THE SMITH-KETTLEWELL

EYE RESEARCH INSTITUTE

RERC

CROSSWATCH

Giovanni Fusco – giofusco@ski.org

June 2015

Introduction

This document describes the software implemented for the CrossWatch project, described in [*Fusco, Giovanni, Huiying Shen, and James M. Coughlan. "Self-Localization at Street Intersections." Computer and Robot Vision (CRV), 2014 Canadian Conference on. IEEE, 2014.*] The goal of the project is to help the self-localization at street intersections of user with visual impairment by using imagery acquired near crosswalks.

The project has two main components: the data acquisition Android app (used to take panoramas at street intersections) and the localization software to be run offline in Matlab. The location of the user is determined by matching the template of the intersection where the user is standing at (using GPS coordinates) with the crosswalk stripes detected in the aerial view generated from the panorama image taken by the user.

Intersection Data

The Android app assumes that the user is standing at the intersection and takes a panorama picture of the scene by rotating the torso (from left to right or viceversa). A frame is saved if there's been an increment in the yaw bigger than a fixed threshold. Each saved image is associated to a text file containing sensors info at the time the picture was saved (see figure 1).

```
Accl:       9.696     0.673     2.168     9.958

Compass:      52.2    -12.6     86.0
Rotation:     0.22231   -0.58907    0.22815
Magnetic:     -33.780    16.836   -22.488
43.935

Rotation Matrix:
   0.20188   -0.60078   -0.77350
   0.07696    0.79705   -0.59899
   0.97638    0.06139    0.20714
longitude = -122.434323
latitude = 37.790788
altitude = 37.95
accuracy = 12.00

accuAcc: 3    accuMag: 3    accuRot: 3

aveFrm = 10    sensorDelay = 1
```

Fig. 1 Example of the sensors data saved with each frame by the Android app

Intersections Templates and simple GIS

In order to localize the user, the system needs to keep a collection of intersections templates, i.e. a set of binary masks in which the crosswalk stripes have been manually segmented (using Google Maps satellite images of intersections as a reference) and a binary mask that specifies the locations of the sidewalks (prior). In order to retrieve these templates at run time, the GPS coordinates of the point in the center of the intersection is used as a key for retrieval (Figure 3).

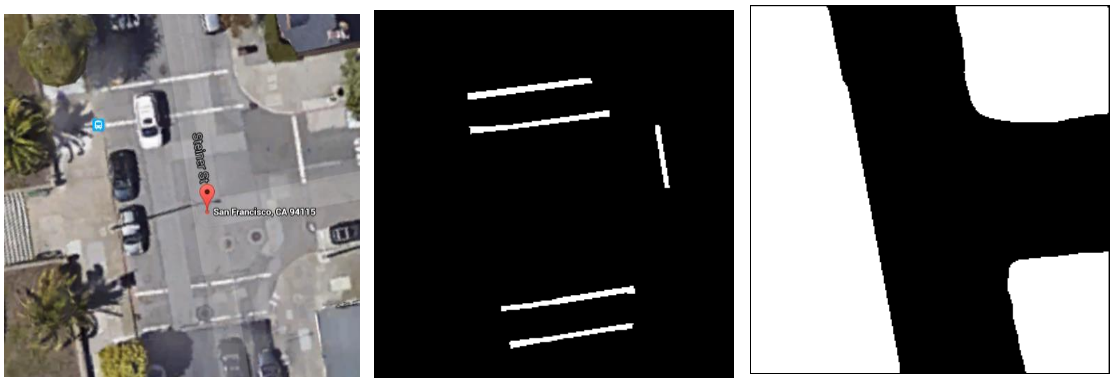

Fig. 2 Example of intersection template (center) and sidewalk area (right). The GPS coordinates corresponding to the pin location (left) are used for retrieval.

```
{
  "intersections": [
    {
      "name": "Fillmore_Washington_SF",
      "lat": 37.791613,
      "long": -122.434428,
    "x": -1,
    "y": -1
    },
    {
      "name": "Steiner_Washington_SF",
      "lat": 37.791409,
      "long": -122.436057,
    "x": 160,
    "y": 140
    }
   ]
}
```

Fig. 3 Example of JSON containing the database of the intersections

Dataset File

To test all the acquired data in batch, we use on a dataset JSON file containing information about the location of the folders, the focal length of the camera used and the height (in meters) at which the camera was held.

```
{
 "dataset": {
   "series": [
      {
         "path": ".\\dataset\\Dec 20, 2013 2;33;19 PM",
         "id": 1,
         "camera": {
             "id": 2,
             "model": "SAMSUNG G4W",
             "focal": [593.1275435,593.9653542],
             "height": "1.53"
         },
         "images": [
            {"filename": "00.jpg"},
            {"filename": "01.jpg"},
            {"filename": "02.jpg"},
                  …
            ]
      },
```

Fig. 4 Example of JSON containing the dataset

Running the Crosswatch software

Setting up

In the folder *config*, the file named *crosswatch_config.m* contains the variables that define the location on the disk of the dataset and the intersection database. The file is currently configured to work with the example dataset included with the source code.

```
%path to the folder containing the data
DATASET_ROOT = pwd;

%path to the JSON file of the dataset
DATASET_JSON = '.\dataset\test_dataset.json';

%path to JSON intersections database
INTERSECTION_ROOT = '.\DB_Intersections';
INTERSECTION_JSON = '.\DB_Intersections\intersections.json';
```

Fig. 5 Crosswatch configuration variables

Getting Started

A demo of Crosswatch can be launched by typing the command *crosswatch* from the Matlab shell*;* the output should be similar to Figure 6.

