Building an Annotated Damage Image Database to Support AI-Assisted Hurricane Impact Analysis

Hao Nan Ou
Data Science and Engingeering Program
The CUNY City College
New York, United States
hou000@citymail.cuny.edu

Sun Ho Ro
Department of Civil and Environment Engineering
Rutgers University
New Jersey, United States
sunhoro@gmail.com

Jie Gong
Department of Civil and Environment Engineering
Rutgers University
New Jersey, United States
jg931@soe.rutgers.edu

Zhigang Zhu
Department of Computer Science
The CUNY City College and Graduate Center
New York, United States
zzhu@ccny.cuny.edu

***Abstract*—Building an annotated damage image database is the first step to support AI-assisted hurricane impact analysis. Up to now, annotated datasets for model training are insufficient at a local level despite abundant raw data that have been collected for decades. This paper provides a systematic approach for establishing an annotated hurricane-damaged building image database to support AI-assisted damage assessment and analysis. Optimal rectilinear images were generated from panoramic images collected from Hurricane Harvey, Texas 2017. Then, deep learning models, including Amazon Web Service (AWS) Rekognition and Mask R-CNN (Region Based Convolutional Neural Networks), were re-trained on the data to develop a pipeline for building detection and structural component extraction. A web-based dashboard was developed for building data management and processed image visualization along with detected structural components and their damage ratings. The proposed AI-assisted labeling tool and trained models can intelligently and rapidly assist potential users such as hazard researchers, practitioners, and government agencies on natural disaster damage management.**

Keywords—annotated image database, hurricane damage assessment, deep learning models, web-based dashboard

I. INTRODUCTION

The total economic loss from the extreme weather events in the United States between 1965 and 2019 was estimated to be $2 trillion, including loss of income and properties destructions [1]. About 64% of the total economic loss is due to hurricanes, resulting in an annual loss of nearly $10 billion [2]. Hurricane Harvey was a disastrous Category 4 hurricane that caused catastrophic flooding and more than 100 deaths in Texas and Louisiana in August 2017 and NOAA estimated that Hurricane Harvey had cost a total economic loss of $125 billion [3]. Even though the yearly rate of natural disaster events shows a nominal growth, the population growth in a hurricane vulnerable zone is rising persistently [4, 5].

Research on hurricane damage prediction, mitigation, and recovery is critical for hurricane damage cycle management. Consequently, many researchers agree that vital information needs to be obtained to improve the buildings' safety and reliability based on the data of damaged and undamaged buildings and their components [6-8]. Conventionally, post-natural disaster damage assessment is conducted in a form of a manual survey by reconnaissance teams. However, given that most natural disaster causes damage to thousands of buildings across a broad region, high cost and long-term data miss were inevitable, especially when the survey should be ideally done within 72 hours of the event to avoid any damage alteration the recovery efforts.

To acquire post-natural disaster damage data on a broader disaster zone with enhancing data quality and quantity, many studies implemented various remote sensing technologies: photogrammetry [9], Unmanned Aerial Vehicle (UAV) [10-14], satellite images [15-17], and Light Detection and Ranging (LiDAR) [18-23]. Despite the viable acquisition of large spatial data, much of data analysis, cleaning, and structuring is still done manually, which is generally time-consuming and expensive. Analyzing these data sets has become increasingly challenging because steps to automate and accelerate data analysis are not at the same pace as collecting data [24]. Additionally, the raw images themselves cannot provide much helpful information to secondary researchers and practitioners even with a significant volume of disaster data that has been collected and achieved for decades.

The majority of building damage assessment relies on damage classification based on damage features from monotemporal data or change detection from multitemporal data [25]. However, component-level building damage studies are intermittent [26-28] or limited on simple component detection that cannot be used to describe the entire building damage attribute [29]. Recently, high-resolution panoramic images from ground-based mobile reconnaissance teams or map service providers have been increasingly used for damage analysis due to their apparent advantages in storing a sequence of 360° images, which enables a multi-angle view along the street [30, 31]. Subsequently, recent studies show various panoramic image processing technics for efficient pre- and post-event building comparison not limited to but including A procedure to extract localized building images from the panoramic images [32], 3D estimation by triangulation [33], and employment of deep learning algorithms for automated structural component detection [34-37].

