CACCIA AL TESORO: DOV'È IL PERSEO?

Scopo

Decrittare mediante il software *Mathematica* un'immagine in formato jpg, che contiene una mappa in cui è indicata la posizione di un quadro misterioso, *il Perseo.*

Materiali e strumenti

- PC col software *Mathematica*
- Smartphone con l'app *Layar* per la realtà aumentata

Mathematica *Layar*

Premessa

Per rappresentare un'immagine digitale, essa viene suddivisa in una matrice di punti chiamati pixel (picture element), il cui colore secondo il modello RGB viene rappresentato mediante una combinazione additiva dei tre colori primari rosso, verde e blu. Usando una profondità di colore a 24 bit (8 per ogni componente), ogni colore primario può essere rappresentato con 256 sfumature possibili espresse da un numero intero tra 0 e 255. Combinando in modo diverso le intensità dei tre colori primari si possono ottenere 256^3= 16777216 colori diversi; questa modalità viene detta TrueColor, ovvero 16 milioni di colori.

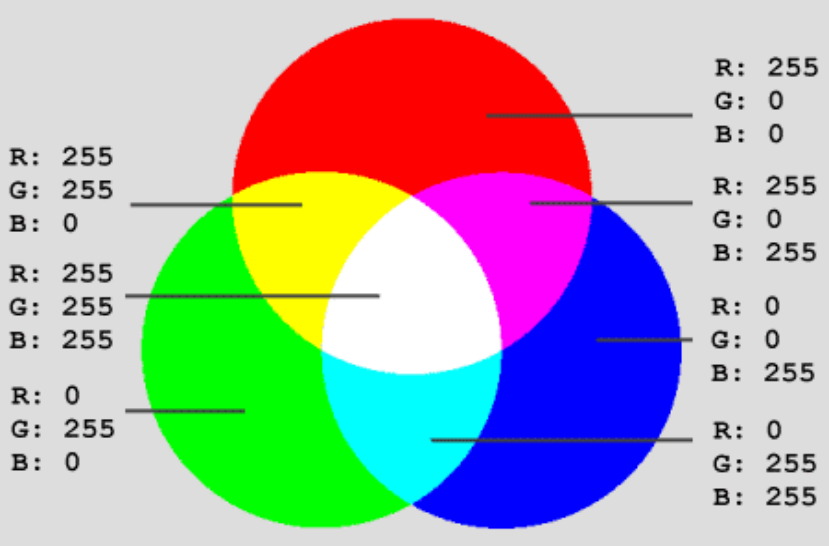

Fig. 1 Modello di colori RGB

Un'immagine in formato jpg è costituita da una matrice di dimensioni diverse a seconda della risoluzione (per esempio se la risoluzione è 800x600 si ha una matrice con 800 colonne e 600 righe); ciascuna cella rappresenta un pixel dell'immagine, il cui colore è rappresentato da una terna di numeri interi compresi tra 0 e 255 $\{n_1, n_2, n_3\}$ corrispondenti alle coordinate RGB. Questo significa che per rappresentare un quadratino rosso nel punto di coordinate [347,435] il contenuto di quell'elemento di matrice dovrà essere {255,0,0}; per avere un quadratino verde il contenuto sarà {0,255,0}, per un quadratino blu il contenuto sarà {0,0,255} e così via.
Per trasformare una immagine a colori in una a scala di grigi bisogna fare una proiezione nello spazio *grayscale*, cioè proiettare la terna $\{n_1, n_2, n_3\}$ in un numero intero tra 0 e 255 che dà la scala del grigio corrispondente.

Fig. 2 Dettaglio a colori e in bianco e nero del quadro "*Hand resist him*" di Bill Stoneham

Questa operazione di trasformazione di un'immagine a colori in una in scala di grigi viene fatta attraverso una funzione di proiezione definita all'inizio del programma:

proj[{x_, y_, z_}] := N[(0.2989*x + 0.5870*y + 0.1140*z)]

che corrisponde ad una media pesata delle coordinate RGB {x, y, z}, che tiene conto della differente percezione con cui l'occhio umano percepisce i colori (formula usata nei comuni programmi di elaborazione delle immagini come Photoshop e Gimp).
Una difficoltà sta nel fatto che alcuni software lavorano con coordinate colore costituite da numeri interi compresi tra 0 e 255, invece altri (come il Mathematica) operano con numeri reali compresi tra 0 e 1. Ovviamente per passare dall'uno all'altro sistema di rappresentazione del colore basta dividere gli interi per 255, mentre l'operazione contraria (cioè passare dai reali tra 0 e 1 agli interi) è un po' più complicata, in quanto bisogna moltiplicare per 255 e arrotondare all'intero più vicino, con il comando Round[x].
Come i vettori, anche le matrici e quindi le immagini possono essere sommate o moltiplicate; ovviamente questa operazione deve essere fatta cella per cella, cioè si sommano o si moltiplicano elementi corrispondenti allo stesso pixel. Naturalmente queste operazioni si possono fare solo tra immagini della stessa dimensione e dello stesso formato, cioè contenenti lo stesso numero di pixel.
L'istruzione seguente:

