Università degli Studi di Catania

Corso di Multimedia

Video Stabilization

Confronto tra approcci classici e deep learning per la stabilizzazione video

Studente: Salvatore Iurato

Febbraio 2026

Indice

1 Introduzione

1.1 Motivazione

La stabilizzazione video è uno dei problemi fondamentali nell'elaborazione digitale di sequenze d'immagini. Quando un video viene acquisito da una telecamera tenuta a mano, montata su un veicolo in movimento o soggetta a vibrazioni meccaniche, il risultato è una sequenza affetta da *jitter*: piccole oscillazioni ad alta frequenza della posizione e dell'orientamento del sensore che rendono il filmato visivamente fastidioso e, in applicazioni più sofisticate, difficile da analizzare automaticamente.

Il problema è rilevante in numerosi contesti pratici: dalla videografia amatoriale ai sistemi di sorveglianza, dalla guida autonoma alla robotica, fino all'analisi sportiva e alle riprese d'azione. In tutti questi scenari è desiderabile separare il *moto intenzionale* della telecamera (panoramiche, zoom, inseguimento di un soggetto) dall'instabilità indesiderata, attenuando quest'ultima senza alterare il contenuto informativo del video.

1.2 Obiettivi del Progetto

Questo progetto ha due obiettivi principali:

1. **Implementare una pipeline completa di stabilizzazione video** che comprenda la stima del movimento inter-frame, lo smoothing della traiettoria stimata e la conseguente correzione geometrica dei frame.

2. **Confrontare sistematicamente** due famiglie di approcci alla stima del moto — metodi classici basati su feature (SIFT) e metodi deep learning (RAFT) — in combinazione con due strategie di smoothing della traiettoria: la Media Mobile e il Filtro di Kalman.

Il confronto viene condotto in modo rigoroso attraverso un insieme di metriche quantitative — RMS displacement, jitter reduction, stability score e fidelity score — i cui risultati sono integrati in una dashboard interattiva realizzata con Streamlit, che consente di esplorare in tempo reale l'effetto di ciascun parametro.

1.3 Approccio Metodologico

La pipeline implementata segue uno schema a tre stadi:

1. **Motion Estimation**: stima del moto globale (dx_t, dy_t, $d\theta_t$) tra coppie di frame consecutivi, utilizzando alternativamente SIFT o RAFT.

2. **Trajectory Smoothing**: integrazione delle stime incrementali in una traiettoria cumulativa $\mathbf{p}_t = \sum_{i=1}^{t}(dx_i, dy_i, d\theta_i)$, seguita da una fase di smoothing con Media Mobile o Filtro di Kalman per ottenere la traiettoria target $\hat{\mathbf{p}}_t$.

3. **Frame Correction**: applicazione della trasformazione affine di correzione $\mathbf{c}_t = \hat{\mathbf{p}}_t - \mathbf{p}_t$ a ogni frame, con zoom adattivo per eliminare i bordi neri introdotti dal warping.

1.4 Struttura della Relazione

La relazione è organizzata come segue. La Sezione 2 illustra il background teorico dei metodi utilizzati: optical flow, SIFT, RAFT, Media Mobile e Filtro di Kalman. La Sezione 3 descrive le scelte implementative e l'architettura del sistema. La Sezione 4 presenta il setup sperimentale e i risultati ottenuti. La Sezione 5 discute i risultati e traccia le conclusioni.

2 Background Teorico

Per comprendere le scelte progettuali del sistema realizzato è opportuno ripercorrere i fondamenti teorici su cui si basa ciascun componente della pipeline. Questa sezione illustra dapprima il concetto di optical flow e i due metodi di stima del moto adottati — SIFT e RAFT — e successivamente le tecniche di smoothing della traiettoria, concludendo con una descrizione delle metriche di valutazione impiegate.

2.1 Optical Flow e Stima del Moto Globale

L'optical flow descrive il campo di velocità apparente dei pixel tra due frame consecutivi di una sequenza video. In modo formale, dato il valore di intensità $I(x, y, t)$ di un pixel alla posizione (x, y) all'istante t, l'ipotesi di conservazione della luminosità impone che lo stesso punto fisico della scena abbia la medesima intensità nell'immagine successiva:

$$I(x, y, t) = I(x + dx,\ y + dy,\ t + dt). \tag{1}$$

Sviluppando in serie di Taylor al primo ordine e dividendo per dt, si ottiene la cosiddetta *equazione del flusso ottico*:

$$I_x\, u + I_y\, v + I_t = 0, \tag{2}$$

dove (u, v) è la velocità del pixel e I_x, I_y, I_t sono le derivate parziali dell'intensità. Poiché l'equazione (2) è una sola equazione in due incognite, il problema è sottoderminato: per risolverlo occorre imporre vincoli aggiuntivi, che differenziano i diversi approcci esistenti in letteratura.

Nella pratica della stabilizzazione video non è necessario calcolare il flusso denso per tutti i pixel: è sufficiente stimare la *trasformazione globale* tra due frame, ossia la tripla $(dx, dy, d\theta)$ che descrive la traslazione e la rotazione della telecamera. Questo problema viene ricondotto alla stima di una trasformazione affine parziale (similarità) tra un insieme di corrispondenze puntuali, individuate con metodi diversi a seconda dell'approccio scelto.

2.2 SIFT: Scale-Invariant Feature Transform

Proposto da Lowe nel 2004, SIFT è uno dei descrittori di feature locali più diffusi e studiati in computer vision. Il suo principio di funzionamento si articola in due fasi principali: il rilevamento dei keypoint e il calcolo del descrittore.

Il rilevamento sfrutta la *scale-space pyramid*: l'immagine viene convoluta con filtri gaussiani a scala crescente, dopodiché si calcolano le differenze tra octave adiacenti (Difference of Gaussians, DoG). I punti di massimo e minimo locali nello spazio DoG —- sia spazialmente che lungo la dimensione di scala — corrispondono a strutture dell'immagine stabili e ripetibili. Ogni keypoint viene localizzato con precisione sub-pixel e gli viene assegnato un orientamento dominante calcolato sull'istogramma dei gradienti nel suo intorno, garantendo l'invarianza alla rotazione.