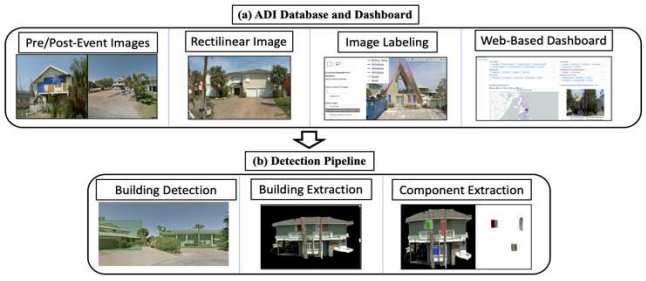

Fig. 1. System workflow.

Even though AI can potentially be used to analyze these data sets automatically, labeled data sets for model training are insufficient at a local level. Therefore, we aim to design an AI assisted labeling tool that can intelligently assist users like hazard researchers, practitioners, and government agencies to create labeled images and point clouds of damaged buildings according to established damage assessment protocols and databases. A system workflow is shown in Fig. 1. The proposed system overcomes these challenges by providing an AI-assisted framework for post-hurricane image data management, with the following key features: (1) An annotated damage image (ADI) database and a web-based dashboard that allow users to visualize damaged building information, damage map, structural component damage rating, and its labeling (Component (a) in Fig. 1). (2) An automatic building and structural component detection pipeline including for building detection, building extraction, and structural component extraction (Component (b) in Fig. 1). (3) A simple building location estimation method using building addresses to process raw images to optimal rectilinear images of the specific buildings from panoramic images (included in Component (a) in Fig. 1).

II. ADI Database and Dashboard

A. Pre and Post Event Data

Two major datasets were used in our Annotated Damage Image (ADI) database. The first data comes from panoramic images of post-Hurricane Harvey, 2017, at Port Aransas and Rockport, Texas, three days after the event. Smart and Sustainable Cities Lab from Rutgers University, accompanied by graduate students from the University of Texas at Austin and Princeton University, utilized a scanning vehicle equipped with a mobile LiDAR scanner and GPS-enabled 360-degree panoramic camera to acquire residential building damage data [38]. Over 60,000 panoramic images were taken during the exploration period. The panoramic images are compiled and each of them is geo-coded to link with pre-hurricane Google Map in the Rutgers-Harvey Portal [39].

The second data set comes from the manual damage assessment protocol results of Hurricane Harvey damages on 533 residential buildings located in Rockport, Texas [40]. The manual damage assessment data includes the damage rating of each structural component. However, the original images used for [40] and the scanning vehicle's panoramic images are not exactly corresponding to each other. Thus, it is necessary to filter out the manual damage assessment data that include buildings overlapping with the panoramic images.

Images from the Google Street View (GSV) are extracted for obtaining pre-event data; Fig. 2 shows an example of pre- and post-event images. The Google Maps Geocoding API was used to extract the latitude and longitude of the 553 buildings and pre-event images from GSV. Since the panoramic images do not contain specific GPS coordinates for each building, it is necessary to translate each building's address into geographic coordinates. An interactive geospatial map, generated using Folium Map package [41], a Python library for visualizing geospatial data, and the scanning vehicle's trajectory, was imported to separate buildings that are not overlapped in the first batch of image data (Fig. 3). This allowed the filtering of 152 buildings that were not covered in the scanning and were left with 401 buildings. The building coordinates in [40] and the scanning vehicle's camera coordinates were taken to calculate the distance between them to identify the nearest panoramic image to the target building.

B. Localizing Buildings in Rectilinear Building Images

Optimal rectilinear images were generated from the panoramic images for a target building extraction (single image with one or at most three buildings). Manually comparing the target buildings in panoramic images to GSV images is time-consuming and prone to human error since the panoramic images cover 360° and include numerous buildings and debris. By generating optimal rectilinear images, we can ensure that the target building is closest to the center of the processed image. Since the image data contain longitude, latitude, and direction angle, we used the address of each building to obtain the geographic information of each building, then used the Google Maps API to obtain the coordinates of each building. In order to match the building with the nearest panoramic image, the Haversine formula [42] has been used to calculate the distance between the building and the panoramic image shooting position.

The rectilinear image generation algorithm requires three parameter inputs: a panoramic image, its heading angle and its projection angle (bearing angle). The heading angle refers to the heading of the camera where the image is taken to the north. The projection angle (bearing angle) is measured between the center of the rectilinear image respecting to the true north and is computed using the geolocations between the heading of the camera and the direction of the target building [43]. After calculating the new heading angle and projection angle, the rectilinear image can be extracted from the image. We utilized the 401 images as the test data to be used in the deep learning model.