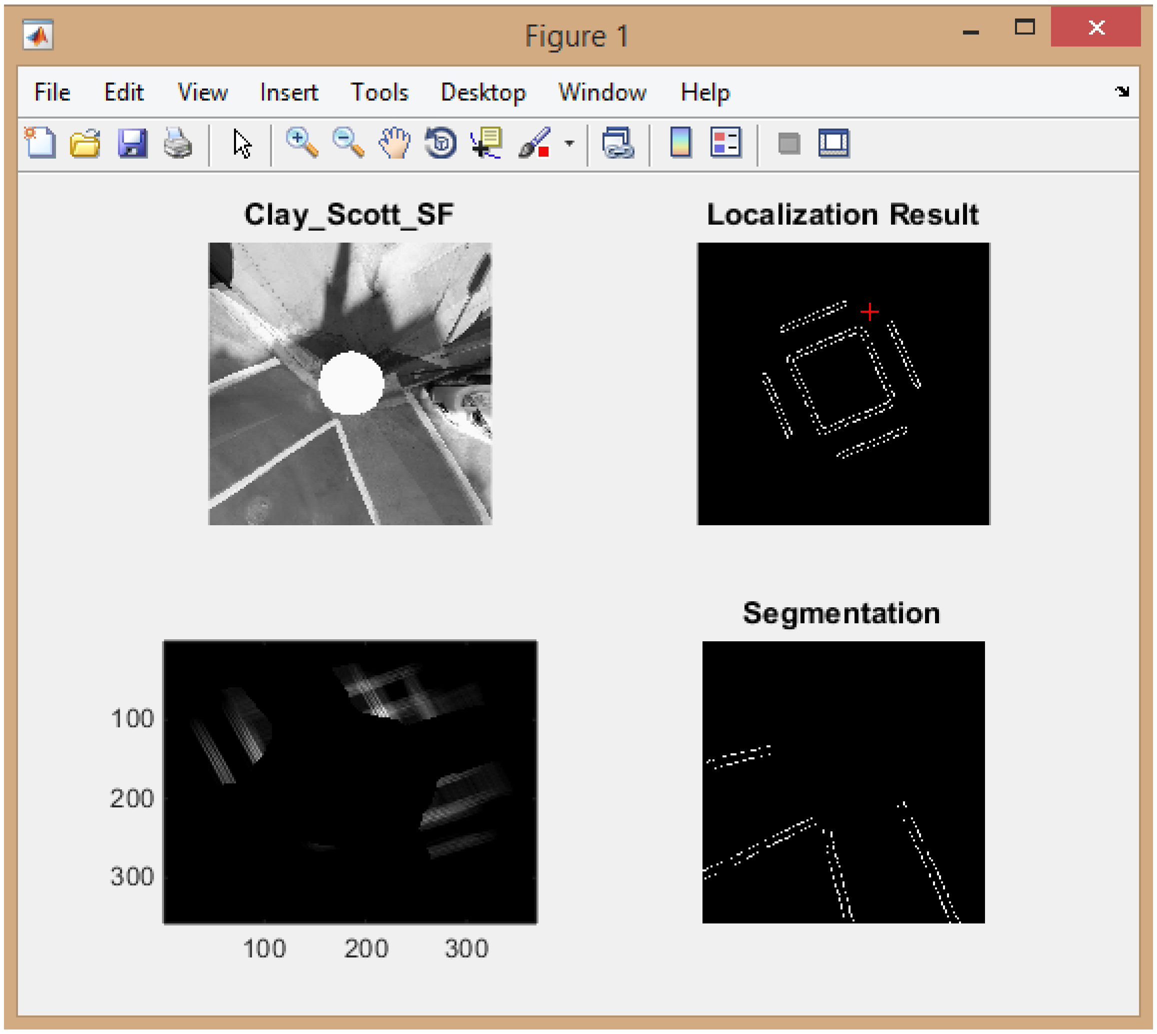

Fig. 6 Output of the Crosswatch demo

Stitching Software

To create reconciled orientation estimates, Crosswatch uses the rotation matrices calculated using OpenCV stitching algorithm. The package includes a Windows 8.1 64 bits binary of the stitching tool. If using a different platform, the source code has been included (Opencv >= 2.4.6 is required).