Il descrittore è un vettore di 128 dimensioni che codifica la distribuzione dei gradienti in una regione 4×4 attorno al keypoint, suddivisa in celle ciascuna con un istogramma ad 8 bin. Questa rappresentazione è robusta a piccole variazioni di illuminazione, traslazioni locali e deformazioni geometriche moderate.

Per stimare il moto tra due frame, si estraggono i keypoint SIFT da entrambe le immagini, si calcolano i descrittori e si effettua il matching con un *Brute-Force Matcher* sulla norma L_2. Al fine di filtrare le corrispondenze ambigue, si applica il *ratio test di Lowe*: una corrispondenza viene accettata solo se il rapporto tra la distanza al primo vicino più prossimo e quella al secondo è inferiore a una soglia (tipicamente 0.75). Le corrispondenze superstiti vengono poi passate a un estimatore RANSAC per calcolare la trasformazione affine parziale, che restituisce la stima di $(dx, dy, d\theta)$ in modo robusto rispetto agli outlier prodotti da oggetti in movimento in primo piano.

Il principale limite di SIFT in questo contesto è la sua dipendenza dalla presenza di strutture texturizzate nell'immagine: scene sfocate, pannelli uniformi o scene d'azione con motion blur pesante possono produrre un numero insufficiente di keypoint, degradando la qualità della stima.

2.3 RAFT: Recurrent All-Pairs Field Transforms

RAFT, introdotto da Teed e Deng nel 2020, rappresenta lo stato dell'arte nei metodi di optical flow basati su deep learning. A differenza degli approcci classici che operano su feature sparse, RAFT stima un flusso *denso* per ogni pixel dell'immagine, sfruttando una architettura interamente differenziabile addestrata end-to-end.

L'architettura si compone di tre blocchi principali. Il *feature encoder* processa entrambi i frame con una rete convoluzionale condivisa producendo rappresentazioni a risoluzione ridotta (1/8 dell'originale). Il *context encoder* processa il solo frame di riferimento e fornisce informazioni contestuali per il blocco di aggiornamento. Cuore del metodo è la *correlation volume*: per ogni coppia di pixel tra i due frame si calcola il prodotto scalare tra le rispettive feature, ottenendo un volume 4D che misura la

somiglianza tra tutti i possibili accoppiamenti. Questo volume viene poi interrogato a livelli multipli di risoluzione (*pooling pyramid*) per gestire efficientemente spostamenti di entità variabile.

La stima del flusso viene raffinata iterativamente da un modulo *GRU* (Gated Recurrent Unit): partendo da un flusso iniziale nullo, ogni iterazione produce una correzione Δf che aggiorna la stima corrente. Grazie al carattere iterativo, RAFT riesce a gestire spostamenti anche molto grandi senza ricorrere alle piramidi gaussiane necessarie nei metodi classici.

Nel contesto di questo progetto il flusso denso prodotto da RAFT viene utilizzato per stimare la trasformazione globale della telecamera. La traslazione (dx, dy) viene estratta calcolando la *mediana* sull'intero campo di flusso: la mediana è intrinsecamente robusta agli outlier (corrispondenti a oggetti in movimento in primo piano) senza necessità di RANSAC esplicito. La componente di rotazione $d\theta$ viene invece stimata applicando RANSAC a un campione uniforme di corrispondenze estratte dal flusso denso, come si descriverà in dettaglio nella Sezione 3.

2.4 Smoothing della Traiettoria

Una volta stimate le trasformazioni incrementali $(dx_t, dy_t, d\theta_t)$ per ogni coppia di frame, queste vengono integrate per ottenere la *traiettoria cumulativa* della telecamera:

$$\mathbf{p}_t = \sum_{i=1}^{t} (dx_i,\ dy_i,\ d\theta_i). \tag{3}$$

La traiettoria $\mathbf{p}_t$ cattura sia il moto intenzionale (panoramiche, inseguimento del soggetto) che il jitter indesiderato. L'obiettivo dello smoothing è produrre una traiettoria $\hat{\mathbf{p}}_t$ che conservi il moto lento e intenzionale attenuando le oscillazioni ad alta frequenza. La correzione da applicare al frame t è semplicemente la differenza $\mathbf{c}_t = \hat{\mathbf{p}}_t - \mathbf{p}_t$.

2.4.1 Media Mobile

La Media Mobile è il metodo più semplice e immediato. Data una finestra di larghezza $2r + 1$, la traiettoria smoothata si ottiene come:

$$\hat{p}_t = \frac{1}{2r+1} \sum_{k=-r}^{r} p_{t+k}. \tag{4}$$

L'implementazione effettiva utilizza una convoluzione con un kernel uniforme, il che corrisponde a un filtro passa-basso con risposta in frequenza $\text{sinc}(f \cdot (2r+1))$. Il parametro r controlla il trade-off tra riduzione del jitter e fedeltà al moto originale: valori grandi producono una traiettoria molto liscia ma appiattiscono anche le panoramiche lente; valori piccoli preservano meglio i movimenti intenzionali ma lasciano filtrare più jitter.

Un limite strutturale della Media Mobile è l'introduzione di un ritardo di fase di r frame: per calcolare $\hat{p}_t$ è necessario conoscere i frame futuri fino a $t+r$. In un sistema offline, come quello implementato in questo progetto, questo non costituisce un problema; in applicazioni real-time sarebbe invece necessario ricorrere a filtri causali.

2.4.2 Filtro di Kalman

Il Filtro di Kalman è un estimatore ricorsivo ottimo per sistemi lineari soggetti a rumore gaussiano. In questo progetto viene adottato un modello a *moto costante* con stato:

$$\mathbf{x}_t = (x_t,\, y_t,\, \theta_t,\, \dot{x}_t,\, \dot{y}_t,\, \dot{\theta}_t)^\top, \tag{5}$$

dove le prime tre componenti sono la posizione cumulativa e le ultime tre le velocità. Il modello di transizione è:

$$\mathbf{x}_{t|t-1} = F\,\mathbf{x}_{t-1}, \qquad F = \begin{pmatrix} I_3 & I_3 \\ 0_3 & I_3 \end{pmatrix}, \tag{6}$$

e l'osservazione è la sola posizione $\mathbf{z}_t = H\,\mathbf{x}_t$ con $H = [I_3 \mid 0_3]$.

