Extraction de contours

1. Obtention d'une image binaire

L'extraction de contours opère sur une image binaire, c'est-à-dire constituée de 0 et de 1. Les zones connexes formées de 1 possèdent un ou plusieurs contours.

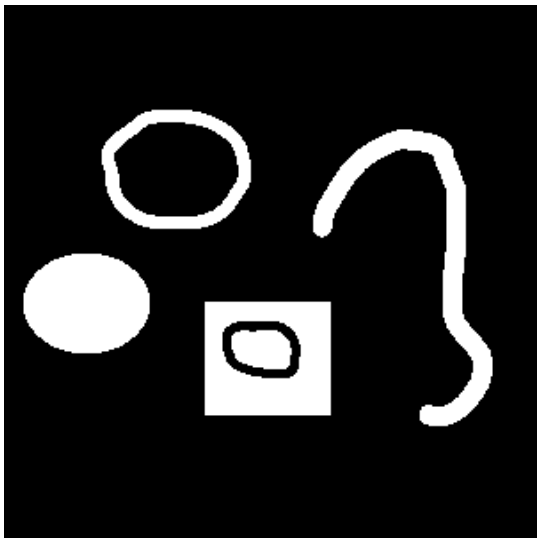
On commence par lire une image en niveaux de gris :

```
Needs['CV`'];
CVOpen[];


img=CVLoadImage['formes1.png',CVLoadImageGrayScale];
s=CVGetSize[img];
```

Si l'on souhaite traiter les informations de couleurs, par exemple isoler des formes d'une certaine couleur, il faut le faire avant l'extraction de contours.

L'image binaire est obtenue en effectuant un seuillage :

```
bin=CVCreateImage[s,IplDepth8U,1];
CVThreshold[img,bin,200.0,255.0,CVThreshBinaryInv];


Image[CVGetImageArray[bin,1,1]]
```



2. Extraction des contours externes

Il faut tout d'abord créer un espace mémoire qui sera utilisé pour stocker les contours :

```
mem = CVCreateMemStorage[0];
```

Les contours externes sont obtenus par

```
cont1=CVFindContours[bin,mem,CVRetrExternal,CVChainApproxNone,{0,0}];
```

Les contours sont fournis sous forme d'une liste chaînée. La liste des points du premier contour est obtenue par :

```
c1=CVGetContour[cont1];
```

Le périmètre d'un contour est obtenu par :

```
p=CVContourPerimeter[cont1]
```

```
266.
```

le contour suivant est obtenu par :

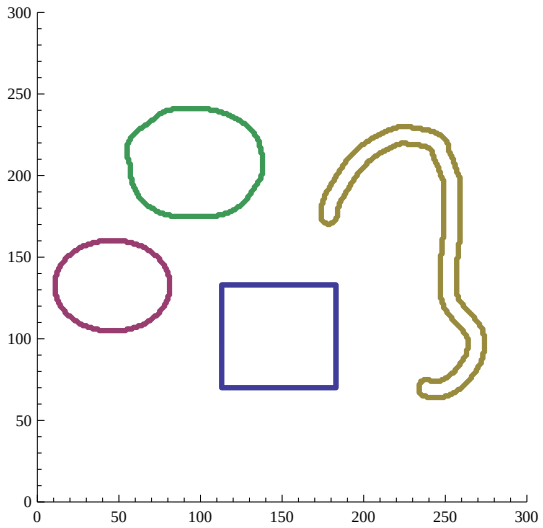
```
cont2=CVGetSeqHNext[cont1];
c2=CVGetContour[cont2];
```

Lorsque le dernier contour de la liste chaînée est atteint, la fonction `CVGetSeqHNext` renvoit 0. La liste des contours est obtenue de la manière suivante :

```
liste={};
cont=cont1;
While[cont!=0,
        c=CVGetContour[cont];
        c=CVFlipY[s,c];
        liste=Append[liste,c];
        cont = CVGetSeqHNext[cont];
      ];
```

