2023.07.07

人工知能
プロジェクト実習 AI-2
(第12回)

コンピュータサイエンス学部
青木　輝勝
Email: aokitms@stf.teu.ac.jp

第12回～第14回の予定

- 第12回～第14回はグループワークを行います．具体的な内容は下記の通りです．
  - **第12回　総合演習(1)**
  - **第13回　総合演習(2)**
    - いずれもグループで議論した上でプログラムを作り上げます．
  - **第14回　プレゼンテーション**
    - 第12回，第13回で開発したプログラム・実験結果についてグループ毎にプレゼンテーションをして頂きます．

本日の内容

- 第12回：総合演習(1)（自然画像中の文字領域の推定）
  - 12.1　自然画像中の文字領域の推定
  - 12.2　本日の課題

本日の内容

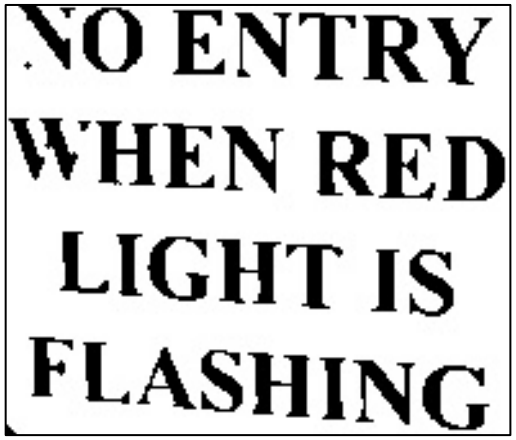
- 第12回：総合演習(1)（自然画像中の文字領域の推定）
  - 12.1　自然画像中の文字領域の推定
  - 12.2　本日の課題

自然画像中の文字領域の推定（1/2）

- 自然画像(写真)の中から文字の領域を抽出する技術．ほとんどの場合，抽出した文字領域を抽出した後にOCR*と呼ばれる技術などを用いて文字の内容を認識する．
- 今回は「文字領域の抽出」までを検討対象とする．

(a)入力画像

(b)文字領域の検出



(c)認識された文字情報

NO ENTRY
WHEN RED
LIGHT IS
FLASHING

*OCR : Optical Character Recognition

自然画像中の文字領域の推定 (2/2)

- 自然画像中の文字領域の推定を行う手法は,
  - ルール型 (人間が文字領域の特徴をルールとして与える)
  - 機械学習型 (多くの事例を与えてルールを学習させる)

  に大別できる.

- 今回はルール型を検討対象とする. ルールについては以下の3項目を踏まえて考えると良い.
  - 画像の前処理(画像サイズやCannyパラメータの最適化など)
  - バウンディングボックスの性質やボックス同士の関係
  - (余裕があれば)バウンディングボックス内の画素情報の性質

基本的な処理の流れ (1/2)

- STEP1:
  - 入力画像をグレー画像に変換する.
  - グレー画像から**輪郭線画像(Canny画像)**または**2値画像**を作る.
- STEP2:
  - 輪郭線画像または2値画像から**輪郭線情報を抽出**する.
- STEP3:
  - 輪郭線情報を用いて, **輪郭線1つずつに対し, バウンディングボックスを生成**し, 画像上に表示する.
- STEP4:
  - 多くの場合, STEP3で大量のバウンディングボックスが生成されてしまうため, **独自ルール**を用いて無駄なボックス(非文字の輪郭線を囲んでいるボックス)を削除してゆく.

基本的な処理の流れ (2/2)

(a)入力画像

(b)Canny画像

(c)全ボックスの表示

(d)ルールによりボックスを削除

本日の内容

- 第12回：総合演習(1)（自然画像中の文字領域の推定）
  - 12.1　自然画像中の文字領域の推定
  - 12.2　本日の課題

第7回講義資料P26

画像内の輪郭線をすべて探し出す

contours, hierarchy = cv2.findContours(img, search, approx)

img : 入力画像(2値画像または Cannyフィルタをかけた後の画像)

search：輪郭線の探索方法の指定

approx：輪郭線の表現(近似)方法の指定

contours：抽出された輪郭線の情報(座標)が格納されるリスト.

hierarchy：抽出された輪郭線の階層構造が格納されるリスト.

輪郭線の探索方法の指定:

cv2.RETR_EXTERNAL：一番外側の輪郭のみ抽出 (通常これで十分).
cv2.RETR_LIST：すべての輪郭を抽出. ただし, 階層構造は持たない.
cv2.RETR_TREE: すべての輪郭線を階層構造を持たせて抽出.

輪郭線の表現方法の指定:

いくつかの方法が定義されているが, 下記が使えれば十分.

cv2.CHAIN_APPROX_SIMPLE

ボックス情報の取得と描画 (1/2)

バウンディングボックス情報の取得

x, y, w, h = cv2.boundingRect(contours[i])

contours[i]：cv2.findContours()で抽出された全輪郭線のうちi番目の輪郭線の情報.

x, y：輪郭線に外接するバウンディングボックスの左上の座標.

w, h：バウンディングボックスの幅(=w)と高さ(=h).

バウンディングボックス情報の描画

box_img = cv2.rectangle(img, (x_tl, y_tl), (x_br, y_br), color, line)

img：入力画像.

(x_tl, y_tl)：バウンディングボックスの左上(Top Left)の頂点の座標.

(x_br, y_br)：バウンディングボックスの右下(Bottom Right)の頂点の座標.

color：(b, g, r) で表す. 例) 赤の場合 (0, 0, 255) .

line：画像に描き込まれるバウンディングボックスの線の太さ. 例) 5

ボックス情報の取得と描画 (2/2)

ボックス情報の取得/描画処理の一例

```
contours, hierarchy = cv2.findContours(canny_img, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
num_contours = len(contours)
for i in range(0, num_contours):
    x, y, w, h = cv2.boundingRect(contours[i])
    print('x = ', x, ' y = ', y, ' w = ', w, ' h = ', h)
    output_img = cv2.rectangle(input_img, (x, y), (x+w, y+h), (0, 255, 0), 3)
output_img_rgb = cv2.cvtColor(output_img, cv2.COLOR_BGR2RGB)
plt.imshow(output_img_rgb)
```

課題12

- 実験用画像20枚(test_img1201.jpg ～ test_img1220.jpg)のうち10枚以上の画像に対し，グループ内で生成した独自ルールを駆使して文字領域を可能な限り正確に抽出せよ (評価は正確さと画像枚数の両方の視点から行う).

- 実用性の観点からは，同一プログラム(パラメータも同一)を実験用画像10枚(以上)に適用することが望ましい. ただし，入力画像のサイズはあらかじめ最適化してもよいこととする.

- 最終回(第14回)の授業にて各グループ毎にプレゼンテーションを行う予定である. このため，実験結果はパワーポイントとしてまとめること(本日moodle上で提出する必要はない).