TP2 : Codage d'un contour

Benjamin Van Ryseghem
François Lepan

29 janvier 2013

1 Codage d'un contour

1.1 Cont

Les points du contour sont stockés dans la variable sous forme d'une liste de nombres complexes.

1.2 En ne conservant qu'un point sur 4, puis un point sur 8, constituer deux autres contours approchant la forme circulaire et les afficher respectivement en bleu et en vert dans la même fenêtre

Voici le code correspondant :

```
nom1 = "cercle-80.txt";
cont1 = rdfChargeFichierContour (nom1);

cont2 = cont1(1:4:$);
cont3 = cont1(1:8:$);
rdfAfficheContour(cont1, 2, "r");
rdfAfficheContour(cont2, 2, "g");
rdfAfficheContour(cont3, 2, "b");
```

2 Descripteur de Fourier

2.1 Pourquoi la fonction rdfDescFourier élimine t'elle parfois un point de la liste des points du contour ?

Pour plus de performance dans le calcule des descripteurs de Fourier

2.2 Vérifier que les descripteurs de Fourier permettent de reconstituer le contour initial

Voici le code correspondant :

```
descFour = rdfDescFourier(cont);
rdfAfficheContour (rdfInverseDescFourier(descFour), 1, "k");
```

2.3 Quel est l'indice de tableau correspondant au descripteur Z0 ?

Il se situe à la moitié de la taille du tableau.

2.4 Expliquer l'utilité de la fonction rdfValeurDescFourier

Cette fonction sert à afficher la valeur d'un point de la liste des descripteurs de Fourier

2.5 À quoi correspond le descripteur de Fourier Z_0 d'une forme décrite par son contour ?

C'est le barycentre de la forme.

On observe que si on met une grande valeur les courbes on tendance a se rapprocher du centre. C'est un peut comme si on augmentait la gravité au centre de la forme.

```
descFour = rdfDescFourier(cont)
descFour((size(descFour,1)/2)) = VALUE;
```

3 Filtrage des descripteurs de Fourier

Voici la fonction rdfAnnuleDescFourier :

```
function vector = rdfAnnuleDescFourier(desc,ratio)
    size = size(desc,1);

    if ratio == 1 then
        vector = desc
    else
        if ratio == 0 then
            vector = zeros(1,size)
        else
            index = int(ratio*size)

            if ratio <= 0.5 then
                s = size/2;
                s1 = size/2 + 1;
                s2 = size/2 + index;
                s3 = size - size/2 - index - 1;
                vector = [ zeros(1,s)  desc(s1:s2)'  zeros(1,s3)]'
            else
                s = index - size/2;
                s1 = size - index;
                s2 = size/2 + 1;
                vector = [ desc(1:s)' zeros(1,s1) desc(s2:$)' ]'
            end
         end
    end
endfunction
```

Lorsque nous réduisons le ratio il se trouve que en effet la forme se simplifie (*c.f.* Fig 1)

FIGURE 1 – Annulation des descripteurs de Fourrier

4 Réduction d'une chaîne de contour

Voici le code que nous avons mis pour rdfAlgorithmeCorde :

```
if real (debut) == real (fin)  then
    d = abs(real(cont) - real(debut))
else
    a = (imag(fin) - imag(debut))/(real(fin) - real(debut))
    b = imag(fin) - a * real(fin)

    if a == 0 then
        d = abs(imag(cont) - b)
    else
        d = abs(a*real(cont) - imag(cont) + b) ./ (sqrt(1+a^2))
    end
end
```

Et on observe bien que si on augmente `dmax` on réduit bien le contour (*c.f.* Fig 2)

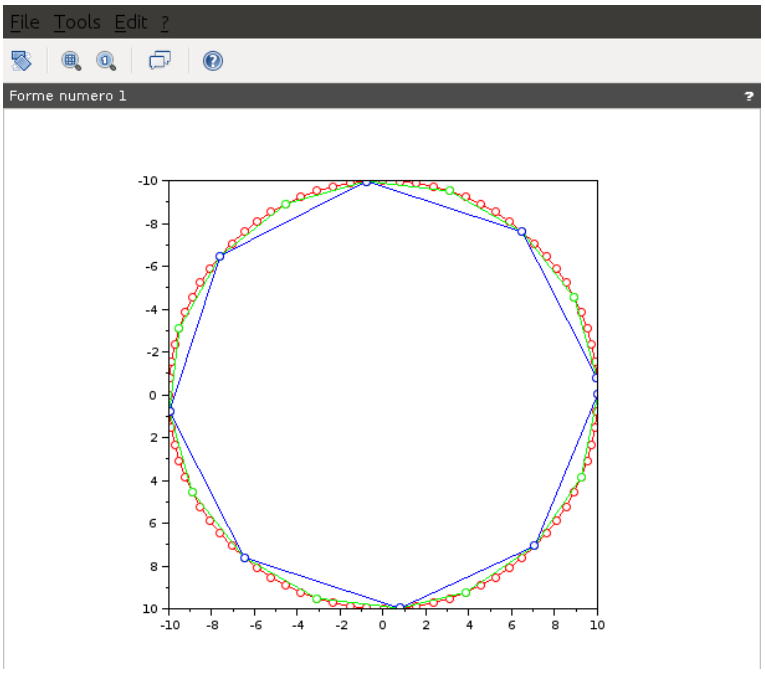

FIGURE 2 – Réduction d'une chaine de contours. rouge : contours initial _ vert : Cordes avec dmax=0.5 _ bleu Cordes avec dmax=1

5 Comparaison des deux approches

Après avoir fait quelques tests nous nous sommes rendu compte que pour des formes connexes (triangles, carres, cercles) l'algorithme des cordes est mieux car beaucoup plus rapide que celui des descripteurs de Fourier. Cependant, dans les deux cas le l'approximation est correcte (*c.f.* Fig 3).

