POLITECNICO DI TORINO

Corso di Laurea
in Matematica per l'Ingegneria

Tesi di Laurea

Filtraggio di immagini

Relatori
prof. Nome Cognome
prof. Nome Cognome
firma dei relatori

. . . . . .
. . . . . .

Candidato
Raffaello Ippolito

firma del candidato

. . . . . .

Anno Accademico 2021-2022

♡

Sommario

Lo studio delle Equazioni alle derivate parziali ed il loro impiego in ambito di filtraggio di immagini.

Ringraziamenti

I candidati ringraziano vivamente il Granduca di Toscana per i mezzi messi loro a disposizione, ed il signor Von Braun, assistente del prof. Albert Einstein, per le informazioni riservate che egli ha gentilmente fornito loro, e per le utili discussioni che hanno permesso ai candidati di evitare di riscoprire l'acqua calda.

Indice

Elenco delle tabelle

Elenco delle figure

If you cannot understand my
argument, and declare
it's Greek to me
you are quoting Shakespeare.

[B. LEVIN, Quoting Shakespeare]

Parte I

Introduzione

Capitolo 1

Storia del filtraggio e principali problemi affrontati

Le immagini hanno un ruolo fondamentale nelle nostre vite, viviamo di immagini, ne guardiamo tutti i giorni in tutti i contesti, la percezione visiva è sempre il nostro primo riferimento senza la quale ci si sentiamo persi.
Viviamo in una società consapevole di ciò e che sfrutta questo aspetto quanto più possibile nei campi più disparati, fotografie, radiografie, ecografie, poster pubblicitari, progetti, etc.
E' quindi un problema sempre di grande interesse cercare di sfruttarle al meglio, a tal fine esistono metodi cosìdetti di "filtraggio", tramite i quali intendiamo migliorare la qualità delle nostre immagini, mettere in risalto determinate caratteristiche o nasconderne altre.

1.1 Storia del filtraggio

Anche prima dell'avvento dei computer esisteva l'elaborazione delle immagini. Esisteva infatti dei procedimenti, detti "mascherature", che servivano a ridurre o esaltare le differenze di luce nella foto.
La mascheratura avveniva durante la fase di stampa su carta, ossia quando il negativo pellicola veniva proiettato sulla carta fotografica mediante l'ingranditore.

Qui, lo "stampatore" usava una serie di metodi per far si che su certe zone della carta andasse più o meno luce di quella che sarebbe arrivata passando attraverso il negativo.

Ad esempio la tecnica nota come "altissimo contrasto" permetteva di avere foto con soli bianchi e soli neri, il viraggio invece serviva a dare un tono di colore alla foto, che restava comunque un bianco e nero: famoso il viraggio tono seppia.

Le prime applicazioni delle immagini digitali sono state sui giornali negli anni 20. Non esistevano veri e propri computer, il segnale era trasmesso tramite telegrafo simulando dei mezzitoni. In particolare, il sistema di trasmissione di immagini via cavo Bartlane era una tecnica inventata nel 1920 per trasmettere immagini di giornali digitalizzate su linee sottomarine tra Londra e New York.

Il sistema di trasmissione di immagini via cavo Bartlane generava sia sul trasmettitore che sul ricevitore una scheda dati perforata o un nastro che veniva ricreato come immagine.

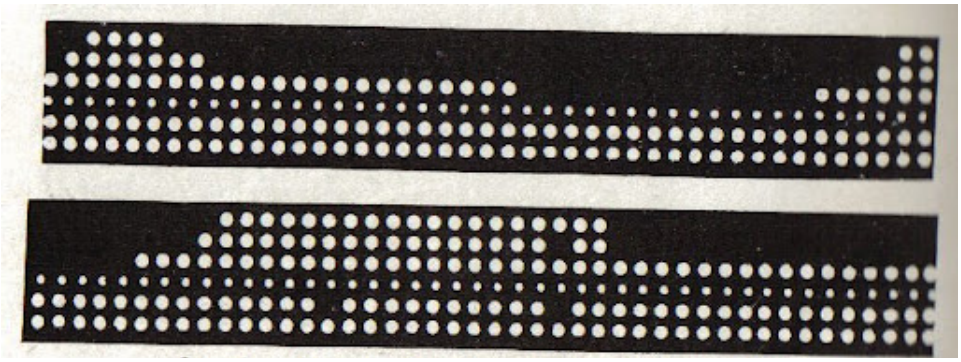

Figura 1.1. Nastro dati codificante un'immagine.

In questo modo riusciva a codificare immagini in bianco e nero in cinque diverse gradazioni di grigio, capacità poi ampliata a 15 gradazioni nel 1929.

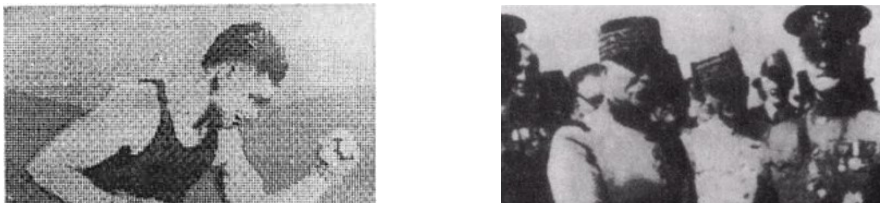

Figura 1.2. La prima immagine risale al 1921 ed è composta da 5 gradazioni di grigio.
La seconda è del 1929 ed è composta da 15 gradazioni di grigio.

