BOOK BUG

VINAY DATTA PINNAKA

COSC 5590 COMPUTER VISION AND IMAGE PROCESSING FALL 2015

Interim Report

Instructor: Dr. Maryam Rahnemoonfar

FUNCTIONALITY

The main functionalities involved in Book Bug are

1. Taking a frame from stream of video from camera of the device.
2. Correction of image to avoid Perspective distortion.
3. Smoothing out the Image and make it a binary image for contour analysis.
4. Draw counters in the image to distinguish between Title, author and edition of the book
5. Making counter masks so that tesseract will extract the title, author and edition number separately.
6. Passing the countered images to Tesseract API so that Data is extracted separately.
7. Store the extracted data to the file or compare the extracted data to contents of the file.

TAKING A FRAME

CONCEPT

User has to take the image of the book cover in order to store the title, edition and other details. So, a frame from the video captured by the camera is taken when user requests it.

PROGRAM IMPLEMENTATION

OpenCV provides the function a class VideoCapture, This class captures video from video files cameras , image sequences. Here is how the class is used.

```
VideoCapture cap(0); // open the default camera
```

When mouse click event is occurred next frame is write to a Mat object using following code

`VideoWriter::write`(const Mat& **image**)

PERSPECTIVE DISTORTION

CONCEPT

Before passing an image to OCR engine we must ensure that the image or the frame taken is free from Perspective distortion. Because OCR can't detect the text accurately if this distortion is present. The algorithm used to avoid this distortion is as fallows

1. Get the edge Map
2. Detect line with Hough transform
3. Getting the corners by finding intersection between lines
4. Determine top-left, bottom-left, top-right, and bottom-right corner.
5. Now apply perspective transform.

PROGRAM IMPLEMENTATION

Using canny edge detector provided by OpenCV we can find the edges of the source image. This edge map image is later used to find line segments using Hough transform. OpenCV function for canny edge detector is.

Canny (Inputimage**, Outputedges, threshold1, threshold2, apertureSize)**

Probabilistic Hough Transform is used to obtain find line segment in binary image.

HoughLinesP (Input**image**, Output**lines**, **rho**, **theta**, **threshold**, **srn**=0, **stn**=0)

Line segments of the edges intersect at corners, once the corners are found out then we can apply Perspective transformation using wrapPerspective method

warpPerspective (src, dst, InputArray M**, dst.size())**

SMOOTHING & THRESHOLDING

Before we apply contours to the image the preprocessing that need to be done is Smoothening the image with GaussinBlur function provided by opencv

GaussianBlur(Inputimg,Outputimg,Imgsize,0)

For thresholding the function used is adaptiveThreshold and the usage is,

adaptiveThreshold(img, img,255,CV_ADAPTIVE_THRESH_MEAN_C, CV_THRESH_BINARY,75,10)

DRAW CONTOURS

Apply dilation to obtain thicken lines in the image. Now find contours in the image using the opencv function findContours and the program implementation is,

findContours(Src**image**, **contours**, **hierarchy**, **CV_RETR_EXTERNAL**, **CV_CHAIN_APPROX_NONE**, Point(0,0))

Now draw the bounding box or rectangle circumscribing each contoured object – each of them frames a text, usage of opencv function is as fallows

boundingRect(InputArray **points)**

OTHER FUNCTIONALITY

Still working on Tesseract API's, and a mechanism to store data and retrieve data has to be implemented. The results expected from OCR engine should be highly accurate because these are critical in data storage and retrieval. Preprocessing process that is to be done before passing to OCR is completed.