



Traffic Light Recognition

Autori: Colacicco Nunziamaria, La Torre Noemi, Sivo Giovanni;

nunziamaria.colacicco@studentmail.unicas.it,
noemi.latorre@studentmail.unicas.it,
giovanni.sivo@studentmail.unicas.it

Ingegneria Informatica - Curriculum Intelligenza Artificiale e Robotica,

Università degli Studi di Cassino e del Lazio Meridionale

18 luglio 2024

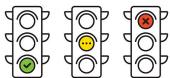
Abstract:

In questo elaborato presentiamo un metodo avanzato di elaborazione delle immagini utilizzando la libreria OpenCV, per la rilevazione e il riconoscimento dello stato semaforico nella città di Cassino (FR). L'approccio utilizzato impiega un accurato processo di pre-elaborazione delle immagini, segmentazione e analisi dei contorni per identificare con precisione ciascuno stato delle luci semaforiche. Attraverso tecniche di filtraggio, bilanciamento del colore e rilevamento dei bordi, il sistema è in grado di isolare e classificare correttamente i segnali luminosi dei semafori. Inoltre, è stato implementato un ulteriore algoritmo per la rilevazione dello stato semaforico anche in condizioni notturne. I risultati ottenuti dimostrano l'efficacia del metodo proposto nel contesto urbano, garantendo un rilevamento affidabile e accurato, anche a fronte di condizioni di illuminazione ridotta.

1 Introduzione

L'industria automobilistica sta evolvendo rapidamente verso veicoli dotati di avanzati sistemi di assistenza alla guida, equipaggiati con sensori come sonar, radar e telecamere. Questi sensori, insieme a tecnologie come sterzo e freni assistiti, trasformano le auto in veri e propri "robot" capaci di percepire e interpretare l'ambiente circostante[1]. Gli ingorghi stradali stanno aumentando notevolmente in tutto il mondo, soprattutto agli incroci, dove i semafori costringono spesso i conducenti ad accelerazioni improvvise quando il semaforo diventa verde; questo comportamento porta a un elevato consumo di carburante e a un aumento dell'inquinamento atmosferico. Inoltre l'esitazione dei conducenti di fronte ai semafori gialli può causare incidenti[2]. Per questo motivo, ricevere informazioni tempestive sullo stato dei semafori riduce il numero di frenate e accelerazioni improvvise, contribuendo anche a diminuire il numero di incidenti agli incroci[3]. È essenziale un'organizzazione intelligente del traffico, dove grazie ai semafori intelligenti si possono fornire supporto informativo sul traffico e facilitare decisioni più consapevoli da parte dei conducenti[4]. I metodi tradizionali di rilevamento dello

stato dei semafori, che utilizzano GPS, telecamere e sensori, presentano numerosi problemi e compromessi. Molti semafori non vengono rilevati a causa della differenza di tempo tra il rilevatore, il tracker, il classificatore, la telecamera e il tempo reale. Inoltre, la presenza di lampioni e altre fonti di luce può generare falsi positivi. L'occlusione parziale o totale dei semafori da parte di altri veicoli o oggetti, nonché le condizioni meteorologiche avverse o un'illuminazione eccessiva, rappresentano ulteriori sfide nel riconoscimento di quest'ultimo[5]. Negli ultimi anni, i progressi nel riconoscimento dei semafori hanno beneficiato enormemente dall'avvento delle reti neurali profonde (DNN) e delle tecniche avanzate di visione artificiale. Le reti neurali convoluzionali (CNN) sono state ampiamente adottate per l'estrazione automatica delle caratteristiche visive e la classificazione dei colori dei semafori, garantendo una maggiore precisione rispetto ai metodi tradizionali basati su caratteristiche progettate manualmente. Ricerche recenti hanno anche esplorato l'integrazione di approcci multi-sensoriali, combinando dati provenienti da telecamere, LiDAR [6] e sensori radar per migliorare la robustezza e l'affidabilità del riconoscimento dei semafori in condizioni ambientali variabili.



2 Obiettivo

L'obiettivo principale del nostro lavoro è stato sviluppare un sistema in grado di riconoscere lo stato semaforico, fornendo alle autovetture le istruzioni appropriate per capire l'azione da intraprendere durante la guida autonoma. Questo implica la capacità di identificare con precisione ciascuno dei tre stati del semaforo (verde, giallo e rosso) e di interpretare correttamente l'output corrispondente, ovvero, proseguire nella guida, rallentare o fermarsi. Per raggiungere questo obiettivo, sono state impiegate tecniche di elaborazione e analisi delle immagini, utilizzando in particolare la libreria OpenCV in C++, con ambiente di sviluppo Visual Studio Code 2022.