Questa è la nascita delle immagini digitali, anche se non sono trattate da computer e quindi codificate in maniera differente. Inoltre in questa fase non abbiamo una vera e propria elaborazione delle immagini, ma solo una trasmissione e la successiva stampa.

Nel 1957, Russell A. Kirsch produsse un dispositivo che generava dati digitali che potevano essere archiviati in un computer; questo utilizzava uno scanner a tamburo e un tubo fotomoltiplicatore.
Negli anni immediatamente successivi questo porto a notevoli sviluppi.

Nel 1960 presso il Jet Propulsion Laboratory, il Massachusetts Institute of Technology, i Bell Laboratories, l'Università del Maryland, e altre strutture di ricerca svilupparono molte delle tecniche di elaborazione digitale di immagini (o della trasformazione di immagini digitali come spesso veniva chiamata) al fine di evitare le debolezze operative delle fotocamere a pellicola per missioni scientifiche e militari.

Queste trovarono poi applicazione in immagini satellitari, immagini medicali, videocitofono, riconoscimento ottico dei caratteri, e miglioramenti fotografici.

Il costo dell'elaborazione in quel periodo era piuttosto alto con l'apparecchiatura di elaborazione. Le cose cambiarono negli anni settanta, quando computer più economici e hardware dedicato furono resi disponibili sul mercato.
Le immagini allora potevano essere elaborate in tempo reale, l'elaborazione digitale sostituisce così i vecchi metodi a pellicola per molti scopi.

Negli anni 2000, grazie all'avvento di computer più veloci, l'elaborazione delle immagini digitali diventò la forma più comune di elaborazione delle immagini e, in generale, divenne il metodo più utilizzato data la sua versatilità e il basso costo.

La tecnologia di elaborazione delle immagini digitali per applicazioni mediche è stata inserita nel 1994 nella Hall of Fame della Space Foundation.

1.2 Principali problemi di elaborazione digitale

Al giorno d'oggi i nostri computer ci permettono una vasta gamma di elaborazioni digitali, ce ne sono alcuni, più semplici ma di notevole interesse, ormai già perfezionati ed altri che sono ancora oggetto di studio.

1.2.1 Rotazioni, riflessioni,etc

Le più semplici in assoluto riguardano "movimenti rigidi" come la traslazione o la rotazione, o anche omeomorfismi come la riflessione. Questi sono molto utili per introdurre i primi concetti matematici, come l'impiego di matrici.
Definiamo una matrice di trasformazione che moltiplicata per un vettore di coordinate ci rstituisca un altro vettore di coordinate. Questo vuol dire che quel punto va spostato dalle coordinate in cui si trovava a quelle appena calcolate. Ad esempio

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

E' una matrice di riflessione lungo l'asse verticale, infatti:

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} -x & y & 1 \end{bmatrix}$$

Cioè ogni punto rimane alla stessa ma cambia la propria x con -x, che è esattamente ciò che intendiamo per riflessione lungo l'asse verticale.

Altri esempi possono essere:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad -\text{Riflessione lungo l'asse orizzontale}$$

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad -\text{Scalamento}$$

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad -\text{Rotazione di un angolo theta}$$

1.2.2 Cambio prospettiva

Una delle trasformazioni più semplici è quella del cambio prospettiva. Una volta compresa la questa trasformazione si è fatto anche un primo passo per parlare di warping, di cui parleremo più avanti.

Supponiamo di avere la foto di un documento e di volerla migliorare in modo da estrarne solo il documento e che i suoi bordi concidano quindi con i bordi dell'immagine.

In genere la cosa più semplice ed affidabile è quella di far scegliere i 4 angoli del documento ad un utente. Volendo automatizzare il processo possiamo però scegliere un'altra strada, ossia estrarre i bordi dell'immagine e cercare tra questi quelli che formano un trapezio, presumibilmente quello sarà il documento.

Ottenuti i 4 angoli, calcoliamo la lunghezza e la larghezza della nostra nuova immagine. Per fare ciò iniziamo con il calcolare la lunghezza dei 4 lati, questi li otterremo banalmente con il teorema di Pitagora. A questo punto prendiamo i lati a due a due che non hanno vertici in comune e ne confrontiamo le lunghezze.
Per ogni coppia prendiamo la lunghezza maggiore (si può anche usare quella minore ma non dilunghiamoci su questa scelta).

Ora siamo in grado di calcolare la matrice di trasformazione. Essa sarà del tipo:

$\begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ c_1 & c_2 & 1 \end{bmatrix}$ dove $\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$ definisce rotazione, scalamento, etc., $\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$ è un vettore di traslazione e $\begin{bmatrix} c_1 & c_2 \end{bmatrix}$ è un vettore di proiezione (che è nullo se il riquadro iniziale e quello finale coincidono).

