ASUPRA UNEI TRANSFORMĂRI ITERATIVE PERIODICE

fiz. Radu Ionicioiu

Institutul de Cercetări în Informatică

Rezumat: Articolul își propune să arate o metodă simplă de codificare a imaginilor grafice și prelucrarea ulterioară în vederea reconstrucției imaginii inițiale. Datorită simplității sale, metoda poate fi folosită cu succes în vederea asigurării confidențialității transmiterii de imagini în domeniul civil (de exemplu, în context economic).

Cuvinte cheie: prelucrare de imagini, algoritmi de codificare

Formularea problemei

Fie dreptunghiul $M=[0,x_0]*[0,y_0]$. Considerăm două transformări $S\theta$ si T, definite de ecuațiile:

$$S\theta, T : M \to M$$

1) $T(x,y)=(x',y')$ unde $x'=x_0/y_0*y$
$y'=y_0/x_0*x$

T este, deci, o transpunere și o scalare. Se observă imediat că $T^2=I$ (transformarea identică pe M).

2) $$S\theta\,(x,y)=(x',y') \text{ cu } x'=f(x,y)+x_0*\begin{cases} 1 & \text{dacă } f(x,y) < 0 \\ 0 & 0 \le f(x,y) \le x_0 \\ -1 & f(x,y) > x_0 \end{cases}$$

$$y'=y$$

unde $f(x,y)=x+(y-y_0/2)*tg\theta$

și $-\theta_{max} \le \theta \le \theta_{max}$, unde avem $tg(\theta_{max})=2*x_0/y_0$

$S\theta$ poate fi descrisă ca o forfecare de unghi θ, insoțită de tăierea părților care ies din cadru și 'lipirea' lor pe laturile opuse (figura 1).

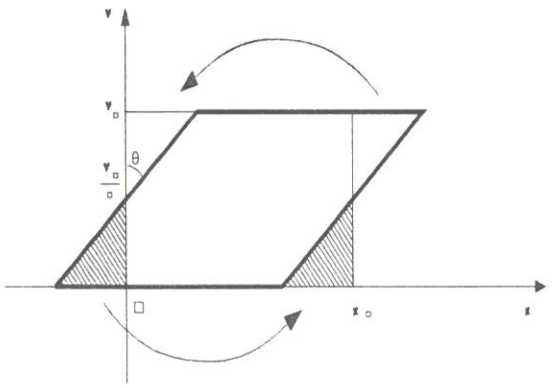

Figura 1.-Transformarea Sθ

În plus, se poate demonstra că avem și $S\theta \circ S\theta = I$, deci, atît T, cît și $S\theta$ sînt transformări punctuale și inversabile.

Efectele celor două transformări asupra unei imagini se pot vedea în figura 2.

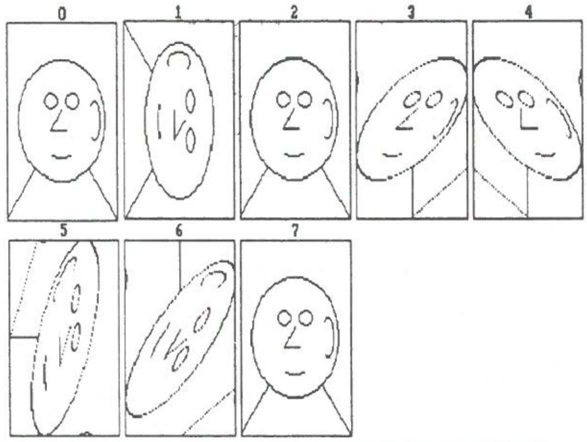

Figura 2.- Efectul diferitelor transformări asupra unei imagini

0.- imaginea inițială	4.- $S\theta$
1.- transformarea T	5.- $U=T\circ S\theta$
2.- $T^2=I$	6.- $S\theta \circ T$
3.- $S\theta$	7.- $S\theta \circ S\theta = I$

În practică se pleacă de la o imagine care se discretizează și se introduce într-o matrice m x n, fiecare element conținînd culoarea pixelului respectiv din plan (analog unei imagini bitmap).

Prin aplicarea iterativă a transformării $U=T\circ S\theta$ pe imaginea inițială, aceasta se deformează progresiv, pînă cînd întreaga imagine devine 'omogenă', orice informație pierzîndu-se aparent.

Apare însă un fenomen extrem de interesant. Dacă se continuă procesul iterativ, aplicîndu-se în continuare **aceeași** transformare U, după un număr n de iterații, imaginea inițială se reface, **fără nici o pierdere de informație** (figura 3).