3 Area geografica di interesse

L'area geografica di interesse è il Centro della città di Cassino(FR), e i semafori presi in considerazione nel nostro dataset sono posizionati in 6 incroci come mostrato in Figura 1



Figura 1: Semafori centro città Cassino (FR).

Di seguito elenchiamo i video dei semafori presenti all'interno del nostro dataset con le rispettive coordinate geografiche:

Dataset Video semafori di giorno:

- Video1: 41°29'20.5"N 13°49'40.9"E
- Video2: 41°29'20.5"N 13°49'40.9"E
- Video3: 41°29'24.1"N 13°50'19.6"E
- Video4: 41°29'24.1"N 13°50'19.6"E
- Video5: 41°29'24.1"N 13°50'19.6"E

- Video6: 41°29'29.2"N 13°50'00.7"E
- Video7: 41°29'29.2"N 13°50'00.7"E
- Video8: 41°29'24.1"N 13°50'19.6"E
- Video9: 41°29'29.2"N 13°50'00.7"E
- Video10: 41°29'26.9"N 13°50'00.2"E
- Video11: 41°29'26.9"N 13°50'00.2"E
- Video12: 41°29'07.4"N 13°49'54.1"E
- Video13: 41°29'07.4"N 13°49'54.1"E
- Video14: 41°29'07.4"N 13°49'54.1"E
- Video15: 41°29'29.2"N 13°50'00.7"E
- Video16: 41°29'24.1"N 13°50'19.6"E
- Video17: 41°29'26.9"N 13°50'00.2"E
- Video18: 41°29'29.2"N 13°50'00.7"E
- Video19: 41°29'33.7"N 13°49'44.0"E
- Video20: 41°29'33.7"N 13°49'44.0"E
- Video21: 41°29'33.7"N 13°49'44.0"E

Dataset Video semafori di notte:

- Video1: 41°29'26.9"N 13°50'00.2"E
- Video2: 41°29'24.1"N 13°50'19.6"E
- Video3: 41°29'07.4"N 13°49'54.1"E
- Video4: 41°29'20.5"N 13°49'40.9"E
- Video5: 41°29'24.1"N 13°50'19.6"E
- Video6: 41°29'07.4"N 13°49'54.1"E
- Video7: 41°29'29.2"N 13°50'00.7"E
- Video8: 41°29'24.1"N 13°50'19.6"E
- Video9: 41°29'26.9"N 13°50'00.2"E
- Video10: 41°29'33.7"N 13°49'44.0"E
- Video11: 41°29'24.1"N 13°50'19.6"E
- Video12: 41°29'29.2"N 13°50'00.7"E
- Video13: 41°29'20.5"N 13°49'40.9"E



- Video14: 41°29'26.9"N 13°50'00.2"E
- Video15: 41°29'29.2"N 13°50'00.7"E
- Video16: 41°29'33.7"N 13°49'44.0"E
- Video17: 41°29'29.2"N 13°50'00.7"E
- Video18: 41°29'07.4"N 13°49'54.1"E
- Video19: 41°29'26.9"N 13°50'00.2"E
- Video20: 41°29'26.9"N 13°50'00.2"E
- Video21: 41°29'33.7"N 13°49'44.0"E
- Video22: 41°29'26.9"N 13°50'00.2"E

4 Configurazione Camera

Per la registrazione dei video, abbiamo utilizzato una fotocamera reflex Nikon D3200 da 24,2 megapixel, con risoluzione video di (1920x1080) pixel a 25 fps per i video di giorno, mentre per i video in notturna una risoluzione di (848x464) pixel a 30 fps. Le riprese sono state effettuate da una posizione fissa, con un angolo di visualizzazione di circa 60 gradi, per catturare l'intero ambiente circostante. La distanza di ripresa dal semaforo varia a secondo dei video. Questi ultimi sono salvati in formato .mp4.

5 Metodologia

5.1 Architettura dell'algoritmo

In Figura 2 viene mostrata la pipeline dell'algoritmo implementata.

5.2 Implementazione dell'algoritmo

5.2.1 Pre-processing

La conversione dallo spazio colore RGB a HSV è utilizzata per isolare i colori in modo più efficace e robusto rispetto all'uso diretto del modello RGB. La separazione nei tre canali H (hue), S (saturation) e V (brightness) permette di operare individualmente su ciascun canale e migliorare le successive fasi per la rilevazione del semaforo.