L'intento è che moltiplicando tale matrice per il vettore $\begin{bmatrix} \text{coordinata x} \\ \text{coordinata y} \\ 1 \end{bmatrix}$ otterremo il vettore $\begin{bmatrix} \text{nuova coordinata x*k} \\ \text{nuova coordinata y*k} \\ k \end{bmatrix}$ eseguendo questa operazione per tutti i punti, costruiamo l'immagine desiderata.

1.2.3 Morphing

Il morphing è uno dei primi effetti digitali sviluppati dall'industria cinematografica e consiste nella trasformazione fluida, graduale e senza soluzione di continuità tra due immagini di forma diversa, che possono essere oggetti, persone, volti, paesaggi.

Il morphing non è altro che l'uso in contemporanea di una dissolvenza incrociata e di un effetto di deformazione chiamato warping (termine inglese che significa appunto deformazione).

Per operare il warping si definiscono sull'immagine di partenza dei "punti chiave" che possono essere uniti tra di loro con delle linee e si definiscono sull'immagine di destinazione i corrispondenti punti e di conseguenza le corrispondenti linee.

Figura 1.3. Processo di morphing con alcuni risultati intermedi.

Durante la dissolvenza dall'immagine iniziale a quella finale, le immagini vengono deformate facendo in modo che ciascun punto chiave si muova lungo il percorso che porta dalla sua posizione nell'immagine di partenza alla posizione del corrispondente punto nell'immagine di arrivo.

Capitolo 2

Nozioni introduttive

Viviamo in un era digitale, le immagini passano generalmente per un calcolatore prima di essere stampate, o in ogni caso possono essere sempre scannerizzate (con strumenti più o meno precisi) così da averne una copia digitale. E' in questa fase che l'immagine subisce il processo di filtraggio, nel calcolatore, quando pè in formato digitale, Per capire cos'è un filtro occorre dunque chiedersi cosa sia un'immagine digitale.

2.1 Immagini digitali

Per capire come codificare un'immagine per memorizzarla in fgormato digitale ci chiediamo prima che cos'è un'immagine.

> Forma esteriore degli oggetti corporei, in quanto viene percepita attraverso il senso della vista, o si riflette – come realmente è, o variamente alterata – in uno specchio, nell'acqua e sim., o rimane impressa in una lastra o pellicola o carta fotografica.
>
> Vocabolario Treccani

Un'immagine viene rappresentata, impressa quindi su superfici, cioè oggetti bidimensionali, di dimensioni finite e le vediamo perchè i nostri occhi percepiscono il susseguirsi di colori diversi. Come codificare tali entità? Come tutti gli oggetti reali, sebbene abbiano dimensioni finite, il susseguirsi delle immagini avviene in una maniera che possiamo considerare come continua. Questo è il primo problema che ci si pone quando si pensa a come codificare delle immagini. La soluzione più largamente utilizzata è anche quella più semplice ed intuitiva, ossia di discretizzare tale distribuzione di colori. Dividiamo l'immagine con una griglia ed ad ogni casella, che d'ora in poi chiameremo "pixel", assegnamo un colore. E' ovvio che così facendo si perdono dei dettagli, la quantità di dettagli che riusciamo a conservare piò variare enormemente, un minimo si perde sempre ma è un prezzo che siamo disposti a pagare.
Facciamo un esempio:

Figura 2.1. Confronto immagine originale e immagine codificate utilizzando una griglia 4x4.

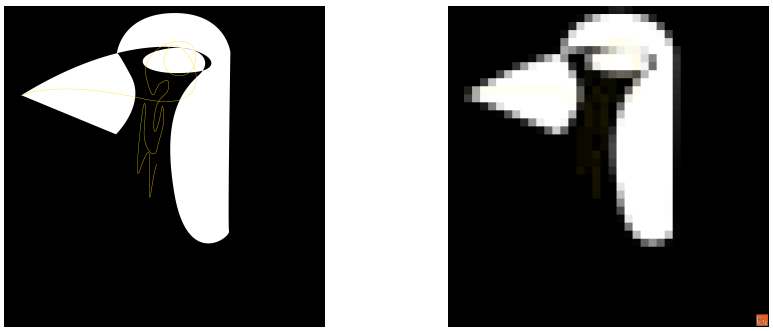

Figura 2.2. Confronto immagine originale e immagine codificate utilizzando una griglia 80x80.

E' semplice vedere come ad una griglia più fitta corrisponda una miglior qualità dell'immagine, questo è il concetto di "risoluzione di un'immagine". Una miglior risoluzione però costa, come anticipato, in termini di memoria. Una griglia 4x4 corrisponde a 16 pixel, ad una griglia 80x80 corrispondono invece 6400 pixel! Questo vuol dire che la seconda immagine pesa 400 volte di più della prima.

Volendo definire in maniera più precisa cosa e' un'immagine digitale, diremmo che quest'ultima è una funzione da $\mathbb{R}^2$ in $\mathbb{R}^3$ cioè, date in input due coordinate, essa restituisce un colore (che è formato da 3 canali RGB). Se però l'immagine è in bianco e nero la questione si semplifica: l'immagine diventa una funzione da $\mathbb{R}^2$ in $\mathbb{R}$, dal momento che per codificare un colore appartenente alla scala di grigio basta un solo canale, il livello di luminosità. Difatti, la riduzione da tre ad un solo canale rappresenta un grosso vantaggio, permettendo di diminuire di due terzi lo spazio di memoria occupato.

