Research Article

A Faulty Solar Panel Detecting and Mapping Method Using a Drone with RGB and Thermal Cameras in Solar Power Plant

Soobin Jeon and Dongmahn Seo

School of Computer Software, Daegu Catholic University, Gyeongsan, Republic of Korea

Article history
Received: 16-11-2024
Revised: 01-01-2025
Accepted: 03-03-2025

Corresponding Author:
Dongmahn Seo
School of Computer Software,
Daegu Catholic University,
Gyeongsan, Republic of Korea
Email: sarum@cu.ac.kr

Abstract: A solar panel generates electricity power using the sun's rays. It is usually located on a rooftop or open field without shadow. This solar panel is hard to be tested by nonprofessionals. Therefore, we propose a detecting system for a faulty solar panel using a drone with an RGB camera and a thermal camera. The drone is applied for collecting aerial videos. The RGB camera is operated to find the location of solar panels with solar panel features, such as the rectangle shape and the surface color of the solar panel. The thermal camera is deployed to search for error spots in solar panels. Since error spots are not able to convert the sun's rays into electricity, they reflect the sun's rays. Therefore, the temperature of the error spot is higher than normal solar panels. The proposed system is tested in two real working solar power plants. We confirmed that the proposed system can be applied without professional knowledge.

Keywords: Automatic Detection, Faulty Solar Panel, Image Processing, Drone, Thermal Image

Introduction

Today, solar energy is attracting attention as renewable energy to replace oil. Solar energy is used for agriculture, industry, households, etc. According to the statistical report (IRENA, 2009), renewable energy's total capacity and production are growing. 2014, the world's solar energy capacity was 179,639 MW, and solar energy production was 192,602 GWh. In 2023, the world's solar energy capacity was 1,418,016, MW and solar energy production was 1,294,481 GWh. The growth of solar energy capacity is almost 8 times, and solar energy production has grown nearly 7 times from 2014-2023. Solar energy keeps growing and is one of the important sources of renewable energy.

As solar energy is growing, the solar panel market is growing. A solar panel is a product that uses solar energy. It converts the sun’s rays to electricity power. The solar panel helps to save energy and to use solar energy efficiently. For efficient usage, where the solar panel is installed is essential. If some small object, such as a leaf, paper, and so on, is placed on the surface of the solar panel, it can decrease the production capacity of the solar panel (Chamberlin *et al.*, 2011). So, the solar panel is placed on a loop or flatland, which does not have a shadow on the solar panel surface (Patil and Asokan, 2016). For this reason, solar panels are exposed to various dangers and are easy to break. The solar panel fault, a hot spot, is not easy to find by a human's eyes. Figure (1) shows a sample of the solar panel image captured by the RGB and thermal camera with a drone. The upper image is the solar panel of the RGB (red, green, and blue) camera, and the lower image is the solar panel of the thermal camera. The left and right red boxes solar panels of the upper image are not different in the RGB. However, the red boxes in the lower image show pretty differently. Because the left red box had the hot spot then, it did not show in the RGB images. This is why finding the error solar panel with only human eyes is difficult. Figure (2) shows an example of the solar panel's hot spot and normal cells.

Three kinds of methods have been studied to find solar panel errors. The first method compares the actual production of the solar panel and the estimated production of the solar panel (Guasch *et al.*, 2003; Harrou *et al.*, 2017; Tao *et al.*, 2017; Zhang *et al.*, 2013; Chao *et al.*, 2019). They need to check all the solar panel adapters that move the energy to the battery. This can be used automatically or not. An automatic method uses the property of AC parameters (Kim *et al.*, 2016; Liu *et al.*, 2010; 2019; Madeti and Singh, 2017; Gokmen *et al.*, 2013; Lu *et al.*, 2018; Pei and Hao, 2019; Ji *et al.*, 2017; Dhanraj *et al.*, 2021). A manual method is to check the production of the solar panel and compare each of them with an expert. However, with the expert, it takes much time to find out because it must check each of the solar power plants' solar panels. A person who uses the solar panel does not know how to find differences between the actual production of the solar panel and the estimated production of the solar panel. Furthermore, users cannot figure out the exact location of the error.

Another one checks the surface of the solar panel from the expert's view. This method must have an expert. The hot spot occurs by a small crack in the surface of the solar panel or a burning spot on the surface. The crack or burning area is more minor than the hot spots. So, even with an expert, figuring out the hot spot area is challenging. Then, it must see the solar panel near the surface with bare eyes. So, the solar panel needs to be separated from the supporting frames.