Per migliorare la visibilità dei dettagli nelle immagini, soprattutto in condizioni di illuminazione non ottimali, abbiamo applicato una correzione gamma. Quest'ultima è implementata con una trasformazione non lineare dei valori di intensità dei pixel. La formula utilizzata è:

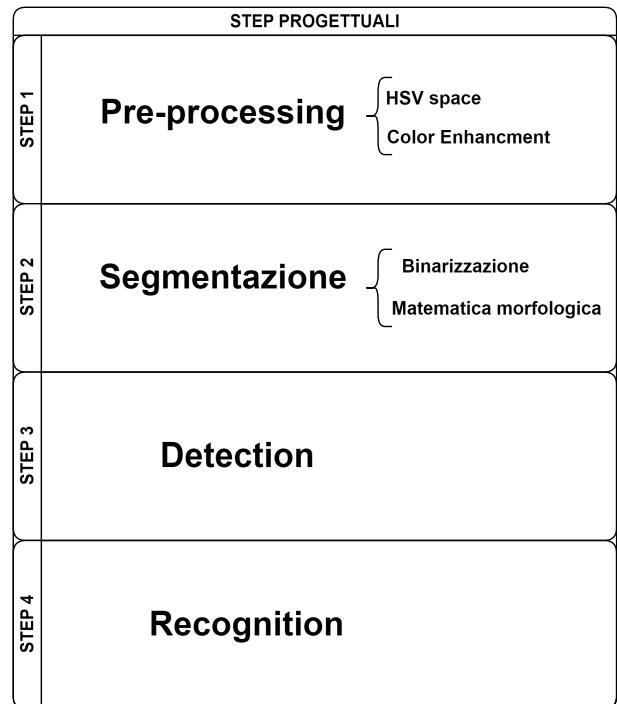


Figura 2: Architettura algoritmo

$$T(k) = c \cdot k^\gamma$$

$$c = (L - 1)^{1-\gamma}$$

dove c è una costante per preservare il range di intensità originale [0, L-1].

Nel nostro caso specifico, abbiamo applicato la trasformazione gamma sul canale (S) si HSV con un fattore

$$\gamma = 0.2$$

incrementando così la luminosità dei pixel scuri e migliorandone la visibilità dei dettagli.

Come mostrato la Figura 3 notiamo che :

Gamma = 1: La trasformazione è lineare e l'immagine di output è identica a quella di input.

Gamma < 1: La trasformazione aumenta la luminosità dell'immagine. I pixel scuri vengono resi più chiari, migliorando la visibilità dei dettagli nelle aree scure. La curva è concava verso l'alto, indicando un aumento della luminosità.

Gamma > 1: La trasformazione diminuisce la luminosità dell'immagine. I pixel chiari vengono resi più scuri, migliorando il contrasto nelle aree chiare e riducendo la sovraesposizione. La curva è convessa verso il basso, indicando una diminuzione della luminosità.

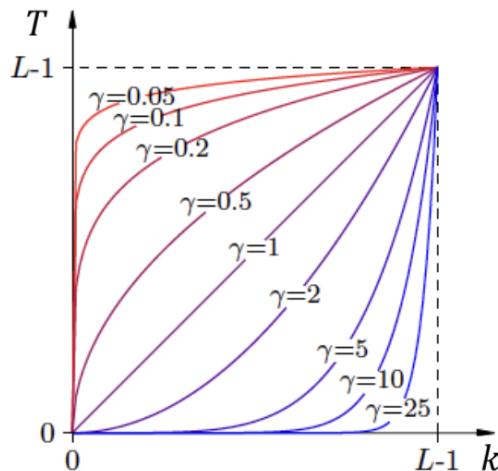


Figura 3: Trasformazione Gamma

Inoltre, utilizziamo per aumentare la luminosità del canale V (brightness) la funzione di OpenCV convertTo mediante un fattore di moltiplicazione (alpha) impostato a 2, ed un termine di bias (beta) impostato a 0, ovvero andiamo a raddoppiare i valori dei pixel nel canale V, aumentando la luminosità complessiva dell'immagine.

