

TP2 : Codage d'un contour

Benjamin VAN RYSEGHEM
François LEPAN

4 février 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");
```

L'exécution du code précédent fournit la Fig. 1, sur laquelle apparaissent trois contours. Il y a d'abord le contour initial d'une forme circulaire représentée en rouge.

Il s'y trouve également le contour vert résultant du codage du contour original en ne conservant qu'un point sur quatre.

Enfin, il y a le contour bleu représentant le codage du contour original en ne gardant cette fois qu'un point sur huit.

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 calcul des descripteurs de Fourier

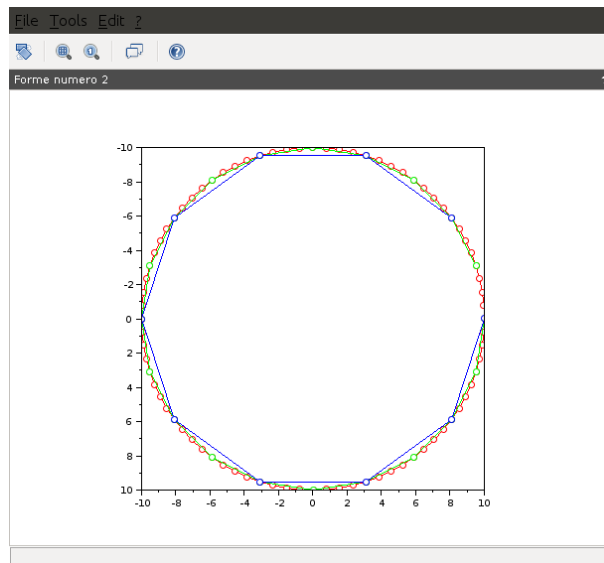


FIGURE 1 – rouge : contours initial — vert : 1 sur 4 — bleu : 1 sur 8

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 Z_0 ?

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 change sa valeur on effectue une translation.

```
nom1 = "cercle-80.txt";
cont1 = rdfChargeFichierContour (nom1);
descFour = rdfDescFourier(cont1)
descFour1 = rdfDescFourier(cont1)
descFour1(1) = 5;
```

```

rdfAfficheContour(rdfInverseDescFourier(descFour), 2, "r");
rdfAfficheContour(rdfInverseDescFourier(descFour1), 2, "g");

```

Le résultat de cette exécution fournit la Fig. 2, on y retrouve les deux contours du cercle. Le premier en rouge est la figure initial et le deuxième contour celui du même cercle mais translaté de 5.

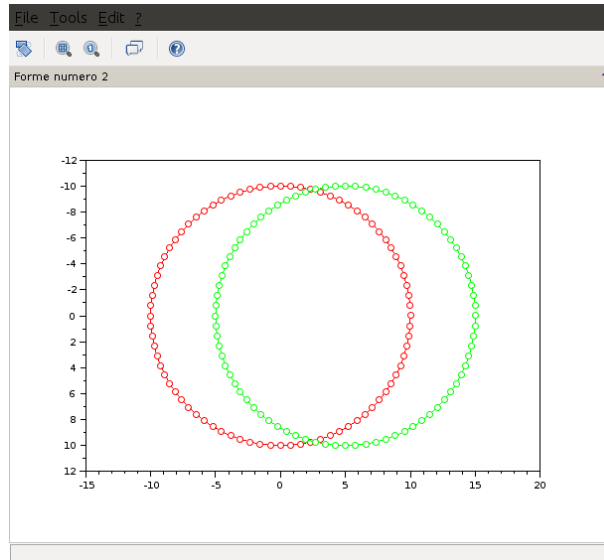


FIGURE 2 – rouge : contours initial — vert : 1 sur 4 — bleu : 1 sur 8

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)
            vector = [desc(1:index)' zeros(1, size-index) ]'
        end
    end
end

endfunction

```

Lorsque nous réduisons le ratio il se trouve qu'en effet la forme se simplifie (*c.f.* Fig. 3)

