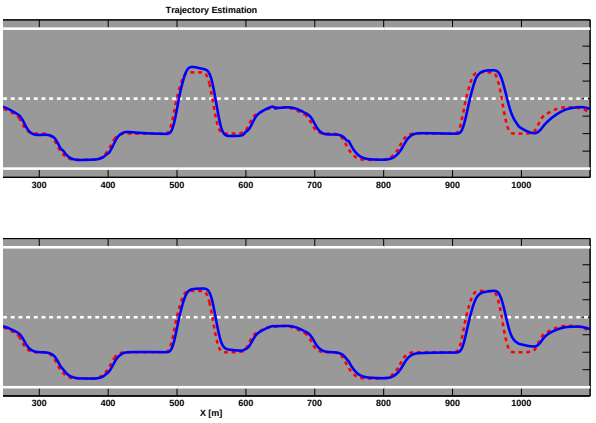

Figura 7.59. Traiettoria su strada

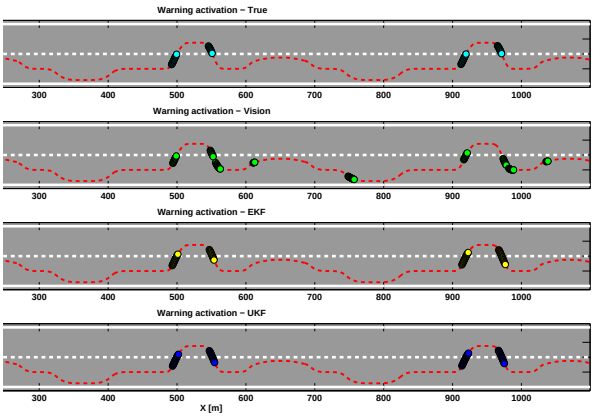

Figura 7.60. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

La differenza tra il filtro Esteso e quello Unscented non è apprezzabile osservando i grafici delle traiettorie su strada riportati in figura (7.59).

L'attivazione degli allarmi di figura (7.60) mostra una risposta identica da parte dei due approcci basati sui filtri di Kalman che non generano falsi allarmi. Al contrario, utilizzando esclusivamente i dati da visione, vengono generati falsi allarmi a 130, 190, 550, 610, 760, 990 e 1070 metri ed inoltre a 960 metri non viene rivelato un attraversamento di corsia.

7.4.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

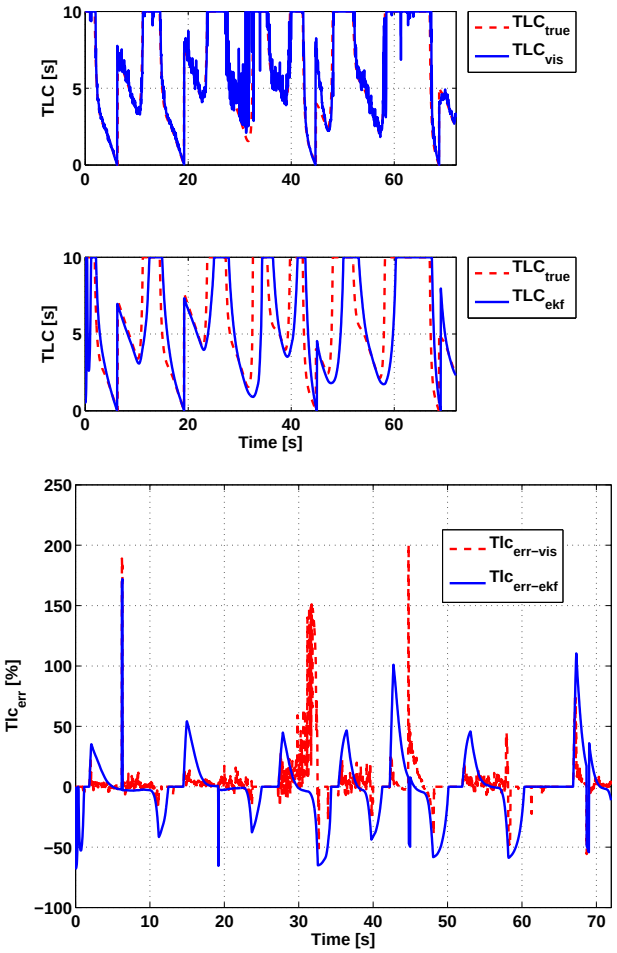

Figura 7.61. Tempo di invasione t_{tlc} EKF

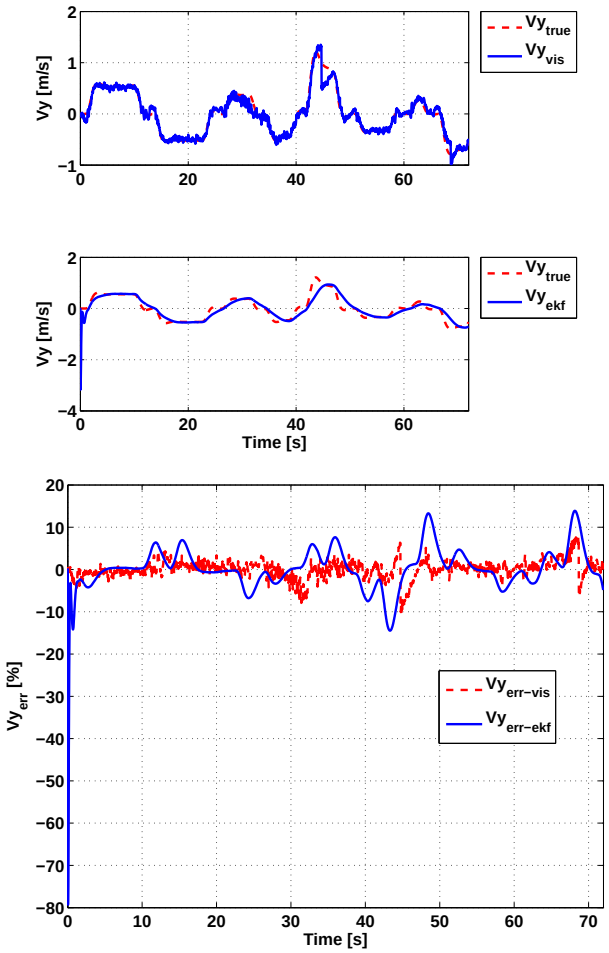

Figura 7.62. Velocità laterale EKF

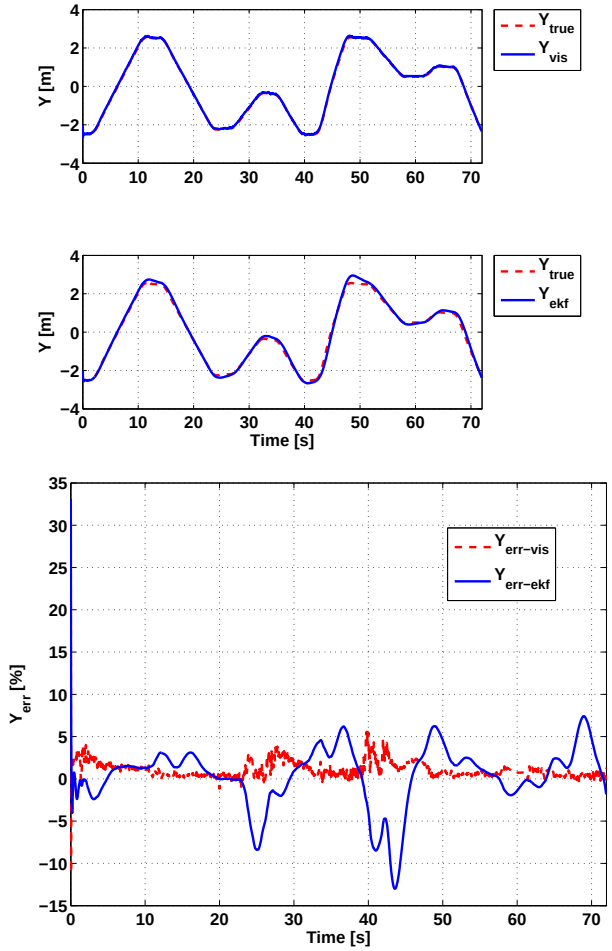

Figura 7.63. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h. La simulazione ha una durata di 72 secondi. Come è visibile in figura (7.61), il t_{tlc} calcolato direttamente dai dati da visione è affetto da elevato rumore. Il filtro di Kalman Esteso, nonostante riduca tale rumore, fornisce una stima del t_{tlc} errata che alterna brevi intervalli a errore nullo ad altri con errore fino al 100%.

Nel grafico relativo alla velocità laterale (V_y)(figura (7.62)), si può notare come l'effetto del filtraggio, pur riducendo la variabilità del segnale, introduca un errore a volte maggiore rispetto a quello ottenuto utilizzando direttamente i dati da visione. L'errore sulla velocità laterale rimane comunque inferiore al 15% per il filtro di Kalman esteso mentre per i dati da visione è minore del

10%.

La posizione laterale in figura (7.63) mostra un comportamento simile: l'errore generato dai dati da visione è sempre inferiore al 5% mentre quello generato dal filtro di Kalman esteso oscilla su valori superiori non superando comunque mai il 12%.

Simulazioni con filtro UKF

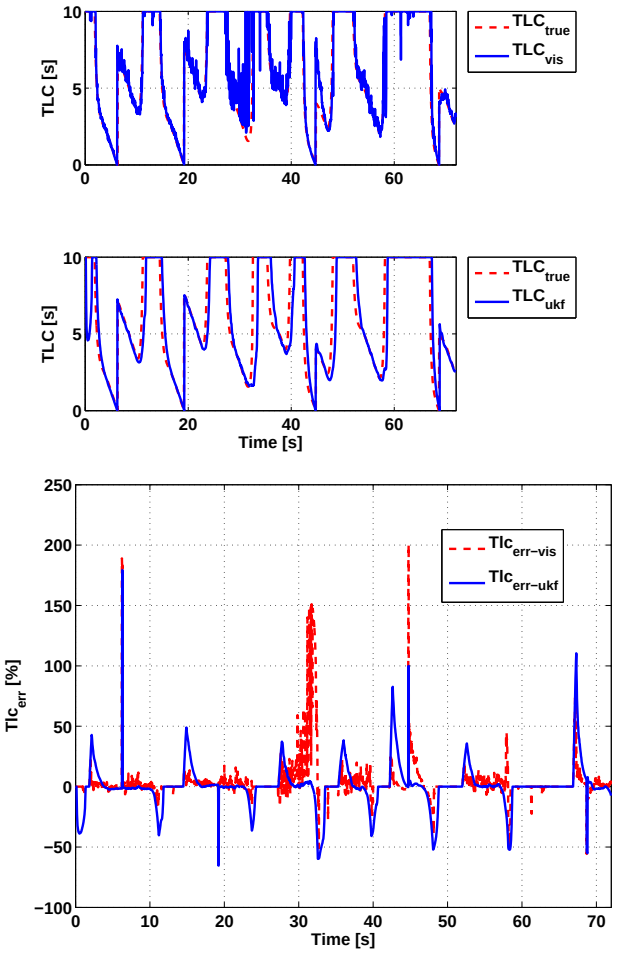

Figura 7.64. Tempo di invasione t_{tlc} UKF

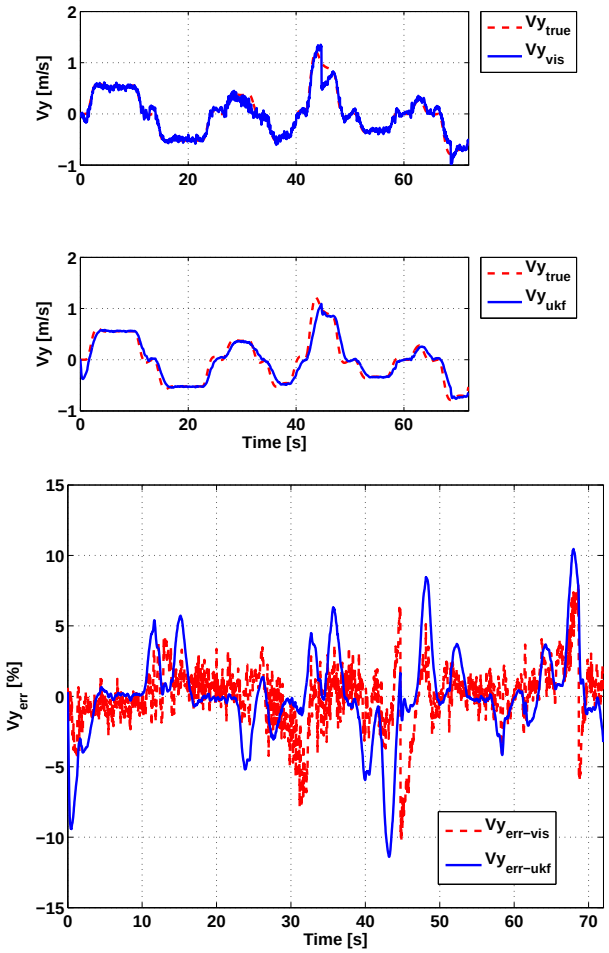

Figura 7.65. Velocità laterale UKF

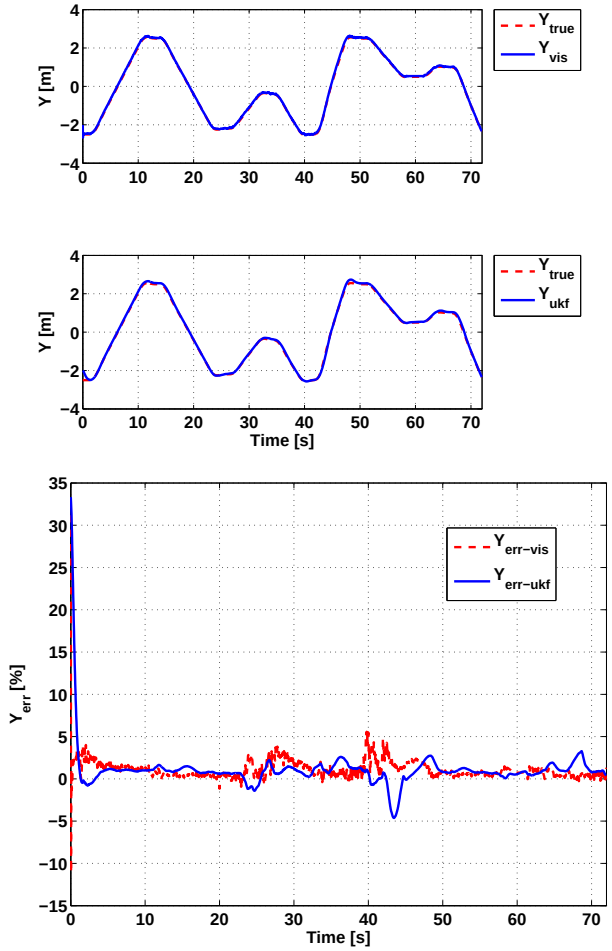

Figura 7.66. Posizione laterale UKF

La figura (7.64) descrive l'andamento del t_{tlc} utilizzando il filtro di Kalman Unscented. Anche in questo caso, come avvenuto con l'uso del filtro di Kalman Esteso, il rumore di misura visibile nei dati di visione viene attenuato. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso, con fasi alterne di rumore nullo e picchi fino al 50%.

La figura (7.65) descrive l'andamento della velocità laterale. In particolare è possibile notare un errore di stima del filtro che oscilla tra $pm10\%$.

Per quel che riguarda la posizione laterale (figura (7.66)), l'errore commesso dal filtro di Kalman Unscented è confrontabile con quello commesso dal sistema di visione: entrambi raggiungono il valore massimo del 5%.

Confronto tra EKF e UKF

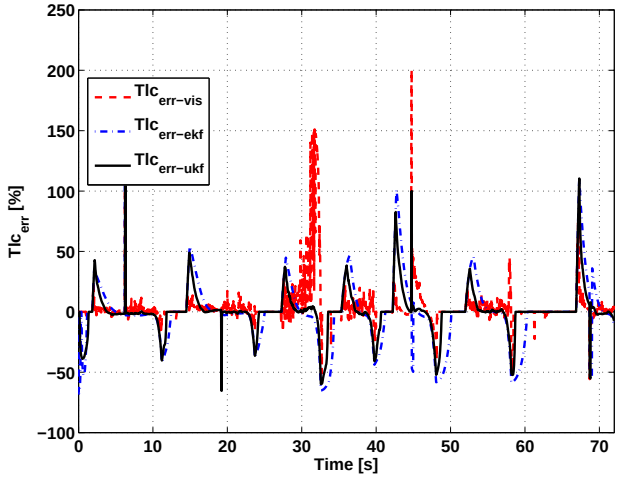

Figura 7.67. Errore sul t_{tlc}

La figura (7.67) riassume gli errori commessi nel calcolo del t_{tlc}. Sia il filtro di Kalman Unscented che quello Esteso riducono il rumore presente nei dati di sola visione e presentano picchi di errore durante tutta la simulazione. Dopo ogni picco, però, il filtro di Kalman Unscented riesce a convergere più velocemente al valore corretto rispetto al filtro di Kalman Esteso.

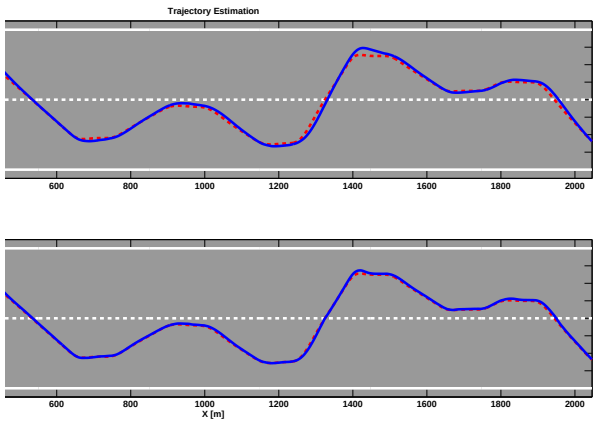

Figura 7.68. Traiettoria su strada

La traiettoria XY visibile in figura (7.68) mostra un buon inseguimento per entrambi i filtri con un leggero vantaggio in termini di precisione per il filtro Unscented.

Gli allarmi generati, riportati in figura (7.69), evidenziano un leggero ritardo nell'avviso da parte dei due filtri di Kalman rispetto alla segnalazione corretta. Per il filtro di Kalman Esteso tale comportamento è leggermente più accentuato. Inoltre nel suo tracciato è presente l'attivazione di un falso allarme a 900 metri. Le attivazioni ottenute dai dati da visione risultano corrette.

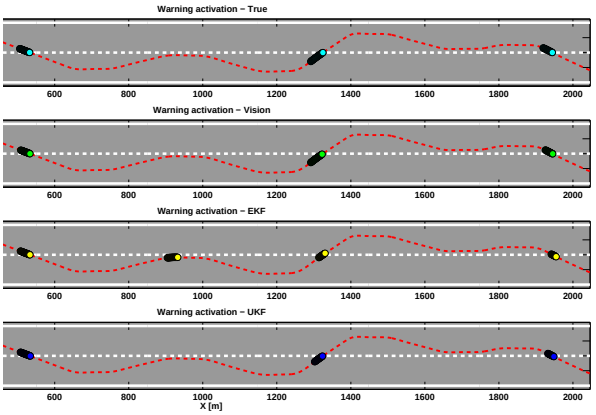

Figura 7.69. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

7.5 Confronti totali

Di seguito è riportato il confronto dei risultati ottenuti per ogni diversa manovra di test dal sistema di visione e dai filtri di Kalman Esteso e Unscented con i diversi modelli del veicolo analizzati nelle sezioni precedenti.

Il confronto è basato sulla radice dell'errore quadratico medio (RMSE Root Mean Squared Error). In particolare $RMSE$ è definito come

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i^*)^2}{n}} \tag{7.2}$$

dove x_i rappresenta il valore calcolato della variabile in esame (Posizione Laterale, Velocità Laterale o t_{tlc}), x_i^* il valore vero ed n il numero totale di campioni della simulazione.

Per ogni manovra di test, il migliore risultato ottenuto per la Posizione Laterale, la Velocità Laterale e per il t_{tlc} è evidenziato in tabella con celle di colore giallo. Il confronto permetterà di selezionare il filtro di Kalman ed il modello del veicolo che permettono di ottenere errori minori.