5.2.2 Segmentazione

La segmentazione è stata utilizzata per isolare le luci del semaforo dal resto dell'immagine. Questo processo permette di identificare e distinguere chiaramente le luci accese. Attraverso la binarizzazione riduciamo l'immagine a due valori di pixel (bianco e nero) in modo da poter gestire solo le regioni di interesse all'interno dell'immagine principale. La soglia di binarizzazione può aiutare a filtrare il rumore o i dettagli irrilevanti nell'immagine. Solo i pixel che superano un certo livello di intensità (soglia) vengono considerati, eliminando così le variazioni di colore e luminosità che non sono pertinenti all'analisi. Utilizziamo una binarizzazione con Triangle sul canale V di HSV, in modo da determinare automaticamente una soglia di binarizzazione ottimale per il nostro dataset di video. In Figura 4, Figura 5 e Figura 6 mostriamo rispettivamente l'output della binarizzazione di alcuni frame dei video 1, video 7 e video 10.

Successivamente, abbiamo applicato operazioni morfologiche sull'immagine binaria per migliorare ulteriormente la qualità della segmentazione, e la rilevazione della luce semaforica che altrimenti non sarebbe stata rilevata a causa di protrusioni e per eliminare eventuali falsi positivi. In particolare nella tabella in Figura 7, mostriamo i risultati nell'utilizzo della matematica morfologica applicata al video 1, al video 7 e al video 10.



Figura 4: Video 1 immagine binarizzata con Triangle video 1



Figura 5: Binarizzazione con Triangle video 7

be stata rilevata a causa di protrusioni e per eliminare eventuali falsi positivi. In particolare nella tabella in Figura 7, mostriamo i risultati nell'utilizzo della matematica morfologica applicata al video 1, al video 7 e al video 10.

L'Erosione (Erode) elimina i piccoli pixel bianchi isolati (rumore) presenti nell'immagine binarizzata. Questo è utile per ridurre le false rilevazioni che vogliamo escludere, inoltre riduce leggermente le dimensioni degli oggetti luminosi segmentati, separando gli oggetti vicini che potrebbero essere connessi. L'elemento strutturante che utilizziamo è un SE di:

Forma Ellittica: Poichè ci interessa preservare meglio la forma degli oggetti arrotondati rispetto a un elemento strutturante rettangolare.

Dimensioni 7x7: Poichè è in grado di rimuovere efficacemente i piccoli rumori e separare gli oggetti vicini, ma non così grande da eliminare completamente gli oggetti di interesse.

L'operazione successiva è un' **Apertura (Open)** che serve per eliminare ulteriori protrusioni ancora presenti nell'immagine binarizzata. Consiste in un'erosione seguita da una dilatazione, rimuovendo ulte-



Figura 6: Binarizzazione con Triangle video 10

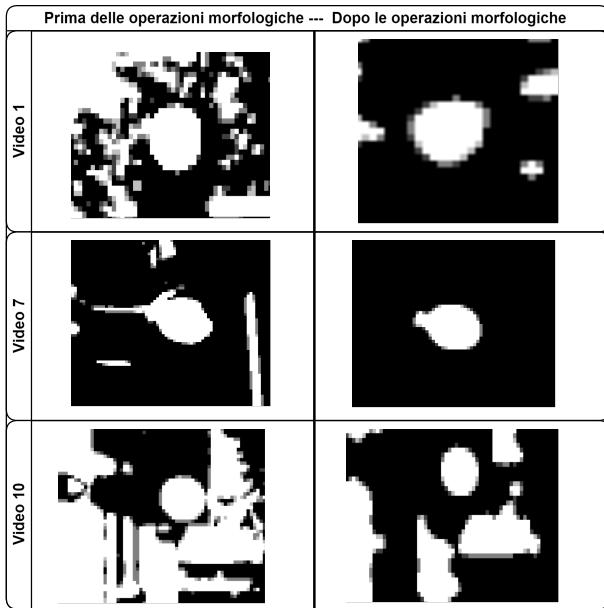


Figura 7: Risultati operazioni morfologiche

riamente i piccoli artefatti non connessi, purificando ulteriormente l'immagine. Nell'operazione di Open, dopo aver effettuato l'erosione, la dilatazione aiuta a ripristinare le dimensioni originali degli oggetti significativi (i cerchi della luce semaforica) preservando le caratteristiche importanti per l'analisi successiva. Anche in questo caso abbiamo utilizzato uno SE di:

Forma Ellittica: Come sopra, preserva meglio la forma arrotondata degli oggetti.

Dimensioni 5x5: Sufficientemente piccolo da non eliminare gli oggetti segmentati rilevanti, ma abbastanza grande da garantire la separazione e la pulizia efficaci delle strutture più piccole.