2.2 Cosa sono i filtri

Una volta capito che un'immagine è una funzione possiamo definire un filtro come una seconda funzione che convoluta alla prima da il risultato richiesto.

Una equazione alle derivate parziali (PDE) esprime una evoluzione, sia u la nostra immagine e u_0 lo stato in cui si trova inzialmente, allora per convoluzione possiamo dire che

$$u(x) = \frac{1}{w(x)} \int \int d(x-\xi)\tilde{d}(u_0(x) - u_0(\xi))u_0(\xi)d\xi. \tag{2.1}$$

$$\text{con } w(x) = \int \int d(x-\xi)\tilde{d}(u_0(x) - u_0(\xi))d\xi$$

Definiti questi concetti siamo pronti ad iniziare la trattazione vera e propria

Parte II

Prima Parte

Capitolo 3

Equazione del calore e metodo Perna-Malik

L'equazione del calore, come intuibile dal nome che porta, è stata formulata per determinare l'evoluzione di un sistema isolato che presenta al suo interno una data distribuzione di calore. Euristicamente, non è difficile pensare che possiamo codificare (pensando ad un'immagine in bianco e nero) i pixel più luminosi come punti "più caldi" e quelli più scuri come punti "più freddi" ed applicare così l'equazione del calore.
vedremo con uno script MATLAB come opera nella pratica.
Per fare ciò opereremo una approssimazione alle differenze finite per il calcolo delle derivate.

3.1 Metodo delle differenze finite

Questo metodo, come anticipato, lo utilizzeremo per calcolare un valore approssimato delle derivate. Il proceimento si rifà molto alla definizione in se di derivata. Decidiamo quindi di approssimare

$$\frac{du}{dx} \approx \frac{u_{i+1}-u_i}{\Delta(x)}$$

Se la derivata seconda è la derivata della derivata, allora approssimiamo

$$\frac{d^2u}{dx^2} \approx \frac{(\frac{u_{i+1}-u_i}{\Delta(x)})_{i+1} - (\frac{u_{i+1}-u_i}{\Delta(x)})_i}{\Delta(x)}$$

$$= \frac{\frac{(u_{i+1}-u_i)_{i+1}}{\Delta(x)} - \frac{(u_{i+1}-u_i)_i}{\Delta(x)}}{\Delta(x)}$$

$$= \frac{\frac{(u_{i+2}-u_{i+1})}{\Delta(x)} - \frac{(u_{i+1}-u_i)}{\Delta(x)}}{\Delta(x)}$$

$$= \frac{\frac{(u_{i+2}-u_{i+1})-(u_{i+1}-u_i)}{\Delta(x)}}{\Delta(x)}$$

$$= \frac{(u_{i+2}-u_{i+1})-(u_{i+1}-u_i)}{\Delta(x)^2}$$

$$= \frac{u_{i+2}-u_{i+1}-u_{i+1}+u_i}{\Delta(x)^2}$$

Per una migliore approssimazione è bene utilizzare un valore di $\Delta(x)$ quanto più basso possibile. Il massimo che possiamo fare è vedere la differenza tra un pixel e quello adiacente ossia $\Delta(x) = 1$, ma allora

$$\frac{d^2u}{dx^2} \approx \frac{u_{i+2}-u_{i+1}-u_{i+1}+u_i}{\Delta(x)^2} = u_{i+2} - 2u_{i+1} + u_i$$

Capiamo che scorrendo tutti gli indici questo è equivalente a $u_{i+1} - 2u_i + u_{i-1}$.
In sintesi utilizzeremo per il calcolo del laplaciamo l'approssimazione

$$\frac{d^2u}{dx^2} \approx u_{i+1} - 2u_i + u_{i-1}$$

3.2 Errore di troncamento

Vale la pena di fare qualche considerazione sull'errore di troncamento che commettiamo adottando queste approssimazioni.
Per definizione, l'errore è la differenza tra il valore esatto e quello approssimato, ossia:

$$\frac{d^2u}{dx^2}\bigg|_{x_i} - \frac{u_{i+1}-u_i}{\Delta(x)} \approx \frac{\Delta(x)^2}{2}\frac{d^2u}{dx^2}\bigg|_{\xi}$$