7.5.1 Percorso su rettilineo a velocità costante

- Modello 1: Cinematico sterzante

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.0311706e+01	6.0201648e+00	4.5458211e+00
Velocità Laterale (v_y)	2.7676446e+01	6.0776260e+00	5.2020850e+00
t_{tlc}	1.7479833e+01	3.4046528e+00	3.1193637e+00

- Modello 2: Cinematico differenziale

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.0311706e+01	6.0387117e+00	4.2058150e+00
Velocità Laterale (v_y)	2.7676446e+01	4.7948401e+00	3.4475369e+00
t_{tlc}	1.7479833e+01	1.7548702e+00	8.0716133e-01

- Modello 3: Cinematico yaw rate

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.0311706e+01	5.7625221e+00	5.6691766e+00
Velocità Laterale (v_y)	2.7674270e+01	6.5811894e+00	4.6142034e+00
t_{tlc}	1.7479759e+01	1.9790944e+00	1.6047037e+00

- Modello 4: Dinamico

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.0311706e+01	9.4971859e+00	9.4055652e+00
Velocità Laterale (v_y)	2.7676446e+01	1.3117265e+01	7.5457676e+00
t_{tlc}	1.7479833e+01	6.1010176e+00	4.6866930e+00

7.5.2 Percorso su rettilineo a velocità variabile

- Modello 1: Cinematico sterzante

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.5768858e+00	2.0449929e+00	1.9088078e+00
Velocità Laterale (v_y)	1.8965093e+01	1.3699709e+01	1.2433078e+01
t_{tlc}	2.1621196e+00	4.1876711e+00	2.5456756e+00

- Modello 2: Cinematico differenziale

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.5768858e+00	2.6238029e+00	1.8049911e+00
Velocità Laterale (v_y)	1.8965093e+01	9.4103898e+00	8.5802598e+00
t_{tlc}	2.1621196e+00	7.2055174e-01	5.0462225e-01

- Modello 3: Cinematico yaw rate

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.5768858e+00	1.4179472e+00	2.1038617e+00
Velocità Laterale (v_y)	1.8924640e+01	8.5372520e+00	6.4413686e+00
t_{tlc}	2.1586201e+00	1.1009729e+00	8.6297366e-01

- Modello 4: Dinamico

	RMSE Visione	**RMSE Stima EKF**	**RMSE Stima UKF**
Posizione Laterale (y)	1.5768858e+00	3.6523079e+00	2.6345199e+00
Velocità Laterale (v_y)	1.8965093e+01	2.6791170e+01	1.8393772e+01
t_{tlc}	2.1621196e+00	5.4430010e+00	3.1593268e+00

7.6 Conclusioni

É stato analizzato l'uso di algoritmi di fusione dei dati per sistemi di avviso di superamento di corsia attigua basati sul calcolo del tempo di invasione di corsia t_{tlc}. Sono stati implementati cinque diversi modelli di vettura. Ogni modello è stato testato sia con filtro di Kalman Unscented che con filtro di Kalman Esteso. I test sono stati condotti su due diverse simulazioni di guida, cercando di simulare il maggior numero di condizioni possibili anche quelle più estreme come, ad esempio, i cambi di corsia durante forti accelerazioni.

L'analisi dei modelli di vettura ha evidenziato quanto segue.

- Il modello cinematico sterzante ha dimostrato accettabili prestazioni nella stima durante cambi di corsia a velocità costante ma sono emerse notevoli difficoltà durante le simulazioni a velocità variabile sopratutto nella stima della velocità laterale.
- Il modello dinamico è il modello più complesso oggetto di studio. Nonostante le sue caratteristiche ha ottenuto scarsi risultati rispetto a tutti gli altri modelli sia nelle simulazioni a velocità costante che in quelle con brusche accelerazioni. Un tale risultato è da imputare a una conoscenza imperfetta dei parametri della vettura (Coefficienti di stiffness, slip angle, etc..) che porta ad errori maggiori nella stima della posizione e velocità laterale rispetto al semplice modello cinematico.
- Il modello cinematico differenziale ha dimostrato un'ottima capacità di stima in ogni situazione. Un'ulteriore vantaggio, oltre la sua semplicità, è l'uso di pochi dati della vettura. Sono necessari esclusivamente le velocità delle ruote dell'asse posteriore e parametri della vettura facilmente misurabili quale interasse e passo.
- Il modello cinematico yaw rate come il modello cinematico differenziale necessita di pochi dati della vettura: la velocità e lo yaw rate del veicolo. Tale soluzione ha ottenuto risultati comparabili con il modello cinematico differenziale con un leggero vantaggio in situazioni simulative a velocità variabile.

Per quanto riguarda gli algoritmi di fusione dei dati l'analisi ha evidenziato che la migliore stima della posizione e della velocità laterale è stata ottenuta con il Filtro di Kalman Unscendent (UKF) basato sul modello cinematico differenziale della vettura.

- Il filtro di Kalman Unscented in genere ottiene sempre errori percentuali inferiori rispetto al filtro di Kalman Esteso. A velocità costante l'errore medio su posizione (y) e velocità (V_y) è rispettivamente del $1 \div 2\%$ e 0.1%, con valori massimi di $10 \div 12\%$ e $3 \div 4\%$. In condizioni di forti variazioni di velocità invece l'errore medio aumenta a $4 \div 5\%$ per la posizione e $1.6 \div 3\%$ per la velocità, con valori massimi del 18% e del $1.5 \div 18\%$ rispettivamente.
- Per quanto riguarda il filtro di Kalman Esteso, nei test effettuati a velocità costante l'errore massimo sulla posizione è del $12 \div 15\%$, mentre quello

massimo sulla velocità è del 3%. Entrambi vengono ottenuti durante l'attraversamento della linea centrale. Durante la fase di guida all'interno della carreggiata l'errore sulla posizione è in media compreso tra il $3.5 \div 6\%$ mentre sulla velocità tra $0.25 \div 0.8\%$. Errori medi nell'ordine del $5 \div 10\%$ per la posizione laterale e del $3 - 4\%$ per la velocità sono stati ottenuti nelle simulazioni con forti variazioni di velocità (brusche frenate o forti accelerazioni). I valori massimi in queste condizioni sono del $10 \div 20\%$ e del $2 \div 20\%$ rispettivamente.

Un ulteriore conferma di questi risultati è fornita dai confronti effettuati sulla radice dell'errore quadratico medio (RMSE) calcolata per tutte le simulazioni effettuate.

8

Risultati con occlusioni

L'ambiente stradale in condizioni reali mette a dura prova l'acquisizione di immagini da telecamera. Sono possibili brevi accecamenti dovuti a bagliori o controluce, fino ad arrivare a qualche secondo di completa occlusione dovuta a grossi automezzi che precedono il veicolo o a brevi tratti di strada priva di segnaletica orizzontale.

Le successive simulazioni mirano ad evidenziare la robustezza dei due migliori algoritmi di data fusion precedentemente selezionati a temporanea assenza di dati provenienti dal sistema di visione. Per ogni simulazione sono previsti quattro intervalli di diversa durata in modo da considerare il maggior numero possibile di scenari. Per le due simulazioni effettuate le occlusioni sono mostrate in tabella 8.1.

Occlusione	Durata $[sec]$	Intervallo di simulazione $[sec]$
1	0,5	10 ÷ 10,5
2	2	14 ÷ 16
3	6	25 ÷ 31
4	15	45 ÷ 60

Tabella 8.1. Occlusioni Simulazioni 1, 2

8.1 Modello 2: Cinematico Differenziale

8.1.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

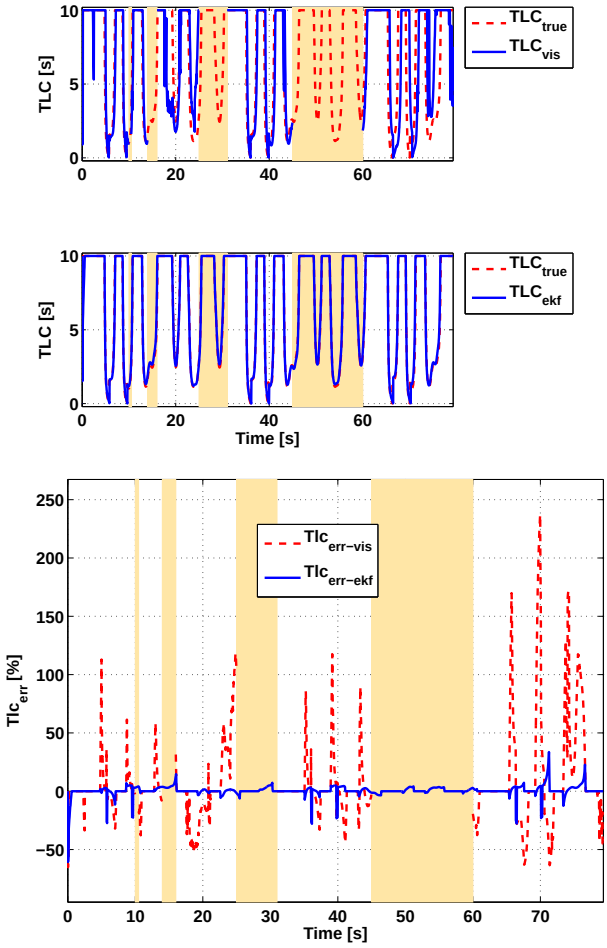

Figura 8.1. Tempo di invasione t_{tlc} EKF

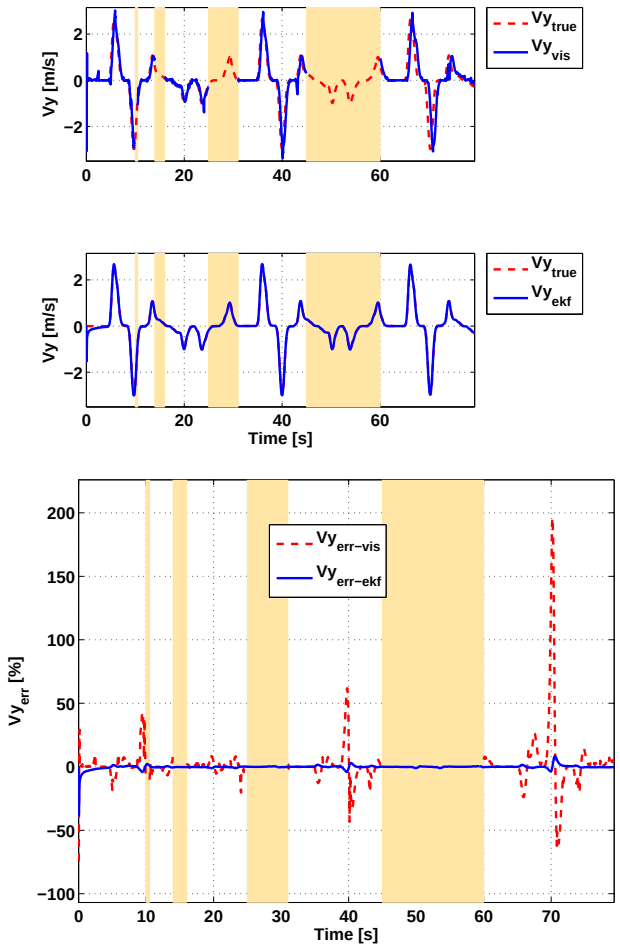

Figura 8.2. Velocità laterale EKF

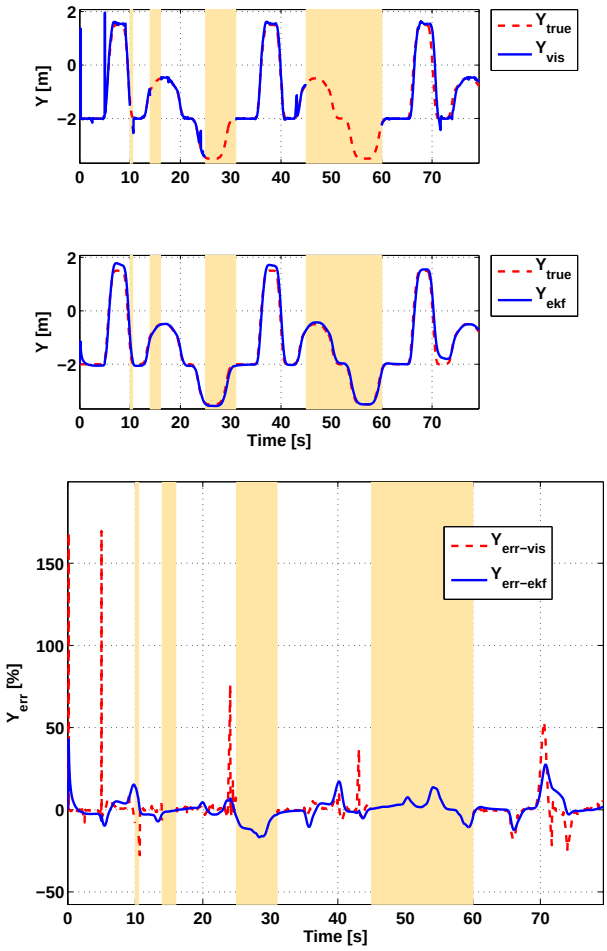

Figura 8.3. Posizione laterale EKF

Il t_{tlc} calcolato con filtro di Kalman Esteso e modello cinematico differenziale è visibile in figura (8.1). La simulazione, effettuata a velocità costante di $50Km/h$, ha una durata di 80 secondi.

Il t_{tlc} stimato con filtro di Kalman Esteso presenta dei picchi di errore solo in corrispondenza dei 6 cambi di corsia. Per tutto il resto della simulazione, anche durante gli intervalli di occlusione della telecamera, l'errore rimane inferiore al 5%.

Per quanto riguarda la velocità laterale (figura (8.2)), l'errore rimane inferiore all'1%, anche durante le occlusioni. In corrispondenza di 10, 40 e 70 secondi di simulazione sono presenti delle brevi escursioni nell'errore fino 4%. La stima della posizione, invece, presenta un errore leggermente più alto che

si attesta intorno a ±15% come visibile in figura (8.3).

Simulazioni con filtro UKF

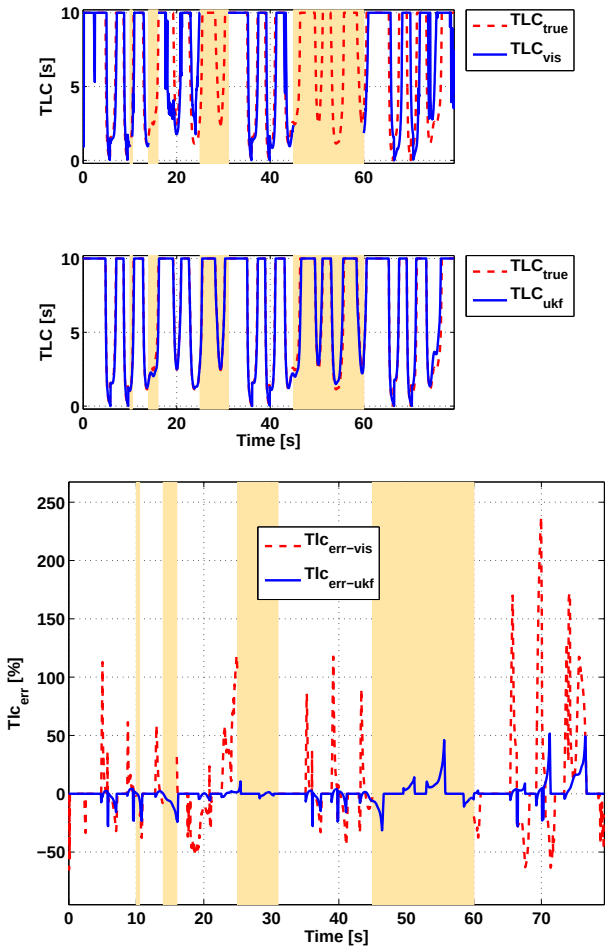

Figura 8.4. Tempo di invasione t_{tlc} UKF

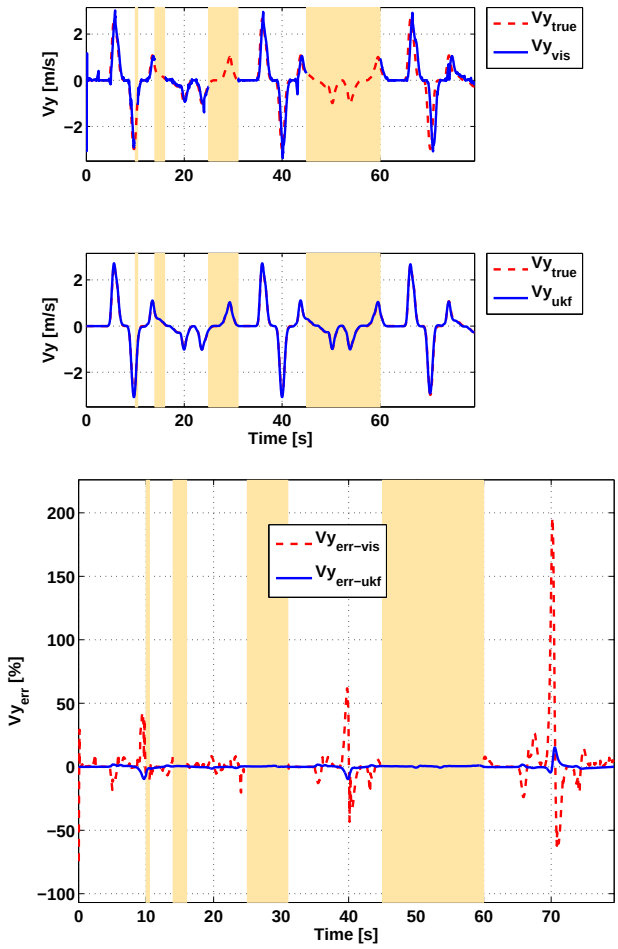

Figura 8.5. Velocità laterale UKF

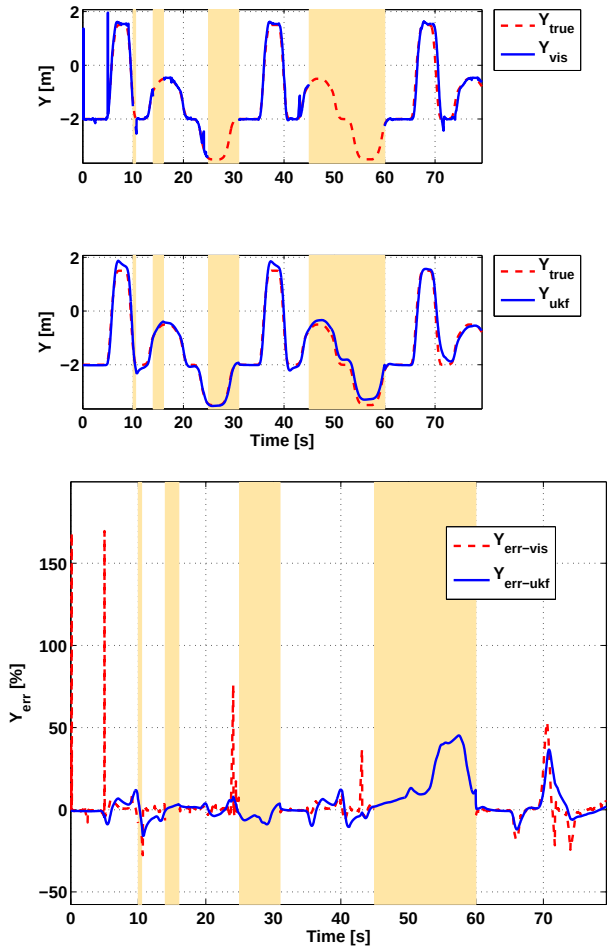

Figura 8.6. Posizione laterale UKF

La figura (8.4) mostra il t_{tlc} ottenuto con il filtro di Kalman Unscented e occlusioni nei dati del sistema di visione. Il filtro stima il t_{tlc} con frequenti picchi di errore anche durante gli intervalli di occlusione. In particolare durante il primo e il secondo intervallo di occlusione si raggiunge il 20% di errore, durante il quarto si arriva anche al 50%.

