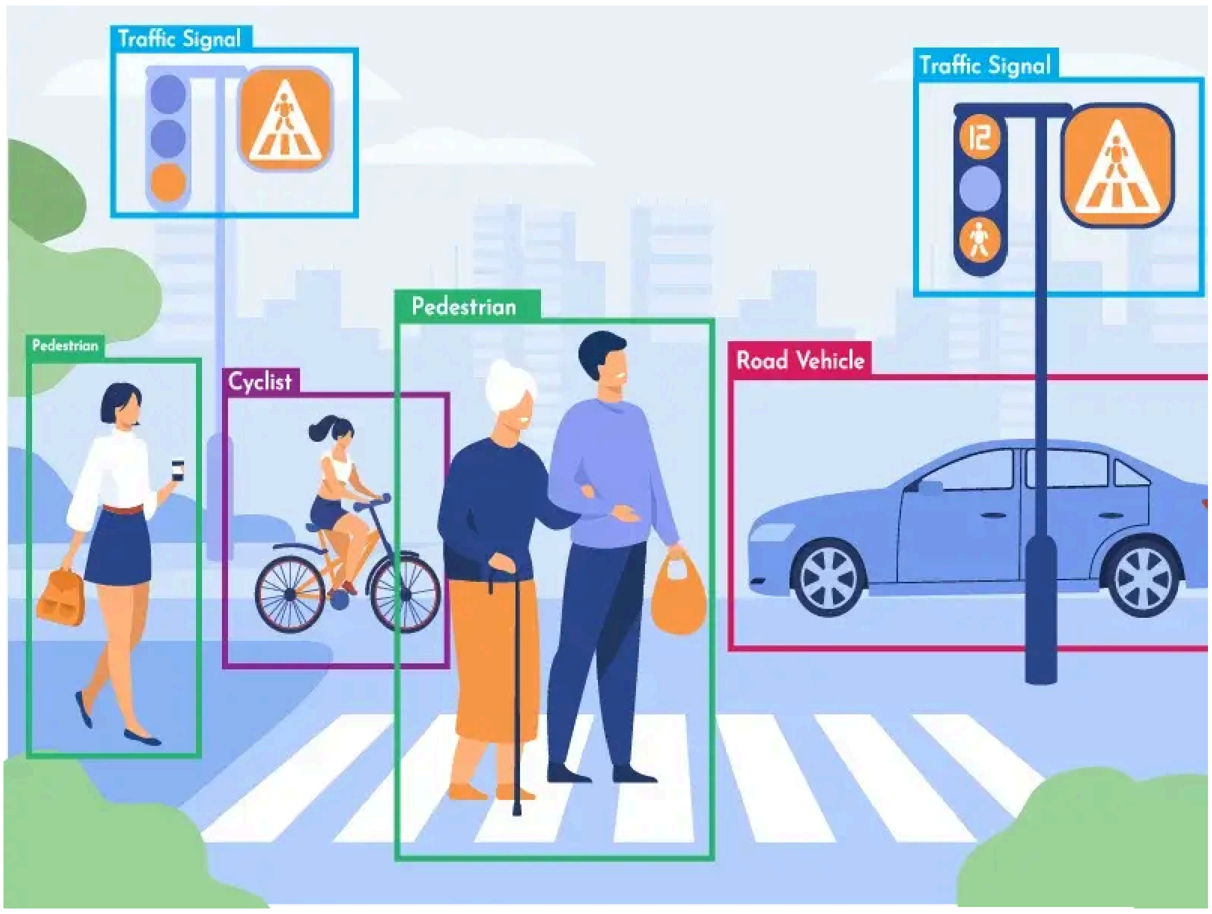
Table of contents

Lecture 10

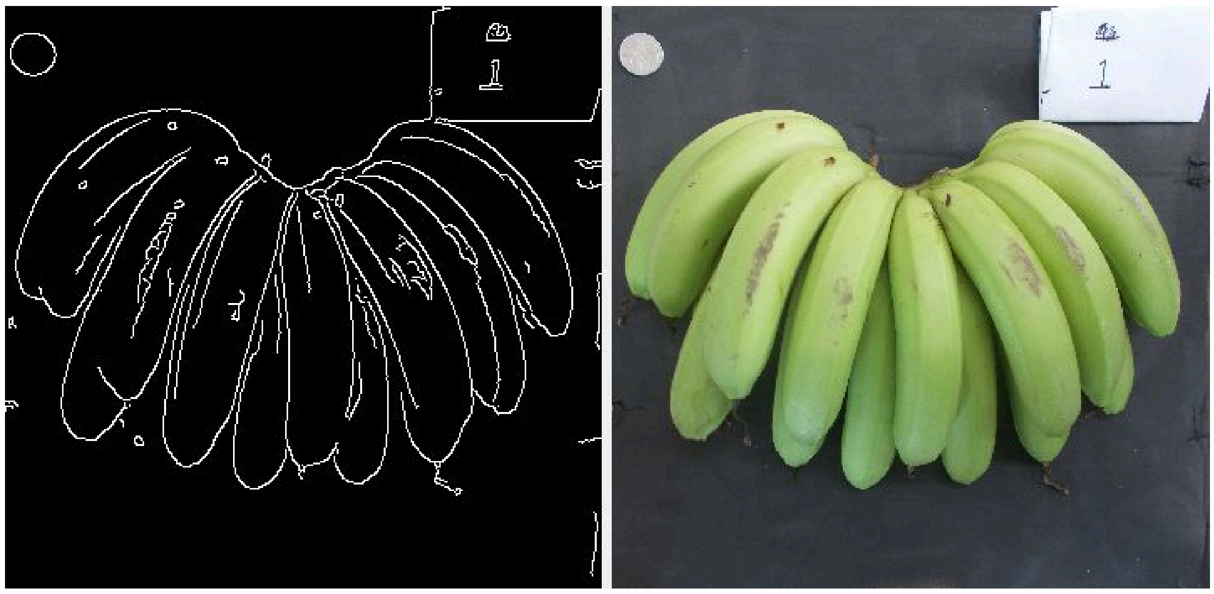


SISTEMA DE INGLES

Topics:

1. Contour

Contour



Stack overflow

What are contours?

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity.

The contours are a useful tool for shape analysis and object detection and recognition.

Steps:

1. **Convert to grayscale and apply other necessary preprocessing**
2. **Apply thresholding method or edge detection method**
3. **Find contours**
4. **Draw contours for visualization (optional)**

syntax:
contours, hierarchy = cv2.findContours(image, mode, method, offset=None)
cv2.drawContours(target_img, contours, contourIdx, color, thickness)

contourIdx: index of the desired contours, if -1 then it will use all contour points

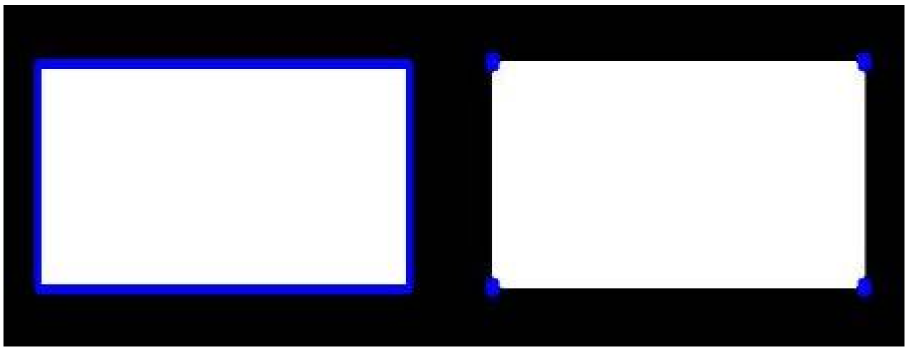
Mode: Contour retrieval modes

cv.RETR_EXTERNAL
cv.RETR_LIST
cv.RETR_TREE

Learn more about the modes here.

Method: Contour approximation method
    cv.CHAIN_APPROX_NONE
    cv.CHAIN_APPROX_SIMPLE
Lear more about the approximation methods here.

Contour Approximation Method


OpenCV

cv.CHAIN_APPROX_NONE vs cv.CHAIN_APPROX_SIMPLE

Contour Features

1. Moments

Image moments help you to calculate some features like center of mass of the object, area of the object etc. Check out the wikipedia page on Image Moment

cv2.moments(cnt)

Applications:

- **Centroid Calculation**: Moments can be used to find the centroid of an object in an image.
- **Shape Analysis**: Moments help in characterizing the shape of objects, which can be useful in object recognition tasks.
- **Feature Extraction**: Moments can serve as features for machine learning models in image classification tasks.

2. Contour Area

Calculates a contour area. The returned area and the number of non-zero pixels, if you draw the contour using drawContours or fillPoly , can be different.

