Contour Detection

A contour is a closed curve joining all the continuous points having some colour or intensity, they represent the shapes of objects found in an image.

**OpenCV** provides a comprehensive set of functions for image processing and computer vision tasks, and contour detection is one of the functionalities it offers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Pros.** | **Cons.** | **Use Case** |
| findContours() | Widely used and versatile.  Can handle various types of contours. | May include noise, especially in complex images. | General-purpose contour detection.  Object recognition and tracking. |
| Canny Edge Detection + findContours() | Helps to preprocess images by highlighting edges. | Sensitive to parameter tuning. | Edge detection and contour extraction.  Useful to ID object boundaries. |
| Douglas-Peucker algorithm (approx.) | Reduces the number of points in a contour while preserving its shape. | Requires a tolerance parameter for approximation. | Shape simplification for efficient processing.  Gesture recognition. |
| Contours with moments | Provides information about the shape and distribution of the contour. | Sensitive to noise. | Object recognition based on moments.  Image analysis for shape characteristics. |
| Contours w/ convex hull | Describes the overall shape of a contour. | May overestimate the shape in concave areas. | Hand gesture recognition.  Defining a simplified shape around a set of points. |
| Contours w/ minimum enclosing circle | Provides a compact representation of the contour. | Sensitive to outliers. | Object tracking based on circular shape.  Anomaly detection in circular objects. |
| Contours w/ minimum enclosing ellipse | Provides a compact representation of the fitted ellipse.  Useful to approx. complex shapes with ellipses. | Sensitive to outliers in the input contour points.  May not be optimal for irregularly shaped objects. | Object tracking and recognition based on ellipses.  Shape analysis in computer vision applications. |
| Contours w/ minimum enclosing rectangle | Represents the minimum area rectangle that encloses the contour. | May not be optimal for irregularly shaped objects. | Object detection in images.  Character recognition/ text localization. |

**Other libraries**/packages offering contour detection capabilities.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Pros.** | **Cons.** | **Use Cases** |
| Scikit-Image/ measure | User-friendly interface.  Good for scientific/research applications. | May not have the same extensive set of functionalities as OpenCV. | Scientific/biomed. image analysis.  Feature extraction and measurement in images.  Contour finding and segmentation. |
| Mahotas | Efficient and fast, especially for large images. (C++)  Contains a variety of image processing functions. | Smaller community/support compared to OpenCV. | CV applications, especially where speed is crucial.  (Canny Edge Det.)  Features include filtering, morphology, and contour finding. |

Code structures:

1. findContours():

contours, \_ = cv2.findContours(image, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

1. Canny Edge Detection + findContours():

edges = cv2.Canny(image, threshold1, threshold2)

contours, \_ = cv2.findContours(edges, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

1. Contour approximation using Douglas-Peucker algorithm:

epsilon = 0.02 \* cv2.arcLength(contour, True)

approx = cv2.approxPolyDP(contour, epsilon, True)

1. Contours with moments:

contours, \_ = cv2.findContours(image, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

for contour in contours:

M = cv2.moments(contour)

1. Contours with convex hull:

hull = cv2.convexHull(contour)

1. Contours with minimum enclosing circle:

(x, y), radius = cv.minEnclosingCircle(contour)

center = (int(x),int(y))

radius = int(radius)

cv.circle(img, center, radius,(0,255,0),2)

1. Contours with minimum enclosing ellipse:

ellipse = cv.fitEllipse(contour)

cv.ellipse(img, ellipse, (0,255,0), 2)

1. Contours with minimum enclosing rectangle:

rect = cv2.minAreaRect(contour) [rotated]

Let (x,y) be the top-left coordinate of the rectangle and (w,h) be its width and height.

x,y,w,h = cv.boundingRect(cnt) [straight]

cv.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)

1. Scikit-Image:

from skimage import measure

contours = measure.find\_contours(image, level)

1. Mahotas:

import mahotas as mh

contours, \_ = mh.find\_contours(image, threshold)