Among randomly selected sample images, the image taken with the closest distance is set to be the front view of the building, and we denote this image as PN-Center to indicate that it captured the shortest distance between the camera and the target building. Since all image data have unique serial numbers, each previous and next from PN-Center is set to be PN-Left and PN-Right (Fig. 4). Then, based on the approach proposed by [7], the following steps are used to generate rectilinear image: (1) Obtain the heading angle of the panoramic image; (2) Obtain the camera location (latitude and longitude) of the images; (3) Compute the direction of projection using a bearing angle formula by two geographic coordination between the camera and the target building. From Fig 4, we generated three rectilinear images RT-left, RT-center, RT-right (Fig. 5).

Fig. 2. Pre- and post-event images: extracted from Google Street View and the panorama image database, respectively.

Fig. 3. Scanning vehicle trajectory in blue/red lines and building locations in [40] represented in blue icons.

Fig. 4. (a) PN-Left; (b) PN-Center; (c) PN-Right.

Fig. 5. (a) RT-Left; (b) RT-Center; (c) RT-Right; (d) Manually captured.

C. *Image Labeling*

Labeled image data set is essential to train the automatic building detection algorithm model. We used 553 pre-event building images from GSV (Google Street View) via Google API, and manually labeled each structural component. Two types of image labeling tools were used in this study: (1) Labelme, a graphical image annotation tool written in [44]; (2) Amazon Rekognition Custom Label [50], which allows users to add, change, or remove labels from a dataset.

D. *The Web-Based Dashboard*

We created a web-based dashboard for storing the processed data and its visualization (https://haonan15.herokuapp.com). The dashboard allows users to label the images, access a user-interactive map, and visualize labeled structural components and their damage rating. The first interface, a building map webpage (Fig. 6a), allows users to look for a specific group of buildings by applying filters (e.g., price range, stories type, siding type, area range, elevated and non-elevated building, and roof type). The building image and information are shown as users click the specific building marks on the interactive map. The second interface, the image uploader (Fig. 6b), which allows users to upload images to MongoDB with necessary building information (e.g., building address, latitude, longitude, and comments). Lastly, the third interface, the image annotator (Fig. 6c), allows users to label the building components (e.g., doors, windows, roofs, walls), where the coordinate of each label shape can be saved to MongoDB.

III. Building and Component Detection Pipeline

The building and component detection pipeline includes four steps: building detection, building extraction, and structure component detection (Fig. 7). We will discuss each of the steps in the following subsections.

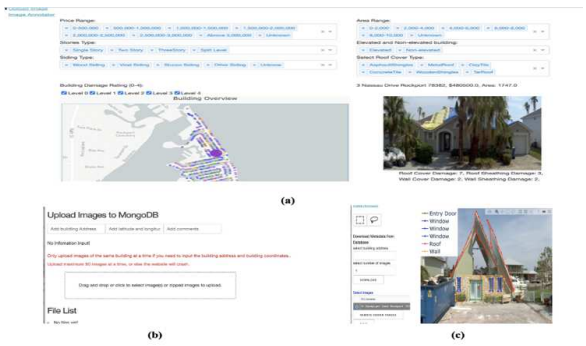

Fig. 6. (a) Building map page; (b) Uploader page; (c) Annotator page.

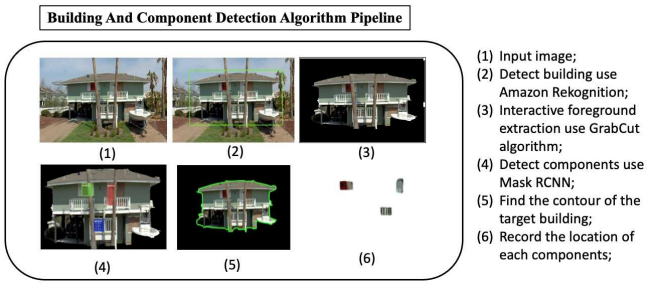

Fig. 7. Building and component detection algorithm pipeline

Fig. 8. The bounding box of detected buildings.

A. *Building Detection*

After labeling structural components of the 553 pre-event buildings, Amazon Rekognition's pre-trained dataset was used to generate a bounding box of detected buildings. We split the images to 444 training images and 109 test images to start the training with this model. Based on the model performance on the test image (Fig. 8), a simple piece of Python code was written to filter out bounding boxes with a confidence level of 0.8 or higher to ensure accuracy. When two or more buildings were detected, the building located closer to the center of the image was set to be the target image.