5.2.3 Detection

Per rilevare e estrarre i contorni degli oggetti nell'immagine binarizzata, abbiamo utilizzato la funzione `cv::findContours` di OpenCV con l'opzione `RETR_-`

`LIST`, che recupera tutti i contorni degli oggetti. In Figura 8 e Figura 9, mostriamo i contorni di tutti gli oggetti rilevati che sono stati disegnati con la funzione di Open CV `cv::drawContours` in un particolare frame per il video 1 e per il video 7.



Figura 8: Video 1 Contorni oggetti rilevati con `findContours`



Figura 9: Video 7 Contorni oggetti rilevati con `findContours`

Il set iniziale di contorni generato utilizzando la funzione `cv::findContours` rileva anche contorni che non corrispondono ai semafori. Pertanto, applichiamo una serie di criteri di filtraggio per eliminare i falsi positivi e mantenere solo i potenziali semafori candidati. Gli step di filtraggio eseguiti sono mostrati graficamente in Figura 10.

L'ordine delle operazioni di filtraggio utilizzata segue una logica dal generale al particolare, partendo da filtri geometrici semplici e veloci, passando per controlli più specifici basati su proprietà geometriche e cromatiche, fino a raggiungere una valutazione complessiva della qualità del rilevamento. Analizziamoli nel dettaglio:

Vincoli sul Rettangolo di Contorno: Ogni contorno viene prima racchiuso all'interno di un rettangolo di

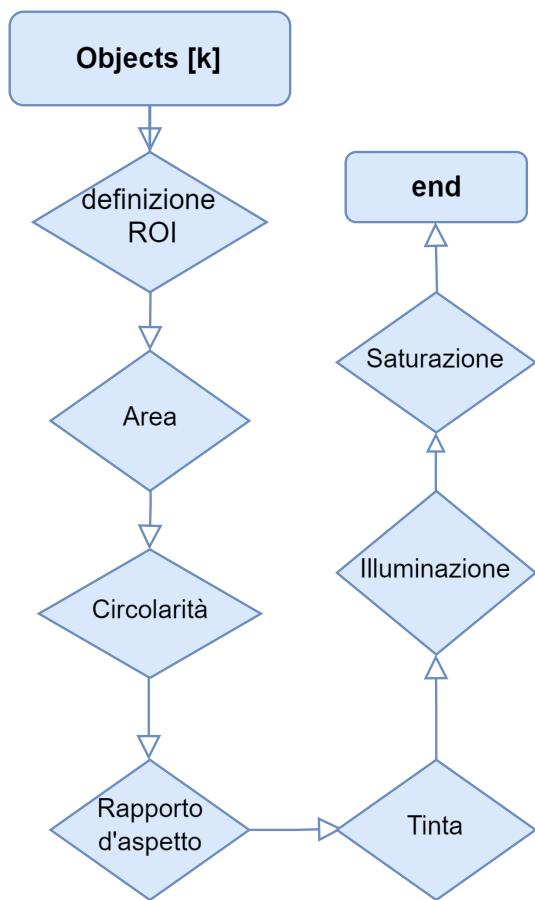
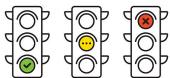


Figura 10: Operazioni di filtraggio

contorno utilizzando la funzione `cv::boundingRect`. La larghezza del rettangolo di contorno viene verificata rispetto a valori minimi e massimi predefiniti (`min_width` e `max_width`). I contorni che non soddisfano questi vincoli vengono esclusi dal controllo successivo.

Calcoli di Area e Perimetro: Per ciascun contorno rimanente, calcoliamo l'area del rettangolo di contorno (`area_brect`), l'area dell'oggetto (`area`) e il perimetro dell'oggetto (`perim`). Utilizziamo rispettivamente per calcolare area e perimetro dell'oggetto le funzioni OpenCv `cv::contourArea` e `cv::arcLength`.

Rapporto di Area Rettangolare: Viene calcolato il rapporto tra l'area del contorno dell'oggetto e l'area del rettangolo che racchiude l'oggetto. Gli oggetti che hanno un contorno con un rapporto al di fuori dell'intervallo (0.55-0.85) vengono scartati. Questo passaggio aiuta a eliminare forme che non si adattano strettamente al loro rettangolo di contorno, come quelle allungate o irregolari.

Vincoli sull'Area: L'area del contorno dell'oggetto

viene ulteriormente vincolata a rientrare in un intervallo specificato (`min_area` a `max_area`). Questo assicura che vengano considerati solo i contorni di una certa dimensione, ovvero quelli con l'area che più si avvicina a quella dei semafori.