Altfel spus, există un număr natural n, astfel încît avem

$$(T\circ S\theta)^n=I$$

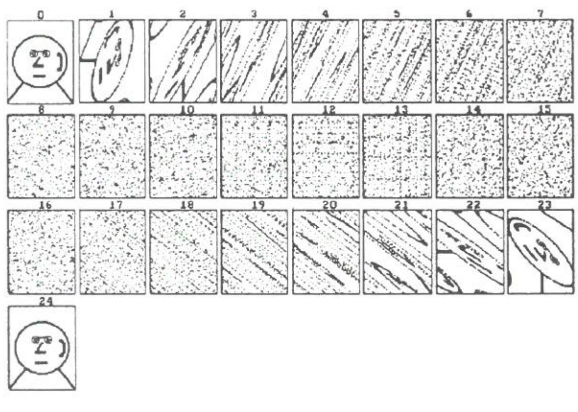

Figura 3.- Refacerea imaginii inițiale după aplicarea succesivă a transformării T∘Sθ

- dimensiunea imaginii: 72x72 pixeli
- $\theta=45°$; imaginea se reface după n=24 de iterații, $(T\circ S\theta)^{24}=I$

Un efect neaşteptat este faptul că n (deci numărul de iteraţii după care se reface imaginea iniţială) depinde puternic şi, în acelaşi timp, foarte neregulat, de **dimensiunile** imaginii iniţiale (păstrînd unghiul de forfecare θ constant). În tabelul alăturat se poate observa aceasta dependentă pentru cazul unei imagini pătrate (NxN pixeli) şi a unghiului $\theta=45^{\circ}$.

Dimensiunea (pixeli)	42x42 pixeli	43	44	45	46	47	48
n (nr de iteraţii)	48	88	30	120	48	32	24

Se poate arăta uşor că n nu depinde de ordinea transformărilor (deşi Sθ şi T nu comută, $S\theta \circ T \neq T \circ S\theta$), deci avem

$(S\theta\ T)^n=I <=> (T\ S\theta)^n=I$

Refacerea imaginii iniţiale dintr o imagine codificată ('omogenă') se poate face în două moduri. Primul constă în aplicarea, în continuare, a transformării $U=T\circ S\theta$ pînă cînd se ajunge la iteraţia n pentru care avem $(T\circ S\theta)^n=I$ (metoda descrisă anterior). Al doilea mod constă, evident, în aplicarea inversei $U^{-1}=S\theta\circ T$ de un număr de ori egal cu cel al iteraţiei după care a rezultat imaginea codificată. Cel de-al doilea algoritm se poate aplica în cazul unei imagini care necesită un n mare pentru refacere (de exemplu, pentru o imagine de 70x70 pixeli, numărul de iteraţii este n=240).

Un alt efect interesant, care apare însă numai pentru anumite valori x_0, y_0 şi θ, este cel de 'dedublare' a imaginii. Această dedublare -în unele cazuri chiar multiplicare – apare la o iteraţie care este un multiplu al unui divizor al lui n. Tot în acest context, pentru valori particulare x_0, y_0 şi θ la iteraţia n/2 (n este întotdeauna par, deoarece pentru a reveni la imaginea iniţială după n iteraţii, trebuie să avem un număr par de transformări T, avînd în vedere că $T^2=I$) poate să apară o inversie a imaginii iniţiale faţă de centrul acesteia.

În practică se preferă imagini pentru care $x_0/y_0 \not\approx 1$, deci quasi-patratice, aceasta pentru evitarea deformării imaginii în urma aplicării transformării T. Pentru imagini avînd raportul x_0/y_0 mult diferit de unitate, apare un efect de 'pixelation' (imaginea pare formată din blocuri).

Algoritm

Prezentăm în continuare o scurtă descriere a algoritmului care realizează aceste transformări.

```
PROGRAM
 BEGIN
   Iniţializare(nx,ny,θ);
   Tablou(nx,ny);
   repeat
     begin
       Shear(θ);
       XtoY;
       Grafic;
     end;
   until (condiţie);
 END.
```

Procedura **Iniţalizare** setează dimensiunea matricii (tabloului) ce va fi procesat şi ungiul θ. **Tablou** desenează imaginea (sau citeşte de pe disk un fişier bitmap) şi iniţializează matricea cu imaginea dorită. **Shear(θ)** şi **XtoY** realizează cele două transformări (Sθ, respectiv T) asupra imaginii stocate în matrice, iar **Grafic** desenează imaginea. Bucla se repetă pînă cînd este îndeplinită **condiţie**.

Concluzii

Metoda prezentată este simplă, putînd fi usor implementată în orice limbaj [2]. Se observă de asemenea că algoritmul nu depinde de tipul imaginii (alb–negru sau color), viteza de prelucrare fiind aceeaşi în ambele cazuri.

Datorită avantajelor arătate, considerăm că metoda prezentată poate constitui nucleul unui viitor sistem de codare/decodare a imaginilor grafice.

Bibliografie

1. **CRUTCHFIELD, J.**: Chaos, Sci.Am., Dec. 1986, p.38.
2. **IONICIOIU, R.**: Rutina pentru transformări iterative periodice-implementare în Turbo Pascal 6.0, TR ICI/1993.
3. **WEISKAMP, K., ş.a.**: Power Graphics Using Turbo Pascal, J. Wiley & Sons, New York,1989.