Real-time Image Processing in A Multi-UAV System for Structural Surveillance Through IoT Platform

Jose Falcon, Mehrube Mehrubeoglu[*], Pablo Rangel

Texas A&M University – Corpus Christi, Electrical Engineering Program, Department of Engineering, Corpus Christi, Texas, USA 78412-5797

ABSTRACT

The use of location and instruction markers for multi-path planning enhancement in any set of unmanned aerial systems' tasks is crucial to the coordination and effectiveness of the individual unmanned aerial vehicles (UAVs). This research implements OpenCV algorithms that allow multiple UAVs to use ArUco markers to receive data related to location and instruction for the purposes of multi-path planning. OpenCV algorithms are utilized to develop vision-based solutions that will enhance the real-time capabilities of the UAVs. The final goal for the multi-drone system entails inspecting and surveying objects for structural damage and applying the developed image processing algorithms to collected images to determine the significance of damage. This project utilizes OpenCV and Python libraries for multi-drone pathway planning by collecting, transmitting, and displaying real-world industrially valuable data over the network infrastructure as an application of Internet of Things (IoT).

Keywords: unmanned aerial vehicle, UAV, OpenCV, ArUco marker, Internet of Things, real-time image processing

1. INTRODUCTION

In recent years, the availability of and ease of operation with unmanned aerial systems have grown. With this growth, automated applications for path-planning designed for single and multi-unmanned aerial vehicle systems through image processing have also increased. There have been many papers published on methods with which unmanned aerial vehicles (UAVs) can be operated and their path planned[1-3]. Most UAVs come equipped with a camera setup that can be used with OpenCV for image processing[4,5]. The application described in this paper focuses on an automated structural damage inspection system based on the camera/vision sensors of the unmanned aerial vehicle.

This project uses two types of unmanned aerial vehicles (DJI Ryze Tello and DJI Tello EDU, SZ DJI Technology Co., Ltd.) equipped with camera/vision sensors for real-time image processing for structural damage detection. The hardware and functionality of these two drones are very similar, however the Tello EDU black has an updated firmware with improved capabilities for future swarm implementations. Visual markers (ArUco) are used as instructional tags for the UAV system[6,7]. These markers are used to support UAV path-planning to guide the UAV towards the inspection area. UAV's intermediate structural damage assessments based on the structural damage characterization, such as magnitude, is then displayed and recorded.

In this work the structural damage assessments are displayed via the network infrastructure as an application of IoT. IoT's involvement in robotic applications include improved navigation, device surveillance, and UAV operations[2]. Through IoT-based UAV, the user is provided with valuable industrial data remotely, based on the imagery and characterization of damage. The main goal of this project is to provide an easy-to-deploy UAV and tag system for measuring safety of structures through damage inspection and concrete crack characterization. IoT tool is used to share the UAV imagery and results remotely for further analysis.

The rest of this paper is organized as follows: Section 2 summarizes additional related work, and research objectives. Section 3 presents the methodology, including marker detection, UAV navigation, and real-time image processing for defect detection. Implementation is described in Section 4. Results and Analysis are discussed in Section 5. Conclusions are provided in Section 6.

[*]Email: ruby.mehrubeoglu@tamucc.edu; Phone: (361) 825-3378

2. RELATED WORK

There have been many applications for marker identification, concrete crack isolation and IoT implementations for UAV systems[8-10], the three areas of focus of this paper. Marker identification has been linked to increased accuracy and efficiency of UAV landing and tracking[11-15,7]. Path planning is a key component in the development of autonomous mobile robots. Path planning is combined with motion planning and decision making to enable higher levels of autonomy on a mobile robot, and visual markers, such as ArUco, assist motion instructions. Vision-based concrete crack detection methods include image processing and deep learning algorithms[16-21]. IoT used with UAV systems provides remote and real-time access to data and systems through the Internet in a variety of applications from surveillance of systems[22], environmental monitoring[23], precision agriculture[24,25], aerial semantic segmentation[26,27], rehabilitation[28,29] to communication and navigation[1,30]. IoT applications represent a growing research field and market.

This research aims to combine these three areas of UAV application, namely, marker detection for real-time path planning, image processing for concrete defect detection, and IoT implementation for UAV image display and access. This project serves towards the final goal of a deployable self-detecting multi-UAV system for remote inspection of structural integrity of tall concrete structures through crack detection and characterization.

3. METHODOLOGY

In this section the UAV system and how navigation is achieved using ArUco markers is explained. Both marker detection and surface defect or crack detection are performed real-time using the UAV's live image feed. IoT interface displays images captured by the UAV as well as image processing results.

3.1 UAV System and Navigation

The UAV systems are composed of a DJI Ryze Tello and DJI Tello EDU drones, with three degrees of freedom of motion, including longitudinal, altitudinal and yaw. 4x4 ArUco markers are used for navigational commands for the UAV. Figure 1 demonstrates the UAV navigation capabilities and the marker axes used in marker detection. Once the marker is detected and identified, the navigation command associated with that marker ID is executed such that the UAV moves along the horizontal plane (longitudinal motion), vertical plane (altitudinal motion), or around the vertical axis (yaw motion).