Les contours ont été modifiés avec la fonction `CVFlipY` de manière à se superposer à l'image originale (origine en haut à gauche).

```
ListPlot[liste,PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```

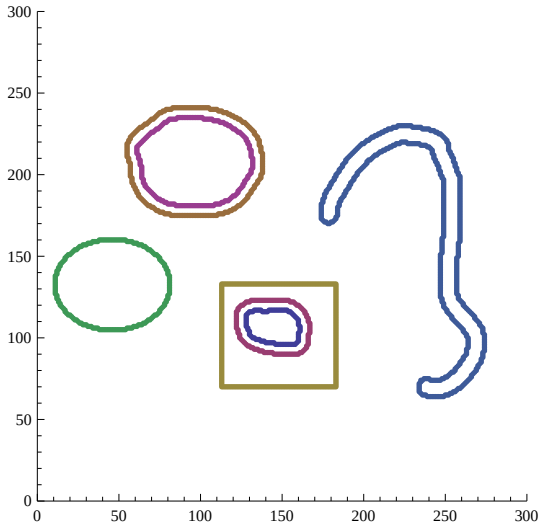


3. Extraction de tous les contours

3.a. Liste de contours

Les contours peuvent être obtenus sous forme d'une liste chaînée :

```
cont=CVFindContours[bin,mem,CVRetrList,CVChainApproxNone,{0,0}];

liste={}
While[cont!=0,
        c=CVFlipY[s,CVGetContour[cont]];
        liste=Append[liste,c];
        cont = CVGetSeqHNext[cont];
      ];


ListPlot[liste,PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



L'inconvénient de cette méthode est l'absence d'information sur le caractère interne ou externe de chaque contour.

3.b. Structure à deux niveaux

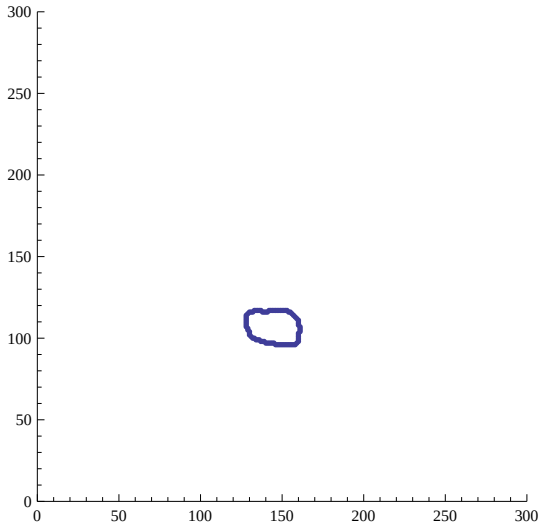
Une structure à deux niveaux est obtenue par :

```
cont=CVFindContours[bin,mem,CVRetrCcomp,CVChainApproxNone,{0,0}];
```

Le premier niveau contient les contours externes. Ce niveau est balayé de manière horizontal, au moyen de la fonction `CVGetSeqHNext`.

Considérons par exemple le premier contour :

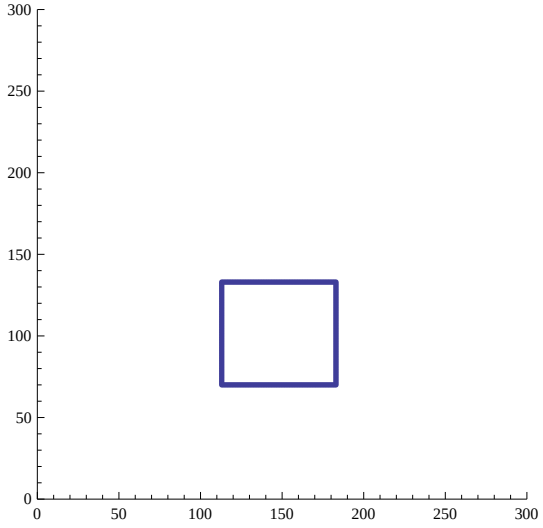
```
c=CVFlipY[s,CVGetContour[cont]];


ListPlot[c,PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



Il s'agit du contour d'une zone connexe (blanche) qui n'avait pas été détecté avec l'option `CVRetrExternal`.

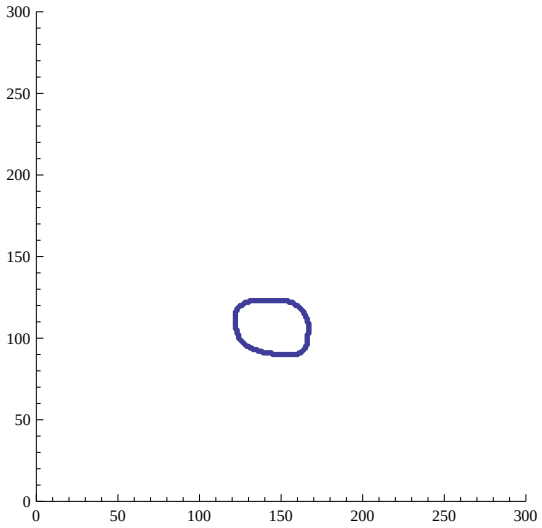
Voyons le 2ième contour :