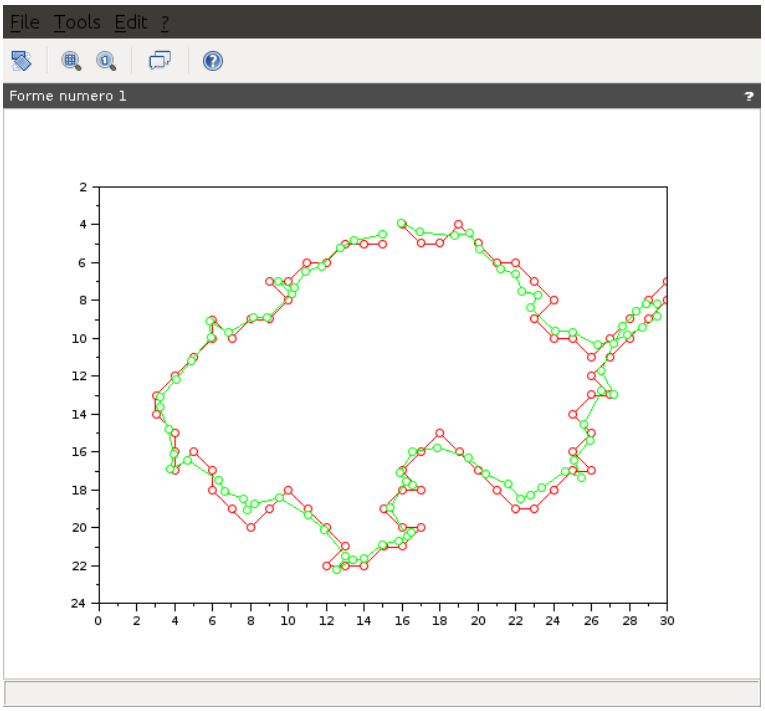

FIGURE 3 – rouge : contours initial _ vert : Fourier avec ratio=0.6

Mais pour ce qui est des formes non connexes (patatoïde) l'algorithme de Fourier est bien plus précis. On observe sur la figure Fig 4 les deux algorithmes et on voit bien que l'algorithme des cordes fait des segment qui sont en dehors de la figure.

En conclusion nous dirons que l'algorithme des cordes n'est pas efficace pour faire de la reconnaissance de formes complexes comme un visage ou une patate (images réelles) contrairement à l'algorithme de Fourier. Par contre il est beaucoup plus rapide pour des formes géométriques (images générées) comme un triangle ou un carré.

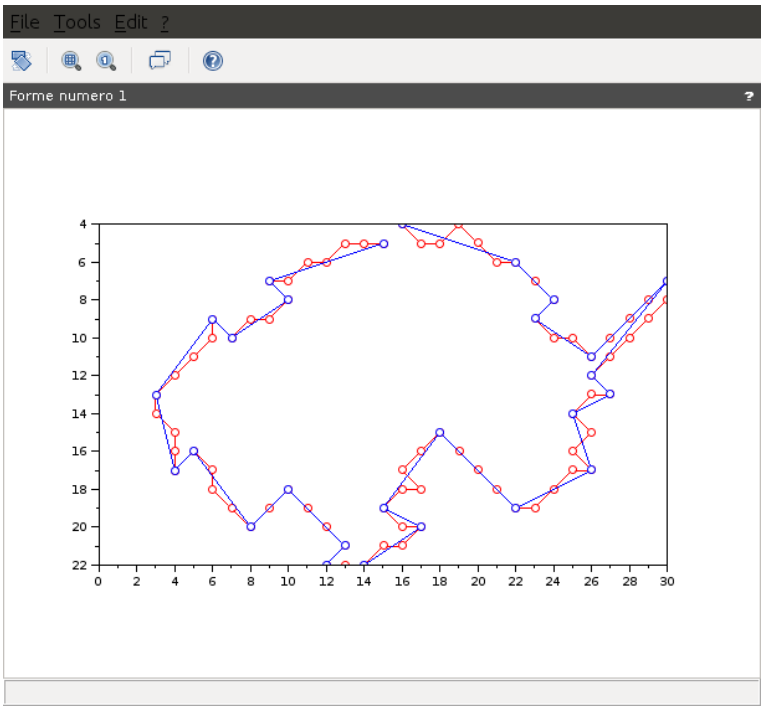

FIGURE 4 – rouge : contours initial _ bleu : Cordes avec dmax=1