

Esercizio Gatto di Arnold: invertire la
trasformazione della mappa di Arnold date due
immagini ottenute dopo k e $k + 1$ passi
Corso di LSMC, a.a. 2019-2020

Cristian Soppio
559597

28 gennaio 2022

1 Descrizione del problema

Date le due immagini 4, ottenute ripetendo una trasformazione con 12 e 13 passi, vogliamo recuperare l'immagine originaria.

2 Descrizione della sperimentazione

Per risolvere il problema abbiamo cercato nella prima immagine dei pixel unici, ovvero con una tonalità di colore singolare, questo l'abbiamo fatto tramite lo script 1 e la function `numero`. Infine ricavando con il comando `find(H)` abbiamo trovato i pixel unici e con la function `trova` le loro immagini tramite la trasformazione, nella seconda immagine. Infine con un semplice conto di algebra lineare, scegliendo due vettori linearmente indipendenti, e grazie alle funzioni `arnold2k` e `trovaesponente` abbiamo ricavato l'immagine originale.

3 Script e function

Si riportano di seguito le function utilizzate nella sperimentazione.

Function 1

```
function t=numero(i,j)
A=imread('img_4_12.png');
%p=zeros(1,3);
p(1)=A(i,j,1);
p(2)=A(i,j,2);
p(3)=A(i,j,3);
```

```

t=0;
for k=1:1024
    for h=1:1024
        if p(1)==A(k,h,1)&& p(2)==A(k,h,2)&& p(3)==A(k,h,3)
            t=t+1;
        end
    end
end
end

```

Function 2

```

function [e,f]=trova(i,j)
A=imread('img_4_12.png');
B=imread('img_4_13.png');
for h=1:1024
    for k=1:1024
        if A(i,j,:)==B(h,k,:)
            e=h;
            f=k;
        end
    end
end
end
end

```

Function 3

```

function B = arnold2k(A, k, M)
%ARNOLD2 Applica k passi della mappa di Arnold.
%
% Questa funziona applica K passi della mappa del gatto di Arnold
% all'immagine rappresentata dalla matrice A. Se viene specificato
% anche il parametro opzionale M (una matrice 2x2), questo viene
% utilizzato al posto di [2 1 ; 1 1] per definire la mappa.
m = size(A, 1);
if ~exist('M', 'var')
    M = [ 2 1 ; 1 1 ];
end
N = eye(2);
for j = 1 : k
    N = mod(M * N, m);
end
% Costruiamo una matrice 2 x m^2 che contiene tutti gli indici (i,j)
II = zeros(2, m^2);
II(2, :) = kron(1:m, ones(1,m));
II(1, :) = kron(ones(1,m), 1:m);

```

```

IIh = 1 + mod(N * (II - 1), m);
% Costruiamo un vettore 1 x m^2 che corrisponde agli indici in IIh
Ih = [ 1 , m ] * (IIh - [ 0 ; 1 ] * ones(1, m^2));
B = A;
for j = 1 : size(B, 3)
    %for i1 = 1 : m
    %    for i2 = 1 : m
    %        B(Ih(i1), Ih(i2), j) = A(i1,i2,j);
    %    end
    %end
    B(Ih + (j-1)*m^2) = reshape(A(:, :, j), m*m, 1);
end
end

```

Function 4

```

function m = trova_esponente(M, m)
%PERIODO Determina l'ordine di M modulo n.
% M matrice simmetrica 2x2, ne possiamo avere al massimo n^3
N = eye(2, 2);
for i = 1 : m^3
    N = mod(M * N, m);

    if all( N - eye(2) == 0, 'all' )
        m = i;
        return;
    end
end
error('Non esiste m tale che M^m = I');

end

```

Script 1

```

A=imread('img_4_12.png');
B=imread('img_4_13.png');
H=zeros(1024,1024);
for i=1:1024
    for j=1024
        t=numero(i,j);
        if t==1
            H(i,j)=t;
        end
    end
end
end

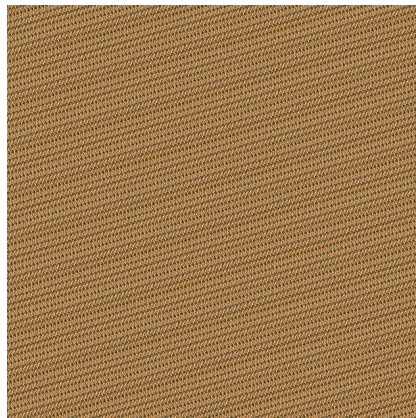
```

Script 2

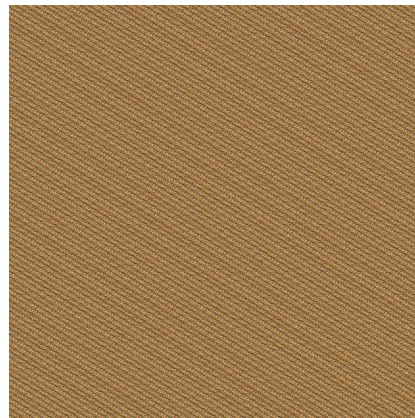
```
M=[6,1;11,2];
```

```
E=arnold2k(A,trova_esponente(M,1024)-12,M);  
imagesc(E);
```

4 Immagini



(a) immagine dopo 12 passi



(b) immagine dopo 13 passi



(c) immagine originaria