Risulta quindi evidente che l'errore dipende da $\Delta(x)^2$. Ad un'attenta analisi possiamo vedere che tra i metodi di approssimazione in avanti, in indietro o centrale, i primi due sono uguali tra di loro, quello centrale invece è diverso, esso dipende da un $\delta(x)^3$ e non da un $\delta(x)^2$, questo vuol dire che per $\delta(x) < 1$ funziona meglio, per $\delta(x) > 1$ funziona peggio. nel nostro caso $\delta(x) = 1$ quindi la scelta è indifferente. Ma procediamo con ordine.
Per brevità di notazione poniamo $\Delta(x) = h$.
Dagli sviluppi di Taylor in x_j di $u(x_{j\pm1}) = u(x_j \pm h)$ fino al terzo ordine, si ottiene che

$$u''(x_j) = \frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2} - \frac{1}{12}u(4)(\xi_j)h^2$$

dove ξ_j e un punto opportuno in $(x_{j-1}, xj + 1)$. Quindi, considerando il problema modello con condizoni di Dirichlet, per $j = 1, ..., N - 1$, si può scrivere che

$$\frac{-u(x_{j+1}) + 2u(x_j) - u(xj - 1)}{h^2} + \frac{1}{12}u(4)(\xi_j)h^2 + \sigma(x_j)u(x_j) = f(x_j).$$

Se introduciamo la notazione $\tau_j = \frac{1}{12}u(4)(\xi_j)h^2$ (errore di troncamento locale) e poniamo:
$\boldsymbol{u} = [u_x...u_{N-1}]^T$
$\boldsymbol{\tau_h} = [\tau_x...\tau_{N-1}]^T$
$\boldsymbol{b_h} = [(f(x_1) + \frac{g_0}{h^2}, f(x_2), ..., f(x_{N-2}), f(x_{N-1}) + \frac{g_1}{h^2}]^T$
possiamo allora scrivere in forma matriciale, $A_h\boldsymbol{u} = \boldsymbol{b_h} + \boldsymbol{\tau_h}$ dove
$A_h = \frac{1}{h^2}tridiag(-1,2,-1) + diag(\sigma_1, ..., \sigma_{N-1})e\sigma_j = \sigma(x_j)$.

Il metodo alle differenze finite consiste allora nel determinare un'approssimazione u_h di u andando a risolvere il sistema lineare $A_hu_h = b_h$. Osserviamo che $\boldsymbol{u_h}$ risulta ben definito in quanto A_h è una matrice non singolare.

Risultando che τ_h tende a zero quando h tende a zero, il metodo dicesi consistente. In particolare nel nostro caso, come osservato in partenza, risulta $\tau_h = O(h^2)$.
Tuttavia la consistenza non assicura da sola la convergenza del metodo.
Per studiarne laconvergenza dobbiamo considerare il comportamento dell'errore $\boldsymbol{e_h} = \boldsymbol{u_h} - \boldsymbol{u}$ quando h tende a zero. Dato che risulta $A_h\boldsymbol{e_h} = \boldsymbol{\tau_h}$, e quindi $\boldsymbol{e_h} = A_h^{-1}\boldsymbol{\tau_h}$ possiamo scrivere:
$||\boldsymbol{e_h}|| \geq ||A_h^{-1}|||\boldsymbol{\tau_h}||$
Vogliamo allora far vedere che, lavorando in norma infinito, siamo in grado di trovare una costante che, per ogni h, maggiora $||A_h^{-1}||$.
A questo scopo osserviamo che si può dimostrare che sia A_h che la matrice
$A_{0h} = \frac{1}{h^2}tridiag(-1,2,-1)$ hanno inversa non negativa, e si ha:

$$A_{0h}^{-1} - Ah^{-1} = A_{0h}^{-1}(A_h - A_{0h})A_h^{-1} \geq 0$$

Per le ipotesi su σ questo implica $A_h^{-1}|| \leq A_{0h}^{-1}||$ osserviamo che $||A_{0h}^{-1}|| = max_j(A_{0h}^{-1}\boldsymbol{e})_j$ dove $\boldsymbol{e}$ indica il vettore di tutti 1.
Osservando che la soluzione esatta del problema $-u'' = 1, u(0) = u(1) = 0$, è il polinomio di secondo grado $\phi(x) = \frac{x(1-x)}{2}$, possiamo concludere che $A_{0h}^{-1}\boldsymbol{e})_j = \phi(x_j)$ e quindi che
$A_h^{-1}|| \leq A_{0h}^{-1}|| \leq max_{0<x<1}|\phi_x|$.

Questo risultato di stabilità ci permette di concludere che l'errore e_h ha lo stesso ordine dell'errore di troncamento $\boldsymbol{\tau_h}$ e di conseguenza che il metodo è convergente del secondo ordine. Si noti che l'uniforme limitatezza della norma di A_h^{-1} implica che il metodo numerico sia stabile

3.3 L'equazione del calore

Prendiamo in esame l'equazione del calore

$$\begin{cases} \frac{\partial u}{\partial t}(t,x) - \Delta u(t,x) = 0 \; x \in \mathbb{R}^2, t \geq 0 \; . \\ u(0,x) = u_0(x) \; . \end{cases} \tag{3.1}$$

L'equazione del calore, come intuibile dal nome che porta, è stata formulata per determinare l'evoluzione di un sistema isolato che presenta al suo interno una data distribuzione di calore. E' banale pensare che con il passare del tempo il calore si distribuisca, tendendo per un tempo infinito ad una distribuzione uniforme.

Euristicamente, non è difficile pensare che possiamo codificare (pensando ad un'immagine in bianco e nero) i pixel più luminosi come punti "più caldi" e quelli più scuri come punti "più freddi" ed applicare così l'equazione del calore.
Otteniamo quindi un'immagine sempre più "liscia", otteniamo di fatto una sfocatura, e per un numero di iterazioni idealmente infinito ci ritroveremmo con una distribuzione uniforme di colore, ossia una tinta unita.