Verifica della Circolarità: La circolarità di ciascun contorno viene calcolata utilizzando la formula

$$c = \frac{4\pi \cdot \text{area}}{\text{perim}^2}$$

I contorni degli oggetti con un valore di circolarità inferiore a una certa soglia (`min_circularity`) vengono esclusi. Questo aiuta a mantenere solo quei contorni che sono approssimativamente circolari, caratteristica tipica dei semafori.

Verifica del Rapporto di Aspetto: Viene calcolato il rapporto di aspetto del rettangolo di contorno (larghezza divisa per altezza). I contorni con un rapporto di aspetto al di fuori dell'intervallo (0.7-1.6) vengono scartati. Questo criterio assicura che i rettangoli di contorno non siano né troppo allungati né troppo schiacciati.

Verifica della Gamma di Tonalità: Viene calcolato il valore medio della tonalità all'interno di ciascun contorno dall'immagine in rappresentazione HSV. La tonalità media viene quindi raddoppiata per tenere conto dell'intera scala di tonalità della ruota HSV in OpenCV. I contorni vengono filtrati in base al fatto che la loro tonalità media rientri negli intervalli predefiniti per le tonalità di verde, giallo e rosso, che corrispondono ai colori tipici dei semafori.

Regolazione della Tonalità dei Pixel: Per i contorni che superano la verifica della tonalità, i valori di tonalità dei pixel all'interno del rettangolo di contorno vengono assegnati ad un livello di tonalità:

- I pixel con una tonalità nell'intervallo del verde vengono impostati a un valore di tonalità di 110.
- I pixel con una tonalità nell'intervallo del giallo vengono impostati a un valore di tonalità di 30.
- I pixel con una tonalità nell'intervallo del rosso vengono impostati a un valore di tonalità di 0.

Vincoli sull'Illuminazione e Saturazione: Viene calcolato il valore medio dell'illuminazione (canale V di HSV) e della saturazione (canale S di HSV) all'interno di ciascun contorno. I contorni con un'illuminazione media inferiore a un minimo predefinito (`min_illumination`) o una saturazione media inferiore a



un minimo predefinito (`min_saturation`) vengono esclusi. Questi vincoli aiutano ad assicurare che gli oggetti rilevati siano sufficientemente luminosi e saturi, caratteristiche tipiche dei semafori accesi.

Calcolo del Punteggio di Rilevamento: Infine, per ciascun contorno che supera tutti i filtri sopra elencati, viene calcolato un punteggio di qualità (`godness`, g) basato su circolarità, illuminazione, saturazione e area normalizzata. Viene calcolato un punteggio finale (`score`) sottraendo all'indice di bontà (`godness`) una componente di distanza di tonalità normalizzata. Questo punteggio indica la probabilità che il contorno sia un semaforo, con punteggi più alti che rappresentano una probabilità maggiore. I contorni rimanenti, che superano tutti i criteri di filtraggio, vengono memorizzati in `frame_detections_objects` come oggetti pertinenti alle caratteristiche di un semaforo. I range numerici dei parametri utilizzati sono mostrati nella tabella in Figura 11.

Parametri
$100 < \text{area} < 2500$
$\text{circolarità} > 0.65$
$0.7 < \text{rapporto d'aspetto} < 1.6$
$\text{illuminazione} > 252$
$\text{saturazione} > 100$

Figura 11: Parametri

5.2.4 Recognition

Dopo aver filtrato i contorni iniziali nella fase di detection, abbiamo calcolato delle metriche di qualità per ciascun oggetto. Queste metriche vengono ordinate in ordine decrescente per considerare prima i contorni con le migliori probabilità di rappresentare una luce di un semaforo.

Consideriamo solo i primi due candidati con un punteggio di qualità superiore a 0.7 per ridurre ulteriormente il numero di contorni da analizzare.

Per ciascun contorno selezionato, calcoliamo il colore medio (Hue) e assegniamo un colore (verde, giallo o rosso) basato su intervalli predefiniti presi dalla ruota HSV. Il punteggio di qualità viene ricalcolato per assicurarsi che corrisponda al punteggio ordinato inizialmente. Si memorizza la rilevazione e si disegna il contorno sull'immagine di output.

Per migliorare la robustezza del sistema ai cambiamenti di stato del semaforo, viene mantenuta una finestra temporale di rilevazioni. Questa finestra raccolge i risultati dei voti ponderati dei colori nei frame recenti. All'interno di questa finestra, si esegue una votazione maggioritaria: il colore che appare più frequentemente con il punteggio più alto nei frame recenti viene determinato come il colore attuale del semaforo. Questo colore viene quindi visualizzato sull'immagine di output.