B. *Building Extraction*

The GrabCut algorithm [45] was implemented to perform interactive foreground extraction. The algorithm works as the following. Initially, the bounding box containing the objects in the image is defined. All the pixels in the area outside the bounding box are regarded as background pixels, while everything inside the bounding box was identified as mostly including the foreground region, the building in this case. For each pixel outside the bounding box, initialize the pixel label to 0, which is the background pixel; for each pixel in the bounding box, initialize the pixel label to 1, that is, the pixel that may be the target. According to the pixels marked as 0 or 1, Gaussian mixture model (GMM) of foreground and background are established respectively. GMM is used to classify background and foreground, in which undefined pixels are marked as possible foreground or background. The k-means algorithm cluster the pixels belonging to the foreground and background into k classes respectively and establish k Gaussian components in GMM. The weight of each Gaussian component can be determined by the ratio of the number of pixels belonging to the Gaussian component to the total number of pixels. Each pixel in the image is considered to be connected with the surrounding pixels through the virtual edge, and each edge has a probability of belonging to the foreground or background, which is based on its color similarity with the surrounding pixels. Each pixel (the node in the algorithm) will be connected to the previous foreground or background node. After the nodes are connected (possibly with the background or foreground), if the edges between the nodes belong to different terminals (that is, one node belongs to the foreground and one node

belongs to the background), the edges between them will be cut off, which will separate the parts of the image. In the end, the algorithm removes the background and extracts the house in the image by segmenting the foreground region iteratively (Fig. 9).

C. Structural Component Extraction

Major structural components—windows, doors, walls, and roofs—were considered in this study. At first, we wanted to use deep learning model to detect the various parts of the house, such as the roof, walls, doors and so on, to record the damage level of objects, so as to facilitate future research. However, it is difficult to detect the roofs and walls separately due to the irregular surfaces and textures. Thus, windows, garage doors, and entrance doors were segmented in the first round, and the rest are automatically defined as the roofs and walls combined. Mask R-CNN [46], a deep neural network model designed to solve instance segmentation problems [47], was adopted for the structural component extraction. In this model, a bounding box and a segmentation template are generated for each object detected in the image. Three types of objects were primarily detected for this study: (1) doors, (2) windows, (3) garage doors. Initially, a trained model weight from [48], which was trained on a 20,000 image dataset, was used to infer the positions of windows, doors, and garage doors. Fig. 10 shows a result of the Mask R-CNN deep learning model.

When each object is detected, a separate directory is created, and the object is saved as a separate image in a folder, such as the windows folder, the entrance door folder, and the garage door folder. Then, the objects' contour features are defined, and the X and Y coordinates of each object's image contour are calculated (Fig. 11). By having the coordinates of the structural components, we can obtain the coordinates of the roof and the wall by excluding the coordinates of the entrance door, the windows, and the garage door.

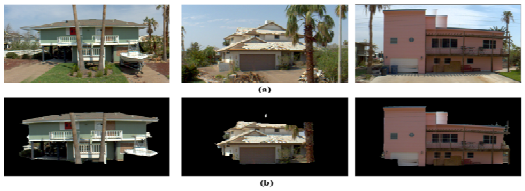

Fig. 9. (a) Original image; (b) Building extraction result.

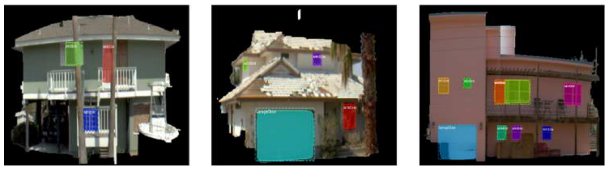

Fig. 10. Detected structural components from the Mask R-CNN model.

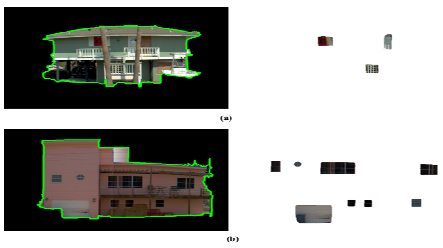

Fig. 11. The contour of the targeted building shown in the green boundary on the left, and detected windows, garage doors, and entrance doors are shown on the right.

