An Initial Study of Automatic Curb Ramp Detection with Crowdsourced Verification using Google Street View Images

Kotaro Hara[1], Jin Sun[2], Jonah Chazan[3], David Jacobs[2], Jon E. Froehlich[1]

[1]Makeability Lab | HCIL
Department of Computer Science
University of Maryland, College Park
{kotaro, jonf}@cs.umd.edu

[2]Department of Computer Science
University of Maryland
{jinsun, djacobs}@cs.umd.edu

[3]Computer Science Magnet Program
Montgomery Blair High School
{jechazan}@gmail.com

Abstract

In our previous research, we examined whether minimally trained crowd workers could find, categorize, and assess sidewalk accessibility problems using Google Street View (GSV) images. This poster paper presents a first step towards combining automated methods (*e.g.,* machine vision-based curb ramp detectors) in concert with human computation to improve the overall scalability of our approach.

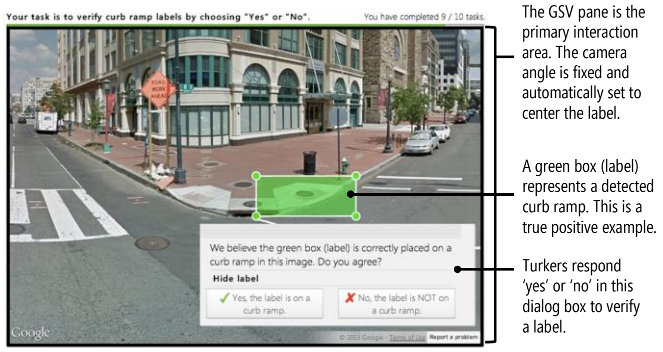
Introduction

Although online maps and routing algorithms continue to improve—*e.g.,* by providing pedestrian and bicycle navigation support—there is little work focused on efficiently obtaining and integrating accessibility data. In previous research [Hara et al. 2013a], we examined whether remote crowd workers could find, categorize, and assess sidewalk accessibility problems using Google Street View (GSV) images. With simple quality control methods (*e.g.,* majority vote), we found that workers could identify accessibility problems with 93% accuracy; however, crowd work is, by nature, labor intensive. Our current explorations, then, are focused on investigating how *automated* methods can be *combined* with human computation to improve the overall scalability, efficiency, and quality of our approach.

While we expect that computer vision and machine learning will provide multiple benefits including location triaging, view selection, and adaptive workflows, in this poster paper, we examine a single use case: automatically detecting accessibility features in GSV images with human verification. In our aforementioned past work, we found that *labeling* tasks were more time consuming than *verification* tasks. Thus, in our work here, crowd workers simply *verify* results from an automatic curb ramp detection algorithm rather than provide their own labels (Figure 1). Curb ramps were selected both because of their visual salience in images (a good starting point) as well as because of their significance to accessibility (*e.g.,* see [3rd Circuit 1993]).

Study Method

We first created a test dataset defined by two geographic regions of interest in Washington DC: a 0.71 km^2 semi-urban mixed residential/commercial area (Mt. Pleasant=*MP*) and a 0.56 km^2 dense urban area with high pedestrian traffic (just east of the White House=*WH*). We built a custom web scraping tool to (i) query the GSV API, (ii) infer the center point of street intersections, (iii) and automatically download the appropriate GSV panoramic image (13312x6656 px) at that intersection location. We also scraped the accompanying 3D-point cloud, which is far coarser than the RGB image (512x256 px; 0.15% of the RGB image).


Your task is to verify curb ramp labels by choosing "Yes" or "No".
You have completed 9 / 10 tasks.
We believe the green box (label) is correctly placed on a curb ramp in this image. Do you agree?
Hide label
✓ Yes, the label is on a curb ramp.
✗ No, the label is NOT on a curb ramp.

The GSV pane is the primary interaction area. The camera angle is fixed and automatically set to center the label.

A green box (label) represents a detected curb ramp. This is a true positive example.

Turkers respond 'yes' or 'no' in this dialog box to verify a label.

Figure 1: We explore initial approaches to scalably detect accessibility problems in GSV images by combining *automated approaches* with *human computation*. This figure shows our validation interface for crowd workers to verify automatically detected curb ramps.

To create ground truth, three members of our research team independently outlined curb ramps in these images using a custom web-based labeling tool. We used majority vote between the three labeled datasets to create a single ground truth dataset (similar to [Hara et al. 2013a]). In all, our dataset contained 118 intersections with 437 curb ramps (160 in MP and 277 in WH).

Automatic Curb Ramp Detection: For the curb ramp classifier, we trained a linear Support Vector Machine (SVM) using LibSVM [Chang and Lin 2011] with a Histograms of Oriented Gradients (HOG) feature descriptor [Dalal and Triggs 2005] from VLFeat.org. For more detail on this general approach, see our workshop paper [Hara et al. 2013b]. To train the classifier, we used 239 positive curb ramp patches (46x19 px) and 4541 negative patches (roughly 50% of our dataset; the remaining 59 intersections are used in our test set: 198 positive and 3762 negative).

