Final Specification

Project Name:

OCR (Optimal Character Recognition) & Translation

Group members:

Annie Lin
annielin@college.harvard.edu
Hillary Do
hillaryjiado@college.harvard.edu
Nhu Nguyen
nhunguyen@college.harvard.edu
Keon Ho (Chris) Lim
klim01@college.harvard.edu

Signatures/Interfaces

- class EdgeImage(image):
    - coords
        - instance var
    - grads
        - also instance var
    - img
        - stores the image!
    - def __init__(self, imgName)
        - override the constructor
        - Takes in an image name
        - Loads the image and stores it in img
        - Computes coords/grad info and stores them in vars
    - @staticmethod (functions henceforth are static methods)
    - def findIntensityGradientX(image)
    - def findIntensityGradientY(image)
    - def processGradients(xGrad, yGrad)
    - def edgeThinning(matrix)
    - def applyHysteresisThreshold(matrix)
    - def extractCoordinates(matrix)
    - def extractGradients(matrix)
- def searchImage(image, template)
    - image and template are of "EdgeImage" types
    - Uses their coords/grads data to search/match
    - spits out coordinates of things found!

Timeline

Week 1 (Goal: Understand Algorithm and Implement) by Friday the 25th

- Finish writing at least three parts of the Template Matching Algorithm in Python based on http://www.codeproject.com/Articles/99457/Edge-Based-Template-Matching
    - Step 1: Find the intensity gradient of the image (Annie)
    - Step 2: Apply non-maximum suppression (Nhu)
    - Step 3: Do hysteresis threshold (Chris)
    - Step 4: Save the data set (Hillary)
- Use edge result to match (Chris)
- Test the written parts using small samples (Everyone)
- Talk to/e-mail TF (Sorry, Aaron)

Week 2

- Finish writing the template matching algorithm
- Write an application that uses the said algorithm that takes in an image as an input and spits out the text
    - Make sure program works for more than one line (Annie)
    - First, begin with just “0”s and “1”s as templates, then gradually expand to all alphanumeric characters
- Add features, which may include:
    - 1) Binary to text feature
    - 2) Compile the translated text, if the text is code
    - 3) Scan url -> opens url on your device
- Last Minute Crying
- Correct last minute bugs
- Finish and hand in to Aaron

Progress Report

- We demonstrated that template matching is a viable option for implementing OCR
    - Please see our sample program, test2.py
    - It takes in an image of 0s and 1s, converts the image into grayscale, templates matches 0s and 1s, then processes the output coordinates to print out text!
- However, this program uses the already built-in implementation of template matching
    - We will write our own template-matching function!
    - This will be the core of our project!

Version Control- We have it set up!

Here: git@code.seas.harvard.edu:~nhunguyen/cs51-2014/nhunguyen-51psets.git