criptomap = ImageResize[Import["C:\\Dati\\PLS\\mappa jpg\\criptoRGB\\criptum.jpg"], {1200, 800}]

oltre ad importare l'immagine contenuta nel file criptum.jpg, la rende anche della dimensione standard scelta, cioè 1200x800. Va precisato che in Mathematica, come in altri linguaggi di programmazione, è sempre opportuno definire il risultato di una o più operazioni con un nome, in modo da poterlo richiamare in altre parti del programma semplicemente digitando il nome ("criptomap" nel nostro caso).
Le operazioni di somma o di moltiplicazione sono, come sappiamo invertibili, cioè si può moltiplicare o sommare un pixel di una certa quantità nella fase di crittazione e successivamente, con le operazioni inverse (sottrazione e divisione) si può ritornare all'originale, a patto naturalmente di conoscere i valori per le quali si sono sommati o moltiplicati i pixel. L'unica accortezza che bisogna avere è che, dato che l'interprete delle immagini di Mathematica è in grado di comprendere solo pixel di colore grigio compresi tra 0 e 1, il risultato dell'operazione che facciamo deve essere necessariamente un numero reale in quest'intervallo e questo si può fare attraverso il modulo del numero (x) ottenuto: in Mathematica il comando è Mod[x,1]. Per camuffare un'immagine si può, ad esempio, moltiplicare tutti i pixel per lo stesso numero, ad esempio per 0.5, ma questo non sarebbe un procedimento molto efficace dato che l'immagine conserverebbe la sua forma, modificando (scurendo in questo caso) solo il suo colore, come mostrato nella figura seguente.

Fig. 3 Immagine modificata (in questo caso scurita) moltiplicando tutti i pixel per uno stesso numero

Un altro procedimento consiste nel sommare il contenuto di ciascun pixel con una quantità variabile, cioè diversa pixel per pixel, per esempio utilizzando il contenuto di una seconda immagine della stessa dimensione. Il risultato, in questo caso, è molto più soddisfacente, come mostrato nella figura seguente.