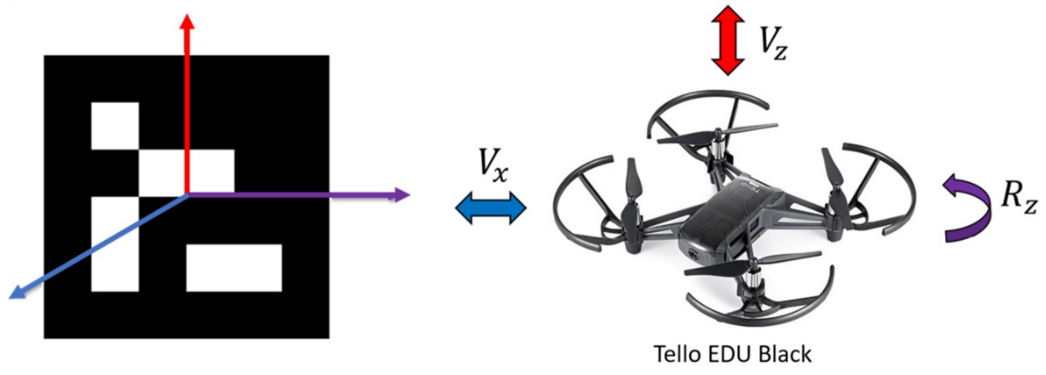


Figure 1. UAV Navigation with ArUco Markers:
V_x = Longitudinal Motion, V_z = Altitudinal Motion, R_z = Yaw Motion

3.2 Marker Detection and Identification

The ArUco marker detection method used in this work is a result of the algorithm-based detection process built into OpenCV. A variation of OpenCV, *opencv-contrib-python*, has an *aruco* submodule that allows for the creation of a marker dictionary from where markers of varying sizes can be pulled.

The marker detection process is the same for any marker of any size. The corners of the detected ArUco marker must be fit to a line and flattened so that parameters of each marker can be measured in correct aspect ratio. Using OpenCV functions, each detected marker is first divided into a cell matrix to determine if the marker fits in any given ArUco library. Each detected marker possesses classification layers that allow for the determination of parameters such as marker ID, marker orientation, corner points, center point, and pose estimation using OpenCV. A line-fit 4x4 ArUco marker with classification layers is displayed in Figure 2. The region of interest (ROI) for the marker is dependent on the dimension of the marker but, as mentioned before, the process for each parameter estimation follows the same process.

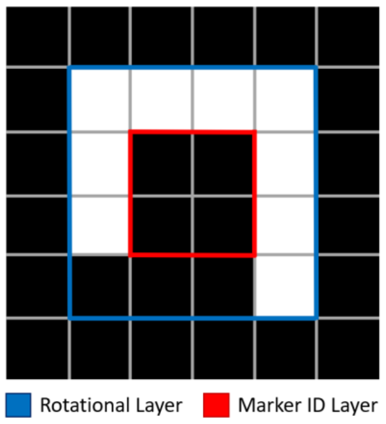


Figure 2. Marker (ID = 10) classification diagram.

Rotational layer of each marker, represented by the outer bound (bounding box) of the bright pixels, or the 4x4 area bounded by the blue frame outside the border of the bright pixels, helps conduct other processes such as pose estimation. The marker ID layer, shown as the inner box, or red frame, is dependent on the dimension of the marker, and allows for the used OpenCV algorithm to lookup the marker from the created marker dictionary. Bounding box help determine the spatial orientation of the bright and dark pixels within the outer bounding box that includes the 4x4 pixels. The spatial orientation is then used to determine the marker ID.

3.3 Defect Detection

Image processing for the captured image containing the structural defect or crack starts with morphological operations to enhance the defect in the image for better characterization. Before any morphological operations are applied, Gaussian blur is applied to each captured image using the *GaussianBlur* function to smooth potential high spatial frequency noise. The image is then converted to gray scale using the *cvtColor* function. Both functions are available in the OpenCV module. These operations enhance the image to avoid misclassification of any textural features present on concrete or any other inspected medium that might adversely affect the proper characterization of the defect. Then the *threshold* function is used to convert the gray scaled image into a binary format. This function uses Otsu's adaptive thresholding algorithm, with the appropriate flag. A crack-filling structuring element is created using the *getStructuringElement* function. The outputs of the threshold and structuring element functions are used in function parameters along with the input image to conduct the morphological operations, specifically, morphological erosion and dilation.

Erosion, the first morphological operation, decreases noise, or pixels that do not belong to the main crack defect, present on the borders of the binary crack image. Erosion uses the structuring element to eliminate any pixels that do not have a 1 behind them in the binary image. What this means is that the structuring element reduces the size of the crack along the crack border as well as noisy pixels, which may or may not be connected to the crack. The size of the structuring element determines how many pixels are removed from the crack or away from the crack that appear as crack (bright pixels in Figure 3). After this noise reduction through erosion, morphological dilation operation is needed to fill in small holes, and to bring the object (crack as defect) back to its original size, since erosion shrinks the crack's visible or surface area. The dilation process increases the surface area of the object (crack) and closes small holes so proper magnitude determination can be made. The process of erosion and dilation is pictured in Figure 3.

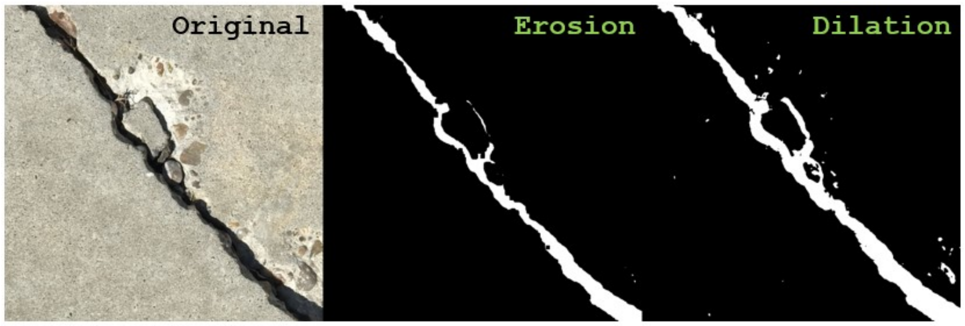


Figure 3. Original crack image, erosion, and dilation sample.

