FLOOD DEPTH ASSESSMENT WITH LOCATION-BASED SOCIAL NETWORK DATA AND GOOGLE STREET VIEW - A CASE STUDY WITH BUILDINGS AS REFERENCE OBJECTS

Boyuan Zou[1,2], Bo Peng[2], Qunying Huang[2†]

[1]Department of Computer Science, University of Wisconsin-Madison, Madison, Wisconsin
[2]Spatial Computing and Data Mining, Department of Geography, University of Wisconsin-Madison, Madison, Wisconsin
[†] Email: qhuang46@wisc.edu

ABSTRACT

Flood damage accurate assessment, such as flood depth, is very helpful for disaster relief. However, most of existing methods have some limitations, either the expensive data requirement (e.g., hydrological observations), or only producing a rough assessment with location-based social network (LBSN) data. To obtain an accurate and real-time flood damage assessment, this paper firstly collects pairs of pre- and post-flood images for the same location from LBSNs and Google Street View respectively. Next, buildings were segmented in images using machine learning methods (e.g., Mask R-CNN). Finally, the rectifying and building height extraction were developed to calculate the accurate value of the flood depth by comparing the paired images. The method was evaluated by using the real datasets generated from flood disasters in Japan and US. Experiment results show that the proposed method provides an accurate and real-time flood disaster assessment.

Index Terms—Damage assessment, Flood impact assessment, social media, machine learning, computer vision

1. INTRODUCTION

While disasters come in many forms, flooding is one of the most increasingly frequent natural disasters. This is because the flood disasters can be raised along with many other different disasters (e.g., tsunamis and earthquakes), other than only heavy rainfall and river erosion. The flood disasters can cause the loss both in economy and humans' lives. As the most frequent natural disaster, flooding accounts for more than 75% of federally declared disasters in the US [1]. Global flood losses are projected to reach $52 billion by 2050, up from $6 billion in 2005 [2]. Flood impact assessment, including identification of inundation extent and depth, is critical in both real-time and longer-term applications [3]. For example, such assessments can be used to calibrate and validate the models used for floodplain mapping (e.g., FEMA's RiskMAP program) and to help with allocating post-disaster infrastructure reconstruction resources. "Traditional" assessment of flood depth and extent requires highly trained domain experts, who visually inspect damages, high water marks, etc. [4], or intensive manual annotation of remotely-sensed images to train classification models for identifying damaged features [5] [6] [7].

In particular, hydraulic simulation models have been developed to estimate the inundation level or flood water depth [8]. However, these models require careful calibration using expensive stream gage observations, which are often sparsely distributed over a flooding area and fail to estimate flood water depth with high spatial resolution [1] [9]. As an alternative, computer vision techniques have recently been applied for flood area detection and water level measurement with images of flooding "scenes" recorded by cameras [10] [11]. However, such models are often case-by-case efforts with "handcrafted" input features, leading to poor generalizability. Additionally, these techniques cannot easily assimilate large-scale datasets, such as crowdsourced images (e.g., photos from location-based social networks [LBSN]).

To our best knowledge, few studies have investigated flood disaster accurate assessment methods using LBSN based images for a specified area. For example, Meng et al. investigated the flood depth estimation based on state-of-the-art deep learning technique and publicly available camera images from the Internet and social media [12]. Specifically, human objects are detected and segmented from flooded images with Mask R-CNN [13] to infer the floodwater depth. However, this method with human objects as reference can hardly measure the flood more than the height of a human object (i.e., 2 meters).

To overcome the above limitations, this study proposes a real-time method of flood disaster assessment based on the real-time analysis and comparison from images between LBSN and Google Street View and uses buildings as reference objects. The method was evaluated with the real datasets related to flood disasters in Japan and US. Experiment results show that this method provides an accurate and real-time flood disaster assessment. Major contributions of this study include the following:

(1) The proposed method rectifies the raw image data by leveraging Uncalibrated Stereo Image Rectification [14] collected from LBSN and Google Street View and generates a pair of pre- and post-disaster images related to flood disaster for the same location. This enables to make the real-time comparison between a pair of images. **(2)** The values of flood

depth computed by comparing from image pair for flood disaster assessment demonstrate the superiority of our model with respect to accuracy and real-time performance.

