

Implementazione in C++ con OpenCV dell'algoritmo Exposure Fusion di Tom Mertens

Stefano Russo (1864451)

1 Introduzione

Exposure Fusion è una tecnica per fondere immagini con diversi tempi di esposizione senza passare per un'immagine HDR, fondendo i punti migliori delle varie immagini. L'implementazione è basata sull'implementazione originale di Tom Mertens in Matlab:
<https://github.com/Mericam/exposure-fusion/>
Si presuppone che le immagini siano già allineate.

2 Implementazione

2.1 Caricamento delle immagini

Le immagini vengono caricate in 64F_C3, cioè double con 3 canali, e vengono scalate di 1.0/255 per far rientrare i valori in [0, 1].

2.2 Generazione delle weight maps per ogni immagine

Prendiamo come esempio quest'immagine:



Vengono generate le weight map in base a tre parametri calcolati per ogni pixel: contrasto, saturazione e qualità dell'esposizione. È possibile impostare il peso dato a ciascun parametro modificando l'array weights.

Contrasto Si prende il valore assoluto dei valori generati applicando l'operatore di Laplace sull'immagine in grayscale. È necessario convertire da 64F_C3 (double con 3 canali) a 8U_C3 (uchar con 3 canali) scalando di 255 prima di convertire in grayscale in quanto la funzione cvtColor non supporta immagini in 64F. Successivamente si può riconvertire in 64F_C1 e scalare di nuovo di 1.0/255, per poi procedere con l'applicazione dell'operatore di Laplace. Nell'implementazione è stato aggiunto un Gaussian Blur prima di applicare Laplace per ridurre il rumore come suggerito qui:

<https://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm>

Saturazione Si prende la deviazione standard dei canali RGB per ogni pixel.

Qualità dell'esposizione Si usa una curva Gaussiana applicata al colore di ogni pixel separatamente: $\exp\left(-\frac{(i-0.5)^2}{2\sigma^2}\right)$ dove i è il colore del pixel. Si moltiplicano i valori per ottenere il risultato.

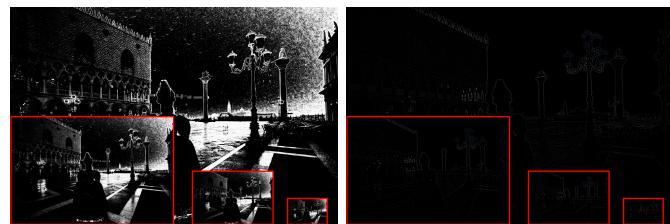


Si moltiplicano le matrici contenenti i parametri elemento per elemento e si normalizzano le matrici.



2.3 Generazione delle piramidi di Laplace e di Gauss

Vengono generate le piramidi di Gauss per ogni weight map e le piramidi di Laplace per ogni immagine



Si genera la piramide di Laplace del risultato comando livello per livello le piramidi di Laplace delle immagini moltiplicate per la piramide di Gauss delle weight map, ottenendo così una media pesata.