The mathematical modeling for erosion of a binary image, A, by the structuring element, B, is represented in Equation (1)[31], where z represents the translation vector. E refers to the integer grid of the binary image, and B_z is the translation of structuring element, B, by the translation vector, z. Dilation of a binary image is denoted in Equation (2), where B^s is B symmetric, and all other variables are the same as (1).

$$A \ominus B = \{z \in E \mid B_z \subseteq A\} \tag{1}$$

$$A \oplus B = \{z \in E \mid (B^s)_z \cap A \neq \emptyset\} \tag{2}$$

To quantify the difference between original and resultant binary images after morphological operations, mean square error (MSE) and structural similarity index measurement (SSIM)[32] were used as image comparison tools. MSE is defined by Equation (3), where m and n are the number of pixels or data points in the two images, I represents the original binary image, (i,j) represents the index (pixel location) for image I, and K is the binary image after I has undergone morphological operations. The two corresponding pixel values in I and K is subtracted, squared and summed to obtain the single MSE value representing image similarity, or dissimilarity that is directly proportional to the value of MSE. MSE's resultant value will result in a 0 if there are no differences between I and K, and will be non-zero otherwise. SSIM is defined by Equation (4) where μ represents the mean luminance for each respective image, σ is the standard deviation for all the pixels in each respective image, and C_1 and C_2 are constants to maintain stability if the denominator is to have a value of 0. SSIM's resultant values are represented in a range from -1 to 1, where a 1 represents no changes and -1 represents maximum change.

$$MSE = \frac{1}{m * n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{3}$$

$$SSIM = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4}$$

3.4 IoT Application

In this implementation, the UAV-captured and processed images are uploaded a Microsoft Azure IoT hub. Azure is equipped with a dashboard that allows users to view all files located in the container of the IoT hub as well as the number of files uploaded in instances of time. IoT hub ID and storage ID are needed for each device ID associated with the hub. This application provides access and viewing of results without the need for being in the viewing range of the operating device. In addition, IoT application allows remote and off-line access to data and results for further assessment of UAV-captured structural defects.

4. IMPLEMENTATION

In a single UAV system, the operating device can connect directly to the UAV for operation. However, in a multi-UAV system, the number of UAVs and the operating device, in this case a Surface Book Pro used to start UAV motion, must be connected to a network router using the SDK for the UAV and a Wi-Fi connection. Then the UAVs' operation algorithms can be sent over the wireless connection. Using the OpenCV algorithms for marker identification and instruction and the algorithms for image processing detailed in Sections 3.2 and 3.3, the UAVs are ready for motion via the path planning operations. With the recognition of location A (marker 0) that dictates the UAV to adopt a certain motion that involves scanning the area for defects, and with B (marker 10) that informs the UAV's next motion plan to return to the ground for example, the UAV is set to apply the image processing techniques on captured images of the area of interest in concrete cracks while it is moving, in this example, from marker 10 to marker 0. (Note: A and B discussed here should not be confused with morphological quantities *A* and *B* described in Equations (1) and (2).) These images are processed in real-time and then sent to Azure IoT hub where any user with the container login can view the captured images. Figure 4 displays the general workflow for the UAV after the system is turned on and the start command is initiated.

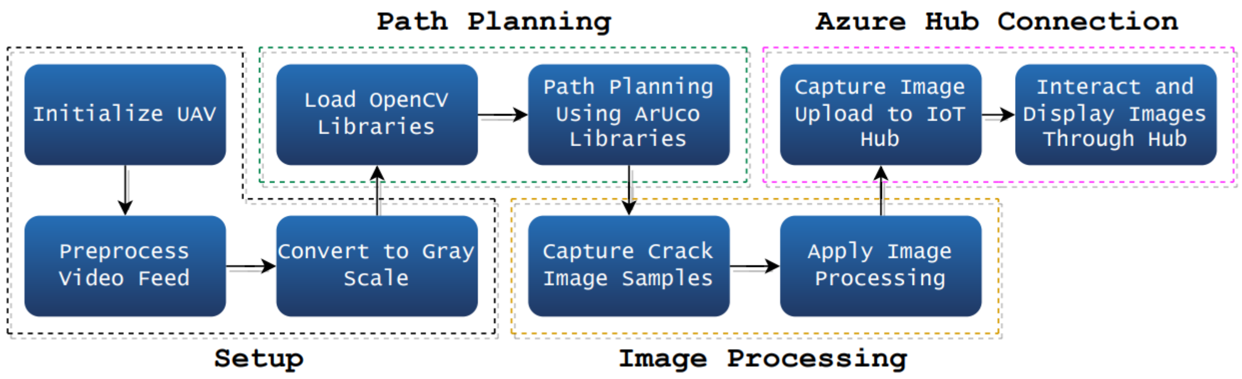


Figure 4. System Workflow: Block diagram for setup, path planning, image processing, and IoT implementation.

Figure 5. a) illustrates the implementation setup that includes communications, marker detection, motion planning based on marker, image capture and image processing, and motion planning based on second marker to complete the scanning cycle of the structure of interest adopted for the experiments. Figure 5. b) is the extended representation of detection protocol using multiple UAVs.

We point out here that with the wireless connection between UAVs and network router using the SDK method, each UAV is designated with their individual IP address. Using the individual IP address for each drone, packets can be sent to each UAV individually. Packets are a system of instructions that are predefined and can be sent to the UAVs for operation. Using an ArUco marker system that has been designated to transmit specific instruction, a packet can be created to transmit the path-planning instructions to the UAV system. Similarly, a packet containing instructional data related to image capture and processing techniques can be transmitted to the UAV system for autonomous operation. It is important to mention that packets made for transmitting path-planning instructional data must be preprogrammed to avoid collision between UAVs while maintaining a relative proximity so that the area of interest for processing can be fully framed without gaps in the image capturing process.