2. METHODLOGY

2.1. Data Collection

This paper develops a workflow of flood depth estimation based on camera photos and machine learning techniques using buildings as reference. Within this framework, the flooded building photos (i.e., post-flood photos; Fig 1a) from news and social media (e.g., twitter) are first collected. Then, the exact location information of each building is extracted from geo-tags (i.e., coordinates) attached each photo or by natural language processing techniques through analyzing descriptive information (e.g., the text or the name) of the photos. Next, with the location information, we collected the pre-flood photo of the same building from Google street view (Fig 1b). Note the photos are selected based on the availability of location information, image quality, and obstacles (e.g., trees) in front of the buildings. In particular, only photos with location information are selected. Also, since the image quality will affect the feature extraction and feature matches between the post- and pre- flood photos, images with low quality (i.e overexposure or low resolution) are discarded. Moreover, if the building is partially blocked by obstacles, the building segmentation algorithm (i.e., Mask R-CNN; [13]) may fail to get the accurate mask of the building. Even with those restrictions, our method is still applicable for large-scale flood depth estimation as a large number of photos could be taken from different angles by the citizens during a disaster. Besides, to derive the depth of flood of an area, it is not necessary to calculate the flood depth of each building of that area, only a few buildings are sufficient.

2.2. Building Segmentation

Next, Mask R-CNN [13] is designed to segment the buildings in the photos. However, it is difficult to find a suitable dataset for training the Mask R-CNN [13]. The building images in most training datasets are remote sensing images instead of street view images. While this paper uses the Cityscape dataset [15], the training result is not ideal even though various hyperparameter (e.g., model training epochs and learning rate) tuning strategies are tested. Indeed, Cityscape dataset provides a great result in detecting certain objects (e.g., cars). However, it does not work well for building detection due to several reasons: (1) Many obstacles (trees, and billboards) can block the buildings; (2) The labelling quality of the building cannot be guaranteed, leading to the loss of some general building structure features (e.g., wall, corner, roof), and therefore some masks of building will not even be detected. To improve the model segmentation result, the authors relabeled 500 of those training data. While the result was improved, it is still not accurate enough with a small sample size. As such, for the experiment purpose, we manually labeled several building masks (e.g., Fig 2), which in turn were used for subsequent water depth inference steps (Section 2.3 – 2.4). Nevertheless, we will focus on optimizing the image segmentation for a further improved result in future work (Section 4).

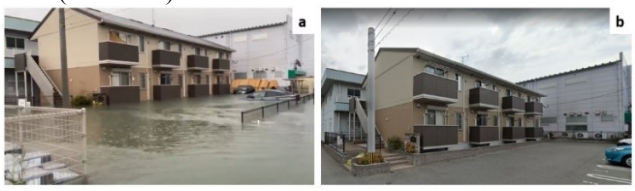

Fig 1. Flood Image pair in Japan: post-flood image in Japan collected from Twitter (a), and corresponding pre-flooded image collected from Google street view (b).

Fig 2. Manually labeled mask of building in Fig 1.

2.3. Uncalibrated Stereo Image Rectification

Due to the compared images were taken by different cameras at different positions and different angles, it is necessary to rectify them to the same angle to make them comparable. Specifically, SIFT algorithm [16] proposed by Lowe is used to collect feature points. Compared to other algorithms, SIFT is slow but more precise and it is scale-invariant. In other words, it is robust in changes like affine distortion or change in 3D viewpoints. After detecting the feature points and creating descriptor, we use FlannBasedMatcher [17] to compute matched features. However, there are lots of miss matches in the results. Correspondingly, Lowe proposed a filter algorithm using a distance ratio test to eliminate false matches [16]. If the distance ratio between two nearest matches of a considered key point is below a threshold, it is treated as a good match [18]. The higher the ratio is, the more matches the filter algorithm keeps, but with the lower accuracy. After filtering, we used those matched feature points to compute the fundamental matrix for image rectification [14]. With matched feature points and fundamental matrix as parameters for the Uncalibrated stereo image rectification algorithm [14], we get the two rectification homography matrix and put them in cv.initUndistortRectifyMap function to get two mapping matrices for each image. Each mapping matrix maps each image's original x or y coordinate to new coordinate. With those mapping matrices, we remap the images and obtain the rectification result, making two images under the same scale and thus comparable (Fig 3).