Vediamo con uno script matlab come opera nella pratica.

```
Im=imread('parrot.jpeg');    %Apro l'immmagine
Im=rgb2gray(Im);             %La trasformo in bianco e nero
Im = imnoise(Im,'gaussian');%Aggiungo del rumore

%---Definisco le costanti e le condizioni iniziali

[ny, nx, ~]=size(Im)         %Dimensioni dell'immagine
dt=0.25;                     %Passo temporale
u=double(Im);                %Copia dell'immagine originale su cui
                              lavorare
T=3                          %Tempo, ossia T/dt + 1 definisce il numero
                               di iterazioni da eseguire
k=0.5;

%---Mostro l'immagine originale
imshow(uint8(Im))
title('immagine originale');

%---Metodo eq del calore
u=double(Im);
for t = 0:dt:T
   u_xx = u(:,[2:nx nx],:) - 2*u + u(:,[1 1:nx-1],:);  % derivata
                                                        seconda lungo x
```

```
21    u_yy = u([2:ny ny],:,:) - 2*u + u([1 1:ny-1],:,:);  % derivata
                                                            seconda lungo y
22    u= u + k*dt*(u_xx+u_yy);
23    temp=u;
24 end
```

Provando a cambiare il tempo, ossia il numero di iterazioni, osserviamo come un maggior lasso di tempo produca immagini più sfocate.

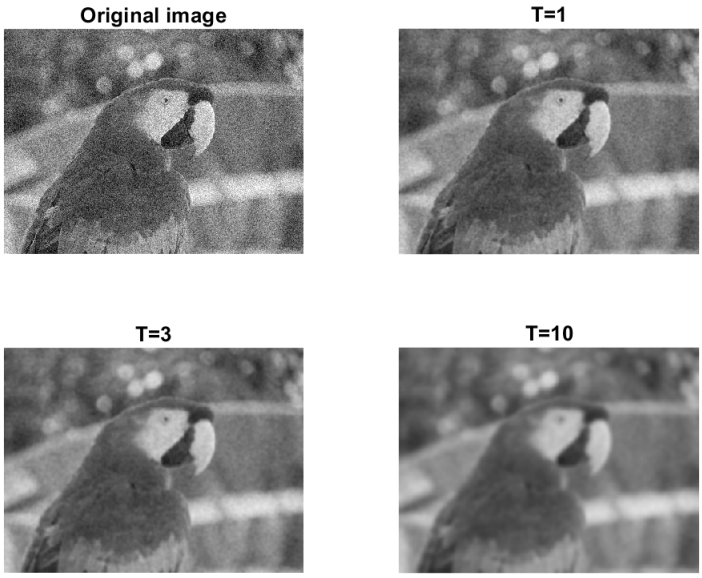

Figura 3.1. Effetti dell'eq del calore nel tempo.

Questo metodo però non è poi molto utile in generale, sì, il rumore viene eliminato o almeno ridotto ma si perdono importanti dettagli, esistono procedimenti come il metodo Perona-Malik che sono decisamente più utili. L'idea del metodo Perona-Malik è di applicare l'equazione del calore nelle regioni in cui il colore è più uniforme, così da preservarne i bordi

3.4 Rilevamento dei bordi

Operando una derivata in una data direzione, per il significato in sè di derivata, questa assume valori più elevati quando la variazione è elevata, e assume valori nulli quando non c'è variazione in quella direzione. Per questo motivo, applicata ad un' immagine, ne rileviamo i bordi!

Presa una tinta unita la derivata sarà quindi nulla in ogni suo punto (è intuitivo, se un'immagine è una funzione che, date due coordinate restituisce un colore, allora una tinta unita è una funzione costante ed in quanto tale ha derivata nulla).

Operando una derivata seconda in una data direzione, per il significato in sè di derivata seconda, questa assume valori più elevati quando la concavità è più stretta, e assume valori pressocchè nulli quando non ci sono concavità (possiamo pensare alle concavità come a dei picchi o dei ventri, su di una immagine vuol dire chiazze di colore diverso).
Presa una sfumatura di colore che varia in maniera lineare, la derivata seconda sarà nulla in ogni suo punto, la derivata prima sarà invece costante. Scriviamo un piccolo script su MATLAB che ci mostri questo aspetto

```
%---Operazioni preliminari
Im=imread("nome_immagine.png"); %Apro l'immmagine

[ny, nx, ~]=size(Im)        %Memorizzo le dimensioni dell'immagine
u=double(Im);               %Copia dell'immagine originale su cui
    lavorare
h=80;                       %Definisco un parametro che usero' per
                             enfatizzare i bordi in fase di stampa


%---Calcolo tutte le derivate
u_x =  u(:,[1 1:nx-1],:) - u;                       %derivata
                                                     prima lungo x
u_xx = u(:,[2:nx nx],:) - 2*u + u(:,[1 1:nx-1],:);  %derivata
                                                     seconda lungo x
u_y =  u([1 1:ny-1],:,:) - u;                       %derivata
                                                     prima lungo y
u_yy = u([2:ny ny],:,:) - 2*u + u([1 1:ny-1],:,:);  %derivata
                                                     seconda lungo y
u_xy = u_x([1 1:ny-1],:,:) - u_x;                   %derivata
                                                     seconda mista

%---Stampo i risultati
figure()
subplot(2,3,2),text(0.3,0,nome,'FontSize',20); axis off
subplot(2,3,4), imshow(Im)
title('Immagine originale')
subplot(2,3,5), imshow(uint8(h*abs(u_x)))
title('h*u_x')
subplot(2,3,6), imshow(uint8(h*abs(u_y)))
title('h*u_y')

figure()
subplot(2,3,2),text(0.3,0,nome,'FontSize',20); axis off
subplot(2,3,4), imshow(Im)
title('Immagine originale')
subplot(2,3,5), imshow(uint8(h*abs(u_x + u_y)))
title('h*(u_x + u_y)')
subplot(2,3,6), imshow(uint8(h*abs(u_xx + u_yy)))
title('h*(u_xx + u_yy)')
```