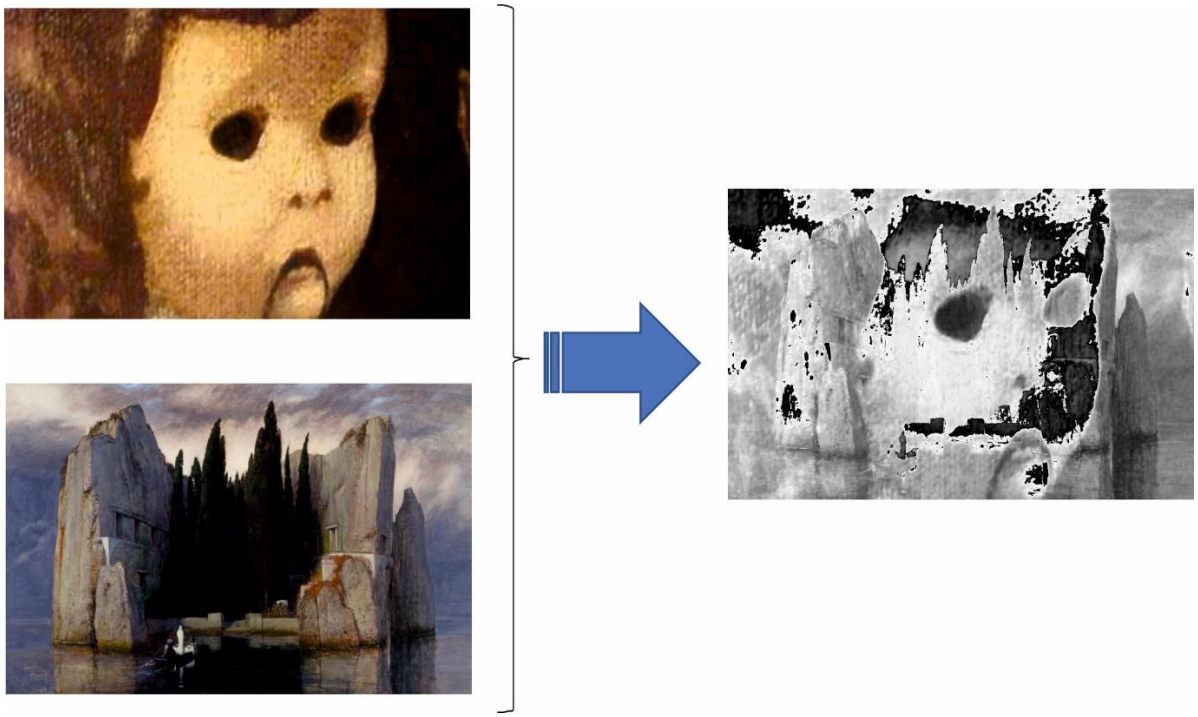

Fig. 4 Immagine criptata (a destra) sommando pixel per pixel all'immagine originale
(in alto a destra) un'altra immagine avente le stesse dimensioni (in basso a destra)

Ovviamente per poter eseguire la decrittazione, in questo caso, bisogna conoscere il file jpg che è stato sommato all'immagine originale.

Costruzione dell'algoritmo di decrittazione

- Vi è stata spedita all'indirizzo e-mail del vostro gruppo (es. 5c1@liceorecanati.org) un archivio compresso crittato (es. 5C1.zip) contenente cinque immagini in formato jpg: la mappa crittata (es. 5C1.jpg) e i 4 *quadri maledetti* necessari per decrittarla: "*Hand resist him*" di Bill Stoneham, "*L'isola dei morti*" di Arnold Böcklin, "*Scudo con testa di Medusa"* di Carvaggio, "*Il trionfo della morte*" di Pieter Brugel.

- La password per aprire l'archivio crittato è il codice alchemico ottenuto mediante il Bot Telegram in seguito all'apertura del caveau nell'esperimento di robotica.
- Le coordinate RGB ottenute dai quattro esperimenti sono i coefficienti moltiplicativi da usare insieme ai quadri nell'algoritmo di decrittazione col software Mathematica.
- Per la crittazione è stato usato un sistema misto costituito sia da moltiplicazioni per quantità fisse (ottenute dalla proiezione delle coordinate RGB risultate dagli esperimenti) sia da somme sui pixel di altre immagini, cioè i *quadri maledetti* RGB1, RGB2, RGB3 e RGB4 contenuti nell'archivio.

Ecco i punti essenziali necessari per realizzare il programma di decrittazione…

- (*Funzione di proiezione dallo spazio RGB allo spazio grayscale*)
  *proj[{x_,y_,z_}]:=N[(0.2989*x+0.5870*y+0.1140*z)];*

- (*Coordinate RGB normalizzate ottenute dagli esperimenti, da inserire nel programma*)
  RGB1={n_1,n_2,n_3}/255;
  RGB2={n_1,n_2,n_3}/255;
  RGB3={n_1,n_2,n_3}/255;
  RGB4={n_1,n_2,n_3}/255;

- (*Otteniamo i coefficienti moltiplicativi, costanti, nello spazio *grayscale**)
  c1=proj[RGB1];
  c2=proj[RGB2];
  c3=proj[RGB3];
  c4=proj[RGB4];

- (*Importiamo l'immagine della mappa criptata*)
  criptomap=ImageResize[Import["C:\\Users\\Ale\\Desktop\\criptoRGB\\criptum.jpg"],{1200,800}];

- (*Importiamo le immagini usate nell'operazione di crittazione*)
  fig1=ImageResize[Import["C:\\Users\\Ale\\Desktop\\criptoRGB\\RGB1.jpg"],{1200,800}];
  fig2=ImageResize[Import["C:\\Users\\Ale\\Desktop\\criptoRGB\\RGB2.jpg"],{1200,800}];
  fig3=ImageResize[Import["C:\\Users\\Ale\\Desktop\\criptoRGB\\RGB3.jpg"],{1200,800}];
  fig4=ImageResize[Import["C:\\Users\\Ale\\Desktop\\criptoRGB\\RGB4.jpg"],{1200,800}];

- (*Estraiamo i dati numerici dai pixel, dato che le immagini di partenza sono a colori otterremo delle *matrici di dimensione 1200x 800 di coordinate {R,G,B}**)
  datcmap=ImageData[criptomap];
  datfig1=ImageData[fig1];
  datfig2=ImageData[fig2];
  datfig3=ImageData[fig3];
  datfig4=ImageData[fig4];