In figura (8.5) è riportato il grafico della velocità laterale. In particolare quella stimata con il filtro di Kalman Unscented presenta un errore minore dell'1% tranne a 10, 40 e 70 secondi in cui ha delle escursioni fino al 4%. Il verificarsi delle occlusioni non influenza la precisione della stima.

Per quel che riguarda la posizione laterale (figura (7.24)), quella stimata presenta un errore intorno al 10% nella prima metà della simulazione. Nella

seconda metà, in particolare all'interno dell'ultima occlusione, l'errore aumenta fino ad arrivare al 50%, per poi ritornare a valori più bassi quando i dati di visione sono nuovamente disponibili.

Confronto tra EKF e UKF

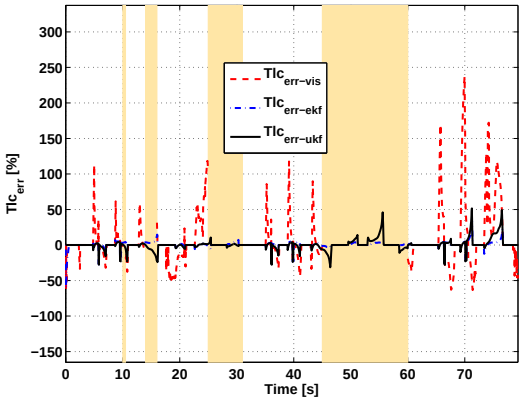

Figura 8.7. Errore sul t_{tlc}

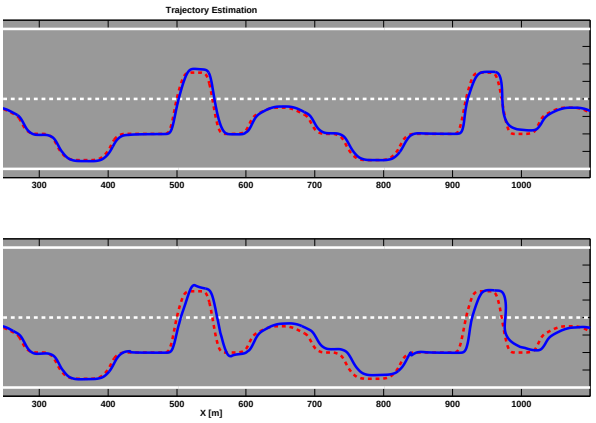

Figura 8.8. Traiettoria su strada

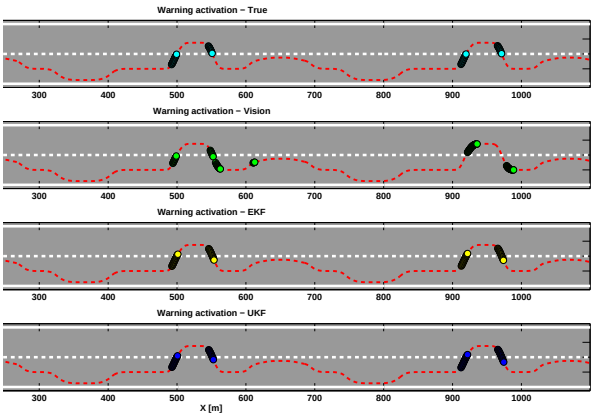

Figura 8.9. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

Il confronto sul t_{tlc} di figura (8.7) mette in evidenza il vantaggio dell'uso dei filtri di Kalman rispetto ai soli dati da visione. I due filtri di Kalman Unscented e Esteso hanno un comportamento molto simile e per buona parte della simulazione l'errore commesso coincide.

Il leggero vantaggio del filtro di Kalman Esteso è apprezzabile dalla figura (8.8) dove la stima della traiettoria è più precisa.

I due filtri riescono entrambi ad eliminare tutti i falsi allarmi, mentre le attivazioni ottenute con il t_{tlc} calcolato dai dati da visione non sono altrettanto precise (figura (8.9)). In particolare con i dati di visione abbiamo falsi allarmi a 130, 190, 550, 610 e 990 metri. Inoltre a 960 metri non viene rilevato un attraversamento di corsia.

8.1.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

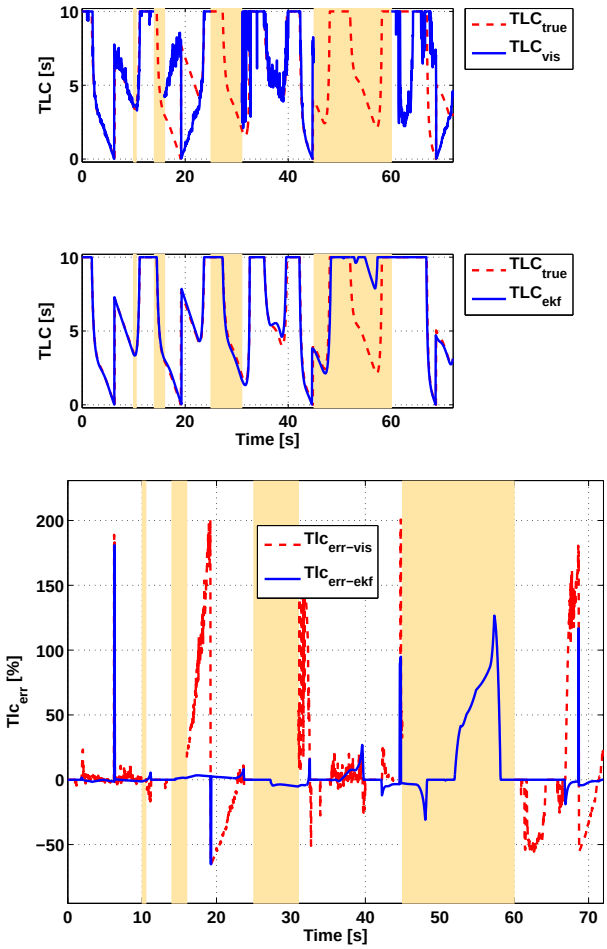

Figura 8.10. Tempo di invasione t_{tlc} EKF

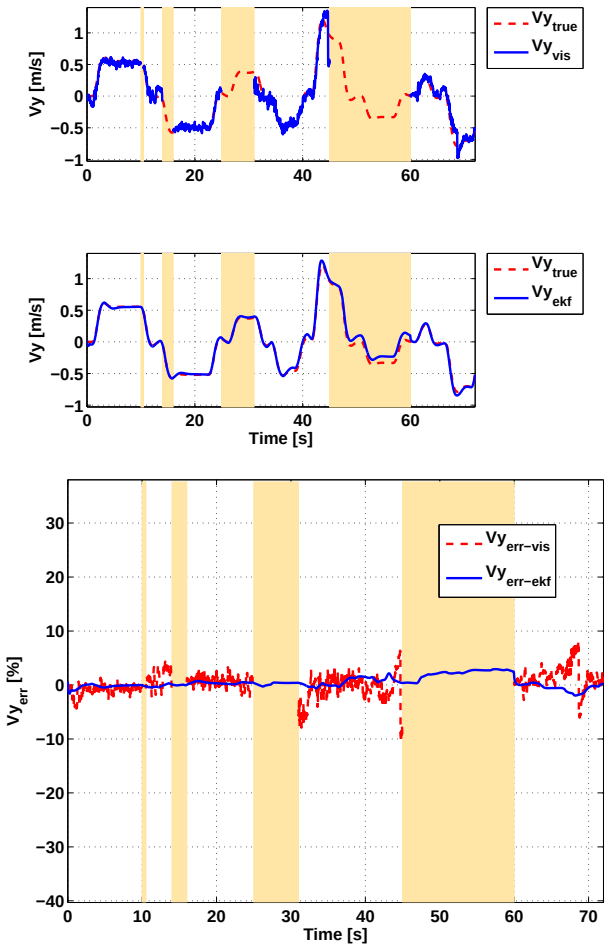

Figura 8.11. Velocità laterale EKF

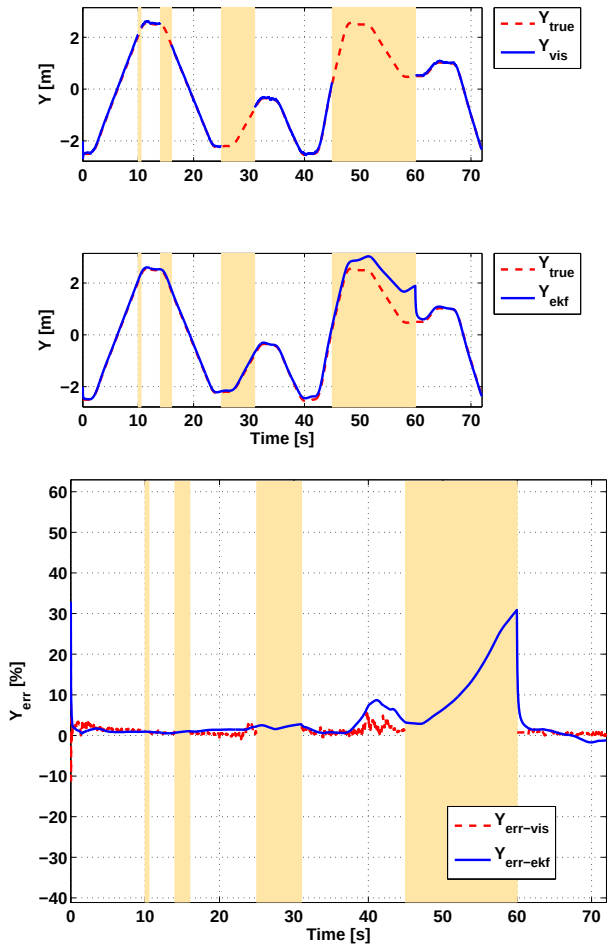

Figura 8.12. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h con brusche frenate e accelerazioni in corrispondenza dei cambi di corsia. La simulazione ha una durata di 72 secondi. Come è visibile in figura (8.10) il t_{tlc} dai dati di visione non è disponibile durante i periodi di occlusione e nei rimanenti è estremamente rumoroso.

Durante i primi tre intervalli di occlusione la stima del t_{tlc} non risente dell'assenza dei dati da visione. Infatti l'errore si mantiene al di sotto del 5% tranne che in corrispondenza dei cambi di corsia in cui sono presenti i consueti picchi di errore. Durante l'ultima occlusione invece, in particolare a partire dal 50-esimo secondo, l'errore aumenta fino a superare il 100% per poi ritornare

a zero al termine dell'occlusione.

Il buon comportamento del filtro di Kalman Esteso è evidente nel grafico relativo alla velocità laterale (V_y)(figura (8.11)). Il filtro elimina completamente l'effetto del rumore introdotto dai dati di visione con un errore di stima inferiore al 3% anche all'interno delle occlusioni.

Maggiori difficoltà sono evidenti nella stima nella posizione laterale (figura (8.12)).In particolare nei primi 50 secondi di simulazione l'errore si mantiene al di sotto del 7% mentre, durante l'ultima occlusione, aumenta progressivamente fino ad arrivare al 31% per poi scendere bruscamente ai valori precedenti. Il maggior contributo all'errore sul t_{tlc} durante l'ultima occlusione è quindi dovuto alla stima della posizione laterale.

Simulazioni con filtro UKF

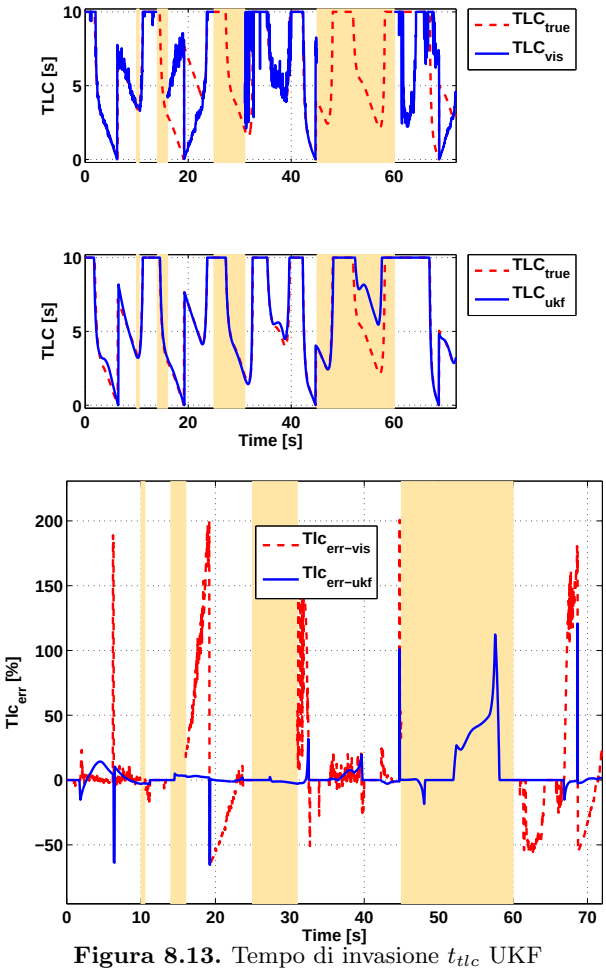

Figura 8.13. Tempo di invasione t_{tlc} UKF

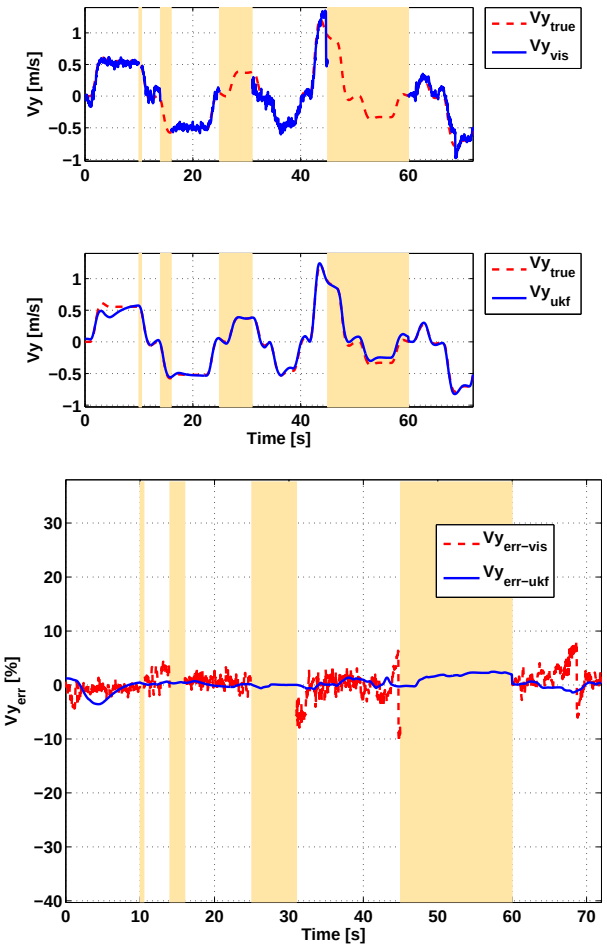

Figura 8.14. Velocità laterale UKF

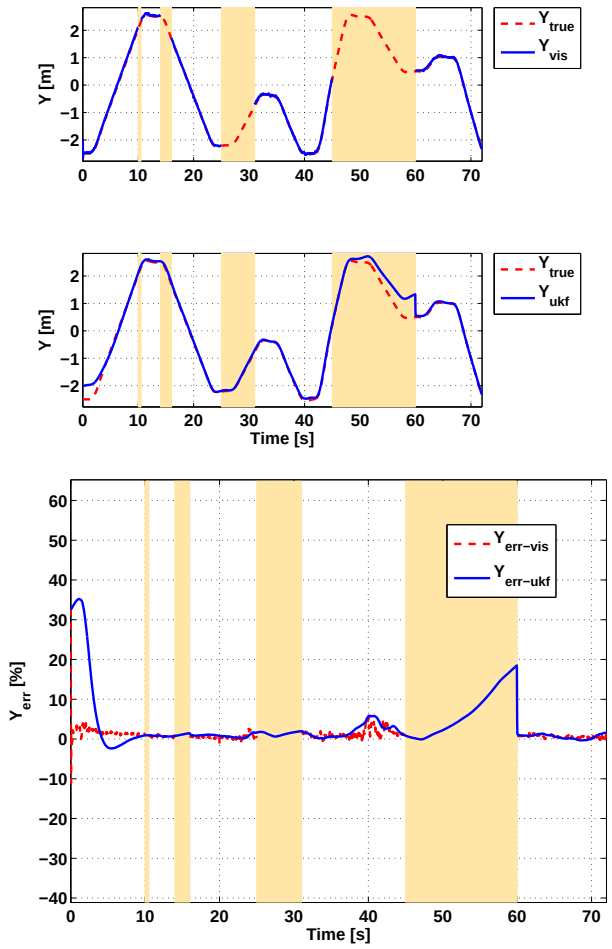

Figura 8.15. Posizione laterale UKF

La figura (8.13) mostra l'andamento del t_{tlc}. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso ma durante l'ultima occlusione riesce a mantenere per un periodo più lungo l'errore al di sotto del 50%.

La figura (8.14) mostra l'andamento della velocità laterale. In particolare l'errore di stima si mantiene al di sotto del 2.5% durante tutta la simulazione, anche negli intervalli di occlusione.

Il filtro di Kalman Unscented riesce a inseguire con precisione anche la posizione laterale (figura (7.31)). In particolare nei primi 50 secondi di simulazione l'errore massimo è del 7%, mentre in corrispondenza dell'ultima occlusione aumenta fino al 20% per poi ritornare ai valori precedenti.

Confronto tra EKF e UKF

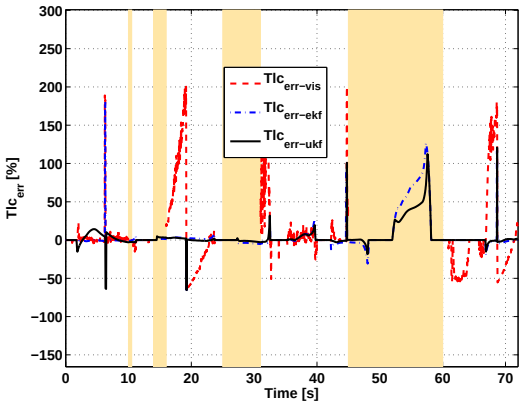

Figura 8.16. Errore sul t_{tlc}

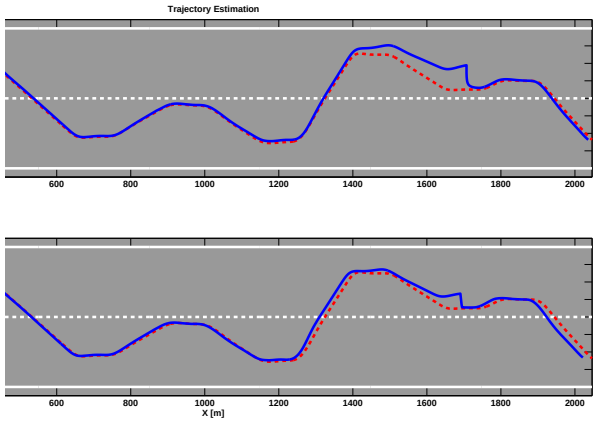

Figura 8.17. Traiettoria su strada

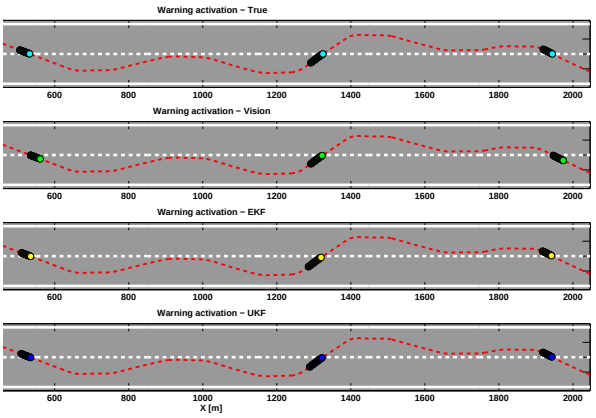

Figura 8.18. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

La figura (8.16) riassume gli errori commessi nel calcolo del t_{tlc}. Sia il filtro di Kalman Unscented che quello Esteso restituiscono un valore di t_{tlc} con un errore di stima dell'ordine del 5% tranne che in corrispondenza dei cambi di corsia e nell'ultima occlusione, in cui il filtro Unscented mostra un leggero vantaggio rispetto al filtro di Esteso.