IV. Results and Discussions

We want to automate the process of getting the positions of the windows, doors, and other components by just inputting the image and getting the coordinates automatically. If we can get the location of these components by using deep learning models and then we proceed to segment the image, it can speed up the researchers' analysis of the degree of damage of each component, and figure out which components can better resist the hurricane in the future. As disaster image data is increasing continuously, this will significantly improve cost efficiency and the overall efficiency of the analysis if we can achieve automation in image data processing. Two deep learning models, Rekognition and Mask-RCNN, were implemented in this paper to detect buildings and extract their structural components. Rekognition was used to detect the building location from the panoramic image. 533 pre-event images from Google Street View were split into 444 train images and 109 test images. During the evaluation, the model made predictions using the test dataset. The returned metrics were F1 score 0.893, precision 0.906, and recall 0.88. All three parameters achieved over 0.85, which indicates a good result. The trained model was used to run inference on the panoramic image dataset. The resulting bounding box coordinates were generated on each image for each building in the JSON files. After extracting the bounding box coordinates, two object-detection deep learning models, Mask R-CNN [48] and Detectron2 [49], were compared for structural component detection and image segmentation. 405 pre-event Google Street View images were split into 356 train images and 49 test images.

To use the Detectron2 model, we trained using "R50-FPN, Cityscapes" as the initial parameter. This is the baseline model trained with Detectron2 and is initialized from COCO pre-training and trained on Cityscapes fine annotations. The trained model resulted in a 0.32 mean average precision (mAP), where mAP is used to evaluate the performance of object-recognition deep learning models, and AP is used to evaluate the accuracy of an object detector on each image. This is obtained by calculating the average precision of all known annotations in all images in the dataset.

For the Mask-RCNN model, it was a step-by-step detection that includes anchor sorting and filtering, bounding box refinement, mask generation, layer activations, weight histograms, logging to the tensor board, and composing the different pieces into the output [46]. The pre-trained model [48], was used to segment the structural components. The same 49 test images were used to evaluate the model, and a 0.52 mAP was returned.

We therefore selected the Mask R-CNN model after comparing the two deep learning models mainly because of the higher mAP. The manually labeled GSV images are very limited and affected a low mAP of 0.32 for Detectron2. In comparison, Mask R-CNN used a pre-trained 20,000 image dataset, and a high mAP of 0.52 is a logical conclusion. Fig. 12 shows the output of inferencing the same post-event image using Detectron2 and Mask R-CNN. However, the result does not necessarily indicate a "better algorithm" as all deep learning model results are heavily dependent on the quality and quantity of the training dataset. As shown in Table I, the mAP of Detectron2 for detecting door and window instances is much lower than Mask-RCNN.

However, we find that Mask-RCNN cannot detect door instances well in this validation image data set, which is why the average accuracy of all instances in the image data is just over 50%. Therefore, we need to collect more data to improve the model's performance and consider relabeling the images.

The Mask-RCNN model used was based on Feature Pyramid Network (FPN) and ResNet101 backbone [46]. It was chosen for its fast training and testing speed, as well as the flexibility and accuracy of the framework. The model can run within about 200 ms per frame on a single GPU. In comparison, training with ResNet-50-FPN on the COCO trainval135k took 32 hours on a single 8-GPU machine, with 0.72 s per 16 image mini-batch, and 44 hours with ResNet-101-FPN [47]. Without tricks, Mask R-CNN surpassed the winner of the 2016 COCO keypoint competition, and at the same time ran at 5 fps [47]. If ResNet-101-FPN model was used for inference, it would take 195ms per image on a Nvidia Tesla M40 GPU machine [47]. We did inference on 49 images without the GPU and saved the locations of all the components detected in the images, the total running time was 3 m and 8 s, which was all most 3.84 s per image.

V. Conclusion, Limitations and Future Work

Automated post-natural disaster image labeling is vital for hazard research as it can be time-consuming and tough to do it manually. This paper provides a method to accelerate automated image labeling and structural component extraction. Three major datasets from post-Hurricane Harvey —60,000 panoramic images, manual damage assessment results, and pre-event Google Street View—were used for training and test datasets. Two deep learning models, Rekognition and Mask R-CNN, were implemented to detect and extract the building and its structural components. Finally, the results were visualized in the user interactive web-based dashboard. The results show that each structural component was successfully extracted from the image and correlated with its damage rating from the manual damage assessment result. Here are a few directions for future research.

(1) This paper focused on the use of panoramic images and Google Street View to find the best view of the particular buildings. As buildings shown on the panoramic images cannot be fully geo-referenced, it will be beneficial to develop a method for live building detection, geo-referencing, and collecting the north angle of the shooting position.