- Iniziamo ora a decrittare la mappa. L'operazione viene svolta in 4 passi successivi, in senso opposto a quello che è stato fatto per la crittazione. Come si è visto, le operazioni sono fatte sui singoli pixel, intesi come tabelle di numeri reali compresi tra 0 e 1. Osserviamo che con il comando *Table[]* definiamo proprio l'insieme dei pixel, indicati dagli indici i (indice di colonna, tra 1 e 800) e j (indice di riga tra 1 e 1200).
- Il singolo elemento di una tabella in Mathematica è indicato dalle doppie parentesi quadrate *[[]]*.
  Ad esempio, datcmap[[i,j]]indica il numero (in grayscale) corrispondente al pixel {i,j} dell'immagine criptum.jpg.

- Osserviamo anche l'operazione di modulo del risultato ottenuto:
  *Mod[datcmap[[i,j]]-7*c1*proj[datfig1[[i,j]]],1]*
  cioè Mod[x,1], dove in questo caso x= *datcmap[[i,j]]-7*c1*proj[datfig1[[i,j]]]*

- Va precisato che in Mathematica possiamo annidare 2 o più operazioni nella stessa riga, risparmiando spazio di scrittura. Le operazioni sono intese dall'interno verso l'esterno, cioè in questo caso, prima la proiezione dei pixel nello spazio *grayscale* (*proj[datfig1[[i,j]]]*, poi la sottrazione tra i 2 pixel {i,j} relativi alle figure *criptum.jpg* e *RGB1.jpg*, dove il pixel relativo a *RGB1.jpg* viene moltiplicato per il coefficiente *7*c1* e infine il modulo.
- La stessa operazione viene ripetuta nei passi successivi, mostrando in output volta per volta i risultati intermedi grazie ad una variabile di appoggio *temp* che richiama volta per volta l'immagine, andando a sostituire la precedente.

```
(*Step 1*)
decrip1=Table[Mod[datcmap[[i,j]]-7*c1*proj[datfig1[[i,j]]],1],{i,1,800},{j,1,1200}];
temp = PrintTemporary[Show[Image[decrip1]]];
Pause[3];
(*Step 2*)
decrip2=Table[Mod[decrip1[[i,j]]-7*c2*proj[datfig2[[i,j]]],1],{i,1,800},{j,1,1200}];
temp = PrintTemporary[Show[Image[decrip2]]];
Pause[3];
(*Step 3*)
decrip3=Table[Mod[decrip2[[i,j]]-7*c3*proj[datfig3[[i,j]]],1],{i,1,800},{j,1,1200}];
temp = PrintTemporary[Show[Image[decrip3]]];
Pause[3];
(*Step 4*)
decrip4=Table[Mod[decrip3[[i,j]]-7*c4*proj[datfig4[[i,j]]],1],{i,1,800},{j,1,1200}];
temp = PrintTemporary[Show[Image[decrip4]]];
Pause[3];
```

- La decrittazione di immagini così complesse produce risultati non sempre ottimali, cioè le immagini possono risultare in alcuni casi leggermente confuse, a causa dell'operazione di *Modulo* che discretizza in modo brusco lo spazio colore (come quando si legge l'orologio: tra le 10.59 e le 11.01 sono passati 2 soli minuti, ma sulla lancetta dei minuti abbiamo un salto di 58 minuti!). Dato che lo scopo deve essere quello di leggere un punto su una mappa, non ci interessa il colore finale dell'immagine decrittata, possiamo renderla in bianco e nero (*Binarize*) e utilizzare preliminarmente un filtro che tende a mediare il contenuto di pixel adiacenti (*StandardDeviationFilter*). Questo dovrebbe rendere più leggibile la mappa.

  ```
  Binarize[StandardDeviationFilter[Image[Table[decrip4[[i,j]],{i,1,800},{j,1,1200}],ColorSpace -> "Grayscale"],2]]
  ```

Conclusione

- Nell'immagine in chiaro della mappa è indicata la posizione dove è stato nascosto *il Perseo*, il misterioso quadro…
- CORRETE A CERCARLO!!!!
- Una volta trovato dovrete scansionarlo con l'app *Layar* per visualizzare una *frase nascosta*..[1]

[1] Se con l'app non riuscite a visualizzare ì dettagli nascosti nell'immagine in realtà aumentata (sono molto importanti le condizioni di luminosità..), cercate informazioni scritte sulla tela….

- Infine inviate una foto del quadro e la *frase nascosta* per e-mail al seguente indirizzo: plsfisica@liceorecanati.org
- Buona caccia e…VINCA IL MIGLIORE!!!