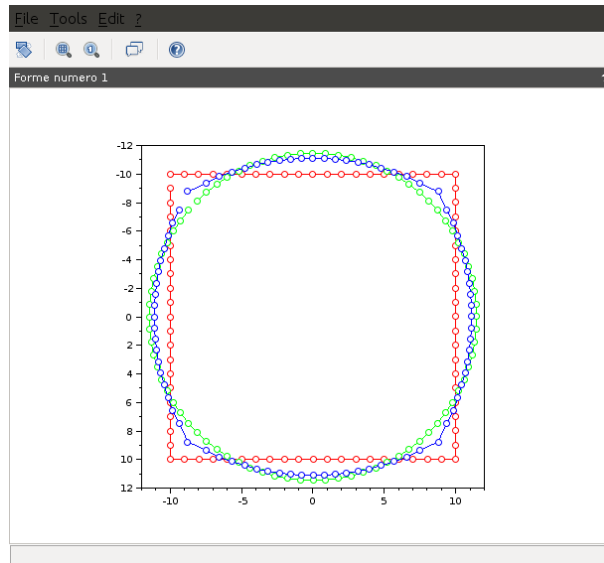


FIGURE 3 – Annulation des descripteurs de Fourier : rouge : contour initial, bleu : ratio = 0.7, vert : ratio = 0.05

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
end

```

Et on observe bien que si on augmente `dmax` on réduit bien le contour (*c.f.* Fig. 4)

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. 5).

Mais pour ce qui est des formes non connexes (patatoïde) l'algorithme de Fourier donne un meilleur aperçu de la forme d'origine. On observe sur la figure Fig. 5 que l'algorithme des cordes fait des segment qui sont moins représentatif de la forme de base.

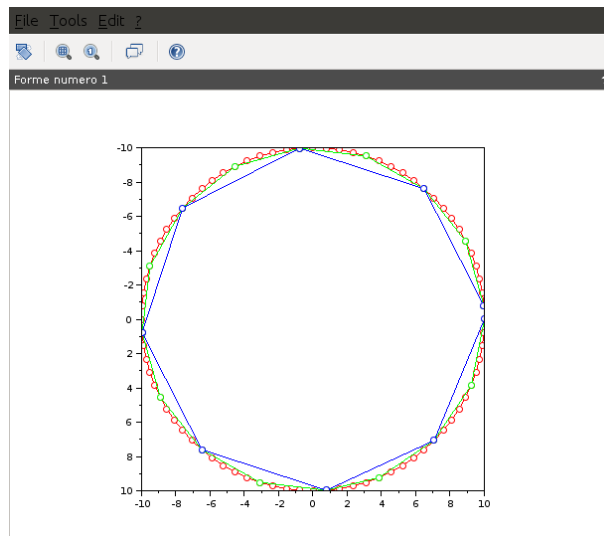


FIGURE 4 – Réduction d'une chaîne de contours. rouge : contours initial — vert : Cordes avec $d_{\max}=0.5$ — bleu Cordes avec $d_{\max}=1$

En conclusion nous dirons que l'algorithme des cordes est moins 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.

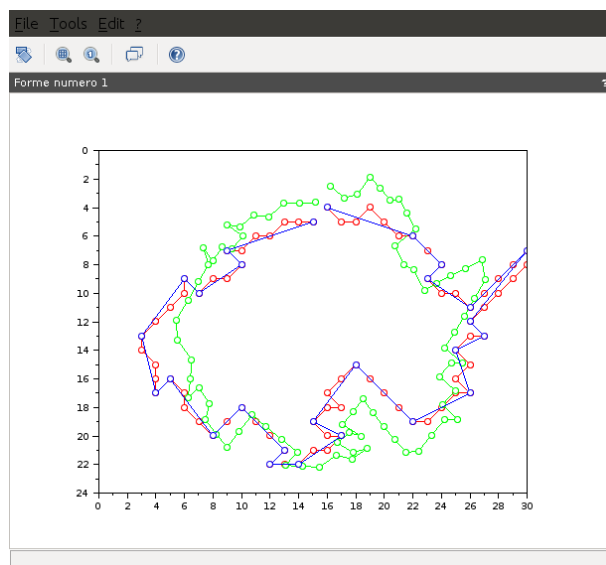


FIGURE 5 – rouge : contours initial — vert : Fourier avec ratio=0.95 — bleu : Cordes avec $d_{max}=1$