Il filtro alterna una fase di *predizione*

$$\hat{\mathbf{x}}_{t|t-1} = F\,\hat{\mathbf{x}}_{t-1}, \qquad P_{t|t-1} = F\,P_{t-1}\,F^\top + Q, \tag{7}$$

e una fase di *aggiornamento*

$$K_t = P_{t|t-1}\,H^\top\Big(H\,P_{t|t-1}\,H^\top + R\Big)^{-1}, \qquad \hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t|t-1} + K_t\Big(\mathbf{z}_t - H\,\hat{\mathbf{x}}_{t|t-1}\Big), \tag{8}$$

dove Q è la matrice di covarianza del rumore di processo e R quella del rumore di misura. Il rapporto Q/R governa il comportamento del filtro: valori elevati di Q (o

bassi di R) portano il filtro a fidarsi quasi ciecamente delle misure, riproducendo fedelmente la traiettoria originale con poco smoothing; valori bassi di Q (o elevati di R) producono una stima vicina al modello di moto costante, ignorando in larga misura le oscillazioni osservate. Rispetto alla Media Mobile, il Filtro di Kalman è intrinsecamente causale e in grado di sfruttare il modello dinamico del sistema, producendo in genere una migliore preservazione dei movimenti intenzionali a parità di attenuazione del jitter.

2.5 Metriche di Valutazione

Valutare la qualità di un algoritmo di stabilizzazione non è banale: una semplice ispezione visiva è soggettiva e non riproducibile. Per questo motivo sono state implementate quattro metriche quantitative, ciascuna volta a catturare un aspetto diverso delle prestazioni.

Il **RMS displacement** misura l'ampiezza media degli spostamenti incrementali lungo la traiettoria smoothata:

$$\mathrm{RMS}_x = \sqrt{\frac{1}{N-1}\sum_{t=1}^{N-1}(\hat{p}_{x,t+1} - \hat{p}_{x,t})^2}. \tag{9}$$

Valori bassi indicano una traiettoria fluida; valori alti segnalano che permane ancora jitter residuo.

La **jitter reduction** quantifica la riduzione percentuale della varianza degli spostamenti incrementali tra traiettoria grezza e traiettoria smoothata:

$$\mathrm{JR}_x = \left(1 - \frac{\mathrm{Var}(\Delta\hat{p}_x)}{\mathrm{Var}(\Delta p_x)}\right) \times 100\%. \tag{10}$$

Lo **stability score** è una media pesata delle jitter reduction sui tre assi, con pesi 0.4 per x, 0.4 per y e 0.2 per θ, rispecchiando la maggiore percettibilità delle traslazioni rispetto alle rotazioni, e produce un valore in $[0, 100]$.

Infine, il **fidelity score** misura quanto la traiettoria smoothata preserva il moto intenzionale, calcolando il coefficiente R^2 nello spazio degli incrementi:

$$R^2 = 1 - \frac{\mathrm{Var}(\Delta p - \Delta\hat{p})}{\mathrm{Var}(\Delta p)}. \tag{11}$$

Lavorare sugli incrementi, anziché sui valori assoluti della traiettoria, elimina l'effetto del trend a bassa frequenza (ad esempio una panoramica lenta) dalla normalizzazione,

rendendo il confronto tra metodi più equo. Un metodo che attenua aggressivamente tutta la traiettoria tenderà ad avere $\Delta\hat{p} \approx 0$, il che implica una differenza $\Delta p - \Delta\hat{p} \approx \Delta p$ e quindi $R^2 \approx 0$; un metodo che segue fedelmente il moto intenzionale mantenendo solo il jitter avrà invece R^2 elevato. Le due metriche — stability score e fidelity score — sono in tensione tra loro e catturano congiuntamente il trade-off fondamentale di qualsiasi algoritmo di smoothing.

3 Implementazione

Il sistema è stato interamente sviluppato in Python 3, sfruttando OpenCV per le operazioni di visione artificiale classica, PyTorch e `torchvision` per il componente deep learning, e Streamlit per l'interfaccia utente interattiva. La scelta di queste librerie rispecchia lo stato dell'arte per ciascun dominio e garantisce una buona portabilità tra sistemi operativi. Il codice è organizzato in moduli distinti con responsabilità ben separate, il che ha facilitato tanto lo sviluppo incrementale quanto la fase di testing.

3.1 Architettura del Sistema

La pipeline di stabilizzazione è suddivisa in quattro moduli principali. `motion_estimation.py` si occupa di ricavare la trasformazione inter-frame da coppie di frame in scala di grigi. `trajectory_smoothing.py` integra tali stime in una traiettoria cumulativa e la liscia con uno dei due algoritmi disponibili. `video_stabilization.py` orchestra l'intera sequenza — dalla lettura del file video all'applicazione delle correzioni geometriche — ed espone due funzioni di alto livello, `stabilize_video_moving_average` e `stabilize_video_kalman`. Infine, `metrics.py` calcola le metriche quantitative descritte nella Sezione 2.5 e salva i risultati in formato JSON per il confronto tra esperimenti successivi.

`app.py`, il punto di ingresso dell'applicazione, non contiene logica elaborativa: il suo ruolo è esclusivamente quello di raccogliere i parametri dall'utente, invocare i moduli sopra descritti e presentarne i risultati attraverso la dashboard Streamlit.

3.2 Stima del Moto con SIFT

L'implementazione SIFT utilizza direttamente `cv2.SIFT_create`, disponibile in OpenCV a partire dalla versione 4.4 a seguito della scadenza del brevetto originale. I quattro parametri principali esposti all'utente — numero massimo di keypoint (`n_features`), soglia di contrasto (`contrast_threshold`), soglia sugli edge (`edge_threshold`) e deviazione standard della piramide gaussiana (`sigma`) — consentono di adattare il rilevatore alla densità di texture della scena.