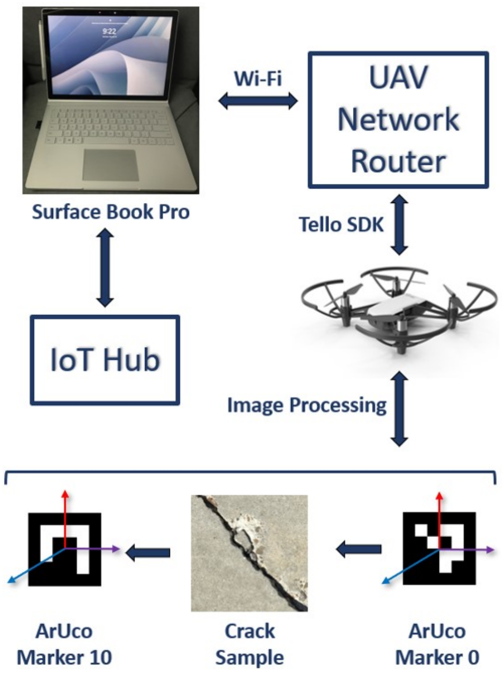


Figure 5. a) Implementation setup

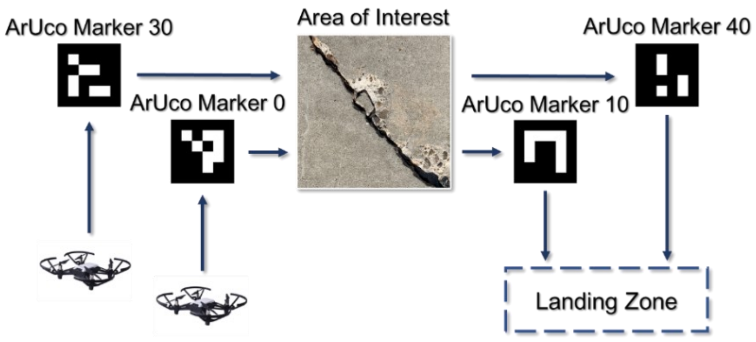


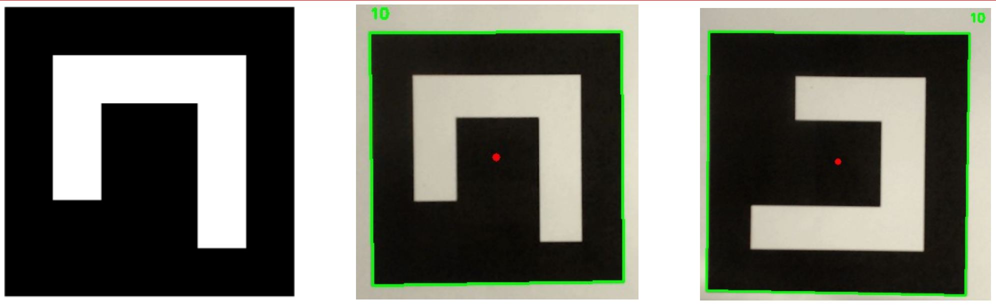
Figure 5. b) Implementation setup for a multi-UAV system with separate ArUco tags and path plan for each UAV.

5. RESULTS AND ANALYSIS

This section summarizes the results for a use case study of concrete structural assessment using the UAV beginning with marker detection followed by defect detection and characterization, and IoT implementation.

5.1 Real-time ArUco Marker Identification and Detection

Marker identification uses a marker detection system and marker library to allow for faster processing times[33]. Using the OpenCV and ArUco marker libraries[34], identification was targeted for ArUco (4x4) markers. This (4x4) marker size allows quicker identification times to reduce the amount of latency present in live video feed and experimental processes. The ArUco markers are mapped into a binary image and pose estimated to return identification results when the markers are rotated or off-center angle. Then, to create bounding boxes for the markers, the OpenCV *findContour* function is used. The results for the ArUco marker identification process are displayed in Figure 6.



a) ArUco marker, ID: 10 **b)** ArUco marker, ID: 10 identification **c)** ArUco marker, ID: 10 identification (marker 90° rotated)

Figure 6. ArUco marker detection and identification.

5.2 Real-time Defect Detection

5.2.1 Image Capture and Morphological Operations

Figure 7 shows the first three steps in concrete damage characterization.

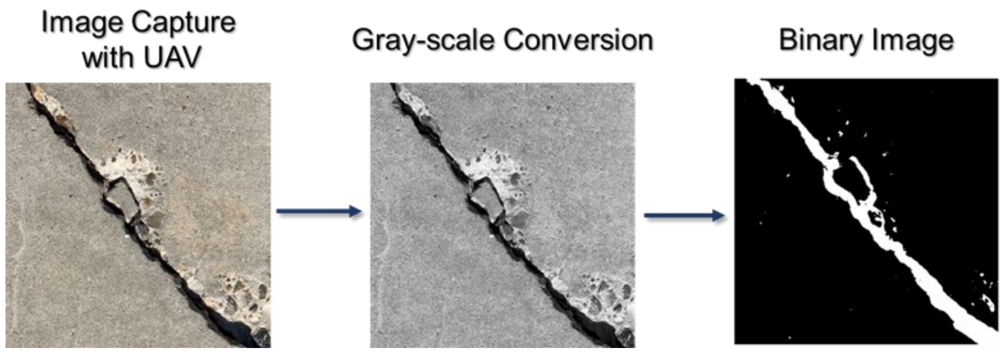
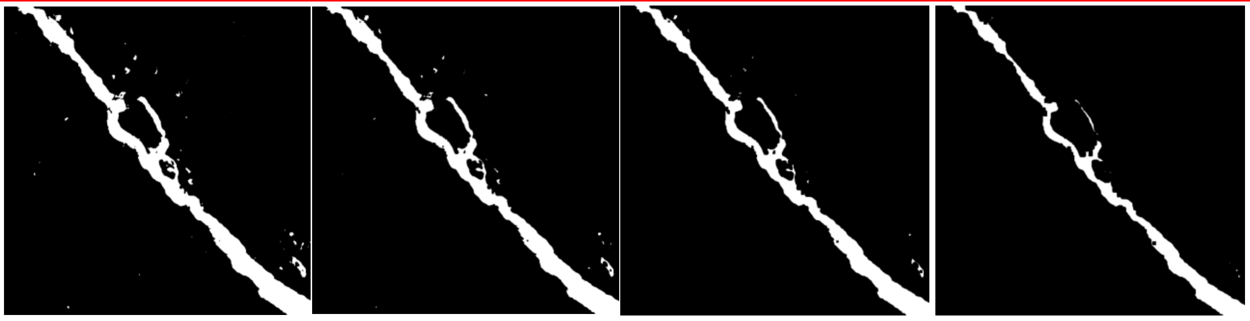


Figure 7. UAV-captured image (left), gray-scale conversion (middle), and binary conversion (right).