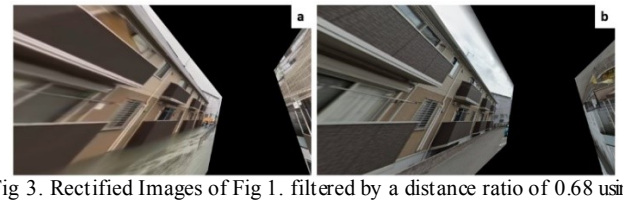

Fig 3. Rectified Images of Fig 1. filtered by a distance ratio of 0.68 using Lowe filter algorithm.

Due to different conditions that the pictures were taken, some images need to be preprocessed before moving to the SIFT step. For example, the flood images are often taken on cloudy day (Fig 4a), whereas Google street view images were usually taken full of sunshine (Fig 4b). This can highly influence the feature point detection and matching, leading to inaccurate results. Herein, we used a Light averaging method [19] by dividing the blocks into size of 16*16 and calculated the average light, to compensate for dark areas and darken bright areas, and then used Gauss Blur at the end to smooth the image. Fig 5a and 5b are the corresponding result after pre-processing images in Fig 4. The rectified results (Fig 5c and 5d) demonstrate that pre-processing works quite well. However, due to many differences mentioned above in the pre- and post-flood image pairs, the feature points match of some image pairs will still not be as perfect as two images taken by the same camera at nearly the same time even after preprocessing. As stated in data collect section, we will discard those problematic data.

Fig 4. Image pair with light difference: (a) The post-flood image of the building after flood collected from News, and (b) The corresponding pre-flood image collected from Google street view.

Since the extracted building masks (e.g., Fig 2) share the same location of the original images (e.g., Fig 1 and 4), applying the same remapping function on these masks can also derive the rectified mask of each building object (e.g., Fig 6) on the rectified images (e.g., Fig 5).

2.4. Building Height Extraction

The last step is to calculate the ratio of height of the building of pre- and post-flood images. Here, we used some vertical lines of the mask of the building (red lines in Fig 7) as the height of the building. Those vertical lines are usually formed by connecting the top and bottom corner of the building. To get the vertical lines, we first use canny edge detector ([20]; *cv.Canny*) to get the edges of rectified masks and then use Hough transform ([21]; *cv.HoughLinesP*) to get the line segments of the edges. Next, we can calculate the angle of each line segment and only keeps vertical lines as final result.

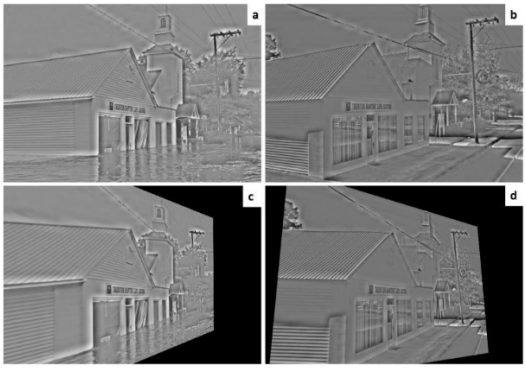

Fig 5. Corresponding images after preprocessing and then rectification: (a) The post-flood image after light average method, (b) The pre-flood image after light average method, and (c) and (d) are rectified (a) and (b).