Una volta estratti i descrittori da entrambi i frame, il matching avviene con un *Brute-Force Matcher* sulla norma L_2, seguito dal ratio test di Lowe con soglia configurabile (default 0.75). Le corrispondenze accettate vengono passate a `cv2.estimateAffinePartial2D` con il flag `RANSAC` e una soglia di reproiezione di 5 pixel. Quando il numero di corrispondenze disponibili scende sotto 4 — il minimo necessario per stimare una trasformazione affine parziale — la funzione restituisce

$(dx, dy, d\theta) = (0, 0, 0)$, assumendo che non vi sia movimento stimabile e preservando il frame corrente senza correzione.

Una scelta implementativa rilevante riguarda la gestione dei keypoint tra frame consecutivi: anziché rieseguire il rilevamento da zero a ogni passo, i keypoint e i descrittori del frame corrente vengono mantenuti in memoria e riutilizzati come riferimento per il frame successivo, riducendo il numero di chiamate a `detectAndCompute` della metà.

3.3 Stima del Moto con RAFT

Il modello RAFT viene caricato tramite `torchvision.models.optical_flow` con i pesi pre-addestrati su Sintel e FlyingChairs (`Raft_Large_Weights.DEFAULT` oppure `Raft_Small_Weights.DEFAULT` per la variante compatta). Per evitare il costo di caricamento a ogni frame, il modello è implementato come *singleton*: alla prima chiamata viene istanziato e mantenuto in memoria per tutta la sessione, venendo riutilizzato per tutti i frame successivi.

La pre-elaborazione dei frame segue esattamente le specifiche ufficiali di `torchvision`: i frame in scala di grigi vengono prima convertiti in RGB (replicando il canale tre volte), poi trasformati in tensori `uint8` con forma (C, H, W). La normalizzazione a virgola mobile nell'intervallo $[-1, 1]$ viene delegata interamente ai `transforms` ufficiali del modello, invocati sulla coppia di frame prima dell'inferenza, nel rispetto del preprocessing atteso dalla rete.

L'inferenza restituisce un elenco di flussi predetti alle diverse iterazioni del modulo GRU; viene utilizzata solo l'ultima predizione, che corrisponde alla stima più raffinata. Il flusso denso risultante, di forma $(H, W, 2)$, viene poi elaborato per estrarre i tre parametri di moto globale. La traslazione (dx, dy) viene stimata come mediana dei valori del flusso sull'intera immagine: rispetto alla media, la mediana è robusta agli outlier provocati da oggetti in primo piano che si muovono in modo indipendente dalla telecamera, senza necessità di applicare RANSAC su ogni singolo frame. Per la rotazione $d\theta$, si campiona una griglia uniforme di punti dal campo di flusso (`num_samples` configurabile, default 200), si costruiscono le corrispondenze $(x, y) \rightarrow (x + u, y + v)$ e si applica nuovamente `estimateAffinePartial2D` con RANSAC per estrarre l'angolo di rotazione dalla matrice affine 2×3.

3.4 Smoothing della Traiettoria

Entrambi i metodi di smoothing operano sulla traiettoria cumulativa, calcolata sommando iterativamente le trasformazioni incrementali stimate:

```
trajectory = np.cumsum(transforms_array, axis=0)
```

Listing 1: Calcolo della traiettoria cumulativa.

La Media Mobile è implementata tramite `np.convolve` con un kernel uniforme normalizzato di larghezza $2r + 1$. I bordi della sequenza vengono gestiti con un padding di tipo `edge` (replica del primo e dell'ultimo valore), una scelta conservativa che evita l'introduzione di artefatti alle estremità del video.

Il Filtro di Kalman è implementato tramite `cv2.KalmanFilter(6, 3)`, con 6 stati e 3 misure. Le matrici di transizione F e di misura H corrispondono esattamente al modello a velocità costante descritto nella Sezione 2.4.2. La matrice di covarianza del rumore di processo Q è diagonale con valori q_{pos} sulle componenti di posizione e $2\,q_{\text{pos}}$ su quelle di velocità: il coefficiente moltiplicativo 2 bilancia la maggiore incertezza sull'accelerazione rispetto alla posizione, producendo un comportamento leggermente più reattivo sui cambiamenti di direzione senza però rinunciare allo smoothing. La matrice R è scalare identità moltiplicata per `measurement_noise`, il parametro principale esposto all'utente per controllare quanto aggressivamente il filtro attenua il jitter.

Una volta ottenuta la traiettoria smoothata $\hat{\mathbf{p}}_t$, la correzione da applicare al frame t è $\mathbf{c}_t = \hat{\mathbf{p}}_t - \mathbf{p}_t$. La funzione `apply_transform_with_zoom` costruisce la matrice affine 2×3 usando `cv2.getRotationMatrix2D`, aggiunge la componente di traslazione e chiama `cv2.warpAffine` con interpolazione bilineare. Il bordo del frame, eventualmente esposto dal warping, viene riflesso (`BORDER_REFLECT`) anziché campionato a zero, a vantaggio della continuità visiva; il fattore di zoom applicato amplia leggermente il frame prima del warping, garantendo che il crop risultante non contenga mai regioni nere.

3.5 Zoom Adattivo

La correzione geometrica sposta i pixel del frame di una quantità proporzionale all'entità del jitter residuo: se la correzione è grande, porzioni del bordo dell'immagine risultano non coperte. La soluzione adottata è uno zoom uniforme — un leggero ingrandimento che "nasconde" i bordi esposti senza distorcere il contenuto — il cui fattore è configurabile dall'utente nell'intervallo $[1.0, 1.3]$. Valori intorno a 1.05–1.10 sono in genere sufficienti per la maggior parte delle sequenze tipiche; valori più elevati sono necessari solo in presenza di shake molto intenso o quando si imposta un smoothing molto aggressivo.

3.6 Dashboard Streamlit

L'interfaccia utente è strutturata come un'applicazione Streamlit a pagina singola. La barra laterale raccoglie tutti i controlli: la selezione del metodo di motion estimation (SIFT o RAFT), i parametri specifici dell'algoritmo scelto, la selezione del metodo di smoothing, il raggio della Media Mobile o i parametri Q e R del Kalman, il fattore di zoom e la risoluzione di elaborazione. L'area principale visualizza, dopo l'elaborazione, tre schede: la prima mostra i grafici delle traiettorie grezze e smoothate sui tre assi; la seconda presenta il confronto video affiancato tra l'originale instabile e la versione stabilizzata; la terza espone le tabelle di metriche per i due metodi di smoothing.