La traiettoria XY visibile in figura (8.17) mostra un buon inseguimento per entrambi i filtri ma anche in questo caso è evidente il miglior comportamento del filtro Unscented durante l'ultimo periodo di occlusione.

Gli allarmi , riportati in figura (7.33), vengono attivati correttamente dai due filtri di Kalman e dal sistema di visione poiché le occlusioni non si verificano durante gli intervalli di attraversamento di corsia.

8.2 Modello 3: Cinematico yaw rate

8.2.1 Percorso su rettilineo a velocità costante

Simulazioni con filtro EKF

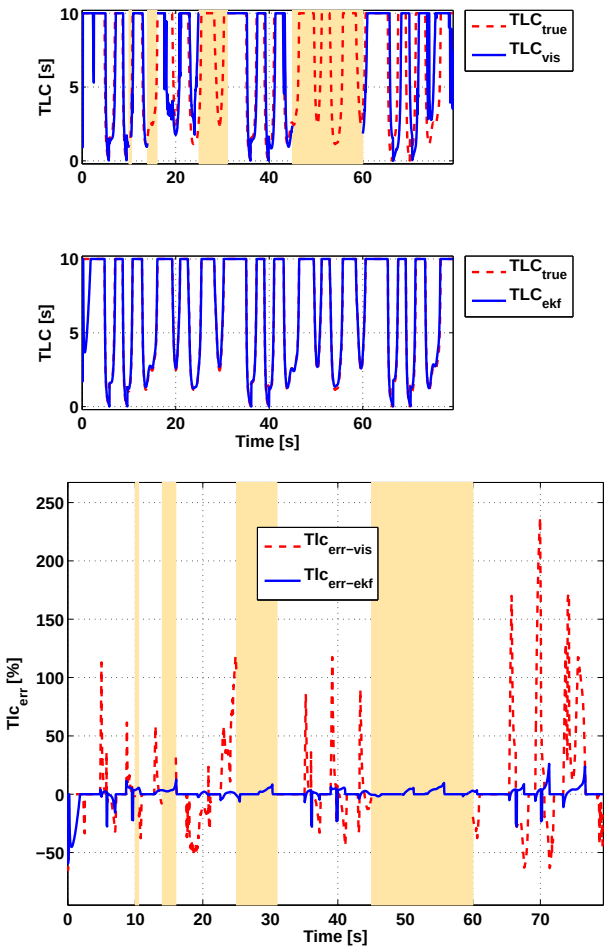

Figura 8.19. Tempo di invasione t_{tlc} EKF

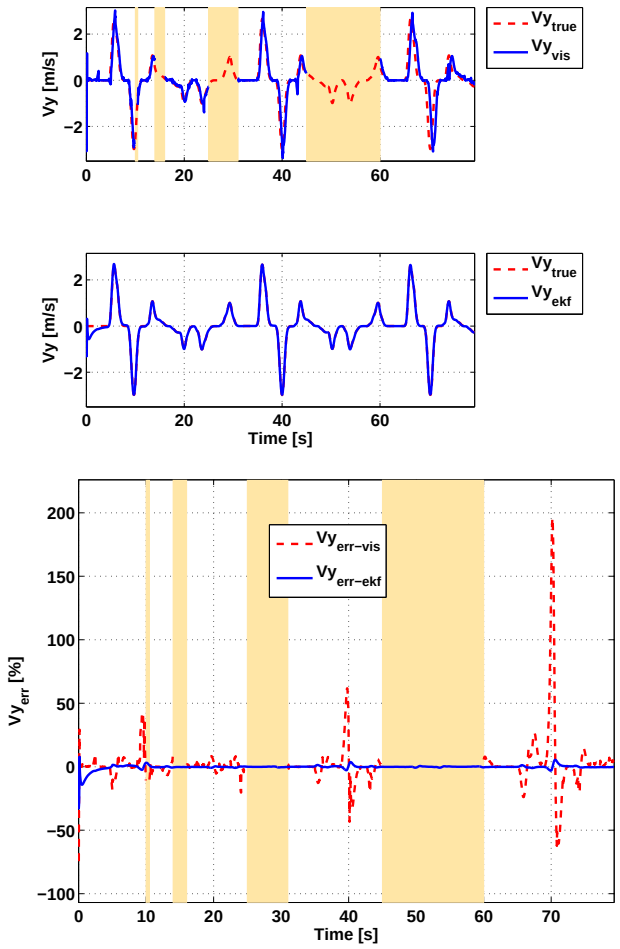

Figura 8.20. Velocità laterale EKF

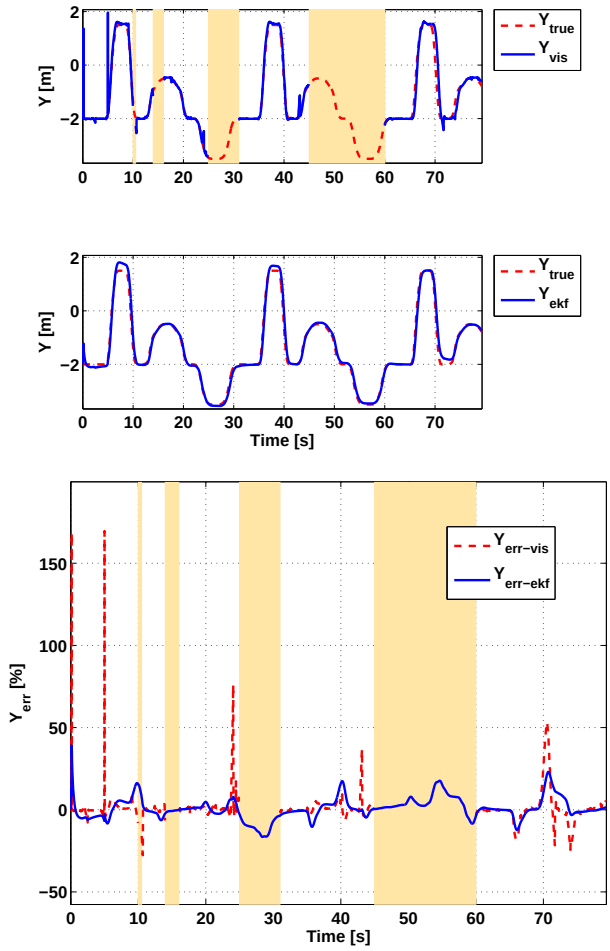

Figura 8.21. Posizione laterale EKF

La figura (8.19)mostra il t_{tlc} calcolato con filtro di Kalman Esteso e modello cinematico differenziale. La simulazione, effettuata a velocità costante di $50Km/h$, ha una durata di 80 secondi.

Il t_{tlc} stimato con filtro di Kalman Esteso presenta dei picchi di errore solo in corrispondenza dei 6 cambi di corsia, nel resto della simulazione, anche durante gli intervalli di occlusione della telecamera, l'errore rimane inferiore al 7%. Sono inoltre evidenti lunghi tratti a errore nullo.

Per quanto riguarda la velocità laterale (figura (8.20)), l'errore rimane inferiore all'1%, anche durante le occlusioni. In corrispondenza di 10, 40 e 70 secondi di simulazione sono presenti delle brevi escursioni nell'errore fino 4%. La stima della posizione, invece, presenta un errore leggermente più alto che

arriva al massimo a $\pm 15\%$ come visibile in figura. (8.21).

Simulazioni con filtro UKF

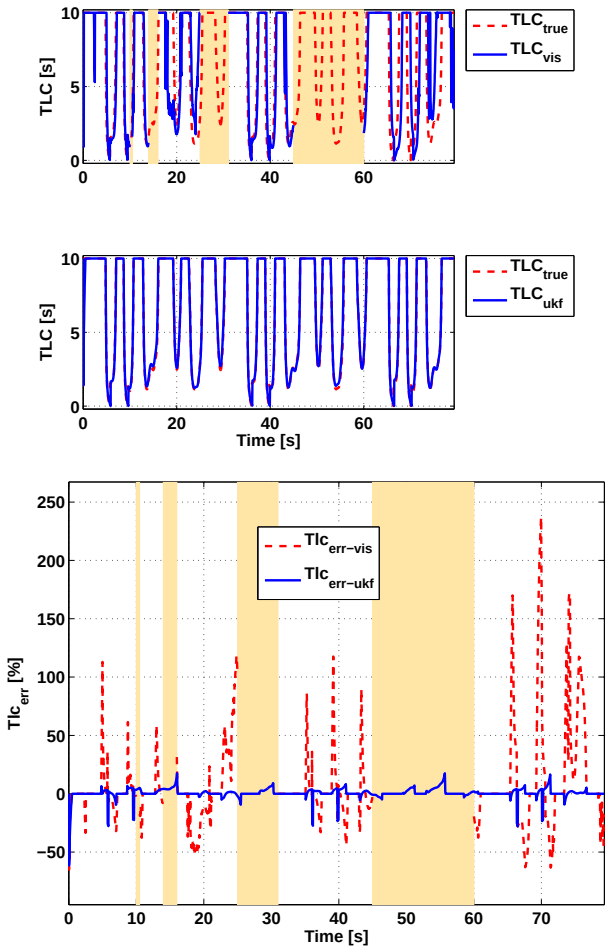

Figura 8.22. Tempo di invasione t_{tlc} UKF

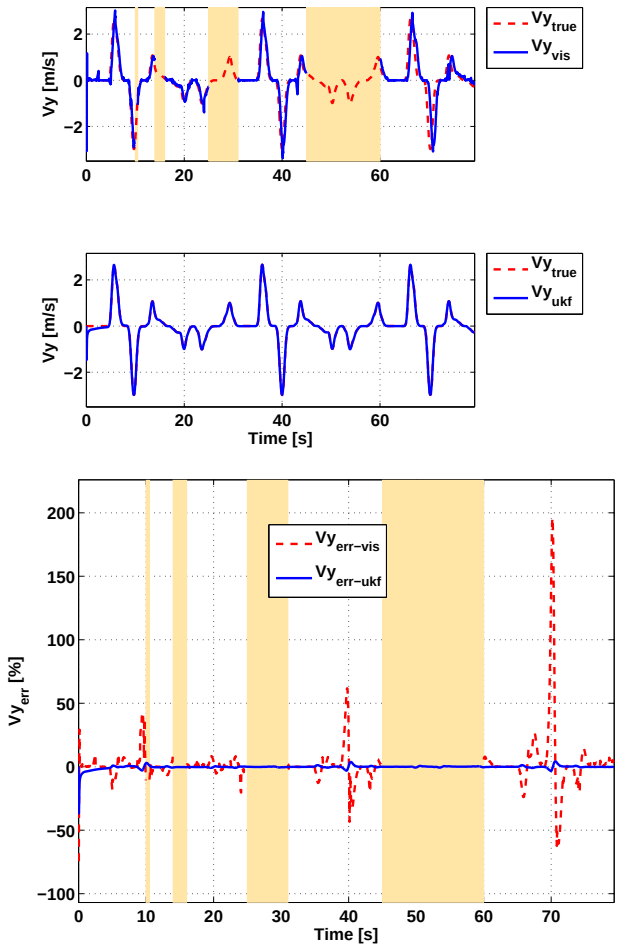

Figura 8.23. Velocità laterale UKF

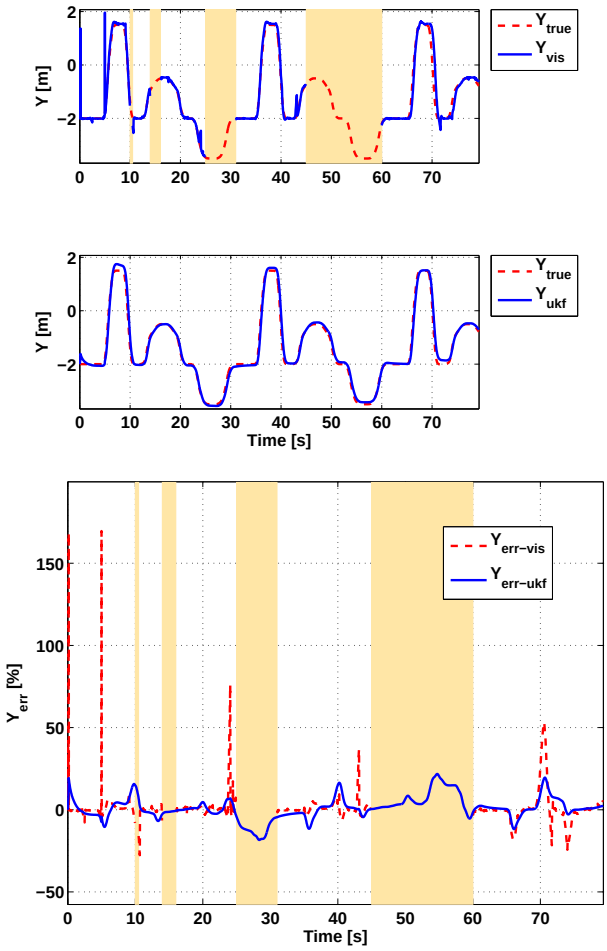

Figura 8.24. Posizione laterale UKF

I risultati sul t_{tlc} ottenuti con filtro di Kalman Unscented e occlusioni nei dati del sistema di visione sono descritti in figura (8.22). L'andamento del grafico è del tutto simile a quello ottenuto con filtro Esteso per cui valgono le stesse considerazioni.

Il grafico della velocità laterale è riportato in figura (8.23) e presenta un errore minore dell'0.6% le escursioni visibili ai secondi 10, 40 e 70 raggiungono valori fino al 4%.

Per quel che riguarda la posizione laterale (figura (7.24)), quella stimata presenta un errore intorno al 15% e solo durante la seconda e terza occlusione raggiunge al massimo il 20%.

Confronto tra EKF e UKF

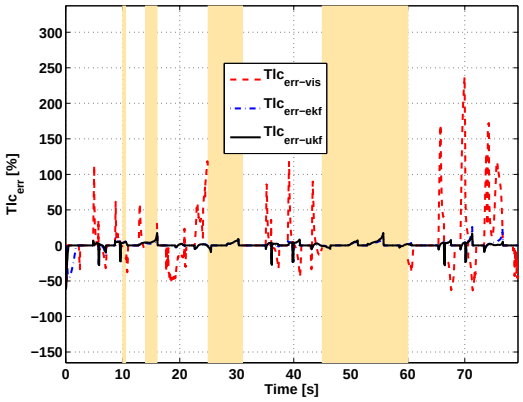

Figura 8.25. Errore sul t_{tlc}

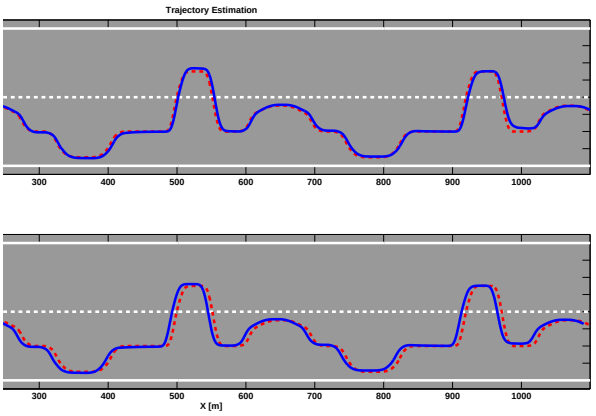

Figura 8.26. Traiettoria su strada

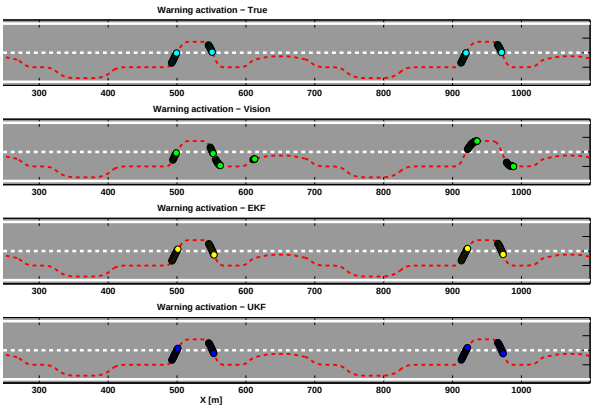

Figura 8.27. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

Il confronto sul t_{tlc} di figura (8.25) mette in evidenza il vantaggio dell'uso dei filtri di Kalman rispetto ai soli dati da visione. I due filtri di Kalman Unscented e Esteso hanno un comportamento molto simile e per buona parte della simulazione l'errore commesso coincide.

Il leggero vantaggio del filtro di Kalman Esteso è apprezzabile dalla figura (8.26) dove la stima della traiettoria è più precisa.

I due filtri riescono entrambi ad eliminare tutti i falsi allarmi, mentre le attivazioni ottenute con il t_{tlc} calcolato dai dati da visione non sono altrettanto precise (figura (8.27)). In particolare con i dati di visione abbiamo falsi allarmi a 130, 190, 550, 610 e 990 metri. Inoltre a 960 metri non viene rilevato un attraversamento di corsia.

8.2.2 Percorso su rettilineo a velocità variabile

Simulazioni con filtro EKF

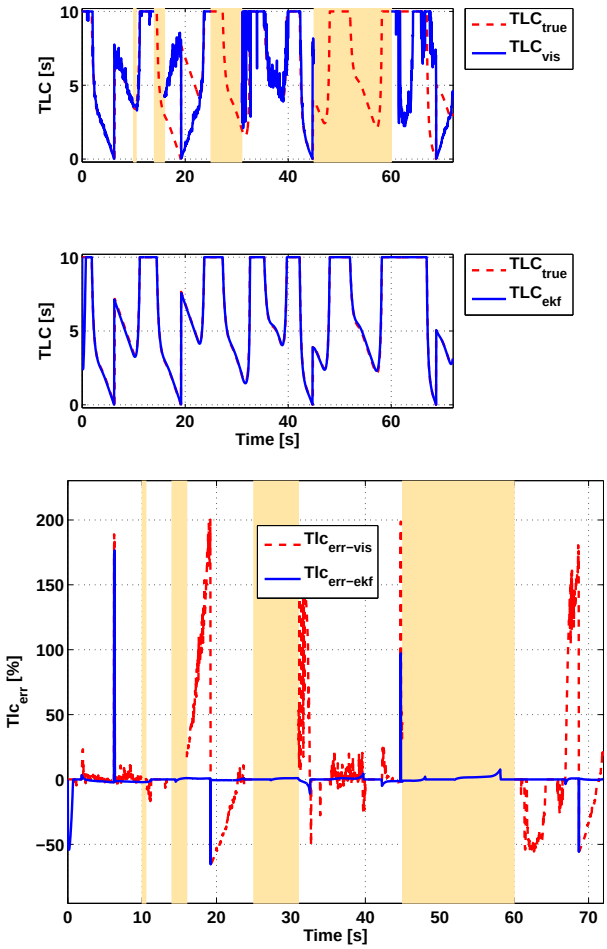

Figura 8.28. Tempo di invasione t_{tlc} EKF

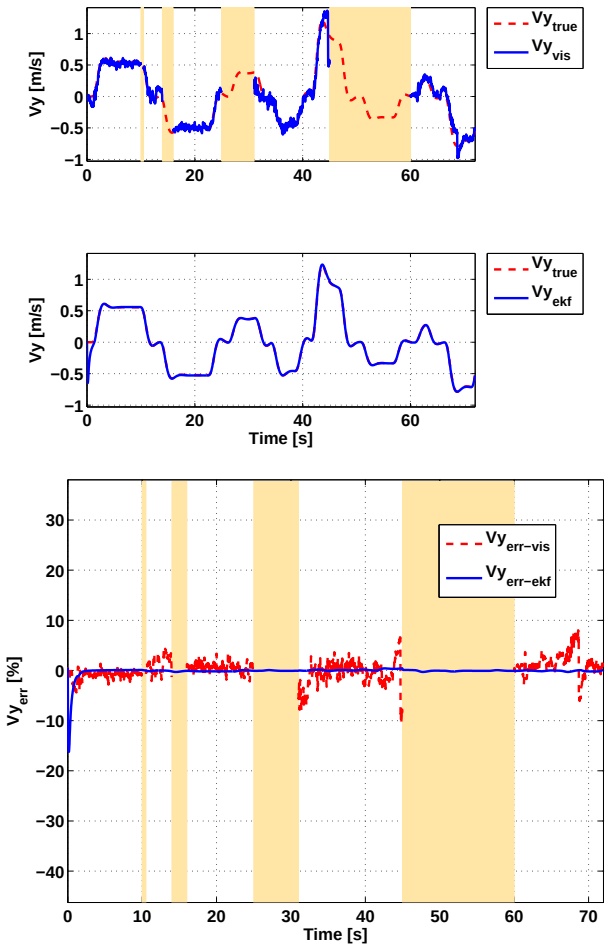

Figura 8.29. Velocità laterale EKF

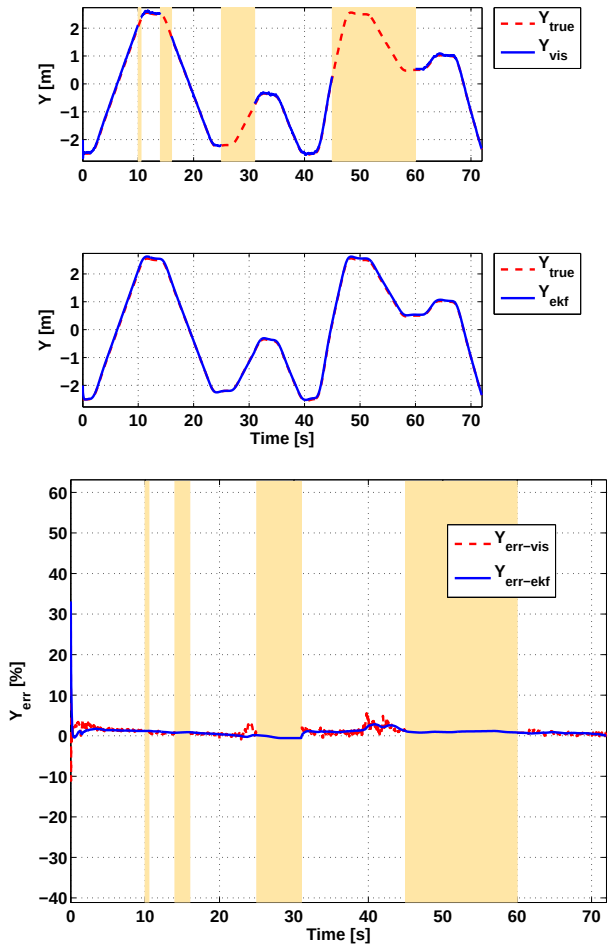

Figura 8.30. Posizione laterale EKF

Il percorso a velocità variabile su rettilineo simula lenti cambi di corsia in autostrada con variazioni di velocità tra 90 e 120Km/h con brusche frenate e accelerazioni in corrispondenza dei cambi di corsia. La simulazione ha una durata di 72 secondi. Come è visibile in figura (8.28) il t_{tlc} dai dati di visione non è disponibile durante i periodi di occlusione e nei rimanenti è estremamente rumoroso.