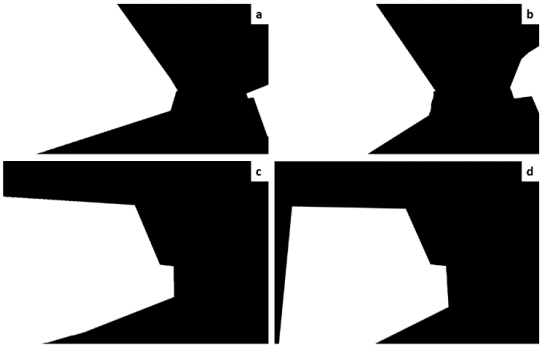

Fig 6. Corresponding mask after rectification. (a) and (b) are corresponding to Fig 1a and 1b, and (c) and (d) corresponding to rectified Fig 4a. and 4b.

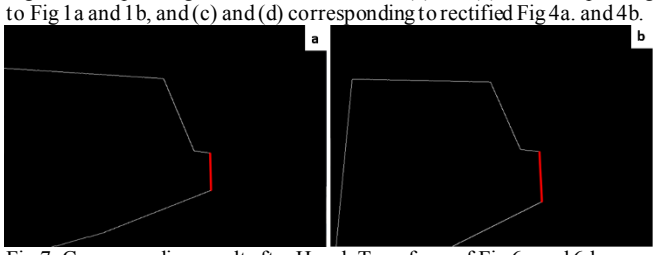

Fig 7. Corresponding result after Hough Transform of Fig 6c and 6d

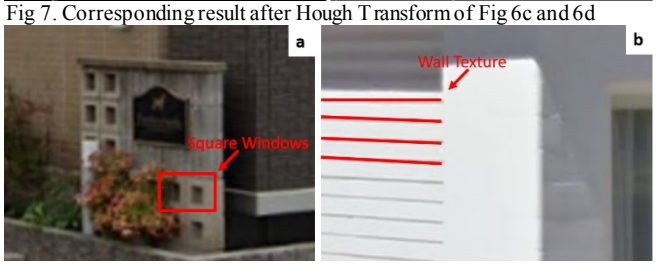

Fig 8. The reference to calculate the real above water ratio for the buildings in Fig 1 (left: square windows as reference) and Fig 4 (right: wall texture as reference)

3. EXPERIMENT RESULT

The experiment results in table 1 indicate that when using Lowe algorithm to get inliers, the best parameter to get the largest number of matches with the highest accuracy of the matches is 0.68 or 0.67. Therefore, we will make the average of those two rates to get the final ratio of the building above the floodwater.

TABLE 1. DETECTED LENGTH (LEN) OF LINES AND ABOVE WATER RATIOS

Fig	Distance ratio in filter	Start point	End point	Len	Ratio	Avg Ratio	Real Ratio
1a	0.68	(1198, 766)	(1239, 627)	144.92	0.825	0.845	0.839
1b		(1107, 801)	(1143, 629)	175.73			
1a	0.67	(1307, 636)	(1322, 797)	161.70	0.866		
1b		(1154, 639)	(1112, 821)	186.78			
4a	0.68	(512, 318)	(514, 410)	92.02	0.741	0.763	0.769~0.808
4b		(518, 316)	(524, 440)	124.14			
4a	0.67	(560, 298)	(593, 422)	128.32	0.786		
4b		(560, 295)	(605, 452)	163.32			

TABLE 2. ABOVE WATER RATIO TO FLOOD DEPTH

Fig	Avg ratio	Real ratio	Wall height (m)	Estimate height (m)	Real height (m)
1	0.845	0.839	6.17	0.955	0.995
4	0.763	0.769~0.808	3	0.710	0.577~0.692

The real ratio of the flooding (i.e., ground truth) is calculated through manually referencing key features of two buildings, for example, the square windows and billboard above the flood water (Fig 8a) or the texture of the wall (Fig 8b). Specifically, we calculated the height of the wall by using the 3D ruler in Google earth pro and calculated the flood depth according to ratio. However, for Fig 4, there is no 3D model of the building. As such, we had to use the altitude difference of the ground and the top of the wall in google street view, which may compromise the result accuracy. The result is shown in table 2.