Un aspetto di particolare cura implementativa riguarda la gestione della sessione: i parametri correnti vengono confrontati con quelli dell'elaborazione precedente in `st.session_state`, e il processing viene rilanciato automaticamente solo quando si rileva una variazione effettiva, evitando ricalcoli inutili durante la normale navigazione dell'interfaccia.

4 Esperimenti

4.1 Setup Sperimentale

Tutti gli esperimenti descritti in questa sezione sono stati condotti sulla medesima sequenza video, acquisita con una fotocamera tenuta a mano. La risoluzione di elaborazione è stata fissata a 1280 pixel di larghezza per entrambi gli esperimenti, mantenendo il rapporto d'aspetto originale.

Le due configurazioni confrontate corrispondono ai due metodi di stima del moto implementati: l'**Esperimento 1** utilizza SIFT con i parametri di default (500 keypoint massimi, soglia di contrasto 0.04, soglia sugli edge 10, $\sigma = 1.6$, ratio test 0.75); l'**Esperimento 2** adotta RAFT Large con 200 punti di campionamento sul flusso denso per la stima della rotazione. Per rendere il confronto equo, i parametri di smoothing sono identici nei due esperimenti: raggio della Media Mobile $r = 30$, e per il Filtro di Kalman $Q = 0.0005$ e $R = 2.0$. Lo zoom è abilitato al 10% in entrambi i casi.

I due metodi operano su hardware diverso: SIFT gira interamente su CPU, mentre RAFT sfrutta la GPU tramite CUDA. La fase di smoothing — avendo complessità $O(N)$ — contribuisce in modo trascurabile al tempo totale in entrambi i casi (circa 0.28 s), concentrando l'interesse sulle prestazioni della sola stima del moto.

4.2 Esperimento 1: SIFT Baseline

Nel primo esperimento la stima del moto ha richiesto complessivamente 40.7 s per l'intera sequenza, con un tempo totale di elaborazione di 41.2 s. La Tabella 1 riporta le metriche principali per i due metodi di smoothing applicati alla traiettoria stimata da SIFT.

Tabella 1: Metriche — Esperimento 1 (SIFT, r=30, Q=0.0005, R=2.0).

Metrica	**Media Mobile**	**Kalman**
Stability Score [0–100]	82.3	45.6
Fidelity Score [0–100]	41.7	67.0
RMS raw x [px]	7.71	
RMS smoothed x [px]	6.71	7.34
RMS raw y [px]	4.58	
RMS smoothed y [px]	3.90	4.24
RMS raw θ [°]	0.133	
RMS smoothed θ [°]	0.028	0.100
Jitter Reduction x [%]	77.3	43.4
Jitter Reduction y [%]	79.1	48.3
Jitter Reduction θ [%]	98.6	44.6
Max offset x [px]	53.5	38.6
Max offset y [px]	25.0	16.2
Tempo motion estimation [s]	40.7	

La Media Mobile raggiunge uno stability score di 82.3, grazie a una forte riduzione del jitter su tutti e tre gli assi — particolarmente marcata sulla rotazione (98.6%). Tuttavia il fidelity score si ferma a 41.7: la convoluzione con la finestra uniforme attenua in modo indiscriminato ogni oscillazione, incluse quelle a frequenza più bassa che possono corrispondere a movimenti intenzionali della telecamera. Lo dimostrano anche i valori del RMS smoothed: l'RMS sull'asse x scende da 7.71 a 6.71 px, ma la riduzione è quasi interamente attribuibile all'attenuazione del jitter piuttosto che a una precisa inseguimento della traiettoria raw.

Il Filtro di Kalman presenta un profilo complementare. Lo stability score scende a 45.6 — la jitter reduction è più modesta, attorno al 44–48% per x e y — ma il fidelity score sale a 67.0, confermando che il filtro riesce a distinguere meglio tra oscillazioni di breve durata e movimenti intenzionali a più bassa frequenza. La componente di velocità nel vettore di stato consente al Kalman di *predire* l'evoluzione della traiettoria invece di limitarsi a mediarla, producendo una stima più fedele al moto originale. L'offset massimo applicato è sensibilmente minore rispetto alla Media

Mobile (38.6 vs 53.5 px su x), a indicare che la traiettoria smoothata non si discosta eccessivamente da quella grezza.