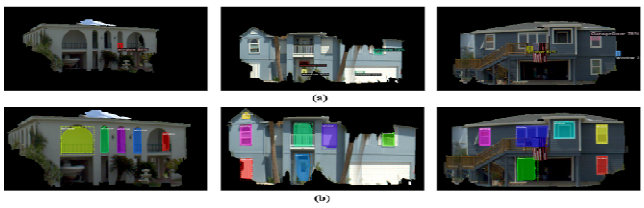

Fig. 12. Output of inferencing post-event image using (a) Detectron2 and (b) Mask R-CNN.

TABLE I. Mean Average Precision of each Test Instances

Category	*Detectron2*	*Mask R-CNN*
Window	0.29	0.6
Door	0.14	0.38
Garage Door	0.53	0.65

Fig. 13. (a) and (b) show buildings with temporal roof tarp and (c) and (d) show buildings with a destroyed roof.

(2) It is not feasible to entirely deliberate the complex and irregular damage texture while manually labeling the initial hurricane-damaged building dataset. For example, as shown in Fig. 13, about 30% of the dataset has severely damaged roofs. Some buildings installed temporal roof tarp, resulting in distinct color contrast to the original roof tiles. Some buildings have completely destroyed roofs, resulting in inadequate pixel occupation. These factors critically interfere with model training. Therefore, a significant number of precisely segmented and labeled data sets for damaged structural components are expected for high model performance and their damage rating detection.

(3) The trained model from this paper only detects major structural components: doors, windows, walls, and roofs. Further training with additional data will enable detailed structural component detection (e.g., handrails, columns, and pipes), patio door and entry doors differentiation, building level classification, and roof type detection. Additionally, while this paper correlates the structural component labeling with the manual damage assessment result, the output data can be used to develop a model for automatic analysis and damage rating assessment of post-hurricane damaged structures.

(4) Based on our dashboard and image database, potential users such as real estate or construction companies can develop applications to analyze suitable materials and designs for building houses that minimize the damages caused by future hurricanes from specific areas. By knowing the damage degree of each housing component, users can further analyze the reasons of why some specific parts of a house are seriously damaged, and whether the geographical location, such as being nearby the seaside, has an impact on the degree of damages on the house. Users who want to purchase a house can evaluate the selected house to see if the current house value is overvalued or undervalued. As more and more data can be uploaded through the database,

Acknowledgment

The work is supported by the National Science Foundation through Awards #1827505 and #1737533, the Air Force Office for Scientific Research (Award #FA9550-21-1-0082), a CCNY CEN Course Innovation Grant from Moxie Foundation, and the Intelligence Community Center of Academic Excellence (IC CAE) at Rutgers (Awards #HHM402-19-1-0003 and #HHM402-18-1-0007).

References

[1] EM-DAT: The international disaster database, Centre for Research on the Epidemiology of Disasters - CRED. [Online]. Available: https://www.emdat.be/database

[2] R. A. Pielke Jr, J. Gratz, C. W. Landsea, D. Collins, M. A. Saunders, and R. Musulin, "Normalized hurricane damage in the United States: 1900–2005," Nat. Hazards Rev., vol. 9, no. 1, pp. 29–42, 2008.

[3] D. J. Frame, M. F. Wehner, I. Noy, and S. M. Rosier, "The economic costs of Hurricane Harvey attributable to climate change," Clim. Change, vol. 160, no. 2, pp. 271–281, 2020.

[4] P. J. Klotzbach, S. G. Bowen, R. Pielke, and M. Bell, "Continental U.S. Hurricane Landfall Frequency and Associated Damage:

Observations and Future Risks," Bulletin of the American Meteorological Society, vol. 99, no. 7, pp. 1359-1376, 2018. doi:10.1175/BAMS-D-17-0184.1.

[5] Corelogic.com, "2019 Storm Surge Report," [Online]. Available: https://www.corelogic.com/downloadable-docs/storm-surge-report_052919-screen.pdf

[6] D. P. Eisenman, K. M. Cordasco, S. Asch, J. F. Golden, and D. Glik, "Disaster planning and risk communication with vulnerable communities: lessons from Hurricane Katrina," Am. J. Public Health, vol. 97 Suppl 1, no. Supplement_1, pp. S109-15, 2007.

[7] A. Lenjani et al., "Towards fully automated post-event data collection and analysis: Pre-event and post-event information fusion," Eng. Struct., vol. 208, no. 109884, p. 109884, 2020.

[8] V. Silva et al., "Current challenges and future trends in analytical fragility and vulnerability modeling," Earthq. Spectra, vol. 35, no. 4, pp. 1927–1952, 2019.

