Mobile Address Tagging for OpenStreetMap

Joel Nelson, Steven Bell, David DeBaun
Department of Electrical Engineering, Stanford University

Objective

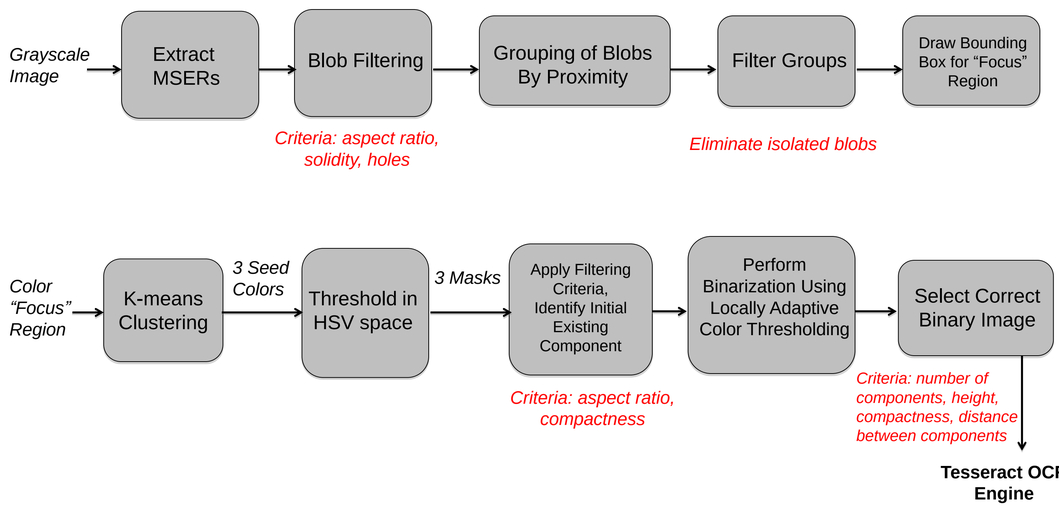
- Develop an Android smartphone application that can tag features in OpenStreetMap (OSM) using the phone's camera and built-in sensors.
- OpenStreetMap is a crowd-sourced mapping project, analogous to Wikipedia for cartography
- While many features such as roads and buildings can be readily drawn from aerial and satellite imagery, a complete map requires that these objects be tagged with their names, addresses and other data
- This has traditionally been a tedious process requiring field notes and hours entering tags on a computer, but a mobile device has the potential to radically accelerate this process

OSM Integration

- To display untagged buildings in a simple and fluid way, we created a custom map using OSM data which displays untagged buildings in red.

- These tiles are hosted by a remote server and transmitted to the application over HTTP.
- When the user selects a building, the application downloads the known data for the building such as the city and zip code. This information is merged with the data gleaned from the image.
- When the user confirms the address, it is sent to the global OSM server using an XML-based API.

Image Processing Algorithm



Application Process Flow

Map View
- Application displays OSM content for user's immediate location, using the GPS
- Buildings that have not yet been tagged are highlighted in red
- User clicks untagged building to tag new data

Camera View
- Application enters camera mode and user takes picture(s) of sign with text to be extracted
- Allows user to retake picture if the result is undesirable

Processing Text Form
- Application processes the image by performing text detection and running the OCR engine to parse the text

OSM Data Form
- Application shows the data extracted from the captured image to be populated in a new tag on OSM
- Gives user opportunity to confirm correctness of the parsed data

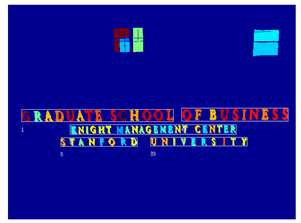
Sample Results

Input image

The input image as captured by the phone's camera.
The Tesseract output is a jumble of symbols and characters.

Extracted MSERS

Dark MSERs extracted from the image. Our solution runs two separate passes, working with both light and MSERS.



Detected Text

The detected text regions are boxed. Note that the window panes were selected as candidate text objects, but rejected because all of the objects have high solidity. Tesseract can now process the selected regions correctly.

References

H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod, "Robust text detection in natural images with edge-enhanced Maximally Stable Extremal Regions," in Image Processing (ICIP), 2011 18th IEEE International Conference on, 2011, pp. 2609–2612.

E. Kim, S. Lee, and J. Kim, "Scene Text Extraction Using Focus of Mobile Camera," 2009 10th International Conference on Document Analysis and Recognition, pp. 166–170, 2009.

S. P. Chowdhury, S. Dhar, A. K. Das, B. Chanda, and K. McMenemy, "Robust Extraction of Text from Camera Images," 2009 10th International Conference on Document Analysis and Recognition, vol. 15, no. 18, pp. 1280–1284, 2009.