**JpegTransformer**

Metodi del Calcolo Scientifico

AA 2022-2023

Progetto 2 [Algebra lineare numerica]

Compressione di immagini tramite DCT

Lecchi Gabriele - 852134

Titta Lorenzo – 852107

**PRIMA PARTE**

**Introduzione**

Java è il linguaggio utilizzato per l’implementazione della nostra versione dell’algoritmo DCT2 (questo algoritmo risulta essere molto utile nell’elaborazione di segnali e immagini particolarmente per la compressione con perdita).

L’algoritmo con il quale lo abbiamo confrontato per verificare risultati e tempistiche fa parte della libreria open source JTransforms disponibile sulla piattaforma GitHub al seguente link: <https://github.com/wendykierp/JTransforms>.

**Confronto degli algoritmi**

Come si può notare dal seguente grafico c’è differenza tempistica sostanziale tra i due algoritmi, la nostra versione esegue l’algoritmo in tempi nettamente superiori all’algoritmo della libreria.  
La nostra versione inoltre presenta un andamento esponenziale ben visibile dal grafico, mentre l’algoritmo fast della libreria di JTransforms per matrici piccole presenta una crescita irregolare al variare della matrice di input.

. Grafico dei tempi di esecuzione di entrambi gli algoritmi

2. Grafico dei tempi di esecuzione dell'algoritmo da noi sviluppato

3. Grafico dei tempi di esecuzione dell'algoritmo preso dalla libreria JTransforms

Per l’algoritmo sviluppato da noi ci siamo fermati a matrici di dimensione 500 x 500 a causa degli elevati tempi di esecuzione; mentre per l’algoritmo preso dalla libreria queste matrici erano troppo piccole per mostrare la crescita esponenziale dei tempi; dunque, abbiamo continuato l’esecuzione fino a matrici di 20.000 x 20.000 dove si può notare in modo più evidente la crescita esponenziale.

**SECONDA PARTE**

**Introduzione**

Per la seconda parte abbiamo pensato di creare un’interfaccia il più intuitiva possibile che permettesse di eseguire le seguenti operazioni:

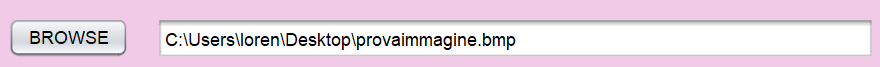
* Scelta immagine (e visualizzazione percorso sul dispositivo)
* Visualizzazione immagine pre-modifiche
* Sezione di impostazione parametri
* Visualizzazione immagine post modifiche

Come ambente di sviluppo ci siamo appoggiati a NetBeans che offre l’utilizzo di un’interfaccia grafica per la programmazione sfruttando il linguaggio Java. All’interno del progetto abbiamo usufruito di JTransforms, una libreria open source che permette di applicare la Discrete Cosine Transform (DCT) a matrici fornite dall’utente.

**Struttura del codice**

Punto 1 – Scelta immagine

Per la scelta dell’immagine abbiamo reso disponibile due possibili iterazioni con l’utente scelta del percorso tramite un bottone che permette all’utente di selezionare direttamente l’immagine o una text box per l’inserimento manuale del percorso di essa.



Nella seguente immagine viene mostrato il codice per il quale tramite l’oggetto JFileChooser si prendere l’immagine dal percorso selezionato, in aggiunta abbiamo inserito un controllo per il quale l’utente possa solamente importare nel programma file bitmap.



Punto 2 - Visualizzazione immagine pre modifiche

Questo passaggio è molto semplice: viene solamente visualizzata l’immagine proposta all’interno di una label, la quale viene adattata allo spazio assegnato alla label tramite il codice seguente:

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Parte 3 - Sezione di impostazione parametri

In questa sezione l’utente interagisce per definire i parametri per l’applicazione dell’algoritmo: il primo è il parametro f (numero intero) che stabilisce la grandezza dei blocchi sulla quale applicare la dct2 (blocchi fxf), mentre il secondo valore d (numero intero) è la soglia di taglio delle frequenze che deve essere compreso tra 0 e (2f-2). Di seguito viene mostrata la parte di interfaccia legata a questa sezione.

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente

Il codice seguente è stato implementato per eseguire dei controlli sull’input affinchè i parametri inseriti dall’utente, anche se errati, possano essere corretti automaticamente dal programma senza causare eccezioni, sia per f sia per d.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Dopo aver eseguito i vari controlli sugli input abbiamo implementato il codice necessario per la risoluzione dei task come segue:

* Parte di codice verde, viene gestita la creazione del blocco dell’immagine, tramite una classe Raster messa a disposizione da java.
* Parte di codice rosso, questa parte di codice ha il compito di dividere i vari blocchi dell’immagine tramite il valore f inserito dall’utente (r.getsamples(…)) dall’apposita textbox dell’interfaccia e poi solo successivamente viene eseguita la dct2 su tutti questi blocchi (dct.forward(…)).
* Parte di codice azzurra, vengono eliminate le frequenze in base al parametro inserito dall’utente, utilizzando due cicli innestati: il primo sul numero di bande ottenute dal raster dell’immagine, mentre il secondo che cicla sull’array dei pixel dell’immagine.
* Parte di codice rosa, si esegue la dct2 inversa per l’array appena modificato tramite i tagli.
* Parte di codice gialla, viene ciclato tutto l’array eseguendo i seguenti passi per ogni valore: viene inizialmente convertito il valore in intero e poi arrotondato secondo i criteri richiesti, 0 se il valore è negativo, 255 se il valore è >255.
* Parte di codice arancione, viene ricreata l’immagine tramite il Raster in scrittura.