[9] A. Alipour et al., "Steer Huriacane Michael: Preliminary virtual assessement team (P-VAT) report," 2018. [Online]. Available: https://doi.org/10.17603/DS2RH71

[10] S. M. Adams and C. J. Friedland, "A Survey of Unmanned Aerial Vehicle (UAV) usage for imagery collection in disaster research and management", Proc. Ninth Int. Workshop on Remote Sensing for Disaster Response, 2011.

[11] S.S. Congress, A.J. Puppala, A. Banerjee, N.H. Jafari and U.D. Patil, "Use of Unmanned Aerial Photogrammertry for Monitoring Low-Volume Roads after Hurrican Harvey," in 12th International Conference on Low-Volume Roads, 2019.

[12] L. Hashemi-Beni, J. Jones, G. Thompson, C. Johnson, and A. Gebrehiwot, "Challenges and opportunities for UAV-based digital elevation model generation for flood-risk management: A case of Princeville, North Carolina," Sensors (Basel), 18(11), p. 3843, 2018.

[13] M. Kakooei and Y. Baleghi, "Fusion of satellite, aircraft, and UAV data for automatic disaster damage assessment," Int. J. Remote Sens., vol. 38, no. 8–10, pp. 2511–2534, 2017.

[14] J. Yeom, Y. Han, A. Chang, and J. Jung, "Hurricane building damage assessment using post-disaster UAV data," in IGARSS 2019 - 2019 IEEE Int. Geoscience and Remote Sensing Symposium, 2019.

[15] J. Doshi, S. Basu, and G. Pang, "From satellite imagery to disaster insights," arXiv [cs.CY], 2018.

[16] T. W. Gillespie, J. Chu, E. Frankenberg, and D. Thomas, "Assessment and prediction of natural hazards from satellite imagery," Prog. Phys. Geogr., vol. 31, no. 5, pp. 459–470, 2007.

[17] N. Said et al., "Natural disasters detection in social media and satellite imagery: a survey," Multimed. Tools Appl., vol. 78, no. 22, pp. 31267–31302, 2019.

[18] I. A. Garcia, M. J. Starek, and T. Chu, "Mobile and airborne lidar scanning of beach elevation change due to hurricane Harvey," in 2020 IEEE Int. Geoscience and Remote Sensing Symposium, 2020.

[19] J. Gong and A. Maher, "Use of mobile lidar data to assess hurricane damage and visualize community vulnerability," Transp. Res. Rec., vol. 2459, no. 1, pp. 119–126, 2014.

[20] M. J. Olsen and R. Kayen, "Post-earthquake and tsunami 3D laser scanning forensic investigations," in Forensic Engineering 2012, 2012.

[21] D. O. Prevatt et al., "Building damage observations and EF classifications from the Tuscaloosa, AL, and Joplin, MO, tornadoes," in Structures Congress 2012, 2012.

[22] D. N. Whiteman and Nasa Reports Server, Raman lidar measurements of water vapor and cirrus clouds during the passage of hurricane Bonnie. Bibliogov, 2013.

[23] S. C. Yim, M. J. Olsen, K. F. Cheung, and M. Azadbakht, "Tsunami modeling, fluid load simulation, and validation using geospatial field data," J. Struct. Eng. (N. Y.), vol. 140, no. 8, p. A4014012, 2014.

[24] A. Lenjani, C. M. Yeum, S. Dyke, I. Bilionis, "Automated building image extraction from 360° panoramas for postdisaster evaluation," Comput.-aided civ. infrastruct. eng., 35(3), pp. 241–257, 2020.

[25] Z. Zhou, "Computer Vision-based Assessment of Coastal Building Structures during Huricane Events," Rutgers, New Jersey, USA, 2017.

[26] A. Hatzikyriakou, N. Lin, J. Gong, S. Xian, X. Hu, and A. Kennedy, "Component-based vulnerability analysis for residential structures subjected to storm surge impact from hurricane sandy," Nat. Hazards Rev., vol. 17, no. 1, p. 05015005, 2016.

[27] S. Xian, N. Lin, and A. Hatzikyriakou, "Storm surge damage to residential areas: a quantitative analysis for Hurricane Sandy in comparison with FEMA flood map," Nat. Hazards (Dordr.), vol. 79, no. 3, pp. 1867–1888, 2015.

[28] Z. Zhou and J. Gong, "Automated Analysis of Mobile LiDAR Data for Component-Level Damage Assessment of Building Structures during Large Coastal Storm Events: Automated analysis of mobile LiDAR data," Comput.-aided civ. infrastruct. eng., vol. 33, no. 5, pp. 373–392, 2018.