The image processing techniques detailed in section 3.3 are applied in multiple iterations to increase the effectiveness of the image processing results. For morphological erosion, with each increment or iteration, the kernel size present in the binary image will be reduced until the kernel is no longer visible. With too many iterations the image will begin to lose the useful data that represents the targeted damage, but proper testing of iterations can isolate the valuable data that is present in the images. Iterations for erosion are displayed in Figure 8.

As can be seen, erosion progressively reduces the noisy pixels, but at the expense of the thickness of the crack represented by bright pixels in the binary image of Figure 7. Dilation is therefore applied to the eroded images after proper iterations are completed. Ideally, after random noise is removed during the erosion process the dilation results will provide only an increased area size of the area of interests. Dilation is applied to the last iteration of erosion (8th iteration of erosion in this case) since this iteration provided the least amount of random noise when compared to the original binary image. Ideally the number of morphological erosion and dilation operations or iterations should match if general structural magnitude information is to be kept intact.

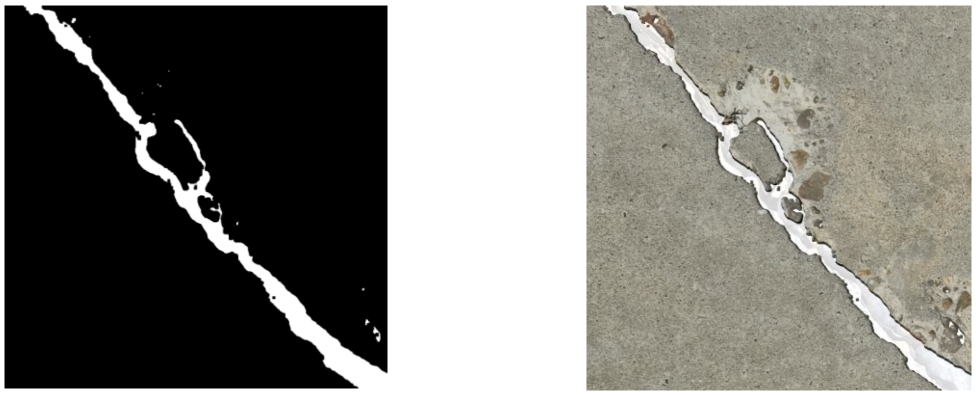


a) Erosion, iteration = 0 **b)** Erosion, iteration = 2 **c)** Erosion, iteration = 4 **d)** Erosion, iteration = 8

Figure 8. Erosion results after 0, 2, 4 and 8 iterations.

Figure 9 (a) depicts the dilation results. Figure 9 (b) shows the dilation results superimposed onto the original color image. Some resurfaced noise in the dilated image after the dilation process is apparent. This can be attributed to the varying ways noise can be introduced in images. One such example is a single pixel that might have remained after multiple erosions that is then dilated. The surrounding conditions when the image was taken (weather and background objects) or the angle in which light is hitting the surface of the captured object will all affect the results of morphological operations.

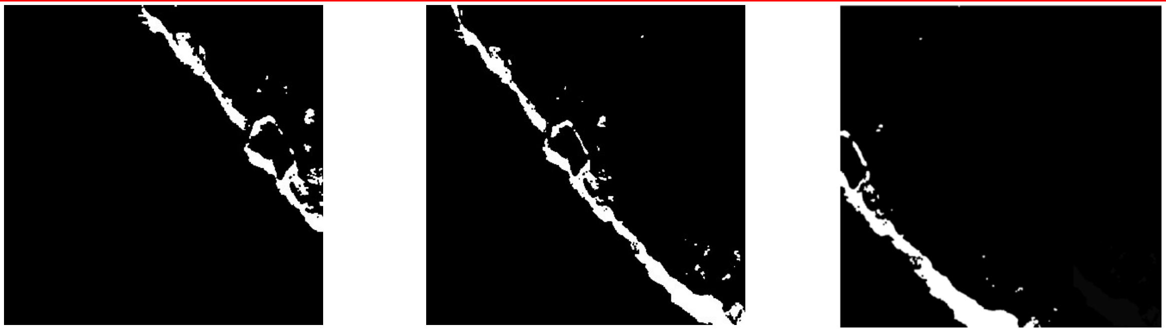
The results of the image processing operations and the original captured structural damage image are superimposed to give a resultant image that provides a clear picture of the surface area that has detected damage in relation to the original surrounding area.



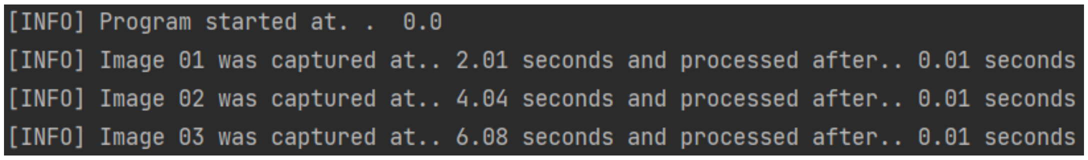
a) Result of final dilation **b)** Superimposed image with detected crack

Figure 9. Output of binary crack image dilation and superimposed image with the crack.