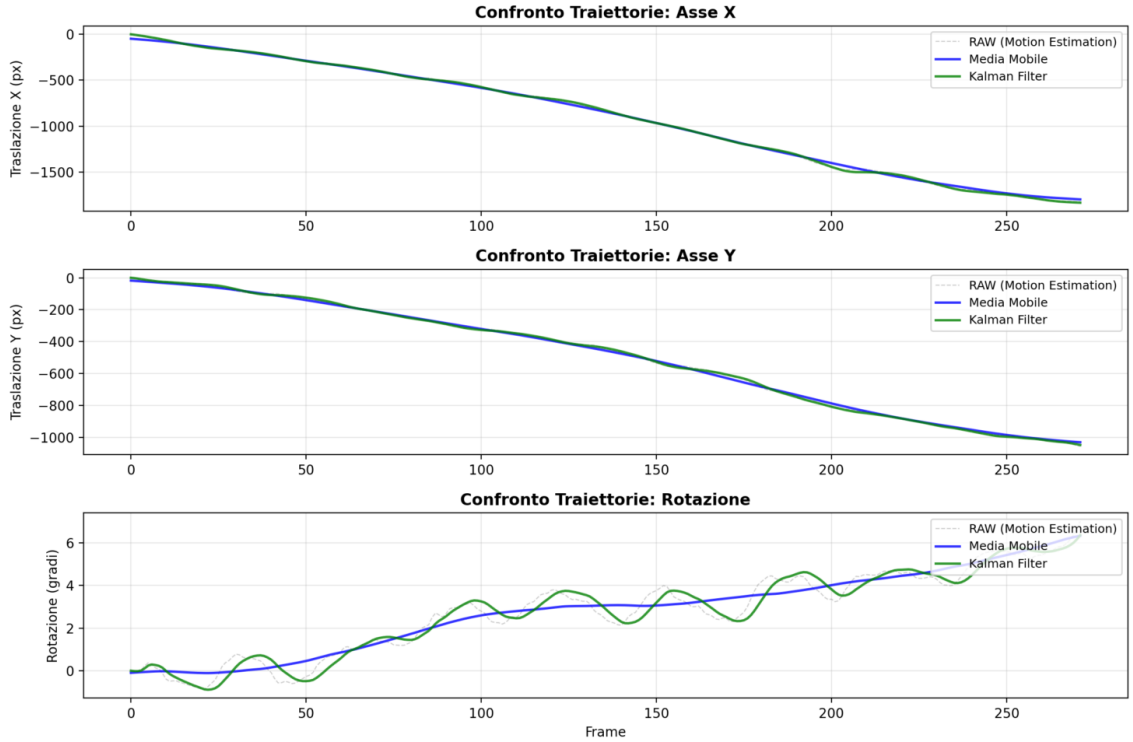

Figura 1: Esperimento 1 (SIFT): confronto tra traiettoria grezza, smoothata con Media Mobile e smoothata con Filtro di Kalman sugli assi x, y e θ.

4.3 Esperimento 2: RAFT Baseline

Il secondo esperimento sostituisce SIFT con RAFT Large mantenendo invariati tutti gli altri parametri. La stima del moto ha richiesto 43.1 s, circa 2.4 s in più rispetto a SIFT: la differenza è attribuibile al tempo di inferenza della rete e alla gestione del tensore su CPU. La Tabella 2 riporta le metriche per questo esperimento.

Tabella 2: Metriche — Esperimento 2 (RAFT Large, r=30, Q=0.0005, R=2.0).

Metrica	**Media Mobile**	**Kalman**
Stability Score [0–100]	95.1	59.2
Fidelity Score [0–100]	26.4	59.0
RMS raw x [px]	2.69	
RMS smoothed x [px]	1.01	1.81
RMS raw y [px]	1.40	
RMS smoothed y [px]	0.311	0.830
RMS raw θ [°]	0.138	
RMS smoothed θ [°]	0.028	0.102
Jitter Reduction x [%]	90.9	57.9
Jitter Reduction y [%]	98.2	67.4
Jitter Reduction θ [%]	97.4	45.6
Max offset x [px]	20.3	14.6
Max offset y [px]	13.0	8.15
Tempo motion estimation [s]	43.1	

Il primo elemento che emerge con evidenza è l'RMS raw di RAFT: 2.69 px sull'asse x contro i 7.71 px di SIFT. Un valore così basso non indica necessariamente che il video originale sia più stabile, ma piuttosto che RAFT produce stime del moto globale intrinsecamente meno rumorose. Il flusso denso, calcolato sull'intera immagine con una rete addestrata su milioni di esempi, è molto meno suscettibile ai falsi match che affliggono il pipeline SIFT in presenza di ripetizioni di pattern, zone a bassa texture o motion blur.

Con la Media Mobile, lo stability score raggiunge 95.1 — il valore più alto tra i quattro scenari testati — e la jitter reduction sull'asse y tocca il 98.2%. Il prezzo da pagare è un fidelity score di appena 26.4: poiché le stime RAFT sono già quasi prive di jitter, applicare una finestra media di 30 frame ha l'effetto di eliminare anche i movimenti intenzionali a frequenza medio-bassa, producendo una traiettoria smoothata che si discosta significativamente da quella raw. In altri termini, la Media Mobile è sprecata su un segnale già pulito.

Il Kalman nel contesto RAFT mostra invece un profilo molto più bilanciato. Lo stability score (59.2) supera il corrispondente SIFT (45.6), mentre il fidelity score (59.0) è notevolmente più alto rispetto alla Media Mobile RAFT (26.4) e si avvicina al valore ottenuto dal Kalman su SIFT (67.0). Il Kalman riesce a sfruttare la qualità delle stime RAFT senza distruggerle: il suo modello a velocità costante evita di interpretare come jitter quelle componenti di bassa frequenza che invece corrispondono a movimenti reali della telecamera.