La stima del t_{tlc} durante la prima metà di simulazione presenta, ad esclusione di due picchi improvvisi a 6 e 19 secondi, un errore inferiore al 4% Durante l'ultima occlusione invece, in particolare a partire dal 52-esimo secondo, l'errore aumenta fino al 10% per poi ritornare a zero al termine dell'occlusione.

Nel grafico relativo alla velocità laterale (V_y)(figura (8.29)) il filtro di

Kalman Esteso elimina completamente l'effetto del rumore introdotto dai dati di visione con un errore di stima inferiore allo 0.4% anche all'interno delle occlusioni.

La stima della posizione laterale (figura (8.30)), invece, si mantiene al di sotto del 3%.

Simulazioni con filtro UKF

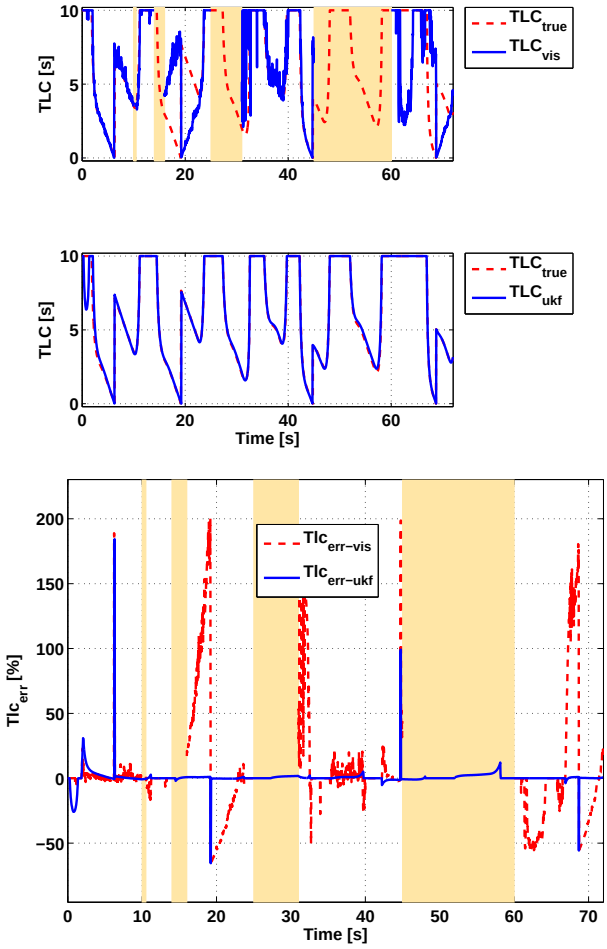

Figura 8.31. Tempo di invasione t_{tlc} UKF

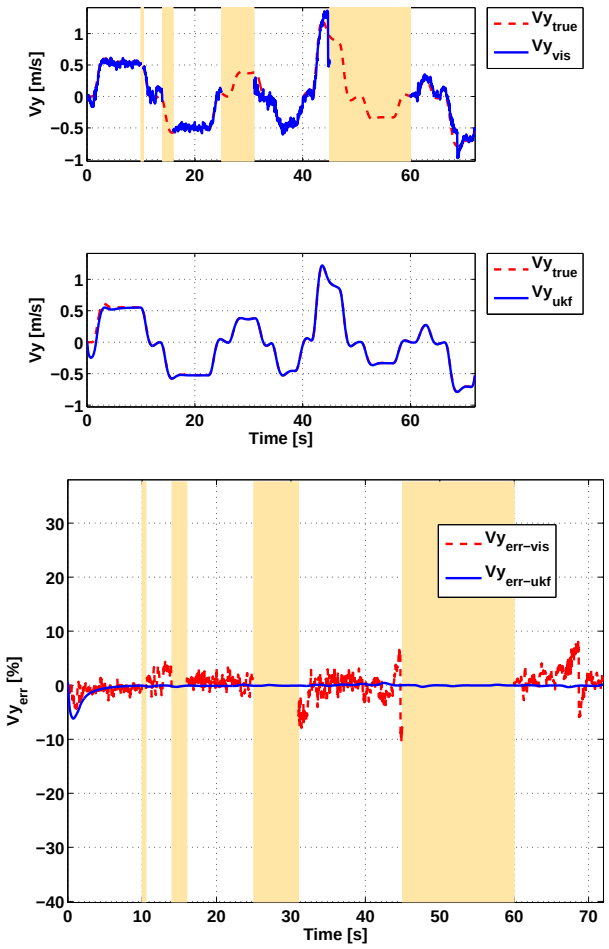

Figura 8.32. Velocità laterale UKF

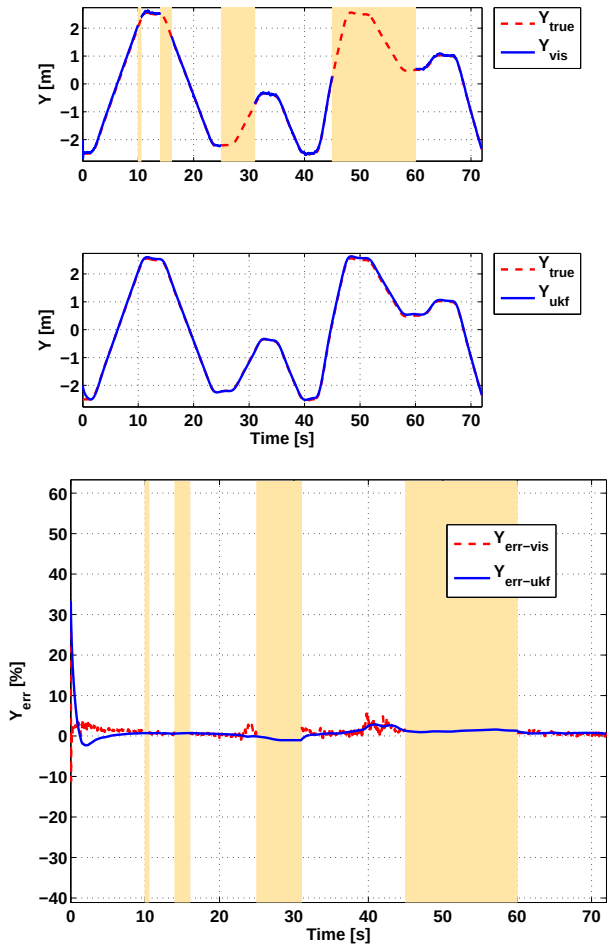

Figura 8.33. Posizione laterale UKF

La figura (8.31) mostra l'andamento del t_{tlc}. Il filtro di Kalman Unscented ha lo stesso comportamento del filtro di Kalman Esteso.

La figura (8.32) mostra l'andamento della velocità laterale. In particolare l'errore di stima si mantiene al di sotto del 0.4% durante tutta la simulazione, anche negli intervalli di occlusione.

Il filtro di Kalman Unscented riesce a inseguire con precisione anche la posizione laterale (figura (7.31)) con valori inferiori al 3%.

Confronto tra EKF e UKF

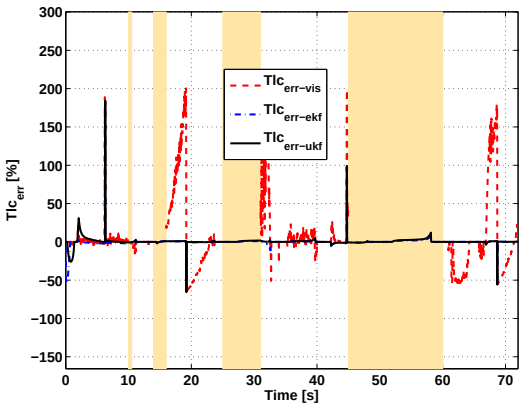

Figura 8.34. Errore sul t_{tlc}

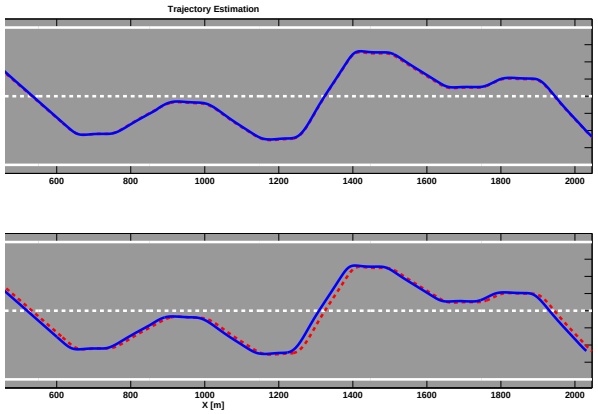

Figura 8.35. Traiettoria su strada

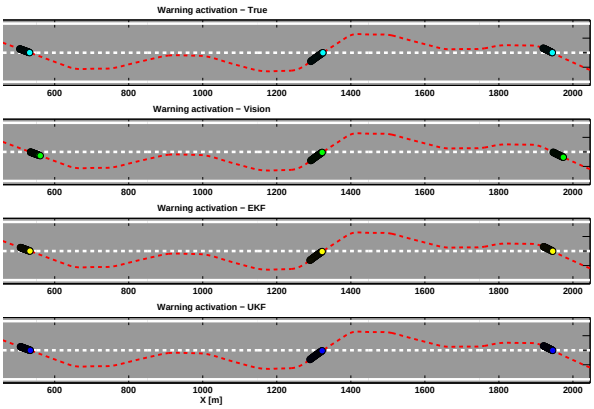

Figura 8.36. Allarme di superamento corsia attigua (soglia t_{tlc} 1.5 sec)

La figura (8.34) riassume gli errori commessi nel calcolo del t_{tlc}. Sia il filtro di Kalman Unscented che quello Esteso restituiscono un valore di t_{tlc} con un errore di stima dell'ordine del 4% il filtro di Kalman Esteso mostra un leggerissimo vantaggio rispetto al filtro Unscented.

La traiettoria XY visibile in figura (8.35) mostra un buon inseguimento per entrambi i filtri ma anche in questo caso è evidente la leggera differenza tra i due filtri.

Gli allarmi, riportati in figura (7.33), vengono attivati correttamente dai due filtri di Kalman e dal sistema di visione poiché le occlusioni non si verificano durante gli intervalli di attraversamento di corsia.

8.3 Confronti totali

Di seguito è riportato il confronto dei risultati ottenuti per ogni diversa manovra di test dai filtri di Kalman Esteso e Unscented con i modelli cinematico differenziale e cinematico yaw rate in presenza di occlusione nella visione.

Il confronto è basato, come nella sezione 7.5, sulla radice dell'errore quadratico medio (RMSE Root Mean Squared Error) definita in (7.2).

In questo caso, a differenza della sezione 7.5, i valori di RMSE riferiti al solo sistema di visione sono stati omessi in quanto poco indicativi. Infatti il sistema di visione smette di funzionare negli intervalli di occlusione e l'RMSE su tutta la simulazione risulta falsato.

8.3.1 Percorso su rettilineo a velocità costante

- Modello 3: Cinematico differenziale

	RMSE Stima EKF	**RMSE Stima UKF**
Posizione Laterale (y)	6.6866980e+00	1.1997420e+01
Velocità Laterale (v_y)	4.7713125e+00	7.0689159e+00
t_{tlc}	1.7526965e+00	2.3445018e+00

- Modello 4: Cinematico yaw rate

	RMSE Stima EKF	**RMSE Stima UKF**
Posizione Laterale (y)	6.6940478e+00	7.1578841e+00
Velocità Laterale (v_y)	6.5591796e+00	4.6726805e+00
t_{tlc}	1.9868037e+00	1.6056110e+00

8.3.2 Percorso su rettilineo a velocità variabile

- Modello 3: Cinematico differenziale

	RMSE Stima EKF	**RMSE Stima UKF**
Posizione Laterale (y)	7.6254222e+00	7.5276283e+00
Velocità Laterale (v_y)	2.2570040e+01	1.5194083e+01
t_{tlc}	1.1924786e+00	1.1674070e+00

- Modello 4: Cinematico yaw rate

	RMSE Stima EKF	**RMSE Stima UKF**
Posizione Laterale (y)	1.3847833e+00	2.1281509e+00
Velocità Laterale (v_y)	8.5038301e+00	6.5158644e+00
t_{tlc}	1.0999255e+00	8.6325248e-01

8.4 Conclusioni

In questa sezione è stato analizzato il comportamento degli algoritmi di fusione dei dati in caso di occlusione temporanea del sensore video. Le simulazioni sono state effettuate con gli algoritmi di fusione basati su modello cinematico differenziale e sul modello cinematico con yaw rate.

- Entrambi i filtri hanno riportato ottime prestazioni nelle simulazioni eseguite a velocità costante, riuscendo a fornire dati corretti anche durante l'occlusione più lunga.
- Maggiori difficoltà di stima sono state riscontrate in condizioni di cambi di corsia a velocità variabile da parte del modello cinematico differenziale, soprattutto nell'ultima occlusione in cui il filtro di Kalman Unscented ha ottenuto i risultati migliori.

- Entrambi i filtri basati sul modello cinematico con yaw rate hanno mostrato un'ottima capacità di stima in presenza di occlusioni e con simulazioni a velocità variabile. L'inseguimento di posizione e velocità laterale con questo modello è risultato nettamente più preciso rispetto a quello ottenuto con il cinematico differenziale.
- In tutte le simulazioni l'errore maggiore è stato commesso nella stima della posizione laterale.
- Lo studio effettuato conferma i notevoli vantaggi dell'uso di algoritmi di fusione dei dati per sistemi LDWS. É evidente, infatti, come un sistema basato solo sui dati del sistema video durante le temporanee occlusioni debba interrompere il supporto al guidatore, mentre integrandolo con algoritmi di data fusion riesca a tollerare occlusioni anche maggiori di 10 secondi.

9

Implementazione Hardware

9.1 Implementazione e Ottimizzazione su DSP

9.1.1 Scheda di sviluppo

Il sistema LDWS è stato implementato su piattaforma hardware basata su processore Texas Instruments $OMAP3530$. Il processore ad alte prestazioni OMAP3530 è basato sull'architettura $OMAP^{TM}3$ che include alcuni sottoinsiemi progettati per supportare l'alta velocità e le applicazioni multi-thread. L'OMAP3530 integra un core Superscalar *ARM Cortex-A8*, un sottosistema *IVA2.2* con core DSP (Digital Signal Processor) C64x+, un acceleratore grafico 2D/3D, un sottosistema display, un camera ISP (Image Signal Processor), un sistema di interconnessione ad alta prestazione e periferiche industrial-standard. L'apparecchiatura include tecniche di power-management utilizzate da sistemi operativi evoluti come WindowsCE, Linux, Symbian OS e Palm OS.

La piattaforma di sviluppo utilizzata in laboratorio è un computer-on-module basata su processore OMAP3530 sviluppata e prodotta da Gumstix inc.. In particolare il prodotto utilizzato appartiene alla serie Overo e presenta le seguenti caratteristiche tecniche:

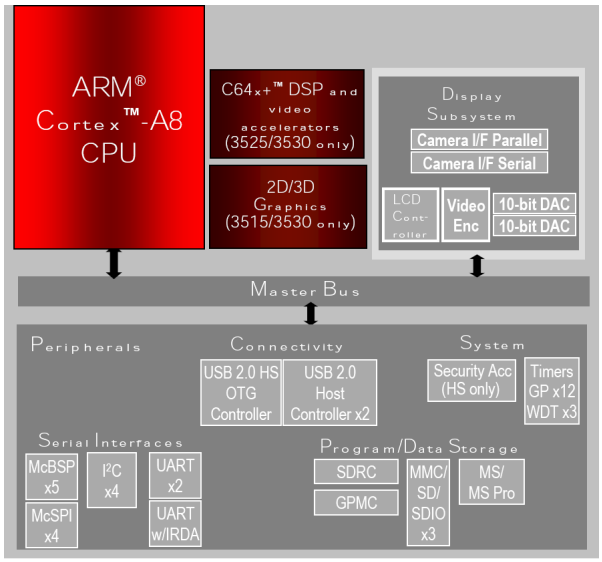

Figura 9.1. Architettura OMAP3530

Processore	OMAP 3530 che integra nello stesso package: ARM Cortex-A8 CPU, Digital signal processor (DSP) C64x+, POWERVR SGX per l'accelerazione grafica 2D e 3D
Clock	600 MHz
Prestazioni	fino a 1200 Dhrystone MIPS
Memoria	256MB RAM e 256MB Flash
Caratteristiche	I/O pin compatibili con tutti i moduli overo basati su OMAP 35x-based slot scheda di memoria microSD I2C, PWM (6), A/D (6), 1-wire, UART, SPI, Camera in, Extra MMC, Cuffie, Microfono, batteria di backup, USB OTG, USB HS Host

Il modulo Overo Gumstix ha dimensioni molto ridotte pari a $58x20\ mm$.

Le interfacce hardware standard aggiuntive, ad esempio USB, connettore di alimentazione, HDMI e ethernet sono fornite utilizzando schede di espansione prodotte dalla stessa Gumstix. Ogni scheda di espansione ha dimensioni leggermente maggiori rispetto al modulo processore in modo che vi sia spazio per i vari connettori di I/O.