Vediamo con diverse immagini molto semplici se otteniamo i risultati attesi.

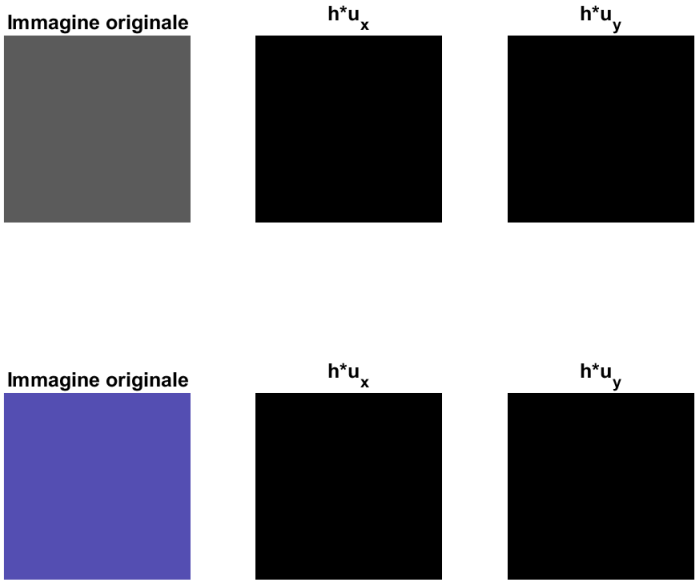

Figura 3.2. Derivate parxiali di una tinta unita.

Possiamo vedere come con un'immagine a tinta unita (che sia essa in bianco e nero o a colori) le derivate sono nulle, quindi lo saranno anche gradiente e laplaciano. Confermiamo inoltre che questi principi valgono sia a colori che in bianco e nero.

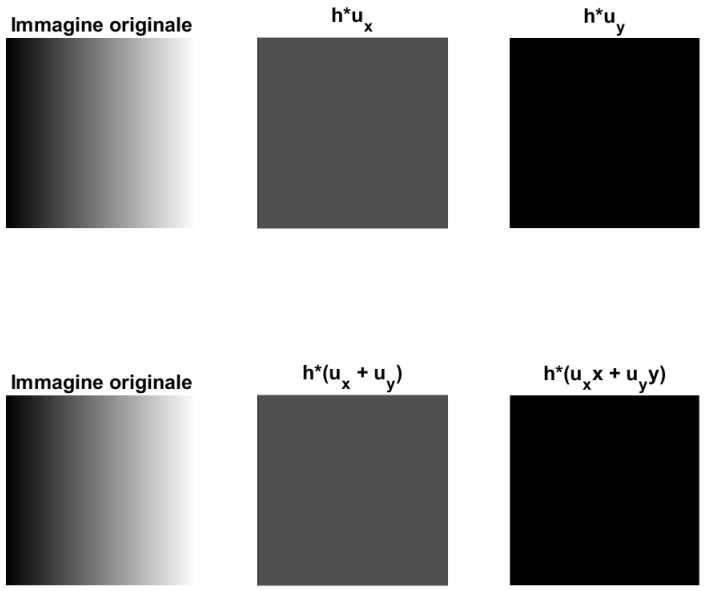

Figura 3.3. Derivate parziali, gradiente e laplaciano di una sfumatura orizzontale.

Guardando invece ad una immagine che presenta una sfumatura lineare lungo l'asse x, la derivata lungo x assume un valore costante mentre la derivata lungo y è nulla, proprio perchè lungo y non c'è variazione mentre lungo x c'è una variazione costante.
Ovviamente, date queste premesse, il gradiente sarà costante uguale u_x (siccome $u_y = 0$) e quindi il laplaciano sarà nullo. Il fatto che in entrambi questi esempi il laplaciano sia nullo è un buon segno, lo useremo per rilevare i bordi ed in queste immagini non ve ne sono, quindi è giusto che il laplaciano sia nullo.