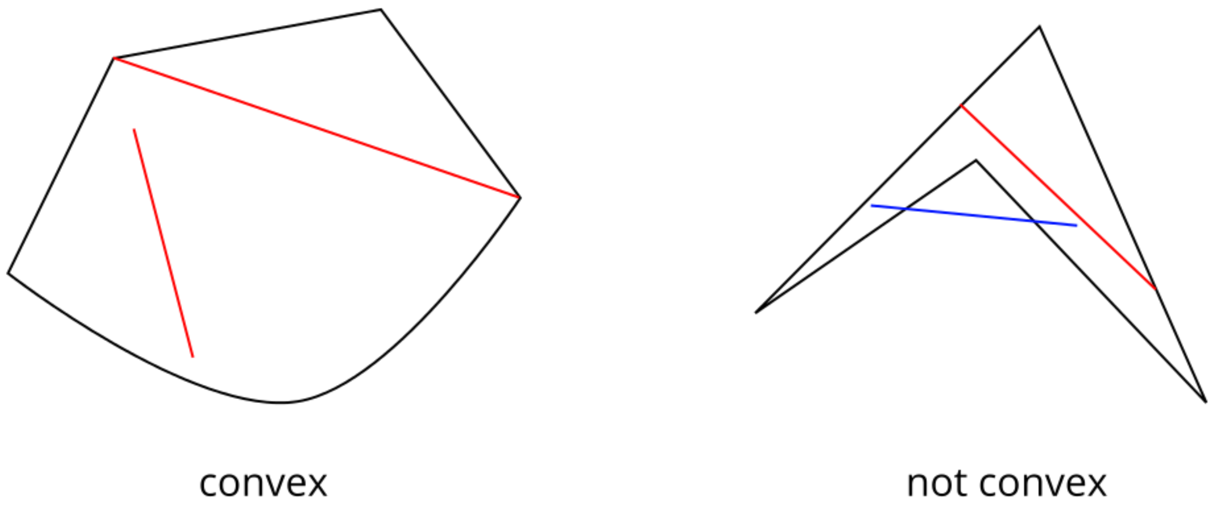
area = cv.contourArea(cnt)

3. Contour Approximation

It approximates a contour shape to another shape with less number of vertices depending upon the precision we specify. It is an implementation of Douglas-Peucker algorithm.

4. Convex Hull

The **convex hull** of a set of points in a Euclidean space is the smallest convex shape that can enclose all the points. It can be thought of as the shape formed by stretching a rubber band around the outermost points of the set.
convex curves are the curves which are always bulged out, or at-least flat, if it is bulged inside, it is called convexity defects.



Underground mathematics

Syntax:
hull = cv.convexHull(points[, hull[, clockwise[, returnPoints]]])

Arguments details:

- ***points*** *are the contours we pass into.*
- ***hull*** *is the output, normally we avoid it.*

- ***clockwise :*** *Orientation flag. If it is True, the output convex hull is oriented clockwise. Otherwise, it is oriented counter-clockwise.*
- ***returnPoints :*** *By default, True. Then it returns the coordinates of the hull points. If False, it returns the indices of contour points corresponding to the hull points.*

You can simply use the syntax below avoiding optional arguments.
hull = cv.convexHull(cnt)

Applications:

The convex hull is a powerful geometric concept with several important use cases in computer vision and image processing. Here are some of the main applications and how it can be helpful for image data analysis and feature detection:

1. Shape Analysis:

- **Characterization of Shapes**
- **Shape Comparison**

2. Object Detection and Recognition:

- **Bounding Shapes**
- **Feature Extraction**

3. Image Segmentation:

- **Region of Interest (ROI) Extraction**
- **Contour Simplification**

4. Collision Detection:

- **Physics Simulations**

5. Path Planning:

- **Robotics**

6. Feature Detection:

- **Keypoint Localization**
- **Shape Matching**

5. Bounding Rectangle

- **Straight Bounding Rectangle:** It is a straight rectangle, it doesn't consider the rotation of the object.

Function: **cv.boundingRect(cnt)**

- **Rotated Rectangle:** bounding rectangle is drawn with minimum area, so it considers the rotation also.
  Function: **cv.minAreaRect(cnt)**

6. Minimum Enclosing Circle

It is a circle which completely covers the object with minimum area.
Function: **cv.minEnclosingCircle(cnt)**

7. Fitting a Line

Fits a line to a 2D or 3D point set.
Function:
vx, vy, x, y = cv2.fitLine(points, distType, param, reps, aeps)

points: set of points that you want to fit a line to. (eg. cnt)

distType: distance type used to fit the line. Common options include:
cv2.DIST_L2: This uses the Euclidean distance (L2 norm) to minimize the distance from the points to the line.
cv2.DIST_L1: This uses the L1 norm, which minimizes the sum of absolute differences.
cv2.DIST_L12: This is a combination of L1 and L2 norms.
Purpose: It determines how the fitting process evaluates the distance between the points and the line.

param: not commonly used and is typically set to 0. It is a placeholder for additional parameters that may be needed for certain distance types.

reps: specifies the accuracy of the fitting. It represents the distance threshold for considering a point as being close to the fitted line. A smaller value means a more accurate fit, while a larger value allows for more deviation from the line.

aeps: used to specify the relative accuracy of the fitting. It is a threshold for the fitting process to determine when to stop iterating. A smaller value results in a more precise fit.