Nonostante le sue piccole dimensioni, la scheda Overo è un computer a tutti gli effetti su cui è possibile eseguire un sistema operativo Linux o Windows CE e può essere programmato per eseguire una vasta gamma di funzioni in quasi ogni settore compresa la sicurezza, controllo accessi, IT, medico, aviazione,

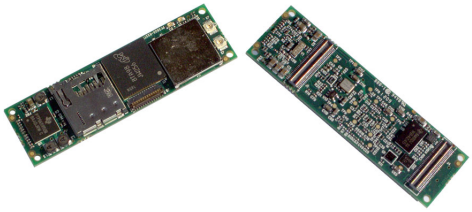

Figura 9.2. Modulo Overo

Figura 9.3. Scheda di espansione Tobi

robotica e l'istruzione, per citarne alcuni.

La piattaforma utilizzata si compone di:

- Modulo Overo
- Scheda di espansione Tobi fissata sul lato inferiore del modulo Overo attraverso i due connettori a 70 pin. Questa scheda di espansione prevede porte di I/O e una presa di ingresso alimentazione. In particolare presenta le seguenti caratteristiche:

Connettori	10/100baseT Ethernet USB OTG mini-AB USB host standard A DVI-D (HDMI) Ingresso audio stereo audio Uscita audio stereo Console seriale USB Input/Output digitali con logica 1.8V through-holes
Alimentazione	5V input
Dimensioni	105mm x 40mm

- Sistema operativo Linux basato su una distribuzione Embedded Linux "Angstrom"

9.1.2 Codice ARM

In questo paragrafo viene descritto il codice C++ scritto in Eclipse CDT per il sistema LDWS. Il programma LDWS che viene eseguito all'interno del sistema operativo Angstrom sulla piattaforma embedded esegue i passi descritti dal diagramma di flusso riportato in figura (9.4)

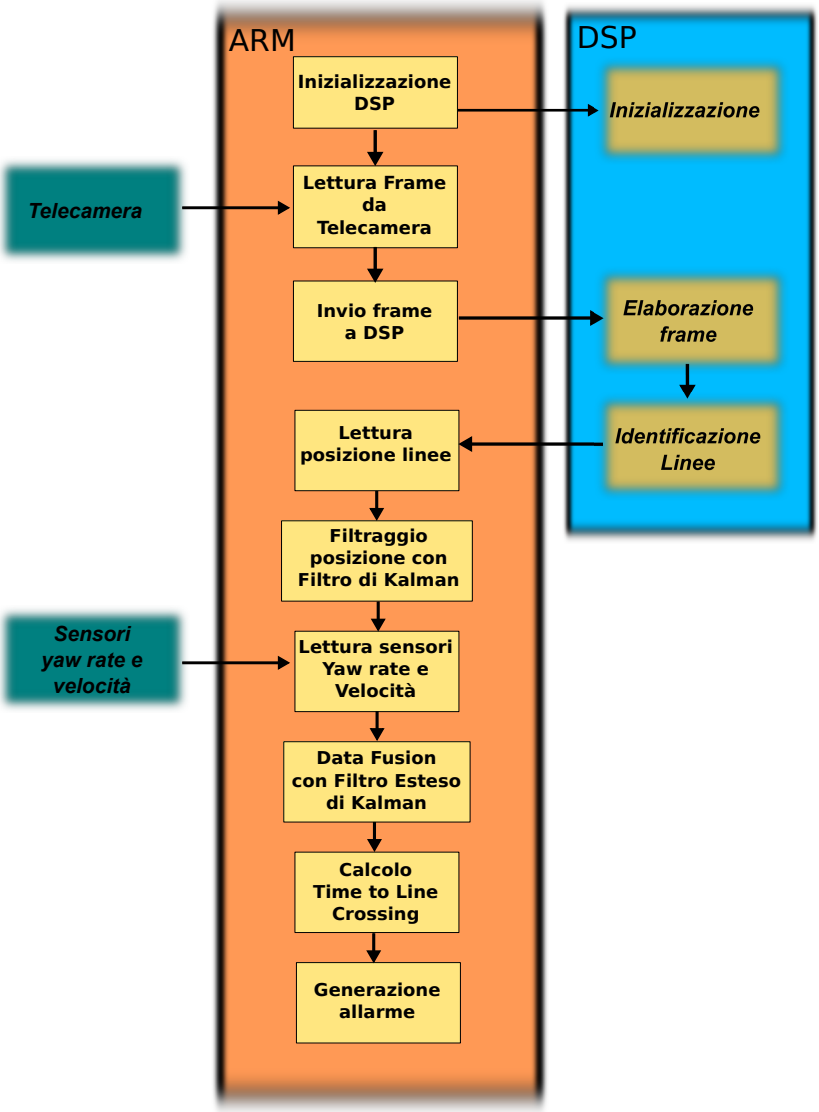

Figura 9.4. Diagramma di flusso codice ARM

La prima operazione è l'inizializzazione del DSP utilizzando le API fornite dal driver DSPLink. A questo punto il programma entra in un ciclo continuo eseguito per ogni frame acquisito. Il frame letto dalla videocamera viene scritto nella memoria condivisa con il DSP che elabora il frame e invia i risultati al processore.

Il processore Arm attende l'elaborazione e appena disponibili legge le informazioni sulla posizione delle linee della carreggiata rispetto all'autovettura. Le posizioni della linee sono filtrate con un filtro di Kalman per inseguire la loro posizione su frame successivi. L'algoritmo, quindi, legge i valori del sensore di yaw rate e velocità del veicolo e li passa insieme con la posizione delle linee al filtro di Kalman Esteso utilizzato per la fusione dei dati. Dopo la fusione dei dati sono disponibili la velocità laterale e la posizione laterale del veicolo all'interno della carreggiata che sono utilizzati per il calcolo del tempo di invasione di corsia attigua e per la generazione dell'allarme al guidatore.

OpenCV

Le operazioni di acquisizione e pre-elaborazione delle immagini nonchè la visualizzazione dei risultati ottenuti sono state eseguite utilizzando la libreria open source di computer vision OpenCV disponibile su
`http://SourceForge.net/projects/opencvlibrary`. La libreria è scritta in C e C + + e gira sotto Linux, Windows e Mac OS X, inoltre è attivo lo sviluppo di interfacce per Python, Ruby, Matlab, e altri linguaggi. OpenCV è stato progettato per ottenere efficienza computazionale con particolare attenzione per le applicazioni in tempo reale. OpenCV è scritto in C ottimizzato e può usufruire dei vantaggi offerti dai processori multicore attuali, un'ulteriore ottimizzazione su architetture Intel, è possibile utilizzando le librerie commerciali Intel Integrated Performance Primitives (IPP), che consistono in routine ottimizzate a basso livello per differenti tipi di algoritmi. OpenCV utilizza automaticamente la libreria IPP appropriata runtime se tale libreria è installata.

Uno degli obiettivi di OpenCV è quello di fornire un'infrastruttura di computer vision semplice da usare che aiuta le persone a costruire rapidamente sofisticate applicazioni di visione.

La licenza open source per OpenCV è stata strutturata in modo tale che è possibile costruire un prodotto commerciale utilizzando tutte o parte delle funzioni OpenCV. Non esiste alcun obbligo di diffondere il proprio prodotto come opensource o rendere i miglioramenti apportati alla libreria di pubblico dominio. Grazie a queste condizioni di licenza libera, c'è una vasta comunità di utenti che comprende persone provenienti da grandi aziende (IBM, Microsoft, Intel, Sony, Siemens e Google, per citarne solo alcuni) e centri di ricerca (come Stanford, MIT, CMU, Cambridge, ed INRIA).

La libreria OpenCV contiene oltre 500 funzioni che si estendono in molte aree della visione, tra cui l'ispezione di prodotti in catene di montaggio, l'analisi di immagini mediche, di sicurezza, interfacce utente, calibrazione

delle telecamere, visione stereo e robotica. Inoltre poiché la visione artificiale e apprendimento automatico spesso vanno di pari passo, OpenCV contiene anche una completa libreria di Machine Learning (MLL), questa sottolibreria è focalizzata sulla statistica e sul riconoscimento di pattern e sulla clusterizzazione.

OpenCV è volto a fornire gli strumenti di base necessari per risolvere i problemi di visione artificiale. In alcuni casi, le funzionalità di alto livello nella libreria sono sufficienti per risolvere i problemi più complessi in computer vision. Anche quando questo non è il caso, i componenti base della libreria sono abbastanza completi per trovare una soluzione a quasi tutti i problemi di computer vision. In genere, nella fase iniziale del progetto si cerca di utilizzare il maggior numero possibile di componenti della libreria e in seguito analizzati i suoi punti deboli si cercano soluzioni inserendo nuove porzioni di codice scritte appositamente.

OpenCV è sostanzialmente strutturato in cinque componenti principali, quattro dei quali sono indicati in figura (9.5). Il componente CV contiene l'elaborazione di base dell'immagine e gli algoritmi di computer vision di alto livello; ML è la libreria di machine learning, che comprende molti classificatori statistici e strumenti di clustering. HighGUI contiene le funzioni di ingresso uscita per la memorizzazione e il caricamento di video e immagini infine, CXCore contiene le strutture e i contenuti di base.

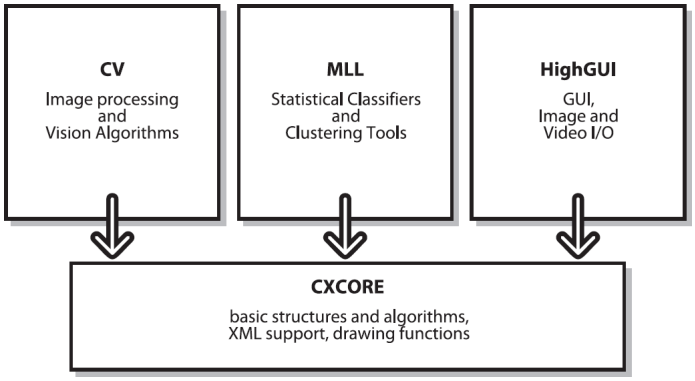

Figura 9.5. Strutture di base OpenCV

La figura (9.5) non comprende CvAux, che contiene algoritmi sperimentali di segmentazione dello sfondo e riconoscimento facciale. CvAux comprende:

- *Eigenobject*, una tecnica di riconoscimento computazionalmente efficiente basata su template matching
- 1D e 2D modelli nascosti di Markov, una tecnica di riconoscimento statistico basata sulla programmazione dinamica

- Embedded HMM (le osservazioni di un genitore HMM sono essi stessi HMM)
- riconoscimento dei gesti da visione stereo
- Estensioni di triangolazione di Delaunay, sequenze, e così via
- Visione Stereo
- Forma abbinamento con contorni regione
- inseguimento di occhi e bocca
- inseguimento 3D
- Ricerca di scheletri (linee centrale) di oggetti in una scena
- Segmentazione primo piano e sfondo
- Video sorveglianza
- Funzioni in C++ di calibrazione della videocamera

Alcune di queste funzioni saranno integrate in *CV* nelle versioni future, altre probabilmente no.

9.1.3 Codice DSP

In questo paragrafo viene descritto il codice DSP scritto in Code Composer Studio per il sistema LDWS. Il programma LDWS che viene eseguito dal DSP ed esegue i passi descritti dal diagramma di flusso riportato in figura (9.6)

Appena il DSP riceve il comando di inizializzazione dal processore arm tramite DSPLink crea lo spazio di memoria condiviso per lo scambio dei dati e crea la connessione con il processore per lo scambio dei messaggi di stato. Nella fase di inizializzazione è stata inoltre inserita la creazione delle matrici utilizzate per l'inversione prospettica che dipendono dai parametri di installazione del sistema come altezza della telecamera, risoluzione, angolo di visione. Il ciclo principale del programma legge ogni qualvolta diviene disponibile un nuovo frame dalla memoria condivisa. Utilizzando le matrici di inversione calcolate nella fase di inizializzazione viene calcolata l'immagine invertita prospetticamente che fornisce una vista dall'alto della strada. A questa immagine invertita si applicano tutti i successivi passaggi necessari ad identificare le linee di demarcazione della carreggiata. Innanzitutto l'immagine viene binarizzata, poi viene effettuata la ricerca delle discontinuità e quindi delle linee tramite le tecniche descritte nel Capitolo 2. Ottenuta la rappresentazione matematica della posizione delle linee all'interno dell'immagine invertita il programma calcola la posizione delle linee in coordinate reali rispetto al veicolo. I dati sulle linee vengono a questo punto inviati al processore ARM e il DSP si mette in attesa di un nuovo frame da elaborare.

ImgLib

Il codice DSP scritto per l'elaborazione delle immagini utilizza un insieme di funzioni gratuite liberamente distribuite da Texas Instruments. La libreria di elaborazione video e immagini (imglib) per la famiglia di DSP

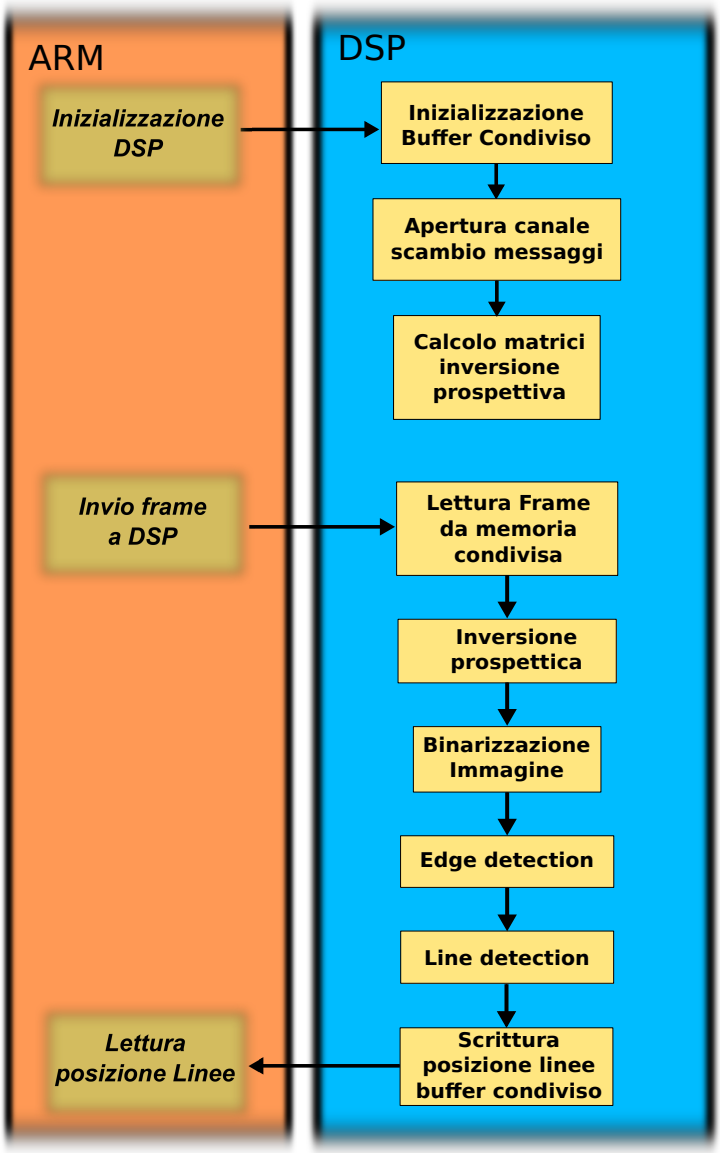

Figura 9.6. Diagramma di flusso codice DSP

TMS320C64x fornisce un set di funzioni C, ottimizzate a livello assembler comunemente utilizzate nelle applicazioni di imaging e in applicazioni tempo reale computazionalmente onerose in cui la velocità di esecuzione ottimale è essenziale.

Utilizzando queste routine, è possibile raggiungere velocità di esecuzione notevolmente maggiori rispetto a un codice equivalente scritto in linguaggio ANSI C standard. Inoltre, fornendo funzioni DSP pronte all'uso, TI imglib può significativamente ridurre i tempi di sviluppo di applicazioni video o di elaborazione delle immagini.

Ogni funzione in imglib è progettata per produrre le migliori prestazioni possibili ottimizzando le risorse disponibili ed evitando i potenziali conflitti di risorse. Pertanto, quando si sviluppa un sistema che utilizza imglib, è importante capire overhead potenziali connessi alla gerarchia di memoria, al fine di valutare e migliorare le prestazioni effettive del sistema. Il codice sorgente è libero e permette di modificare le funzioni in relazione a specifiche esigenze. La raccolta di routine software è organizzata in tre diverse categorie funzionali:

- Compressione e decompressione - Questa sezione contiene le funzioni che sono applicabili a standard di compressione come JPEG, video MPEG e H.26x.
- Analisi dell'immagine - Questa sezione fornisce le funzioni applicabili all'analisi standard di immagini, analisi dei bordi e dei contorni, operazioni di dilatazione ed erosione, calcolo dell'istogramma, calcolo dei contorni con il metodo di Sobel, ecc.
- Operazioni di filtraggio e conversione di formato - In questa sezione sono raccolte le funzioni che si applicano per filtrare o convertire un'immagine, in particolare sono utilizzabili funzioni per: convoluzione e correlazione con filtri generici, algoritmo di Floyd-Steinberg per il calcolo di errori di diffusione, filtraggio mediano, saturazione e conversione tra spazi di colore (ad es. YCbCr in RGB) ecc.

9.2 Test funzionali in laboratorio algoritmo LDWS

I test funzionali in laboratorio sono stati eseguiti sulla piattaforma embedded utilizzando video e dati dei sensori generati con l'aiuto del simulatore software Carsim 8.0 prodotto dalla Mechanical Corporation. I video sono stati elaborati offline caricandoli sulla memoria esterna microSD della scheda Overo e visualizzati utilizzando l'interfaccia HDMI su un monitor per PC. In questo caso la porzione di codice necessaria per la lettura dei frame dalla telecamera è stata sostituita con un codice equivalente che legge i frame da un file video in formato *.avi*.

9.2.1 Carsim 8.0

Carsim è un pacchetto software per la simulazione dinamica di veicoli, permette l'animazione e l'analisi del comportamento di un veicolo sottoposto a determinati ingressi di controllo. I modelli matematici sono completamente non lineari, ogni vettura è caratterizzata da più di centocinquanta equazioni differenziali che permettono di ottenere dati molto simili alla realtà.

I modelli possono essere utilizzati in simulazioni HIL (Hardware In the Loop) per il test di nuovi algoritmi di controllo. Inoltre è completamente interfacciabile con l'ambiente Simulink di Matlab. Per ogni modello è possibile definire la dinamica e i parametri di ogni componente: ruote, sospensioni, impianto frenante, cambio, aerodinamica, superficie stradale e molto altro. Eseguita una simulazione risultano disponibili per l'analisi più di seicento variabili del veicolo (forze, accelerazioni, velocità, posizioni, angoli etc.).

CarSim predice le prestazioni dei veicoli in risposta agli input di controllo del guidatore (sterzo, acceleratore, freni, frizione, e spostare) in un dato ambiente (geometria stradale, coefficienti di attrito, vento). Con prestazioni, si intende il movimento dei veicoli, le forze e i momenti coinvolti nell'accelerazione, la manovra, e la frenata. Possono essere simulate quasi tutte le prove di un veicolo che potrebbero essere condotte in pista o su strada. È possibile studiare i cambiamenti nel comportamento del veicolo dovuti alla modifica di qualsiasi parametro del veicolo, ingresso di controllo o ambiente di guida. È possibile aggiungere ulteriori elementi e sistemi al veicolo come ad esempio ABS, controllo di trazione, controllo della stabilità e sviluppare nuovi algoritmi.

9.2.2 Configurazione Carsim 8

9.2.3 Veicolo

Le prove effettuate sono state eseguite simulando un'automobile del segmento *C* a cui appartengono auto commerciali come VW Golf, Alfa Romeo 147, Audi A3, Fiat Bravo. Carsim permette di configurare ogni aspetto meccanico e dinamico del veicolo utilizzato nelle simulazioni. Come visibile in figura (9.7).

Innanzitutto sono state definite le dimensioni fisiche del veicolo altezza, larghezza, interasse, posizione del centro di massa etc. Inoltre tramite la schermata di configurazione è possibile accedere alla configurazione dei parametri di aerodinamica, trasmissione, motore, sistema frenante e di sterzo. Le caratteristiche delle sospensioni e di ogni singolo pneumatico ad esse connesse sono configurabili tramite altrettante schermate di configurazione.