For an image processing application to be considered real-time, image capture and the image processing must happen simultaneously or at a time interval that is negligible. This application uses a two-second interval for capturing structural damage images where the UAV and hence the camera are in a left-to-right (longitudinal) motion and the image processing algorithms begin immediately after the image capture. Each iteration of image capture and image processing workflow is included with a *time.time()* Python function that records the time in seconds since each epoch, which is January 1, 1970, at 00:00:00 where time begins. This value is subtracted by itself at the start to get a baseline value of 0 for the start time. The real-time image processing and read out time results can be seen in Figure 10.



a) Processed Image 01 **b)** Processed Image 02 **c)** Processed Image 03



```
[INFO] Program started at. .  0.0
[INFO] Image 01 was captured at.. 2.01 seconds and processed after.. 0.01 seconds
[INFO] Image 02 was captured at.. 4.04 seconds and processed after.. 0.01 seconds
[INFO] Image 03 was captured at.. 6.08 seconds and processed after.. 0.01 seconds
```

d) Capture and processing read-out times

Figure 10. Real-time image processing results and processing times for defect detection during UAV flight path after and before detection of ArUco markers.

5.2.2 Defect Assessment

The MSE and SSIM values can be used to assess the effect of the operations that the image has undergone. In this case the MSE and SSIM values were calculated for the original binary image from raw color data and the binary image obtained after eight iterations of erosion, before dilation was applied to bring the object in the image (the crack) back to its original magnitude representation. These values can be used to assess both damage and the effect of image processing on original data. The MSE and SSIM results are shown in Figure 11.

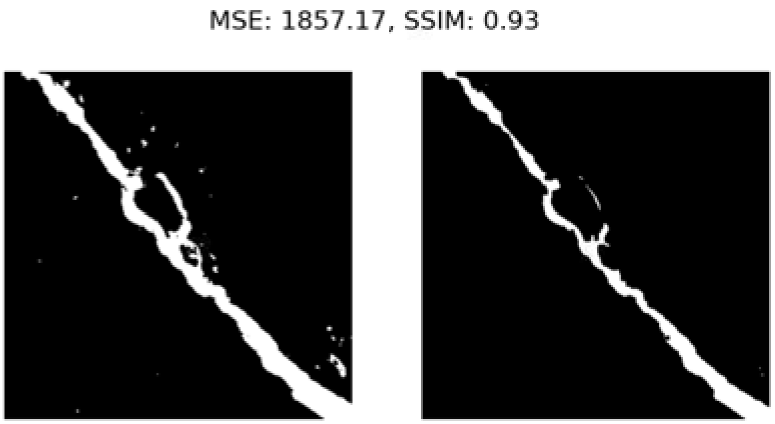
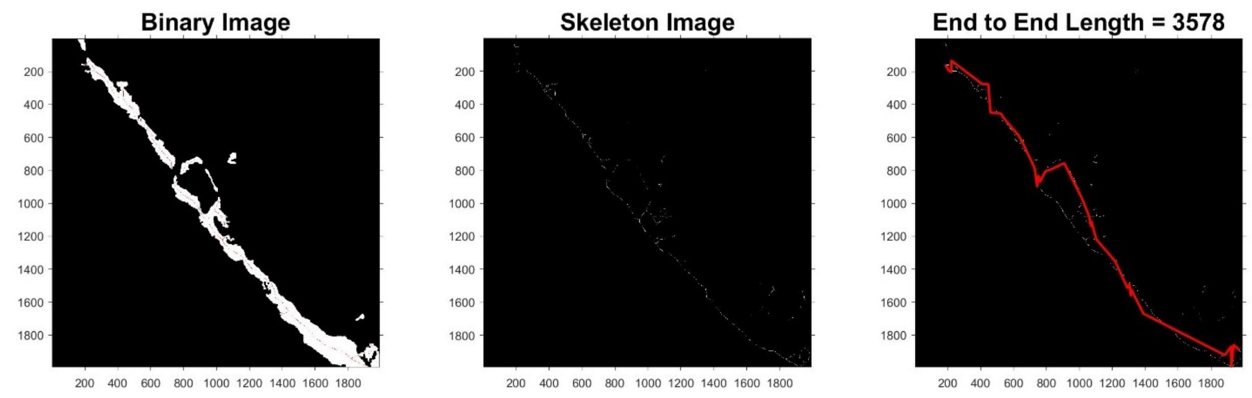


Figure 11. MSE and SSIM image comparison measurements for original (left) and resultant (right) binary images.

Figure 12 shows the characterization of the detected crack in terms of its magnitude. The crack is characterized by first applying skeletonization to the multiple iterations of eroded then dilated image, then adding the number of pixels that belong to the crack end to end. The actual length of the crack can be estimated by multiplying by the conversion factor representing the actual physical distance each pixel represents. Here, the length is reported in pixels.

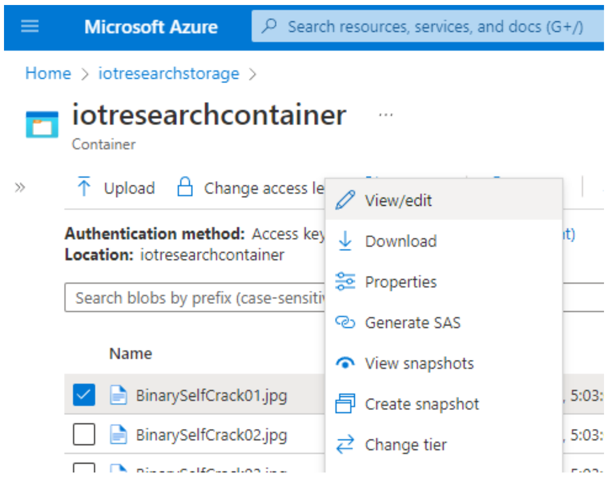


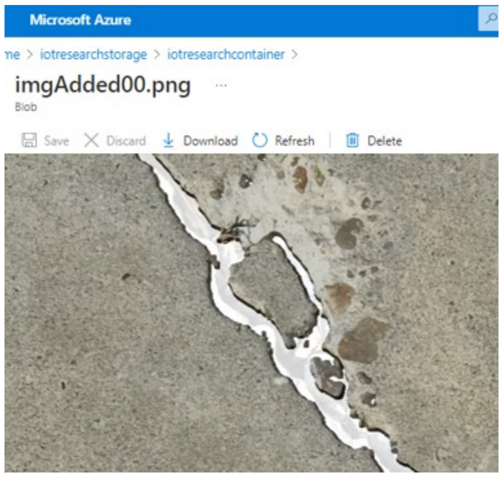
a) b) c)

Figure 12. Estimating defect magnitude. **a)** Binary image after erosion and dilation **b)** Skeletonized image of the crack defect; **c)** End-to-end length of the crack in pixels.