[29] S. Van Ackere, J. Beullens, W. Vanneuville, A. De Wulf, and P. De Maeyer, "FLIAT, an object-relational GIS tool for flood impact assessment in Flanders, Belgium," WATER, vol. 11, no. 4, 2019.

[30] W. Zhai and Z.-R. Peng, "Damage assessment using Google Street View: Evidence from Hurricane Michael in Mexico Beach, Florida," Appl. Geogr., vol. 123, no. 102252, p. 102252, 2020.

[31] W. Zhang, C. Witharana, W. Li, C. Zhang, X. Li, and J. Parent, "Using deep learning to identify utility poles with crossarms and estimate their locations from Google Street View images," Sensors (Basel), vol. 18, no. 8, p. 2484, 2018.

[32] A. Lenjani, C. M. Yeum, S. Dyke, and I. Bilionis, "Automated building image extraction from 360° panoramas for postdisaster evaluation," Comput.-aided civ. infrastruct. eng., vol. 35, no. 3, pp. 241–257, 2020.

[33] Z. Zhu, K. D. Rajasekar, E. M. Riseman, and A. R. Hanson, "Panoramic virtual stereo vision of cooperative mobile robots for localizing 3D moving objects," in Proceedings IEEE Workshop on Omnidirectional Vision (Cat. No.PR00704), 2002.

[34] C. Cheng, A. H. Behzadan, and A. Noshadravan, "Deep learning for post - hurricane aerial damage assessment of buildings," Comput.-aided civ. infrastruct. eng., no. mice.12658, 2021.

[35] T. Czerniawski and F. Leite, "Automated segmentation of RGB-D images into a comprehensive set of building components using deep learning," Adv. Eng. Inform., vol. 45, no. 101131, p. 101131, 2020.

[36] H. Liu, J. Zhang, J. Zhu, and S. C. H. Hoi, "DeepFacade: A deep learning approach to facade parsing," in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017.

[37] D. Rueda-Plata, D. González, A. B. Acevedo, J. C. Duque, and R. Ramos-Pollán, "Use of deep learning models in street-level images to classify one-story unreinforced masonry buildings based on roof diaphragms," Build. Environ., vol. 189, no. 107517, p. 107517, 2021.

[38] "Damage Control," Rutgers.edu. [Online]. Available: https://ucmweb.rutgers.edu/magazine/1419archive/insights/damage-control.html. [Accessed: 29-April-2021].

[39] M. Guo and J. Gong, Rutger-Iris Hurricane Harvey Portal. [Online]. http://rutgersiris.com/harveymap/leaflet/debug/vector/HarveyPortal-with-full-data.html

[40] S. Wengrowski, "Comprehensive damage assessment and analysis of damage mechanisms from Hurricane Harvey." Rutgers University - School of Graduate Studies, 2019.

[41] folium, Python Data, Leaflet.js Maps, Github repository, Github,[https://github.com/python-visualization/folium], 2013

[42] K. Gade, "A non-singular horizontal position representation," J. Navig., vol. 63, no. 3, pp. 395–417, 2010.

[43] D. Ellis, "Calculating the bearing between two geospatial coordinates," Towards Data Science, 19-May-2020. [Online]. Available: https://towardsdatascience.com/calculating-the-bearing-between-two-geospatial-coordinates-66203f57e4b4.

[44] K. Wada, labelme: Image Polygonal Annotation with Python, Github repositor, Github, [https://github.com/wkentaro/labelme], 2016.

[45] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut: Interactive foreground extraction using iterated graph cuts," ACM Trans. Graph., vol. 23, no. 3, pp. 309–314, 2004.

[46] W. Abdulla, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow", Github repository, Github, [https://github.com/matterport/Mask RCNN], 2017.

[47] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in 2017 IEEE Int. Conference on Computer Vision (ICCV), 2017.

[48] S. Tasneeyapant, "Mask_RCNN_ViewAnalysis," Github repository, [https://github.com/songwongtp/Mask_RCNN_ViewAnalysis], 2018.

[49] Y. Wu, A. Kirillov, F. Massa, W. Lo, Y. and R. Girshick, "Detectron2," Github repository, [https://github.com/facebookresearch/detectron2], 2019.

[50] Amazon.com. [Online]. Available: https://aws.amazon.com/rekognition/custom-labels-features/. [Accessed: October 19, 2021.