Il codice utilizza un approccio multi-step per identificare e determinare il colore delle luci dei semafori all'interno di una sequenza video. L'uso di filtri geometrici e cromatici, insieme a una combinazione di voto ponderato per il colore del frame e voto maggioritario nel tempo, garantisce un rilevamento accurato e robusto delle luci semaforiche.

In figura 12 mostriamo il riconoscimento semaforico di giorno per il video 1

In figura 13 mostriamo il riconoscimento semaforico di giorno per il video 7

6 CASO PARTICOLARE : Video notturni

Nel contesto del riconoscimento dei semafori durante le ore notturne, è stato sviluppato un nuovo approccio progettuale per garantire il rilevamento dei semafori anche in condizioni di scarsa illuminazione. Sebbene i passaggi implementativi siano strutturati in modo simile a quelli utilizzati di giorno, alcune modifiche significative sono state apportate per adattarsi alle sfide specifiche di questo scenario.

Dettagli Implementativi:

Pre-processing Iniziale: Durante la fase di pre-processing, viene applicato un filtro di blur gaussiano ai frame iniziali RGB per migliorare la qualità dell'immagine. In particolare, è stato utilizzato un kernel di dimensioni (31x31) per ridurre il rumore e migliorare la nitidezza delle immagini notturne.

```
cv::GaussianBlur(frame_RGB, frame_-  
blurred_out, cv::Size(31, 31),  
0);
```



Figura 12: Riconoscimento semaforico di giorno per il video 1



Figura 13: Riconoscimento semaforico di giorno per il video 7

Trasformazione Gamma: Per migliorare la visibilità delle caratteristiche luminose nei video notturni sul canale V, è stato aumentato a 17 il fattore alpha di intensità della trasformazione gamma. Questo aiuta a enfatizzare i dettagli delle luci dei semafori, rendendoli più distinguibili in condizioni di illuminazione ridotta.

```
gamma_transformation(frame_hsv_chans[2] ,
frame_hsv_chans[2] , 17);
```

Binarizzazione con Metodo Otsu: La binarizzazione dei frame è stata eseguita utilizzando il metodo di Otsu, il quale calcola automaticamente una soglia ottimale basata sull'istogramma dell'immagine. Questo approccio è particolarmente efficace per separare le regioni di interesse, come le luci dei semafori, dallo sfondo nelle condizioni notturne.

```
cv::threshold(frame_hsv_chans[2] , frame_-
binarized_out , 0 , 255 , cv::THRESH_BINARY |
cv::THRESH_OTSU);
```

Operazione Morfologica: Per migliorare la coerenza e la forma dei segmenti binarizzati, è stata eseguita un'unica operazione di apertura morfologica con uno SE di forma circolare e con un kernel (12,12). L'Open consiste in un'erosione seguita da una dilatazione. L'erosione cancella gli oggetti dove non entra lo SE e la dilatazione restaura gli elementi sopravvissuti che si vuole ripristinare.

```
cv::morphologyEx(frame_binarized_-
```



Figura 14: Binarizzazione con Otsu video 7



Figura 15: Binarizzazione con Otsu video 10

```
out,frame_morph_out , cv::MORPH_OPEN,
kernel); con kernel di (12,12)
```

In Figura 14 e in Figura 15 mostriamo la Binarizzazione con Otsu per il video 7 ed il video 10. Dato che dalle immagini binarizzate risulta evidente che gli unici oggetti presenti sono i semafori, non è necessario escludere ulteriori oggetti durante la fase di riconoscimento. Pertanto, non è più necessario



Figura 16: Riconoscimento semafori di notte per il video 7



Figura 17: Riconoscimento semafori di notte per il video 10

calcolare e ordinare le metriche di qualità dei contorni, né selezionare solo i primi due candidati con un punteggio superiore a 0.7 come veniva fatto nella fase di riconoscimento diurna.

Parametri Specifici: Durante la fase di adattamento per il riconoscimento notturno dei semafori, sono stati regolati i parametri relativi alle dimensioni minime e massime delle regioni di interesse, i parametri utilizzati nella fase di filtraggio, nonché i range dei colori considerati. Questi aggiustamenti sono cruciali per garantire un'accurata identificazione delle luci dei semafori in condizioni di bassa luminosità.

