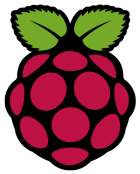
Raspberry Pi OpenCV Caractéristiques et descripteurs de forme



1 le type de forme des figures géométriques simples

La fonction findContours trouve beaucoup trop de points pour déterminer les segments de droite minimaux qui composent la figure. Nous devons donc utiliser une fonction **approxPolyDP** qui nous permettra de réduire le nombre de point utilisé pour définir le contour de la figure.

void **approxPolyDP**(InputArray **curve**, OutputArray **approxCurve**, double **epsilon**, bool **closed**)

- **curve** : Le vecteur 2D des points touvés par findContours
- **approxCurve** : Un vecteur de point qui recevra les points approximés, défini comme suit :
  vector <Point> approx;
- **epsilon** : Une valeur qui représente une précision pour effectuer l'approximation. De façon générale, une valeur de 1% à 3% du périmètre donne de bons résultats. Elle peut être calculée de la façon suivante : arcLength(Contours[i], true) * 0.02
- **closed** : Booléen qui indique si la figure est fermée ou non.

2 Exemple de programme

```
  findContours( image_CNY, contours, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE );
  cout << "Nombre d'objets trouvés :" << contours.size() << endl;
```



```
  for( int i = 0; i< contours.size(); i++ ) {
        approxPolyDP(contours[i], approximation, arcLength(contours[i], true) * precision, true);
    contours[i] = approximation;

    drawContours( drawing2, contours, i, color, 1, CV_AA, hierarchy, 0, Point() );
    imshow("approximation", drawing2);
    cout << "L'objet id : " << i << " est approximé par " << approximation.size() << " segments" << endl;
  }
```

Le nombre d'élément du vecteur « approximation » correspond à l'approximation du nombre de points qui ont été utilisés pour l'identification des segments. Or le nombre de points correspond au nombre de segments et donc au nombre de côté qui constitue la forme.

Un rectangle sera approximé à quatre segments. Un cercle sera approximé à un polygone de n cotés (n dépend de la précision demandée).

le programme complet sur github :
https://github.com/PhilippeSimier/openCV/tree/master/DetectionFormes

3 Caractéristiques (aire, périmètre, centre de gravité, rectangularité)

Périmètre : le périmètre de chaque objet peut être trouvé en utilisant la fonction « arcLength ».

```
Perimetre = arcLength(contours[i], true);
```

Aire : L'aire d'un objet est exprimée par le nombre de pixels appartenant à l'objet.

Aire = contourArea(Contours[i]);

Fonction: contourArea

- Contours – Le vecteur 2D des points qui sont contenu dans un vecteur (vector).
- closed – Booléen indiquant le sens de rotation (horaire ou anti-horaire). Par défaut, la valeur est à « false » et ceci correspond à la valeur absolue de l'aire.

Centre de gravité :

Le centre de gravité d'un objet correspond au point qui permettrait de faire tenir physiquement l'objet en question en équilibre.

Nous utiliserons la classe « moments » qui permet de calculer les moments d'ordre 3 sur des polygones.

Fonction: moments

- Contours – Le vecteur des points du contour de l'objet
- binaryImage – Booléen qui spécifie si l'image est une image binaire ou non

La fonction retourne les descripteurs de moments dans une structure qui contient les éléments recherchés.

Ainsi, la valeur m00 représente l'aire d'un objet et le centre de gravité est calculé comme suit :

x = m10 / m00 et y = m01 / m00

où les valeurs de x et de y représente le point milieu de l'objet.

Rectangularité

La rectangularité est une mesure qui nous permet de déterminer à quel point une forme est de type rectangulaire. Cette mesure permet d'obtenir une approximation de la forme rectangulaire d'un objet. On peut donc savoir si un objet est de type rectangulaire ou s'il s'en éloigne.

Fonction: RotatedRect **minAreaRect**(InputArray points) – Calcule le plus petit rectangle (avec rotation) qui englobe un objet.

- points – les points contenu dans un vecteur (vector).
- La fonction renvoie un objet de la classe RotatedRect

Fonction: Rect **boundingRect**(InputArray points) – Calcule le plus petit rectangle (sans rotation) qui englobe un objet

- points –les points contenu dans un vecteur (vector).
- La fonction renvoie un objet de la classe Rect

```
findContours(ImgResultat, Contours, CV_RETR_EXTERNAL ,
CV_CHAIN_APPROX_SIMPLE);
vector<Point> approx;
Mat ImgResultat2 = Mat::zeros( ImgResultat.size(), CV_8UC3 );


for( int i = 0; i< Contours.size(); i++ ){
   // Test de la rectangularité
   RotatedRect leRect = minAreaRect(Contours[i]);
   double Aire_leRect = leRect.size.width * leRect.size.height;
   double Aire_Rect_Contour = contourArea(Contours[i]);
   double Rectangularite = Aire_Rect_Contour / Aire_leRect;
   cout << "Rectangularite de l'objet  : " << i << " : " << Rectangularite << endl;
   cout << "Angle de la forme # " << i << ": " << leRect.angle;
}
```