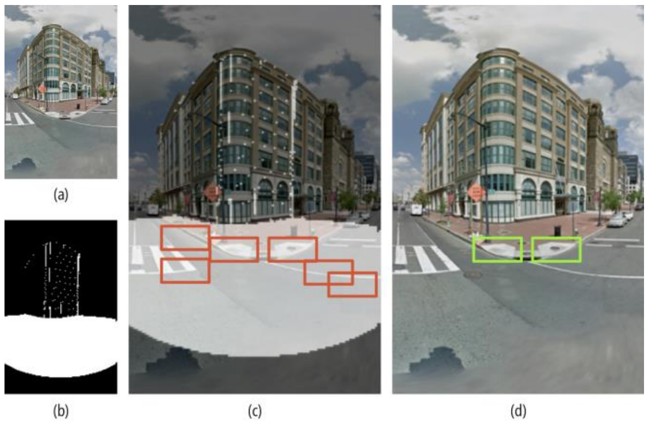


Figure 2: (a) We scraped GSV panoramas of the target intersections then (b) auto-generated a pixel mask based on 3D-point cloud data (from GSV) and street corner areas (from OpenStreetMap). (c) Our curb ramp detector slides a classifier window across the masked area (white) and returns both *false* and *true* positives (red rectangles). (d) The resulting correct rectangles after crowdsourced verification.

Using the trained classifier, we implemented a multi-scale sliding window curb ramp detector. We scanned the classifier across 46x19 px windows of the GSV panoramic image. Following Dalal and Triggs's pyramid approach [2005], this process is repeated across successively shrunken images (at 95%) after each full pass. This provides robustness against curb ramps of varying sizes. To eliminate redundant classifications, we use non-maximum suppression [Malisiewicz et al. 2011], which selects detections with high confidence values and eliminates overlapping classifications (overlap threshold: > 30%).

In an attempt to improve efficiency and accuracy, we also auto-generated image masks to isolate sidewalk intersection pixels. Importantly, our sliding window classifier only passed through these masked pixels (Figure 2b and 2c). The mask was generated via a custom algorithm, which used 3D-point cloud data from GSV to discriminate between streets and non-street elements (*e.g.*, buildings, the sky) as well as building boundary data from OpenStreetMaps to help infer street corner locations.

Human Verification via Mechanical Turk: To investigate the potential of using minimally trained crowd workers for verification, we posted a custom tool to MTurk in July 2013 (Figure 1). In each HIT, turkers had to verify 10 labels. We paid \$0.05 per HIT (\$0.005 per verification). At each intersection, we verified only the top 10 labels with the highest classifier confidence score. To gather consensus, three turkers verified each label. Note: in this paper, we focus only on verifying *false* and *true positives* (false negatives are left for future work).

Results and Conclusion

In total, 26 distinct turkers completed 177 HITs and performed 1,770 verifications (3 verifications for each of the 590 labels). On average, turkers completed 6.8 HITs (*SD*=6.0), which is equivalent to 68 verifications. The median time per HIT was 27.3s (*avg*=31.3s; *SD*=15.0s). We examined two thresholds of correctness: we considered a label correct if 10% of the classified patch overlapped with ground truth as well as a more strict measure: 50% overlap (Figure 3a and b). The detection precision was 22.7% and 9.2% respectively for each threshold without verification, which increased to 90.4% and 43.9% with crowdsourced verification. The precision is not 100% because some labels are ambiguous (*e.g.*, they overlap by some percentage with a curb ramp making it hard for the turker to know if 'yes' or 'no' is the right answer). Low recall indicates that the classifier failed to detect many curb ramps (high false negative rate); future work will attempt to improve classifier performance by using multiple photo angles and top-down satellite imagery.

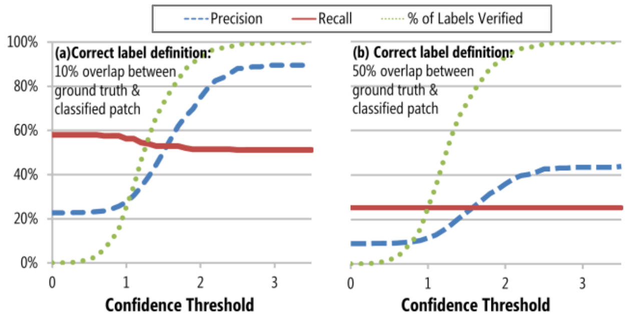


Figure 3: We examined two correctness thresholds: 10% (left) and 50% (right) overlap between ground truth and the classified patch. The x-axis is a different threshold: the classifier's confidence score for each patch. This graph visualizes the result of crowdsourced verification at various confidence score cutoffs (*e.g.*, the result of verifying all labels with a score < 1.5 in the left graph results in ~50% precision).

Though these results are promising, as our initial attempt, our methods require much further development and refinement. Our future work includes improving the performance of the curb ramp detector, investigating how verified labels can *actively* train the classifier, and researching how detection results can be used for crowd scheduling.

Acknowledgements

This work was supported by an NSF grant (IIS-1302338) and a Google Faculty Research Award. We thank Jianxiong Xiao for the GSV 3D-point cloud scraping code.

References

3RD CIRCUIT, C. OF A. 1993. *Kinney v. Yerusalim, 1993 No. 93-1168.* .

CHANG, C.-C. AND LIN, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Trans Intel Sys and Technology 2*, 3, 27:1–27:27.

DALAL, N. AND TRIGGS, B. 2005. Histograms of oriented gradients for human detection. *IEEE CVPR 2005.*, 886–893 vol. 1.

HARA, K., LE, V., AND FROEHLICH, J. 2013a. Combining Crowdsourcing and Google Street View to Identify Street-level Accessibility Problems. *Proc. of CHI'13.*

HARA, K., LE, V., SUN, J., JACOBS, D., AND FROEHLICH, J. 2013b. Exploring Early Solutions for Automatically Identifying Inaccessible Sidewalks in the Physical World Using Google Street View. *Human Computer Interaction Consortium*.

MALISIEWICZ, T., GUPTA, A., AND EFROS, A.A. 2011. Ensemble of Exemplar-SVMs for Object Detection and Beyond. *ICCV.*