5.3 IoT Implementation

The last focus of this project is the IoT application. As mentioned before, this work uses the Microsoft Azure IoT hub as a network space where images are uploaded and available for viewing so long as the user is operating off a device that is included in the device list of the hub. The image uploading process involves locating the connection string, container name, and image source folder. This information is configured in a YAML file and is used as information strings in Python functions that allow uploading files to the hub storage account. Once images are uploaded to the hub, users can view/edit files, view data associated with files, and view devices that uploaded the images to the hub. Moving forward additional functionality can be added to the IoT interface. Figure 13 displays the IoT container available in Azure hub, the images located in the container, and the view screen for the user.





a) Container files and view/edit option **b)** View/edit screen

Figure 13. Azure IoT hub display.

6. CONCLUSION

The detailed methods for implementation and experimental results have allowed successful structural damage inspection guided by a set of ArUco markers for UAV path planning enhancement. ArUco markers provide a fast detection method for UAV path planning for the real-time implementation of structural damage detection, in this case, of concrete cracks. Varying iterations for morphological image processing techniques for noise reduction have also been displayed for isolated structural damage results. The captured images and results were uploaded through the network infrastructure to Azure IoT hub as the application of IoT for this research, enabling remote viewing and access to UAV imagery for further user evaluation. Overall, the system represents an affordable and safer alternative to real world large scale structural damage assessment.

The multi-UAV application was limited to individually-programmed UAVs and ArUco marker instructions that were executed independently (Fig. 5(B)). Coordinated multi-UAV communication and tasks will be incorporated in the next phase of the project. Future work includes determining the accuracy of path planning with Vicon Vantage motion capture cameras. These cameras, along with a UAV equipped with reflective orbs, will produce the three-dimensional path that the UAV underwent during testing. The longitudinal, altitudinal and yaw movements of the UAV will be recorded from the UAV object where the path planning dataset is collected via the Vicon software to map and analyze the UAV's path during testing. Here we only scratched the surface of the potential for IoT implementation. With multiple defect images collected using UAVs, IoT will lend itself to image analytics and AI/deep learning approaches for further defect detection, identification and characterization.

ACKNOWLEDGMENT

This project is supported in part by NSF McNair Scholars Program at Texas A&M University-Corpus Christi. The IoT-related materials are supported in part through the NSF IUSE grant #2044255.

REFERENCES

[1] Hellaoui, H., Chelli, A., Bagaa, M., and Taleb, T., "UAV communication strategies in the next generation of mobile networks," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 1642-1647 (2020).

[2] Mohamed, A. M. A., and AbuElgasim, A. E., "Controlling drone–using IOT platform," in *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, 1-4 (2019).

[3] Goyal, A., Kumar, N., Dua, A., Kumar, N., Rodrigues, J. J. P. C., and Jayakody, D. N. K., "An efficient scheme for path planning in internet of drones," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 1-7 (2019).

[4] Yong, S.-P. and Yeong, Y.-C., "Human object detection in forest with deep learning based on drone's vision," *2018 4th International Conference on Computer and Information Sciences (ICCOINS)*, 1-5 (2018).

[5] Wu, W., Qurishee, M. A., Owino, J., Fomunung, I., Onyango, M., and Atolagbe, B., "Coupling deep learning and UAV for infrastructure condition assessment automation," in *2018 IEEE International Smart Cities Conference (ISC2)*, 1-7 (2018).

[6] Wang, Y., Zheng, Z., Su, Z., Yang, G., Wang, Z., and Luo, Y., "An improved ArUco marker for monocular vision ranging," *2020 Chinese Control And Decision Conference (CCDC)*, 2915-2919 (2020).

[7] Garduno, A., Mehrubeoglu, M., and Rangel, P., "Vision-based Path Planning for Exploration in an Unmanned Aerial Vehicle using OpenCV," *Image Processing and Computer Vision, The 2021 World Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE'21)*, 1-13 (26-29 July 2021) (In print).

[8] Shajahan, N. M., Kuruvila, T., Kumar, A. S., and Davis, D., "Automated inspection of monopole tower using drones and computer vision," in *2019 2nd International Conference on Intelligent Autonomous Systems (ICoIAS)*, 187-192 (2019).

[9] Yu, H., Yang, W., Zhang, H., and He, W., "A UAV-based crack inspection system for concrete bridge monitoring," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 3305-3308 (2017).

[10] Shao, C., Chen, Y., Xu, F., and Wang, S., "A kind of pavement crack detection method based on digital image processing," in *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 397-401 (2019).

[11] Marut, A., Wojtowicz, K., and Falkowski, K., "ArUco markers pose estimation in UAV landing aid system," *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 261-266 (2019). doi: 10.1109/MetroAeroSpace.2019.8869572.

[12] Khazetdinov, A., Zakiev, A., Tsoy, T., Svinin, M., and Magid, E., "Embedded ArUco: a novel approach for high precision UAV landing," *2021 International Siberian Conference on Control and Communications (SIBCON)*, 1-6 (2021). doi: 10.1109/SIBCON50419.2021.9438855.