4. CONCLUSION AND FUTURE WORK.

This work proposed a new method to estimate the flood depth, which provides a more precise result than many other existed researches. The pre- and post-flood buildings in images were collected from LBNSs and Google Street View and extracted for flood depth estimation. Due to the different positions and angles from which the building was mostly captured, these pair images need to be rectified by the Uncalibrated Stereo Image Rectification algorithm [14], which in turn requires using SIFT to get feature points, applying FlannBased Matcher to get feature points matches, and then filtering the matches with specific distance ratio [18]. Then, we can calculate the height ratio of the building above the flood water in the rectified image pair, and the value of flood depth was accurately derived (TABLE 2). However, this research has several limitations, which indicate future research directions. For example, as mentioned above, due to the shortcomings of the current training dataset (i.e., Cityscape [15]), we manually labeled the masks of buildings for this experiment. We will build a new dataset to help train the model automatically.

5. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant 1940091.

6. REFERENCES

[1] USGS. (2019). Flood Inundation Mapping (FIM) Program. Retrieved from https://www.usgs.gov/mission-areas/water-resources/science/flood-inundation-mapping-fim-program?qt-science_center_objects=0#qt-science_center_objects

[2] Hallegatte, S., Green, C., Nicholls, R. J., & Corfee-Morlot, J. (2013). Future flood losses in major coastal cities. *Nature climate change, 3*(9), 802.

[3] Merz, B., Kreibich, H., Schwarze, R., & Thieken, A. (2010). Review article" Assessment of economic flood damage". *Natural Hazards and Earth System Sciences, 10*(8), 1697-1724.

[4] Wagenaar, D., De Bruijn, K., Bouwer, L., & Moel, H. d. (2016). Uncertainty in flood damage estimates and its potential effect on investment decisions. *Natural Hazards and Earth System Sciences, 16*(1), 1-14.

[5] Kolesnikov, A., Zhai, X., & Beyer, L. (2019). Revisiting Self-Supervised Visual Representation Learning. *arXiv preprint arXiv:1901.09005*.

[6] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436.

[7] Liu, X., Sahli, H., Meng, Y., Huang, Q., & Lin, L. (2017). Flood Inundation Mapping from Optical Satellite Images Using Spatiotemporal Context Learning and Modest AdaBoost. Remote Sensing, 9(6), 617.

[8] Leandro, J., Chen, A., & Schumann, A. (2014). A 2D parallel diffusive wave model for floodplain inundation with variable time step (P-DWave). *Journal of Hydrology, 517*, 250-259.

[9] Oubennaceur, K., Chokmani, K., Nastev, M., Tanguy, M., & Raymond, S. (2018). Uncertainty analysis of a two-dimensional hydraulic model. *Water, 10*(3), 272.

[10] Wang, Y., Chen, A. S., Fu, G., Djordjević, S., Zhang, C., & Savić, D. A. (2018). An integrated framework for high-resolution urban flood modelling considering multiple information sources and urban features. *Environmental Modelling & Software, 107*, 85-95.

[11] Yu, J., & Hahn, H. (2010). Remote Detection and Monitoring of a Water Level Using Narrow Band Channel. *J. Inf. Sci. Eng., 26*(1), 71-82.

[12] Meng, Z., Peng, B., & Huang, Q. (2019). *Flood Depth Estimation from Web Images.* Paper presented at the Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Advances on Resilient and Intelligent Cities, Chicago, Illinois, USA.

[13] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). *Mask r-cnn.* Paper presented at the Proceedings of the IEEE international conference on computer vision.

[14] Hartley, Richard I. "Theory and practice of projective rectification." *International Journal of Computer Vision* 35.2 (1999): 115-127.

[15] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [Bibtex]

[16] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision, 60*(2), 91-110.

[17] Relja Arandjelovic. 2012. Three things everyone should know to improve object retrieval. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (CVPR '12). IEEE Computer Society, USA, 2911–2918.

[18] Feature matching with flann. OpenCV. (n.d.). from https://docs.opencv.org/3.4/d5/d6f/tutorial_feature_flann_matcher.html

[19] PENG , X.-bang, & JIANG , J.-guo. (2006). An Image Segmentation Thresholding Method Based on Luminance Proportion. *Electronic Technology & Information Science*, 10–12.
[20] Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence, (6), 679-698.
[21] Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, *15*(1), 11-15.