```
cont=CVGetSeqHNext[cont];
c=CVFlipY[s,CVGetContour[cont]];


ListPlot[c,PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



Il s'agit du contour externe d'une zone connexe qui contient un trou noir (lequel contient la zone précédente). Le contour interne (le bord du trou) est obtenu en explorant le deuxième niveau. Celui-ci est accessible par un déplacement vertical :

```
vnext=CVGetSeqVNext[cont]
c=CVFlipY[s,CVGetContour[vnext]];


ListPlot[c,PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```

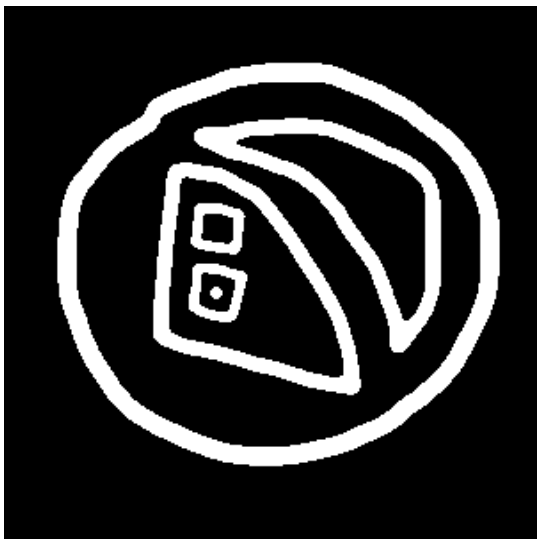


Si la zone connexe contient plusieurs trous, ceux sont accessibles par un déplacement horizontal sur le deuxième niveau.

3.c. Hiérarchie complète

Dans certains cas, l'emboîtement des contours est plus complexe. Considérons l'image suivante :

```
img=CVLoadImage['formes2.png',CVLoadImageGrayScale];
s=CVGetSize[img];
bin=CVCreateImage[s,IplDepth8U,1];
CVThreshold[img,bin,100.0,255.0,CVThreshBinary];


Image[CVGetImageArray[bin,1,1]]
```

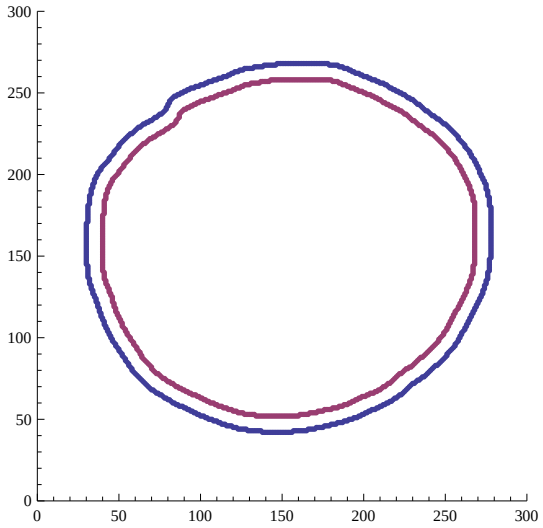


La hiérarchie complète des contours peut être obtenue sous forme d'un arbre par :