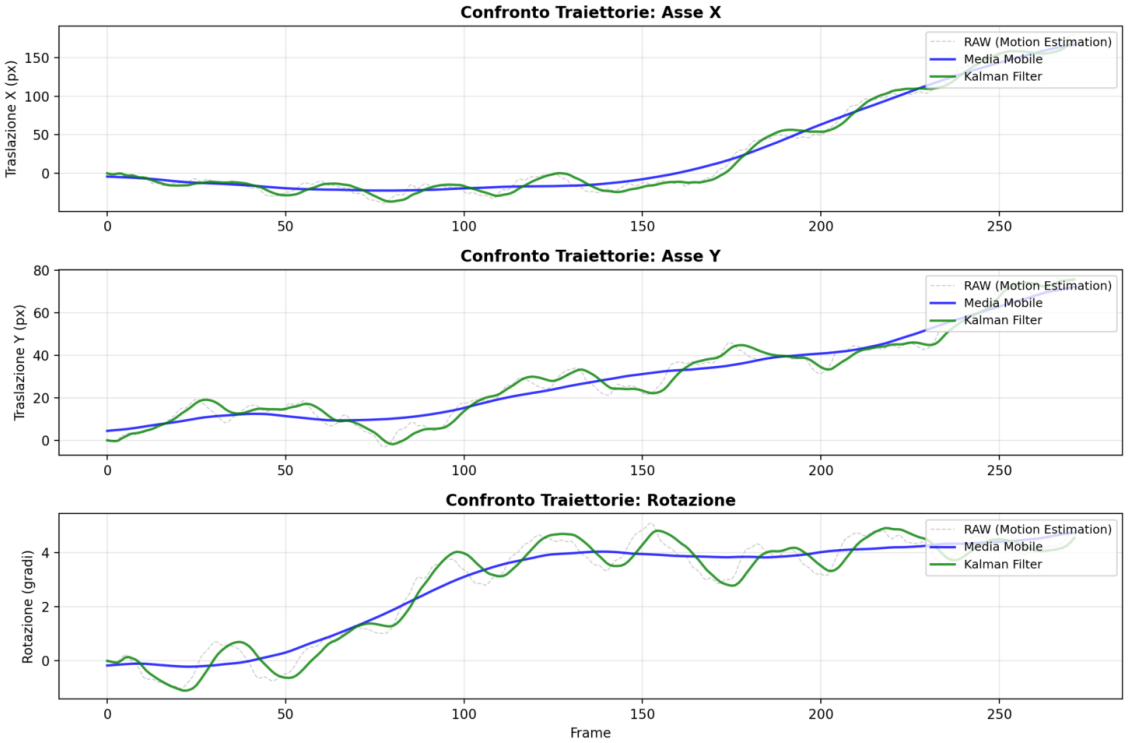

Figura 2: Esperimento 2 (RAFT): confronto tra traiettoria grezza, smoothata con Media Mobile e smoothata con Filtro di Kalman sugli assi x, y e θ.

4.4 Confronto Diretto tra i Quattro Scenari

La Tabella 3 sintetizza i risultati dei quattro scenari (metodo di stima × metodo di smoothing) nelle due metriche di valutazione più rappresentative.

Tabella 3: Confronto complessivo: Stability Score e Fidelity Score.

Motion Est.	**Smoothing**	**Stability**	**Fidelity**
SIFT	Media Mobile	82.3	41.7
SIFT	Kalman	45.6	67.0
RAFT	Media Mobile	**95.1**	26.4
RAFT	Kalman	59.2	**59.0**

La tabella mette in luce due tendenze sistematiche. Prima: a parità di metodo di smoothing, RAFT produce sempre uno stability score più alto di SIFT (+12.8 punti con Media Mobile, +13.6 con Kalman), segno che la qualità della stima del moto ha un impatto diretto sulla qualità del risultato finale. Seconda: a parità di metodo di stima, la Media Mobile supera sempre il Kalman in stabilità ma ne è sempre inferiore in fedeltà; lo scarto in stabilità è di 36.7 punti per SIFT e 35.9 per RAFT, mentre lo scarto in fedeltà è di 25.3 e 32.6 punti rispettivamente, evidenziando che il compromesso stabilità–fedeltà è una caratteristica strutturale dei due smoothing e non dipende dal metodo di stima a monte.

Vale la pena notare il comportamento asimmetrico del fidelity score: con SIFT la Media Mobile ottiene 41.7 contro 67.0 del Kalman, uno scarto di 25.3 punti. Con RAFT, lo stesso confronto produce uno scarto di 32.6 punti (26.4 vs 59.0). Questo suggerisce che quanto più le stime raw sono accurate (cioè quanto meno rumorose), tanto più la Media Mobile è *deleteria* per la fedeltà al moto intenzionale: avendo poco jitter da eliminare, la sua azione di smoothing si abbatte quasi interamente sul contenuto informativo del segnale.

4.5 Costo Computazionale

Come atteso, la fase di smoothing è computazionalmente trascurabile in entrambi gli esperimenti: 0.28 s per la Media Mobile e 0.28–0.30 s per il Kalman, indipendentemente dal metodo di stima a monte. Il carico computazionale è dominato quasi interamente dalla stima del moto: 40.7 s per SIFT (su CPU) e 43.1 s per RAFT (su GPU con CUDA).

Il fatto che RAFT su GPU richieda un tempo paragonabile a SIFT su CPU è rilevante: nonostante la rete abbia milioni di parametri, il singleton del modello viene caricato una sola volta e l'inferenza viene eseguita in modalità `no_grad`, minimizzando l'overhead di PyTorch. Il collo di bottiglia per RAFT è il trasferimento dei tensori tra

RAM e VRAM e la latenza dei kernel CUDA per frame di risoluzione relativamente piccola (1280 px): su sequenze ad alta risoluzione o con GPU più recenti il vantaggio rispetto a SIFT su CPU sarebbe più marcato. SIFT, dal canto suo, benefilcerebbe relativamente poco da una GPU, essendo un algoritmo basato su operazioni sparse difficilmente parallelizzabili.

5 Discussione e Conclusioni

5.1 Interpretazione dei Risultati

I risultati degli esperimenti convergono verso un'indicazione chiara: la qualità del risultato finale dipende in modo cruciale dalla catena completa del pipeline, e la scelta del metodo di stima del moto ha un peso almeno pari a quello del metodo di smoothing. Questo è un punto non scontato: si potrebbe essere tentati di pensare che uno smoother sufficientemente aggressivo possa compensare le imprecisioni della stima del moto a monte, ma i dati mostrano che così non è. Il fatto che la Media Mobile applicata a RAFT raggiunga uno stability score di 95.1 mentre la stessa Media Mobile applicata a SIFT si fermi a 82.3 — pur con parametri di smoothing identici — dimostra che la qualità della traiettoria raw condiziona strettamente la qualità di quella smoothata.

Il trade-off tra stability score e fidelity score merita un'attenzione particolare. Come si è visto, i due indicatori tendono a muoversi in direzioni opposte: ogni aumento dello stability score comporta un costo in termini di fedeltà al moto originale, e viceversa. Questo non è un artefatto del sistema implementato, ma riflette un limite fondamentale di qualsiasi approccio di smoothing lineare: una finestra di integrazione larga elimina più rumore ma degrada più segnale utile. Il Filtro di Kalman si posiziona lungo questa curva di compromesso in modo più vantaggioso rispetto alla Media Mobile, poiché il suo modello cinematico lo rende selettivo rispetto alla frequenza: attenua le oscillazioni rapide mantenendo quelle lente, invece di trattare uniformemente tutte le componenti spettrali come fa la convoluzione con un kernel rettangolare.