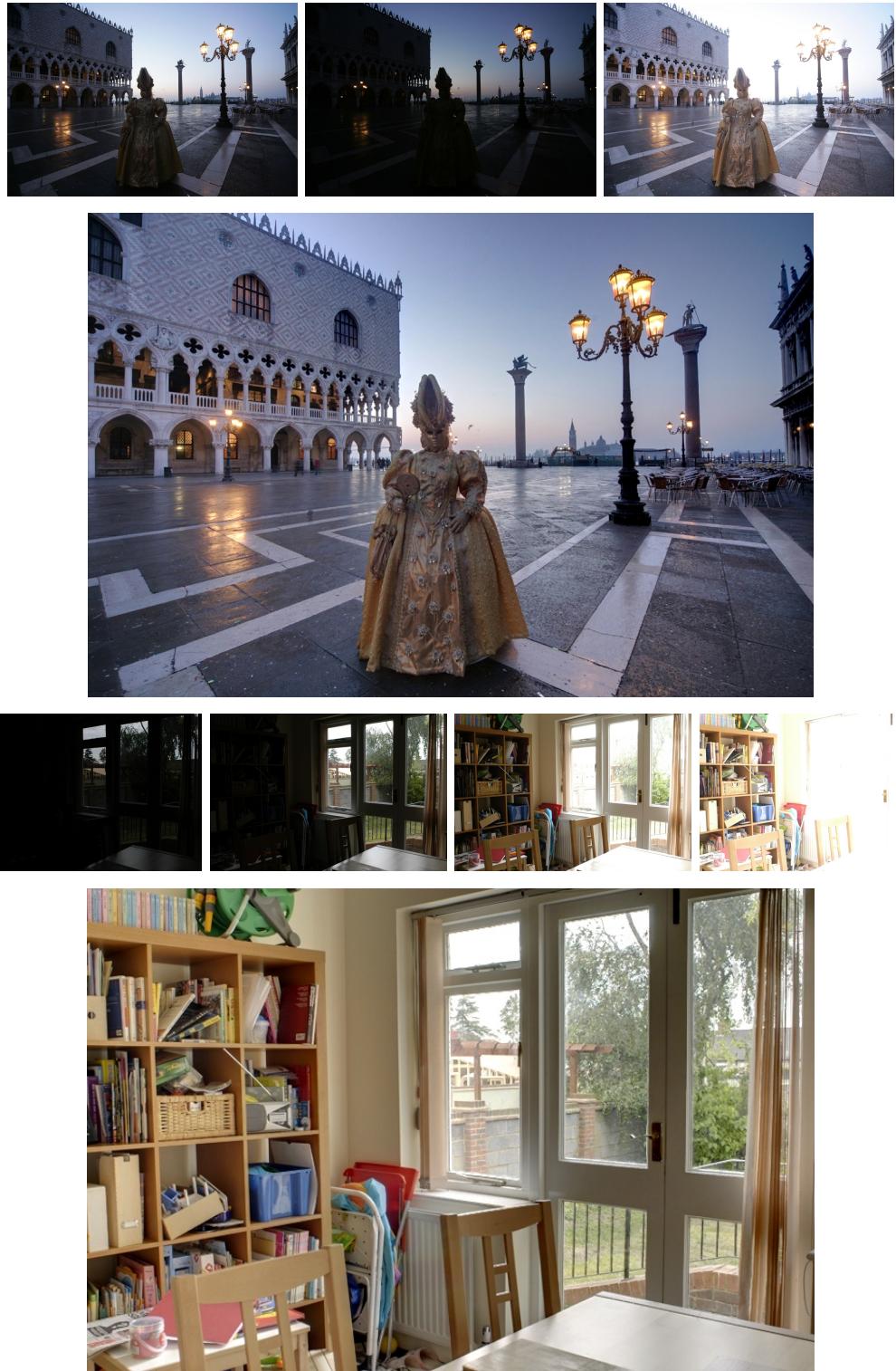


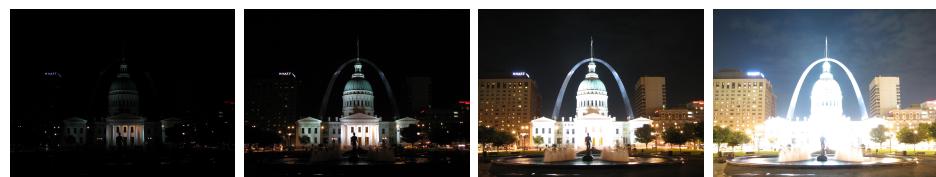
2.4 Collapse della piramide di Laplace del risultato

Si applica la funzione collapse alla piramide di Laplace del risultato, ottenendo l'immagine finale.



3 Risultati





Tutte le immagini sono state create con i parametri del peso impostati ad 1 e $\sigma = 0.2$. Si può notare come l'immagine finale prenda i punti salienti di ogni immagine. Per esempio, nel secondo e nel terzo esempio i dettagli fuori dalla finestra vengono presi dalle immagini con tempo di esposizione minore, mentre quelli dell'interno vengono presi dalle immagini con più tempo di esposizione.

4 Riferimenti

Statua (Risultato 1): Jacques Joffre

House (Risultato 2): Min H. Kim

Window (Risultato 3): Sebastian Nibisz

St. Louis (Risultato 4): Kevin McCoy