```
cont1=CVFindContours[bin,mem,CVRetrTree,CVChainApproxNone,{0,0}];
```

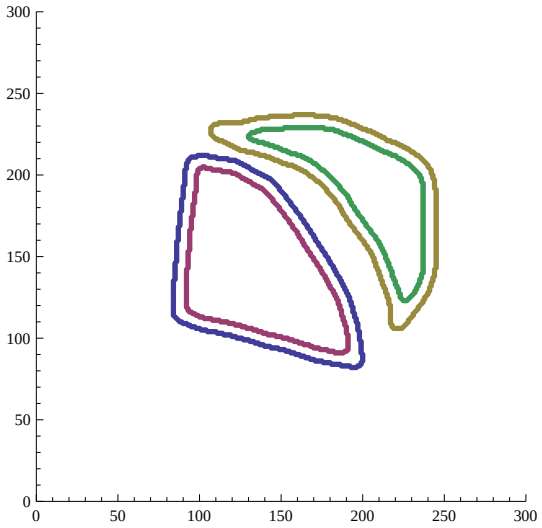
Considérons tout d'abord la racine de l'arbre et son noeud descendant direct, obtenu par déplacement vertical dans la séquence :

```
c1=CVFlipY[s,CVGetContour[cont1]];
cont2=CVGetSeqVNext[cont1];
c2=CVFlipY[s,CVGetContour[cont2]];


ListPlot[{c1,c2},PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



Le niveau suivant de l'arbre contient les contours des deux formes incluses dans la précédente :

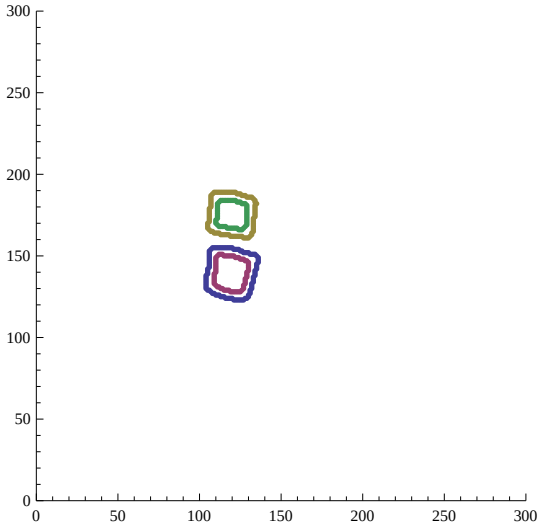
```
cont3=CVGetSeqVNext[cont2];
c3=CVFlipY[s,CVGetContour[cont3]];
cont4=CVGetSeqVNext[cont3];
c4=CVFlipY[s,CVGetContour[cont4]];
cont5=CVGetSeqHNext[cont3];
c5=CVFlipY[s,CVGetContour[cont5]];
cont6=CVGetSeqVNext[cont5];
c6=CVFlipY[s,CVGetContour[cont6]];


ListPlot[{c3,c4,c5,c6},PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



En explorant l'arbre verticalement à partir de `cont4`, on obtient les contours des deux petits carrés :

```
cont7=CVGetSeqVNext[cont4];
c7=CVFlipY[s,CVGetContour[cont7]];
cont8=CVGetSeqVNext[cont7];
c8=CVFlipY[s,CVGetContour[cont8]];
cont9=CVGetSeqHNext[cont7];
c9=CVFlipY[s,CVGetContour[cont9]];
cont10=CVGetSeqVNext[cont9];
c10=CVFlipY[s,CVGetContour[cont10]];
```

```
ListPlot[{c7,c8,c9,c10},PlotRange->{{0,s[[1]]},{0,s[[2]]}},AspectRatio->1]
```



4. Caractéristiques des contours

La fonction suivante permet d'obtenir le rectangle d'aire minimale contenant le contour. On l'applique ici a un des contours obtenus plus haut :

```
rect=CVMinAreaRect2[cont7]

{118.4705810546875, 160.61764526367188, 30.80202293395996, 31.044557571411133,
 75.96376037597656}
```

Les éléments de la liste obtenue sont : les coordonnées du centre du rectangle, sa largeur et sa hauteur, l'orientation (en degrés) par rapport à l'axe Y.

La fonction suivante permet d'obtenir l'ellipse qui s'ajuste le mieux aux points du contour :

```
ellipse=CVFitEllipse2[cont1];

{153.67434692382812, 144.31842041015625, 224.0941925048828, 249.94686889648438,
 80.38348388671875}

CVClose[];
```