In figura 16 mostriamo un output con ciascuno dei tre stati semaforici rilevati per il video 7

In figura 17 mostriamo un output con ciascuno dei tre stati semaforici rilevati per il video 10

7 Risultati e valutazioni

Il sistema di riconoscimento dei semafori ha dimostrato di funzionare efficacemente per i video analizzati. Il metodo di rilevamento e classificazione basato su filtri geometrici e cromatici, combinato con il sistema di voto ponderato e la votazione maggioritaria nel tempo, ha garantito un rilevamento accurato e robusto delle luci semaforiche. Un ulteriore aspetto positivo è stato rilevato durante il test dell'algoritmo con un video in movimento, il Video14. Anche in questo scenario dinamico, il sistema ha mantenuto la sua capacità di riconoscere e interpretare correttamente l'output delle luci semaforiche. Questo

risultato evidenzia la robustezza dell'algoritmo e la sua adattabilità a situazioni di traffico reale, dove la telecamera non è fissa ma in movimento, come potrebbe essere il caso di un veicolo in marcia. Tuttavia, è stato osservato che in alcuni video contenenti due semafori, per alcuni frame, uno dei due semafori non viene rilevato ininterrottamente. Nonostante questa limitazione, il sistema continua a funzionare correttamente, mantenendo l'accuratezza complessiva del riconoscimento del semaforo principale. Questo risultato suggerisce che il sistema è affidabile e capace di gestire situazioni complesse senza compromettere la funzionalità globale. In conclusione, il riconoscimento semaforico si è rivelato robusto e affidabile per tutti i video analizzati, con prestazioni che rimangono soddisfacenti anche in presenza di due semafori, garantendo così un funzionamento continuo e preciso anche a fronte di condizioni di illuminazione ridotta.

8 Possibili Sviluppi Futuri

Eventuali sviluppi futuri possono contribuire a rendere il sistema di riconoscimento dei semafori ancora più efficace, affidabile e versatile, migliorando la sicurezza e l'efficienza del traffico stradale.

- **Espansione del Dataset di Addestramento:** Arricchire il dataset di addestramento con immagini e video provenienti da diverse città e condizioni di traffico per migliorare la generalizzazio-



ne del sistema.

Collaborare con enti pubblici e privati per raccogliere dati reali che riflettano una vasta gamma di scenari di traffico.

- **Rilevamento di Semafori Pedonali e per Biciclette:**

Estendere il sistema per rilevare non solo i semafori stradali per veicoli, ma anche quelli per pedoni e ciclisti.

Integrare il riconoscimento dei segnali acustici associati ai semafori pedonali per migliorare l'assistenza ai non vedenti.

- **Ottimizzazione dell'Algoritmo di Votazione Ponderata:**

Considerare l'implementazione di tecniche di fusione dei dati provenienti da più telecamere o sensori per aumentare la precisione del riconoscimento.

- **Integrazione con Sistemi di Navigazione e Veicoli Autonomi:**

Collegare il sistema di riconoscimento dei semafori con i sistemi di navigazione per fornire informazioni in tempo reale agli automobilisti.

Integrare il sistema nei veicoli autonomi per migliorare la capacità di questi veicoli di interpretare correttamente i segnali stradali e prendere decisioni di guida sicure.

- **Implementazione di Algoritmi di Apprendimento Automatico:**

Integrare tecniche di apprendimento profondo (deep learning) per migliorare l'accuratezza e la robustezza del sistema.

Utilizzare reti neurali convoluzionali (CNN) per il rilevamento e la classificazione dei semafori, permettendo al sistema di adattarsi meglio a diverse condizioni ambientali.

- [3] Chen X. "Sistema di rilevamento abilitato dal modello ibrido adattivo (HMSS) per la stima mobile a grana fine dell'inquinamento atmosferico". In: *IEEE Trans. Mob. Comput.* 21 (2020), pp. 1927–1944.
- [4] Elsagheer S. e AlShalfan K. "Sistema di gestione del traffico intelligente basato sull'Internet dei veicoli (IoV)". In: *J. Avv. Trasporto* (2021).
- [5] Ziyue L. et al. "Un algoritmo migliorato di riconoscimento dei semafori per la guida autonoma in scenari complessi". In: *Int. J. Distrib. Sens. Netw.* (2021).
- [6] Dennis Sprute et al. "3D-LiDAR-based Pedestrian Detection for Demand-Oriented Traffic Light Control". In: *International Conference on Industrial Informatics* (2023).

Riferimenti bibliografici

- [1] Nathaniel Fairfield e Chris Urmson. "Traffic Light Mapping and Detection". In: *Google, Inc.* () .
- [2] Laécio R. et al. "Valutazione delle prestazioni dei semafori cooperativi intelligenti nelle ret". In: *VANET. Int. J. Comput. Sci. Eng.* 24 (2020), pp. 276–289.