Un risultato particolarmente istruttivo emerge dal confronto RAFT + Media Mobile: questo scenario ottiene il massimo stability score (95.1) ma anche il minimo fidelity score (26.4). La ragione va ricercata proprio nella qualità delle stime RAFT. Poiché il flusso ottico denso è già quasi privo di jitter ad alta frequenza, la Media Mobile non trova molto da eliminare su quel fronte; compensa invece abbattendo indiscriminatamente tutta la variabilità del segnale, inclusi i movimenti intenzionali. In pratica, uno smoother così conservativo "pialla" una traiettoria già quasi uniforme, producendo un video con telecamera quasi ferma che potrebbe risultare artificiale e innaturale allo spettatore. Questo suggerisce che la scelta dei parametri di smoothing debba essere adattata alla qualità dell'estimatore a monte: con RAFT è sufficiente un raggio di Media Mobile molto più piccolo, o un Kalman con R ridotto, per ottenere una stabilizzazione efficace senza perdere il moto intenzionale.

5.2 Limiti del Sistema

Il sistema presenta alcuni limiti che vale la pena riconoscere esplicitamente. Un primo limite riguarda la latenza complessiva: più di 40 secondi per processare una sequenza sono accettabili in un contesto offline, ma proibitivi per applicazioni real-time. RAFT beneficia già dell'accelerazione GPU tramite CUDA, il che lo rende competitivo con SIFT su CPU nonostante la maggiore complessità del modello; tuttavia, per avvicinarsi a prestazioni real-time sarebbe necessario ridurre la risoluzione di elaborazione o adottare varianti più leggere come RAFT Small, a scapito dell'accuratezza. SIFT, invece, gira interamente su CPU e non può beneficiare di accelerazione GPU in modo diretto, rappresentando il vero collo di bottiglia in scenari ad alta risoluzione.

Sul piano della robustezza, SIFT mostra una vulnerabilità nota in scene con poca texture o in presenza di motion blur intenso: in questi contesti il numero di keypoint rilevabili può scendere sotto la soglia minima per RANSAC, e la funzione ritorna $(0, 0, 0)$ per l'intero frame, introducendo un'imprecisione nella traiettoria cumulativa che si propaga ai frame successivi. RAFT è strutturalmente immune a questo problema poiché opera su tutti i pixel, ma è soggetto a propri artefatti nelle scene con superfici riflettenti o in condizioni di sottoesposizione severa.

Un altro limite riguarda il modello di moto adottato: il sistema stima e corregge esclusivamente la trasformazione di similarità nel piano dell'immagine (traslazione + rotazione + scala uniforme), ignorando le componenti di moto tridimensionale — variazioni di prospettiva, rollio fuori piano, variazioni focali. Questo modello è adeguato per le tipiche oscillazioni da camera a mano, ma potrebbe rivelarsi insufficiente per sequenze acquisite con droni o con camere montate su veicoli in rapido movimento.

Infine, il fidelity score basato su R^2 fornisce una misurazione utile ma non cattura la qualità percepita dall'osservatore umano. Due traiettorie con lo stesso fidelity score possono produrre risultati visivamente molto diversi a seconda di dove si concentrano i residui di errore: un errore concentrato in pochi frame è più disturbante dello stesso errore distribuito uniformemente lungo l'intera sequenza.

5.3 Linee di Sviluppo Future

Diverse direzioni di sviluppo sembrano promettenti. Sul fronte delle prestazioni, un guadagno immediato sarebbe ottenibile elaborando i frame in batch anziché uno alla volta, riducendo l'overhead dei trasferimenti tra RAM e VRAM per RAFT e sfruttando meglio il parallelismo della GPU. In parallelo, l'adozione di un modello di moto omografico — in grado di stimare la trasformazione prospettica completa tra

due frame — permetterebbe di gestire correttamente le rotazioni tridimensionali e le variazioni di campo visivo.

Sul fronte dello smoothing, un'esplorazione interessante riguarderebbe l'adozione di filtri adattativi che modulino automaticamente i parametri Q e R in funzione del contenuto dinamico della scena: in una sequenza con inquadratura statica è ragionevole imporre un smoothing più aggressivo rispetto a una panoramica, e un sistema di rilevamento automatico del tipo di moto potrebbe guidare questa adattività.

Un'altra direzione è l'integrazione di metriche di qualità percettiva — come SSIM o VMAF — per affiancare le metriche geometriche attualmente implementate, avvicinando la valutazione al giudizio soggettivo dell'utente finale. Questo consentirebbe anche di costruire funzioni di ottimizzazione più allineate con la reale esperienza visiva, aprendo la strada a tecniche di tuning automatico dei parametri.

5.4 Conclusioni

Questo progetto ha realizzato e valutato una pipeline completa di stabilizzazione video che integra due approcci alla stima del moto — il classico SIFT e il modello deep learning RAFT — con due strategie di smoothing della traiettoria, la Media Mobile e il Filtro di Kalman, il tutto accessibile tramite una dashboard interattiva che permette di esplorarne i parametri in tempo reale.

I risultati sperimentali mostrano che RAFT produce stime del moto sistematicamente più accurate di SIFT, con un RMS raw inferiore di quasi tre volte, e che questa qualità si traduce direttamente in stability score più elevati a valle dello smoothing. Il Filtro di Kalman offre il miglior bilanciamento tra riduzione del jitter e preservazione del moto intenzionale, posizionandosi in modo più vantaggioso sul trade-off stabilità–fedeltà rispetto alla Media Mobile indipendentemente dal metodo di stima. La combinazione RAFT + Kalman è quella che risulta più equilibrata nei due indicatori simultaneamente, con uno stability score di 59.2 e un fidelity score di 59.0, mentre RAFT + Media Mobile è preferibile quando l'obiettivo è la massima stabilità e si accetta una perdita di fedeltà al moto originale.

Dal punto di vista didattico, il progetto ha offerto l'opportunità di approfondire concretamente tanto i fondamenti classici della visione artificiale — scala di interesse, invarianza ai descrittori, stima robusta con RANSAC — quanto le architetture deep learning per l'optical flow, verificandone i vantaggi e i limiti in un'applicazione reale. La dashboard realizzata costituisce uno strumento di esplorazione immediata e

intuitiva di questi fenomeni, rendendo visibili e quantificabili differenze che altrimenti rimarrebbero difficili da apprezzare senza strumenti di analisi dedicati.