Higher Institute for Applied Sciences and Technology
Informatics Department
Image Retrieval Lab
2019 - 2020

HIAST

Homework

Image Processing using OpenCV

Overview

OpenCV:
OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. It has over 2500 algorithms, extensive documentation and sample code. It is free for use under the open source BSD license and works on Windows, Linux, Mac OS X, Android and iOS.

Emgu:
Emgu CV is a cross platform .Net wrapper to the OpenCV image processing library. Allowing OpenCV functions to be called from .NET compatible languages such as C#, VB, VC++, ... etc.

Notes:

- A C# project linked to Emgu is provided for you.
- To use Emgu functions, you need to call it from the class **CvInvoke**.

  ```
  example: CvInvoke.Imshow(para).
  ```

- Functions' details are available in *help.pdf* document or *Emgu documentation* [1]*page*.

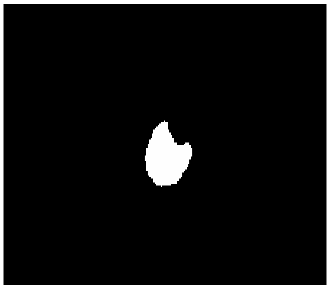
[1] http://www.emgu.com/wiki/files/3.1.0-r16.12/document/html/8dee1f02-8c8a-4e37-87f4-05e10c39f27d.htm

1. Tumor detection

We want to detect the presence of a tumor in a brain MRI scan (MRI: Magnetic resonance imaging), based on the fact that in MRI images the tumor area has a higher intensity level than normal cells.

1. Read the image `mri.jpg` using **imread** function and display it using **imshow** function.
2. Convert the image to gray using **CvtColor** function*.*
3. Reduce the noise using **GaussianBlur**[2] or **Blur** function.
4. Adjust the image contrast using **CvFunctions.ContrastStretching** function, to help you in the later thresholding process.
5. Convert the image to binary using **Threshold** function. (remember the foreground pixels must be white)
6. Enhance the binary image with morphological operation(s) using **MorphologyEx** function.
    - Create the structuring element(s) using **GetStructingElement** function.

Expected processing result:

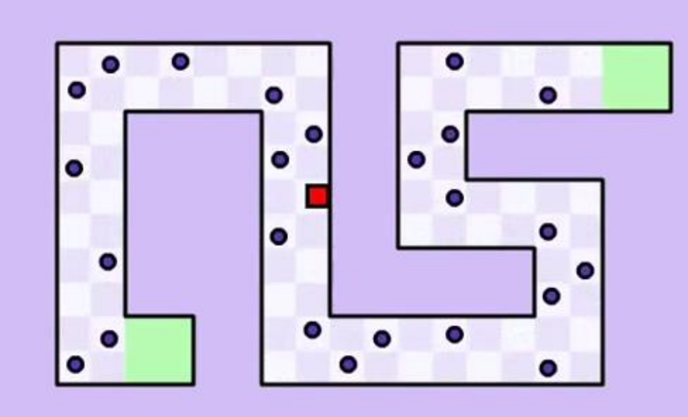


7. Draw a contour around the tumor on the original image using **CvFunctions.DrawContours** function.
8. Display the result of each step using **imshow** function

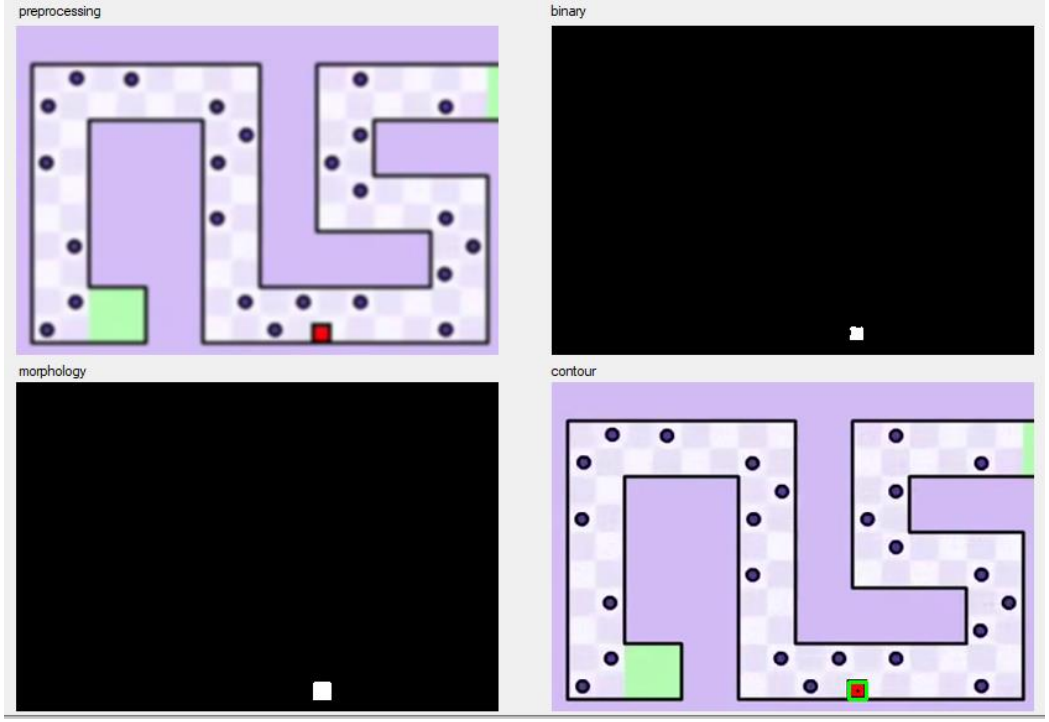
[2] You can start with a filter of 3x3 window and 0.5 deviation in the x direction.

2. Object Tracking

We want to track a red object moving through obstacles in a video.



1. Read video frames using **VideoCapture** class.
2. For each frame:
    a. Reduce the noise and show the result in *preprocessing* window.
    b. Convert the frame to HSV colorspace using **CvtColor** function**.**
    c. Convert it to binary using **inRange** function keeping only the red square and show the result in *binary* window.
    d. Apply morphological operations to enhance binary image and show the result in *morphology* window.
    e. Draw Contour on tracked object using **CvFunctions.DrawContours** function and show the result in *contour* window.


preprocessing
binary
morphology
contour