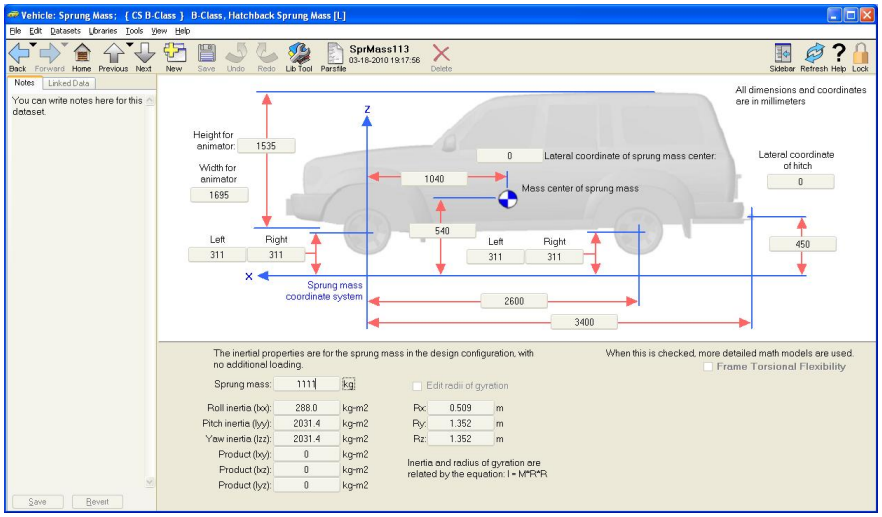

Figura 9.7. Schermata di configurazione del veicolo

9.2.4 Animatore 3D

Carsim include *SurfAnim* un programma per la visualizzazione di animazioni da una videocamera virtuale, che opera in un mondo simulato in 3D. SurfAnim crea un mondo 3D simulato leggendo i files con informazioni su forma, movimento e audio del veicolo insieme alle impostazioni della fotocamera. SurfAnim può essere utilizzato in seguito per riprodurre le animazioni salvate in precedenza e inserirle nelle presentazioni Microsoft PowerPoint o software simili. Si possono inoltre generare file video multimediali in formato *.avi*. Quando una videocamera riprende un veicolo che si muove lungo la strada, trenta volte al secondo, viene registrata l'immagine del veicolo e del mondo rispetto a un particolare punto di vista. SurfAnim permette di configurare un punto di vista esterno o interno. Nel caso di punto di vista esterno, come mostrato in figura (9.8), la videocamera può essere rivolta verso un punto fisso nello spazio in in modo tale che il veicolo sia visibile solo all'interno del campo visivo, oppure puntata verso un punto solidale al veicolo in modo da mantenere l'auto all'interno dell'inquadratura. Nel caso di punto di vista interno, è possibile

posizionare la videocamera all'interno del veicolo e ottenere il campo visivo del guidatore.

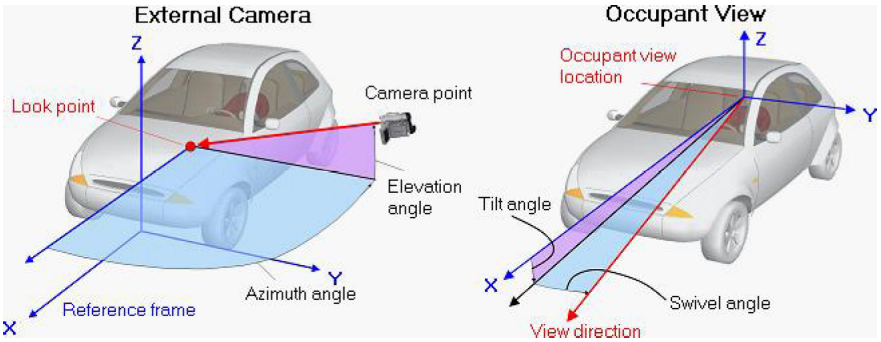

Figura 9.8. Geometria dei punti di vista della videocamera

Al fine di ottenere la vista di una telecamera per LDWS il punto di vista della telecamera è stato fissato in corrispondenza dello specchietto retrovisore nella stessa posizione che occuperebbe la telecamera in una installazione all'interno di un veicolo reale. La figura (9.9) mostra il pannello di configurazione settato con i valori utilizzati.

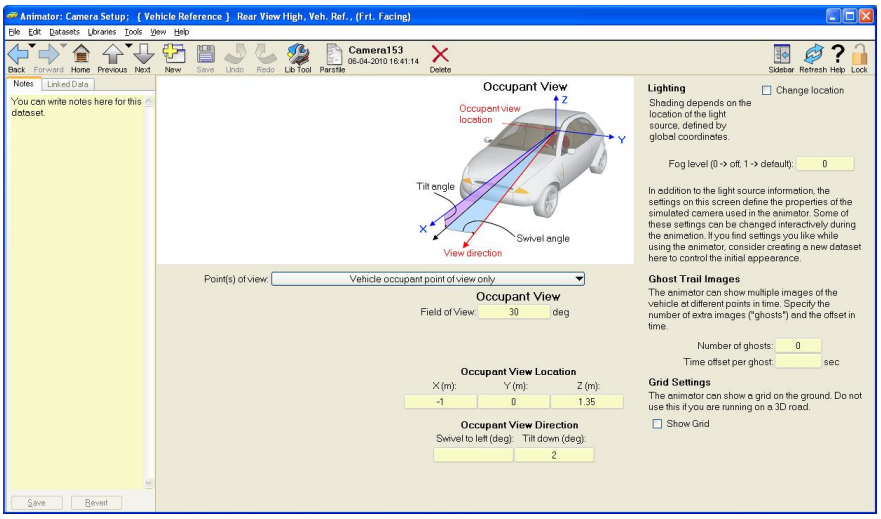

Figura 9.9. Parametri videocamera

9.2.5 Disegno del percorso virtuale

La strada in Carsim è descritta da un insieme di informazioni utilizzate per simulare una superficie 3D, con proprietà variabili di attrito e resistenza al rotolamento dei pneumatici. La serie di dati è utilizzata anche per definire l'aspetto della strada nelle animazioni.

La geometria orizzontale di una strada è fornita come una sequenza di coordinate X e Y che definiscono una linea di riferimento, come ad esempio la linea di mezzeria della strada o il bordo del marciapiede. Viene indicata con S (Stazione) la distanza lungo la linea di riferimento. Quando si descrive la geometria stradale, S è la coordinata longitudinale indipendente. Per ogni dato valore di S, esiste un unico insieme di coordinate X e Y a cui possono essere associati anche altri valori come elevazione, inclinazione, attrito, ecc..

S è definita come una funzione di X e Y, collegando i punti con linee rette, per ogni coppia di coordinate XY, un corrispondente incremento di S è calcolata utilizzando il teorema di Pitagora. Questo incremento viene aggiunto al valore precedente di S:

$$S_i = S_{i-1} + \sqrt{(X_i - X_{i-1})^2 + (Y_i - Y_{i-1})^2} \tag{9.1}$$

Inoltre è possibile definire S usando un'interpolazione spline cubica definendo le derivate $\frac{\partial X}{\partial S}$ e $\frac{\partial Y}{\partial S}$. Ciò fornisce un comportamento realistico, anche quando la geometria stradale non è dettagliata.

La schermata principale per la definizione della strada è mostrata in figura (9.10). Questa schermata è utilizzata per assemblare vari set di dati e ottenere una descrizione generale della strada.

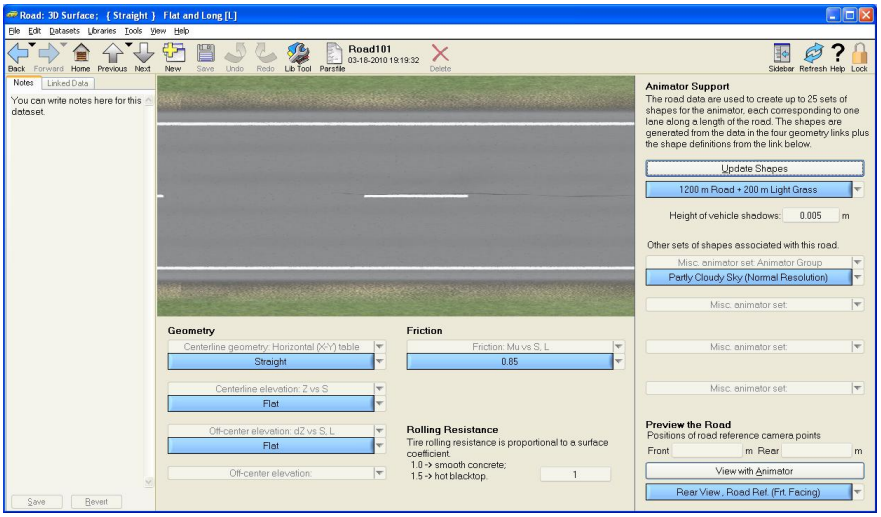

Figura 9.10. Schermata configurazione ambiente stradale

In particolare possono essere configurate separatamente ognuna delle seguenti caratteristiche:

1. Coordinate X-Y della linea centrale. E' possibile definire la geometria orizzontale della strada utilizzando una tabella di coordinate XY. Questa tabella stabilisce automaticamente anche il collegamento tra S e le coordinate globali XY.
2. Coordinate X-Y-Z della linea centrale. Nel caso sia necessario definire un tracciato con una geometria verticale della strada è disponibile una tabella di coordinate XYZ. Anche in questo caso la connessione tra S e le coordinate globali XY è automatica.
3. In alternativa ai due punti precedenti è possibile definire la strada assemblando singoli segmenti di rettilinei o curve disegnati separatamente.
4. L'attrito tra il pneumatico e la strada è definito in funzione di S e di L (Larghezza della strada) ottenendo variazioni di attrito sia longitudinalmente al tracciato che lungo la sezione della carreggiata.
5. E' disponibile una tabella per aggiungere ruvidità alla superficie della strada in corrispondenza della posizione delle gomme.
6. L'animatore utilizza delle texture 2D disegnate sul profilo stradale in dipendenza di S e di L.
7. Gli oggetti lungo il percorso (alberi, guard rail, coni di segnalazione, ecc.) sono disegnati in corrispondenza di coordinate X-Y o $S - L$ assegnando specifiche texture e dimensioni.
8. Il pannello di configurazione permette inoltre di visualizzare il percorso progettato in vista 3D, utile per valutarne velocemente le caratteristiche.

9.2.6 Registrazione video di prova

La fase finale di simulazione prevede l'esecuzione dell'animatore 3D. I dati di dinamica e cinematica del veicolo calcolati su uno specifico percorso vengono elaborati da un modulo grafico che genera la vista della telecamera secondo la posizione impostata. Nel nostro caso il punto di vista è posizionato in corrispondenza dello specchietto retrovisore con vista frontale come mostrato in figura (9.11)

Figura 9.11. Vista frontale videocamera

La vista della telecamera durante tutta la simulazione può essere registrata in un file video.

Figura 9.12. Finestra di registrazione

Il programma mette a disposizione la possibilità di eseguire la compressione del file video in linea tramite un'apposita finestra di selezione.

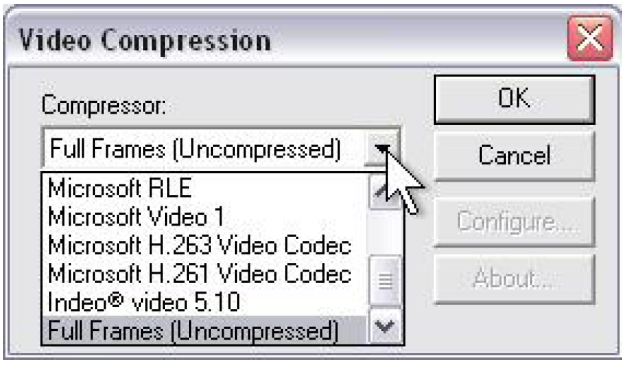

Figura 9.13. Finestra di compressione video

9.2.7 Esecuzione video simulatore su piattaforma embedded - Test con video Carsim

Il video e i dati registrati tramite carsim sono stati salvati sulla scheda microSD della piattaforma embedded basata su Gumstix Overo ed elaborati con il programma LDWS per verificarne il corretto funzionamento. Il video invertito prospetticamente è visualizzato sul monitor collegato alla scheda, inoltre per semplicità di analisi e verifica, sono state disegnate in rosso e in verde le le linee identificate dall'algoritmo LDWS come è visibile in figura (9.14). L'analisi ha evidenziato un'identificazione precisa della posizione delle linee sui video di Carsim e la corretta attivazione degli allarmi in caso di invasione di corsia attigua.

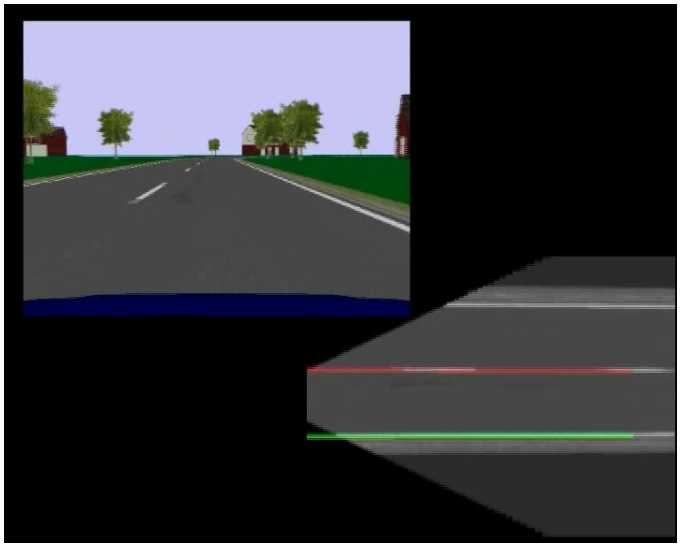

Figura 9.14. Simulazione video carsim in rettilineo

In figura (9.15) viene mostrata la capacità dell'algoritmo progettato di identificare e inseguire correttamente le linee della carreggiata anche in tratti di strada curva.

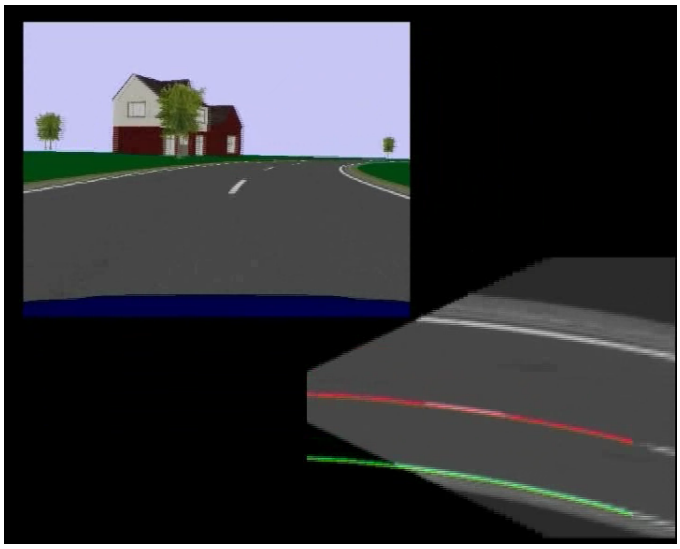

Figura 9.15. Simulazione video carsim su strada curva

Il test funzionale dell'algoritmo LDWS utilizzando un simulatore software e la piattaforma embedded è stato superato con successo. Si è potuto quindi procedere all'installazione e al test a bordo vettura.

9.3 Installazione e test a bordo vettura

9.3.1 Vettura utilizzata

La vettura utilizzata per i test su strada è una Fiat Grande Punto del 2009 (figura (9.16)).

Figura 9.16. Grande Punto

L'auto ha le seguenti caratteristiche:

- Lunghezza 4,03 metri
- larghezza 1,68 m
- altezza 1,49 m
- passo 2,51 m
- peso
- Motore 1.3 multijet turbodiesel da 75 Cavalli

9.3.2 Hardware utilizzato

L'hardware utilizzato per l'installazione a bordo vettura è costituito da tre moduli:

- Videocamera
- Modulo dei sensori inerziali
- Scheda elettronica

La videocamera è connessa alla scheda elettronica tramite interfaccia USB mentre il modulo sensori ha un'interfaccia seriale RS-232.

Videocamera

La videocamera utilizzata è una videocamera CMOS prodotta da Point Grey Research, Inc.. Point Grey Research, Inc. è leader mondiale nello sviluppo

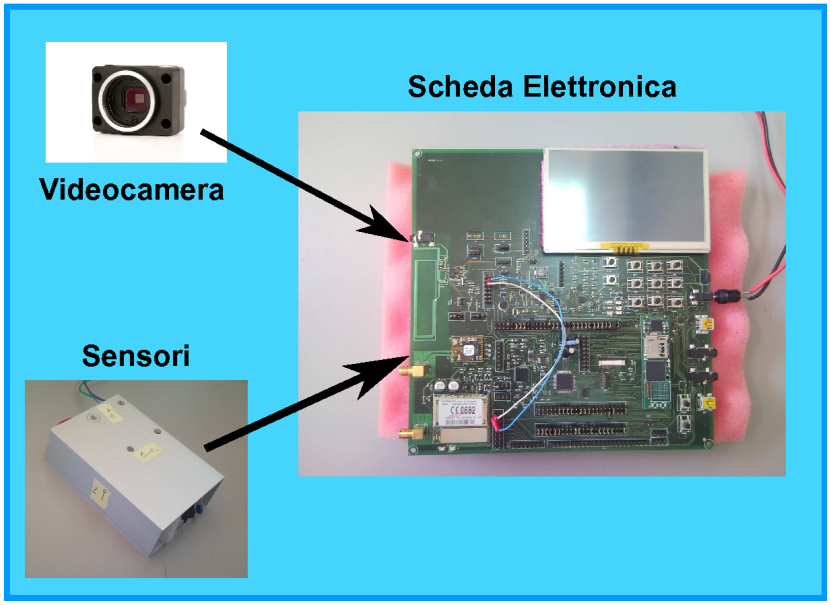

Figura 9.17. Schema a blocchi Hardware

di prodotti video ad avanzata tecnologia per la visione artificiale, visione industriale e applicazioni di computer vision. Con sede a Richmond in Canada, Point Grey, produce e distribuisce telecamere IEEE-1394 (FireWire) e USB 2.0 che sono note per la loro eccellente qualità, prestazione e facilità d'uso. L'azienda fornisce una vasta gamma di hardware, software e un competente supporto tecnico che la colloca tra le migliori aziende nel settore. Il prodotto scelto per le prove a bordo vettura è la telecamera CMOS FireFly MV dotata di interfaccia USB2.0 che permette di trasmettere i dati e fornire l'alimentazione con un unico cavo minimizzando la quantità di hardware di interfaccia e alimentazione. La fotocamera offre inoltre un connettore di I/O a 7-pin per uso generale. Il connettore GPIO può essere utilizzato per sincronizzare la telecamera a dispositivi esterni, come un trigger esterno o un circuito di alimentazione supplementare.

Un FPGA controlla tutte le funzioni della telecamera, inclusa esposizione, risoluzione, frame rate, memoria ecc. e può essere facilmente aggiornato con nuove caratteristiche.

Inoltre è disponibile un kit di sviluppo con il software necessario per una rapida progettazione e prototipazione. Il FireCapture (kit di sviluppo software) permette il controllo della telecamera e l'acquisizione delle immagini utilizzando semplici API scritte in C/C++.