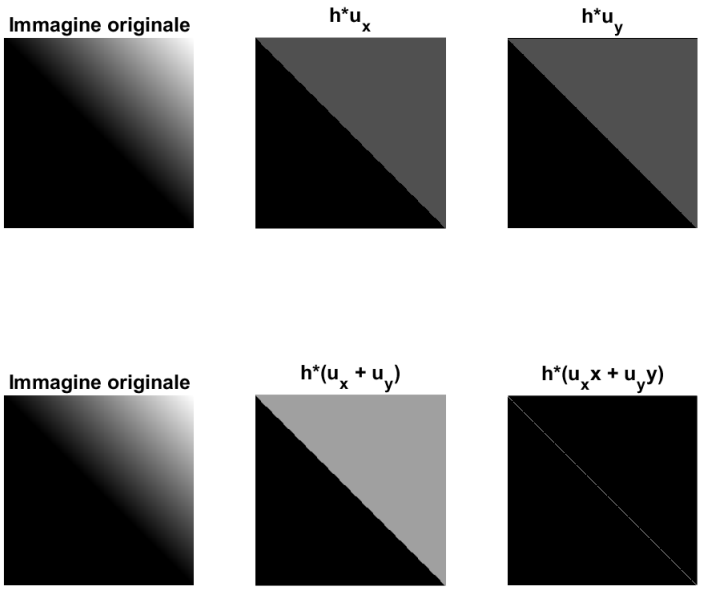

Figura 3.4. Derivate parziali, gradiente e laplaciano di una sfumatura diagonale.

Se prendiamo una sfumatura diagonale ma solo su metà immagine vediamo dei riultati interessanti: entrambe le derivate parziali sono nulle nelle regioni in cui non c'è sfumatura, esattamente come nel caso della tinta unita, ed entrambe sono costanti dove c'è sfumatura (che ricordiamo essere lineare).
Tutto ciò riconferma quanto visto dai punti precedenti, decidiamo quindi di volgere uno sguardo al gradiente ed al laplaciano e noteremo qualcosa di interessante, il gradiente ha un aspetto molto simile alle due derivate parziali, sommando i loro valori è semplicemente più luminoso.
Per quanto riguarda il gradiente la storia cambia. Le derivate seconde sono indicatrici della variazione delle derivate prime, cioè della variazione della variazione del valore della funzione, ma l'unica variazione che hanno le derivate prime è lungo la diagonale. Abbiamo così individuato il nostro primo bordo, cioè la diagonale che divide di fatto due regioni, una in cui il colore è costante ed una in cui sfuma.

Facciamo ancora un esempio, proviamo ad introdurre una semplice figura e vediamo cosa accade.

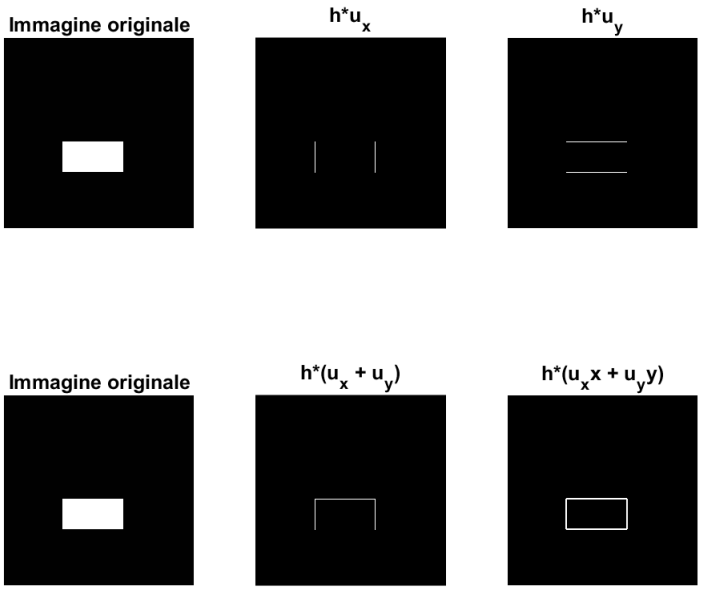

Figura 3.5. Derivate parziali, gradiente e laplaciano di una figura semplice.

Abbiamo importato un rettangolo nero su di uno sfondo bianco, come confermato anche dalla prima sfumatura, la derivata lungo x rileva i bordi verticali, quella lungo y i bordi orizzontali, dalla loro somma (quindi dal gradiente) otteniamo già il bordo del rettangolo. Il bordo così ottenuto è un bordo che idealmente rimarrà inalterato a prescindere dall'ordine della derivata, in particolare quindi anche per derivate seconde, quindi il laplaciano continua a soddisfare la nostra richiesta di determinare i bordi.
Come detto: "Il bordo così ottenuto è un bordo che idealmente rimarrà inalterato a prescindere dall'ordine della derivata" cerchiamo di capire perchè. Presa una striscia di pixel, cioè uno strato della nostra immagine (immaginiamola quindi come una funzione da R in R), otterremo quindi una funzione porta!
La funzione porta non è derivabile in senso classico, ripensando alla definizione di derivata avremmo un valore di +infinito prima e -infinito poi. La sua derivata sarà quindi una coppia di delta di Dirac