[13] Wang, Z., Wang, B., Tang, C., and Xu, G., "Pose and Velocity Estimation Algorithm for UAV in Visual Landing," *2020 39th Chinese Control Conference (CCC)*, 3713-3718 (2020). doi: 10.23919/CCC50068.2020.9188491.

[14] Nair, A. S., Jeyanthy, P. A., Ramesh, L., Kurian, G. M., and Mohamed, S. R., "Autonomous Precision Landing with UAV and Auto charging," *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, 463-466 (2021). doi: 10.1109/RTEICT52294.2021.9573947.

[15] Lebedev, I., Erashov, A., Shabanova, A., "Accurate Autonomous UAV Landing Using Vision-Based Detection of ArUco-Marker," in: Ronzhin, A., Rigoll, G., Meshcheryakov, R. (eds) [Interactive Collaborative Robotics. ICR 2020. Lecture Notes in Computer Science], 12336, 179-188, Springer, Cham. https://doi.org/10.1007/978-3-030-60337-3_18

[16] Paglinawan, A. C., Cruz, F. R. G., Casi, N. D., Ingatan, P. A. B., Karganilla, A. B. C., and Moster, G. V. G., "Crack detection using multiple image processing for unmanned aerial monitoring of concrete structure," in TENCON 2018 - 2018 IEEE Region 10 Conference, 2534-2538 (2018).

[17] Levine, N. M. and Spencer Jr, B. F., "Post-earthquake building evaluation using UAVs: A BIM-based digital twin framework," *Sensors*, 22(3), 873, 1-24 (2022).

[18] Han, D., "Crack detection of UAV concrete surface images", *Proc. SPIE 11139*, *Applications of Machine Learning*, 1113914 (6 September 2019). https://doi.org/10.1117/12.2525174

[19] Lee, J.-H., Yoon, S.-S., Kim, I.-H., and Jung, H.-J., "Diagnosis of crack damage on structures based on image processing techniques and R-CNN using unmanned aerial vehicle (UAV)", *Proc. SPIE 10598*, *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018*, 1059811 (27 March 2018). https://doi.org/10.1117/12.2296691

[20] Mao, Z., Zhao, C., Zheng, Y., Mao, Y., Li, H., Hua, L., and Liu, Y., "Research on detection method of pavement diseases based on Unmanned Aerial Vehicle (UAV)", *Proc. SPIE 11584*, *2020 International Conference on Image, Video Processing and Artificial Intelligence*, 115840L (10 November 2020). https://doi.org/10.1117/12.2580285

[21] Kang, D. and Cha, Y.-J., "Damage detection with an autonomous UAV using deep learning ", *Proc. SPIE 10598, Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018*, 1059804 (27 March 2018). https://doi.org/10.1117/12.2295961

[22] Gupta, A. K. and Johari, R., "IOT based electrical device surveillance and Control System," *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, 1-5 (2019).

[23] Ghosh, S., Ghosh, K., Karamakar, S., Prasad, S., Debabhuti, N., Sharma, P., Tudu, B., Bhattacharyya, N., and Bandyopadhyay, R., "Development of an IOT based robust architecture for environmental monitoring using UAV," *2019 IEEE 16th India Council International Conference (INDICON)*, 1-4 (2019). doi: 10.1109/INDICON47234.2019.9028987.

[24] Quach, C. H., Pham, M. T., Nguyen, T. S., and Phung, M. D., "Real-time Agriculture Field Monitoring Using IoT-based Sensors and Unmanned Aerial Vehicles," *2021 8th NAFOSTED Conference on Information and Computer Science (NICS)*, 492-497 (2021). doi: 10.1109/NICS54270.2021.9701498.

[25] Caruso, A., Chessa, S., Escolar, S., Barba, J., and López, J. C., "Collection of Data With Drones in Precision Agriculture: Analytical Model and LoRa Case Study," in *IEEE Internet of Things Journal*, 8(22), 16692-16704 (2021). doi: 10.1109/JIOT.2021.3075561.

[26] Chakravarthy, A., Sinha, S., Narang, P., Mandal, M., Chamola, V., and Yu, R., "DroneSegNet: AI-driven Robust Aerial Semantic Segmentation for IoT Applications," in *IEEE Transactions on Vehicular Technology*, 1-10 (2022). doi: 10.1109/TVT.2022.3144358.

[27] Anand, T., Sinha, S., Mandal, M., Chamola, V., and Yu, F. R., "AgriSegNet: Deep Aerial Semantic Segmentation Framework for IoT-Assisted Precision Agriculture," in *IEEE Sensors Journal*, 21(16), 17581-17590, (2021). doi: 10.1109/JSEN.2021.3071290.

[28] Bouteraa, Y., and Ben Abdallah, I., "Development of IoT-based robot for wrist rehabilitation," *2020 17th International Multi-Conference on Systems, Signals & Devices (SSD)*, 735-739 (2020).

[29] Moore, A., Mehrubeoglu, M., Moony, A., and McLauchlan, L., Application of IoT-based sensing and signal processing for rehabilitation," *SPIE Defense and Commercial Sensing,* (3-7 April 2022).

[30] Liu, W., "Improvement of navigation of Mobile Robotics based on IoT System," *2021 IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI)*, 69-72 (2021).

[31] Daugherty E. R. and Lotufo, R. A., *Hands on Morphological Image Processing*, Chapters 1-2, SPIE, Bellingham, WA (2003).

[32] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P., "Image quality assessment: from error visibility to structural similarity," in *IEEE Transactions on Image Processing*, 13(4), 600-612 (April 2004). doi: 10.1109/TIP.2003.819861.

[33] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F. J., and Marín-Jiménez, M. J., "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, 47(6), 2280-2292 (June 2014). DOI=10.1016/j.patcog.2014.01.005

[34] *Detection of ArUco Markers*, Tutorial, OpenCV Open Source Computer Vision (Available: https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html. Accessed: 2 Feb. 2022).