La telecamera utilizzata ha le seguenti caratteristiche:

Specifica	Descrizione
Sensore immagine	CMOS 1/3 a scansione progressiva
Otturatore	otturatore globale con tecnologia Micron TrueSNAP
Modello sensore	Micron MT9V022
Risoluzione massima	752 (H) x 480 (V)
Dimensione pixels	6.0 μm x 6.0 μm
Convertitore AD	ADC integrato a 10-bit
Interfaccia Digitale	5-pin Mini-B USB 2.0 per controllo camera, dati video e alimentazione
Velocità di trasferimento	480 Mb/s
Frame rate	752x480 a 61 FPS o 320x240 a 112 FPS (con selezione regione di interesse)
Controllo guadagno	manuale o automatico tra 0 e 12 dB
Velocità otturatore	manuale o automatico tra 0.12 e 512 ms
Gamma	da 0 a 1
Sincronizzazione	con trigger esterno o via software
Tipo di trigger	IIDC v1.31 Trigger Modes 0 e 3
Consumo	da 4.75 a 5.25 V via Mini-B USB 2.0 o connettore GPIO, minore di un (1) Watt
Dimensioni (L x P x A)	24.4 x 44 x 34 mm
Massa	37 g
Montaggio Lenti	CS-mount (con adattatore per 5mm C-mount) o adattatore per microlenti M12

Figura 9.18. Telecamera Point Grey FireFly MV

Scheda elettronica

La scheda prototipo utilizzata per testare il sistema LDWS è stata dettagliatamente descritta nelle sezioni precedenti e integra:

- Modulo processore Overo Gumstix utilizzato anche per i test in laboratorio
- Modulo di posizionamento GPS
- Modulo di comunicazione GSM
- Schermo lcd da 4.3 pollici
- Keypad
- Bussola elettronica

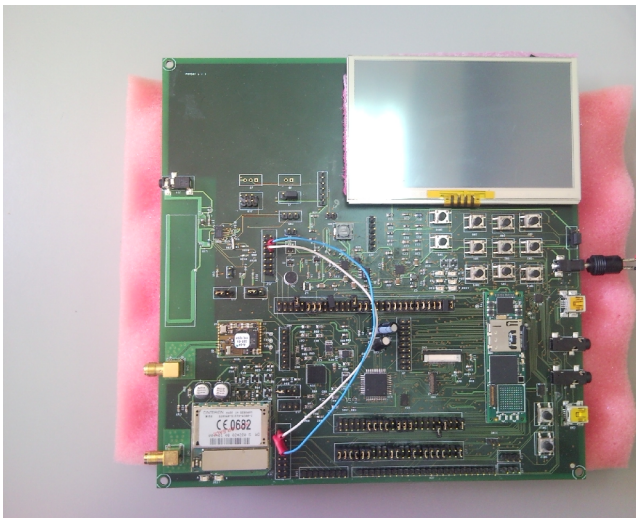

Figura 9.19. Sceda prototipo basata su Gumstix Overo

Sensori Aggiuntivi

La scheda elettronica con display e processore non integra al suo interno un sensore di yaw rate necessario al calcolo del tempo di invasione di corsia attigua del sistema LDWS, per questo motivo si è scelto di aggiungere, tramite connessione seriale, una piattaforma inerziale esterna (figura (9.20))

La piattaforma di misura inerziale (IMU) a sei gradi di libertà (6DOF) è un prodotto della SparkFun Electronics (figura (9.21)). La versione utilizzata prevede un accelerometro a tre assi, un giroscopio a tre assi,e un sensore magnetico anch'esso a tre assi. Ogni canale è selezionabile indipendentemente dall'utente, così come la frequenza di campionamento. Il dispositivo può fornire dati in formato ASCII o binario, utilizzando una connessione Bluetooth o

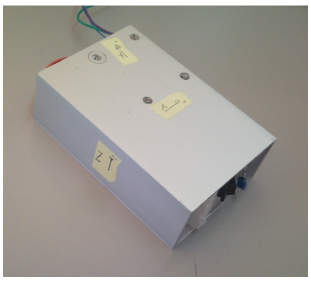

Figura 9.20. Piattaforma Inerziale con involucro protettivo

seriale con logica TTL. Il controllo è fornito tramite un processore LPC2138 ARM7 con abbondante memoria aggiuntiva utile all'implementazione di eventuale codice aggiuntivo. Viene inoltre fornito il codice sorgente del programma precaricato che permette un rapido sviluppo delle specifiche applicazioni

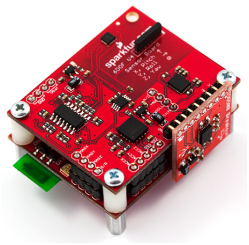

Figura 9.21. Scheda elettronica Piattaforma Inerziale

La IMU è dotata dei seguenti sensori:

- accelerometro a tre assi Freescale MMA7260Q con sensibilità impostabile a $1,5\ g$, $2\ g$, $4\ g$ o $6\ g$
- due giroscopi InvenSense IDG500 da 500 gradi al secondo
- sensori magnetici Honeywell HMC1052L e HMC1051Z

Tutti i sensori sono internamente compensati in temperatura.

Specifica	**Descrizione**
Alimentazione	da $4.2\ V$ a $7\ V$
Consumo	inferiore a $150\ mA$
Risposta in frequenza	Sensori magnetici: $312\ Hz$ Giroscopi: $140\ Hz$ Accelerometero asse X e Y: $350\ Hz$ Accelerometero asse Z: $150\ Hz$

9.3.3 Configurazione vettura

L'installazione a bordo vettura del sistema LDWS ha seguito i seguenti passi:

- Posizionamento e calibrazione della telecamera
- Posizionamento dei sensori aggiuntivi
- posizionamento della scheda elettronica
- Connessione dei moduli Hardware
- connessione dell'alimentazione

Posizionamento telecamera

La videocamera come previsto dal progetto LDWS è stata posizionata al centro del parabrezza in prossimità dello specchietto retrovisore in modo tale che l'asse ottico coincida con l'asse longitudinale del veicolo (figure (9.22) e (9.23)).

Figura 9.22. Videocamera installata dietro il parabrezza

Figura 9.23. Videocamera installata dietro il parabrezza

Il corretto posizionamento è stato ottenuto grazie all'utilizzo di una ventosa per videocamera che grazie agli snodi permette un facile, veloce e preciso orientamento (figura 9.24).

Figura 9.24. Ventosa con attacco 8mm per videocamera

La verifica preliminare è stata effettuata inizialmente connettendo la telecamera a un notebook e controllando che il campo visivo contenga la porzione di strada corretta.

Posizionamento sensori

La piattaforma inerziale esterna utilizzata come sensore yaw rate è stata posizionata il più possibile vicino al centro di massa dell'autovettura al fine di ottenere una corretta misura della velocità angolare della vettura intorno al proprio asse verticale. La figura (9.25) mostra il posizionamento del sensore all'interno dell'abitacolo nel suo involucro protettivo.

Figura 9.25. Posizione sensori inerziali

Connessione e posizionamento scheda embedded

La scheda embedded è stata posizionata in prossimità del cruscotto in modo tale che il display possa essere visibile dal passeggero anteriore (figura (9.27)). L'alimentazione è stata fornita tramite un adattatore DC/DC connesso alla presa 12V dell'accendisigari. L'alimentatore utilizzato permette di selezionare la tensione in uscita tra 1.5/3/4.5/5/6/9/12 Vdc e sopporta correnti fino a $3A$, è inoltre provvisto di protezione per i sovraccarichi, sovra-temperatura e corto circuiti.

Figura 9.26. Alimentatore DC/DC per accendisigari

Figura 9.27. Scheda embedded e alimentatore a bordo vettura

9.3.4 Test a bordo vettura

I test a bordo vettura sono stati effettuati sia su autostrada che su strada statale, di seguito sono riportate due immagini per ogni tipo di strada.

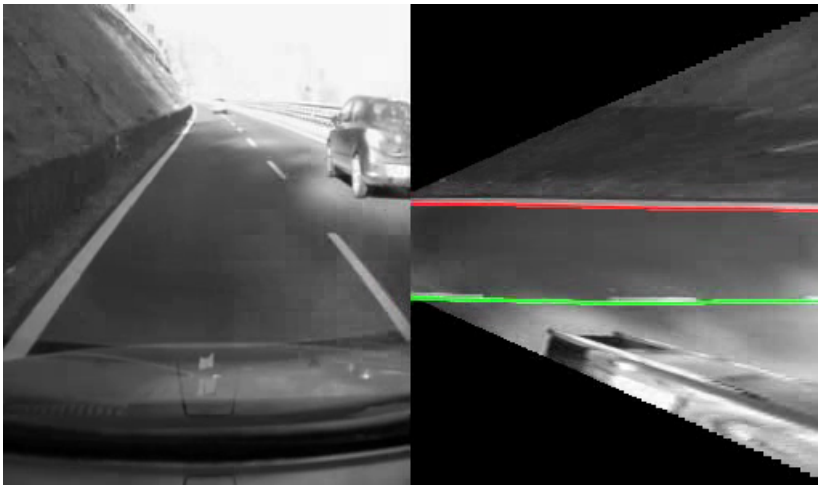

Figura 9.28. Test a bordo vettura Autostrada

Figura 9.29. Test a bordo vettura Autostrada

Figura 9.30. Test a bordo vettura strada Statale

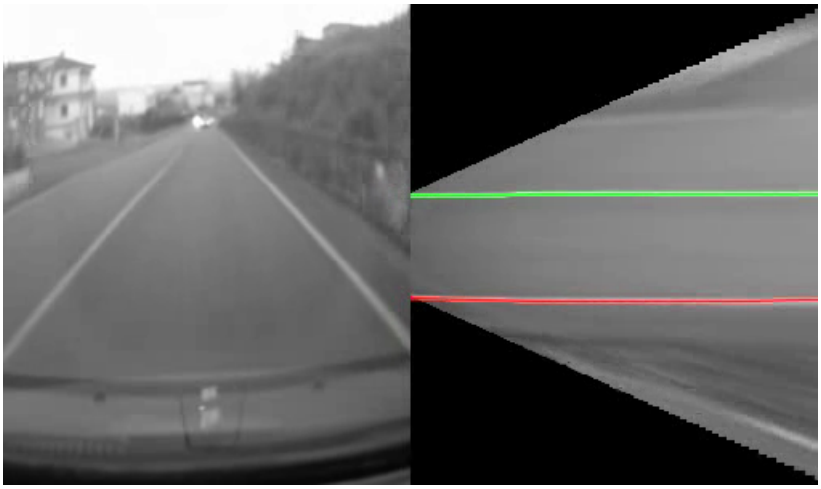

Figura 9.31. Test a bordo vettura strada Statale

Le prove hanno evidenziato il corretto funzionamento del sistema che ha identificato la segnaletica stradale orizzontale anche in presenza di ombre o di temporanei abbagliamenti. L'attivazione degli allarmi è risultata corretta sia in autostrada che su strada statale avvisando sempre il conducente con un anticipo sufficiente per eseguire le manovre correttive necessarie.

10

Conclusioni

In questo lavoro di tesi sono stati descritti in dettaglio metodologie di fusione dei dati per sistemi di avviso di superamento di corsia attigua basati sul calcolo del tempo di invasione di corsia t_{tlc}.
Dopo una breve introduzione, nel **Capitolo 2** è stata descritta la struttura ed il funzionamento di un classico sistema LDWS. Particolare attenzione è stata rivolta agli algoritmi di visione per l'identificazione e il tracking delle linee.
Un'approfondita descrizione del calcolo del tempo di invasione t_{tlc} è stata inserita nel **Capitolo 3**. Il calcolo del t_{tlc} necessita della conoscenza puntuale della posizione e velocità laterali del veicolo. Sono stati, quindi, presentati due approcci diversi: uno basato sull'utilizzo esclusivo dei dati provenienti dal sistema di visione, l'altro basato sul filtraggio di tali dati tramite algoritmi di fusione dei dati. In particolare questi ultimi, basandosi su modelli matematici del veicolo, *fondono* insieme i dati provenienti dalla visione (posizione laterale, orientazione del veicolo rispetto alla strada) con quelli dei sensori a bordo vettura (velocità longitudinale, velocità angolare delle ruote, angolo di sterzo).
Il **Capitolo 4** ha presentato una panoramica sui modelli matematici del veicolo su cui si basano gli algoritmi di fusione dei dati. In particolare, sono stati considerati quattro diversi modelli con complessità e caratteristiche differenti: *Cinematico sterzante*, *Cinematico differenziale*, *Cinematico yaw rate*, *Dinamico*.
Nel **Capitolo 5** sono stati descritti gli algoritmi di fusione dei dati presi in considerazione tra quelli presenti in letteratura: *Filtro Esteso di Kalman*, *Filtro di Kalman Unscented*.
Il **Capitolo 6** descrive le manovre di guida realizzate con il simulatore software Carsim necessarie al testing degli algoritmi sopra descritti.
I risultati delle simulazioni sono presentati nel **Capitolo 7**. L'analisi ha coinvolto ogni combinazione filtro-modello, valutando su tutti i percorsi di test l'errore commesso nella stima di posizione, velocità laterale e t_{tlc}. La stessa analisi, per confronto, è stata effettuata sulla stima calcolata con dati del solo sistema di visione. Sono state considerate tutte le possibili condizioni di guida:

velocità variabili, percorsi in autostrada, manovre brusche etc.. Il risultato di tutte queste simulazioni ha evidenziato che:

- L'algoritmo basato su filtro di Kalman Esteso e modello cinematico differenziale della vettura, pur non ottenendo i risultati migliori, risulta essere comunque un buon compromesso in termini di precisione e prestazioni considerando soprattutto il minor carico computazionale richiesto.
- L'algoritmo basato su filtro di Kalman Unscented e modello cinematico differenziale della vettura ha ottenuto i risultati migliori. La stima di posizione, velocità laterale e t_{tlc} è precisa e affidabile in ogni condizione di guida considerata.
- Risultati comparabili alla soluzione basata su modello cinematico differenziale sono stati ottenuti utilizzando il filtro di Kalman Unscented con modello cinematico yaw rate. Questo approccio ha mostrato un leggero vantaggio in termini di precisione durante simulazioni a velocità variabile.
- Entrambi gli algoritmi sono stati testati simulando situazioni di occlusione temporanea della telecamera per verificarne la robustezza. Nonostante nei test siano state inserite occlusioni di durata anche maggiore a 10 secondi gli algoritmi hanno continuato a fornire informazioni su posizione, velocità laterale e t_{tlc} con precisione anche in queste condizioni.
- I buoni risultati ottenuti hanno evidenziato la possibilità di utilizzo delle soluzioni basate su filtro di Kalman Unscented e modello cinematico differenziale o cinematico yaw rate per la realizzazione di un sistema di avviso di invasione di corsia attigua con ottime caratteristiche di robustezza e precisione.
- I risultati migliori si sono comunque registrati nei tratti a velocità costanti o leggermente variabili indicando che sistemi LWDS basati sui metodi di Data Fusion qui presentati sono maturi per percorsi su strade a scorrimento veloce, tipo autostrade o superstrade, in presenza di segnaletica orizzontale ben evidente e raggi di curvatura elevati. Ulteriori approfondimenti sono invece richiesti per ottenere identici buoni risultati su percorsi a più bassa velocità in presenza di alte accelerazioni e forti curvature.

Il **Capitolo 8** ha presentato i risultati ottenuti dagli algoritmi di fusione dei dati in caso di occlusione temporanea del sensore video. Lo studio effettuato conferma i notevoli vantaggi dell'uso di questi algoritmi per sistemi LDWS. É evidente, infatti, come un sistema basato solo sui dati del sistema video durante le temporanee occlusioni debba interrompere il supporto al guidatore, mentre integrandolo con algoritmi di data fusion riesca a tollerare occlusioni anche maggiori di 10 secondi.
Nel **Capitolo 9** sono state analizzate tutte le fasi necessarie all'implementazione del sistema LDWS su piattaforma hardware embedded e alle prove sperimentali effettuate. E' stata descritta l'architettura hardware utilizzata, con particolare riferimento alle caratteristiche del processore OMAP3530, e del programma LDWS scritto in codice C++ sia nella parte eseguita dal processore che quella su DSP. Successivamente sono riportate le prove sperimentali

in laboratorio sull'algoritmo LDWS eseguendo il software su piattaforma embedded e utilizzando i video e i dati forniti dal simulatore software Carsim. A conclusione delle prove in laboratorio è descritta la procedura seguita per la configurazione e l'installazione dell'hardware LDWS a bordo di una vettura di prova e sono inoltre inseriti i risultati ottenuti in un ambiente stradale reale. L'implementazione su piattaforma embedded dell'algoritmo LDWS ha superato positivamente tutte le verifiche funzionali in laboratorio sia dal punto di vista dell'hardware che del software. Le prove effettuate a bordo vettura hanno dimostrato il corretto funzionamento del sistema in condizioni di uso reale.

Riferimenti bibliografici

1. Martin E. Liggins, David L. Hall,James Llinas, “Handbook of multisensor data fusion : theory and practice”, 2nd ed., CRC Press, 2009
2. S. Majumder, S. Scheding, and H. Durrant-Whyte, “Multisensor data fusion for underwater navigation” Robotics and Autonomous Systems, vol. 35, pp. 97–108, 2001.
3. The Integrated Project PReVENT, http://www.prevent-ip.org/, 2008.
4. J. B. Romine and E. W. Kamen, “Modelling and fusion of radar and imaging sensor data for target tracking” Opt. Eng., vol. 35, no. 3, pp. 659–673, 1996.
5. Macci, D.; Boni, A.; Cecco, M.; Petri, D., “Multisensor Data Fusion”, IEEE Instrumentation and Measurement Magazine, Part 14, pp. 24-33, June 2008.
6. Luo, Ren C.; Chou, Y.; Chen O. , “Multisensor Fusion and Integration: Algorithms, Applications, and Future Research Directions”, Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation, pp. 1986-1991, August 5 2013 8, 2007, Harbin, China.
7. Cheng, H.; Zheng, N.; Zhang, X.; Qin, J.; Wetering, H. (2007), “Interactive Road Situation Analysis for Driver Assistance and Safety Warning Systems: Framework and Algorithms”, IEEE Transactions on Intelligent Transportation Systems, Vol. 8, No. 1, March 2007
8. D. E. Catlin, “Estimation, Control and the Discrete Kalman Filter” , In Applied Mathematical Sciences 71, Springer-Verlag, 1989.
9. Rezaei Ghahroudi, M.; Fasih, A. (2007), “A Hybrid Method in Driver and Multisensor Data Fusion, Using a Fuzzy Logic Supervisor for Vehicle Intelligence”, 2007 International Conference on Sensor Technologies and Applications (SENSORCOMM2007), pp. 393- 398, October 14-20, 2007, Valencia, Spain
10. M. Tsogas, A. Polychronopoulos, A. Amditis, “Unscented Kalman Filter Design for Curvilinear Motion Models Suitable for Automotive Safety Applications”, 7th International Conference on Information Fusion, 2005
11. Gustafsson, F. Gunnarsson, F.Bergman, N. Forssell, U. Jansson, J. Karlsson, R. Nordlund, “Particle filters for positioning, navigation, and tracking”, Signal Processing IEEE Transactions on, Vol. 50, No. 2, pp. 425-437, 2002.
12. “Annual Statistical Report 2008”, European Road Safety Observatory (ERSO), 2008.
13. “Incidenti Stradali - Anno 2007”, Istituto Nazionale di Statistica (ISTAT), 2008.

14. TRACE (TRaffic Accident Causation in Erurope)"Review of crash effectiveness of Intelligent Transport Systems" 2007 - Lane Departure Warning and Control
15. W. van Winsum, K.A. Brookhuis, D. de Waard, "A comparison of different ways to approximate time-toline crossing (TLC) during car driving" , Accident Analysis and Prevention 32, pp. 47-56, 2000.
16. S. Mammar, S. Glaser, M. Netto, "Time to lane crossing for lane departure avoindace: A theoretical study and experimantal setting" , IEEE Trans. on Intellingent Transportation Systsems 7(2), pp. 226-241, 2006.
17. Farrelly J., Wellstead P., "Estimation of Lateral velocity" ,Proceeding of the 1996 IEEE International conference on Control Applications, Dearborn, pp. 552- 557, 1996.
18. A. Y. Ungoren, H. Peng, H.E. Tseng, "A Study on Lateral Speed Estimation Methods" , Int. J. Vehicle Autonomous Systems, Vol. 2, Nos 1/2, pp. 126-144, 2004.
19. T. Witte, A. Wilson "Accuracy of non-differential GPS for the determination of speed over ground", Journal of Biomechanics, Vol. 37, pp. 1891–1898, 2004.
20. P. Muraca, P. Pugliese, G. Rocca, "Convergence analysis of an Extended Kalman Filter using sensor querying and intermittent observations (DRAFT)"
21. P. Muraca, P. Pugliese, "State estimation for distributed Parameter Systems under intermittent observation", 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand, December 9-11, 2009.