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Parte 4 – Visualizzazione immagine post modifiche

Per la visualizzazione finale abbiamo inserito le due immagini una a lato dell’altra, affinchè si possano notare le differenze tra il prima e il dopo dall’applicazione del filtro, come mostrato nella seguente sezione.

**Esecuzione codice e risultati finali**

Di seguito vengono illustrati i risultati dell’applicazione del nostro codice applicate su alcune immagini fornite nella cartella di e-learning.





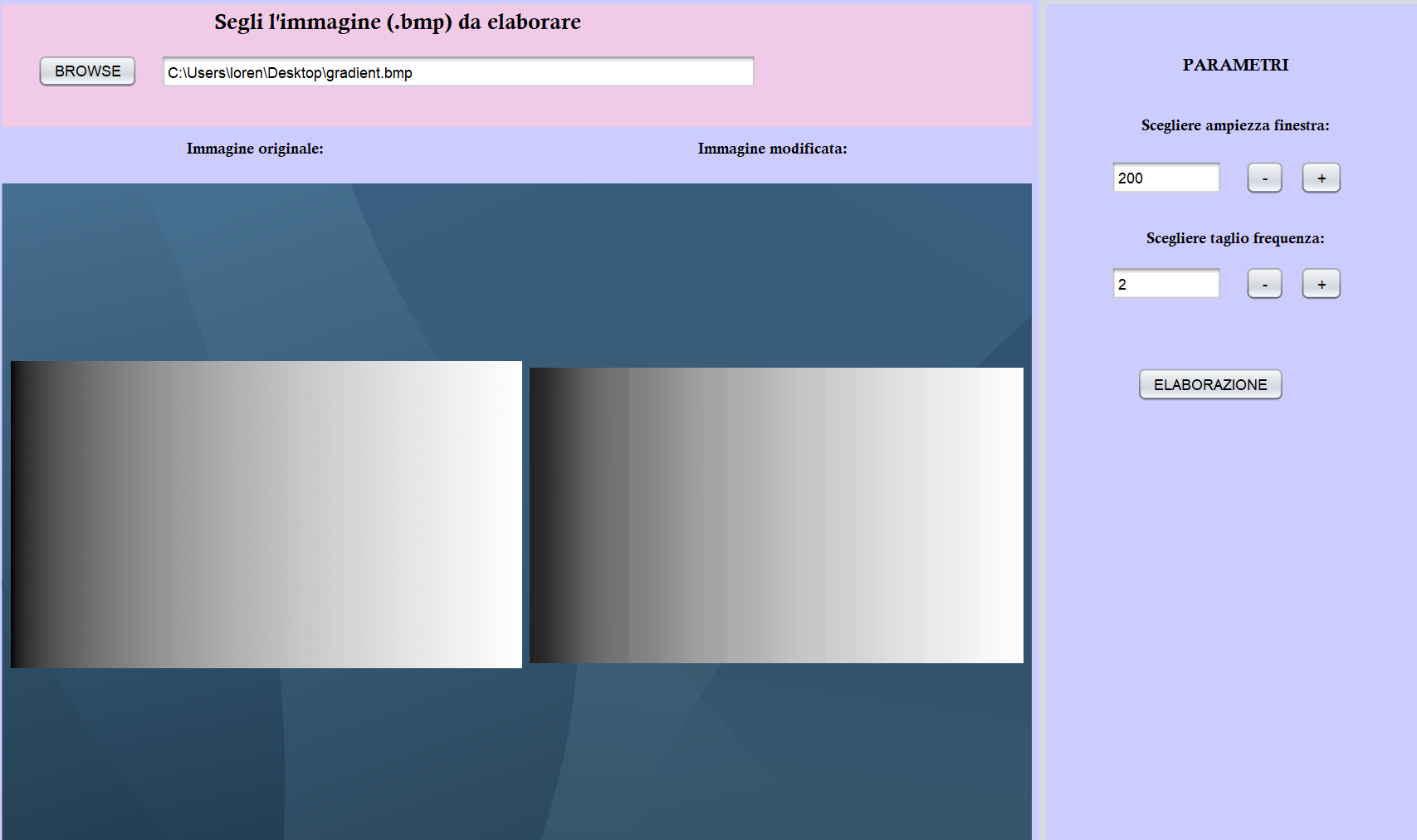


Immagine che contiene testo, schermata, Software multimediale, Sito Web

Descrizione generata automaticamente