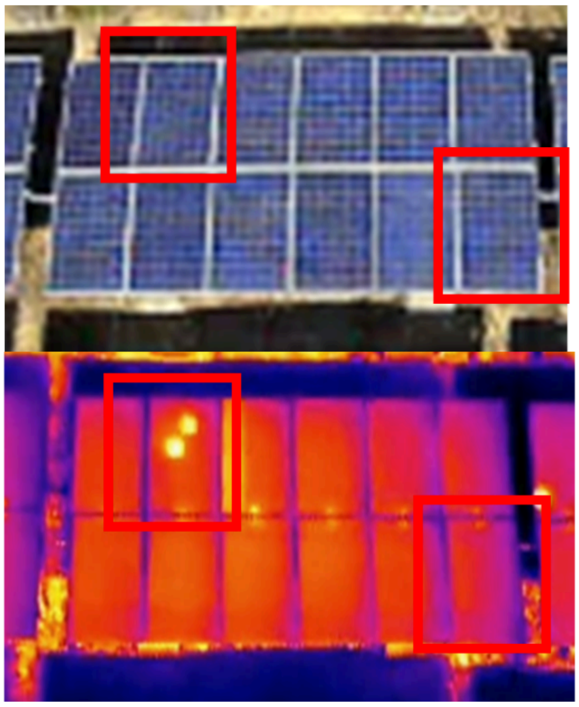


Fig. 1: The sample image of solar panel

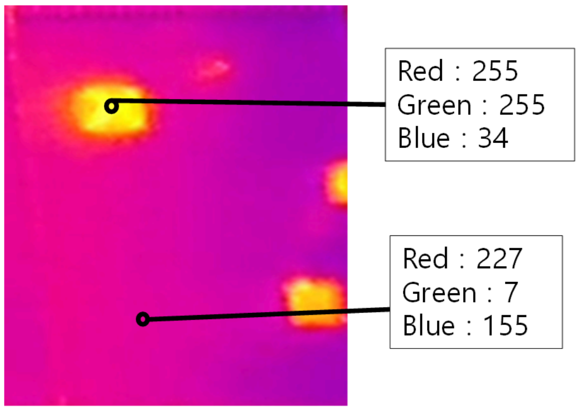


Fig. 2: The sample value of the hot spot and normal cells in the solar panel

The other checks the solar panel with the thermal camera (Hu *et al.*, 2013; Ancuta and Cepisca, 2011; Dotenco *et al.*, 2016). This approach does not need an expert. The hot spot on the surface is expressed as the highlighted spot on the thermal image. It is easy to find the faulty solar panel. It needs to capture a thermal image higher than the solar panels. However, it is hard to capture the thermal images of the solar panel. Most solar power plants are installed on the highest ground. The above method requires much time from experts to check the solar panel.

Machine learning and computer vision are used to detect the solar panels in the image. Computer vision uses a solar panel feature, such as panel shape, color, and so on (Tao *et al.*, 2017; Yao and Hu, 2017; Malof *et al.*, 2015; Carletti *et al.*, 2020). Then, machine learning makes self-learning a machine that detects solar panels in an aerial image (Yuan *et al.*, 2016; Salamanca *et al.*, 2017; Selvaraj *et al.*, 2022). These approaches have good accuracy and a higher detection rate. However, most studies are limited to detecting hot spots inside a single panel. Solar power plants have tens or hundreds of solar arrays installed, and each array comprises tens or hundreds of solar panels. Therefore, it is essential not only to detect defects in solar panels but also to quickly and accurately identify where the defect occurred in the solar power plant and enable repair of the panel. The solar power plants consist of solar panel arrays. Then, the solar panel arrays consist of solar panels. Most solar power plants have plenty of solar panel arrays in a wide area.

To conveniently detect the failure of the solar panel in the solar power plants, we proposed an automatic real-time system based on computer vision with a drone, an RGB camera, and a thermal camera. The drone helps capture the RGB and thermal images easily wherever the solar panel is. Then, the RGB camera helps to find the solar panel and array used by the features of the solar panel. Then, the thermal camera helps to find the faulty solar panel with the hot spot.

Materials and Methods

The proposed system follows the flow chart in Figure (3). When the RGB and thermal video are input, the system figures out the solar panel and solar panel array in each frame of the RGB video. Then, the location of the hot spot area is compared between the RGB and thermal images. When the hot spot is placed on the area of the solar panel, it means that the solar panel has a failure. Additionally, our system generates a map of the solar power plants and saves images that show the locations of faulty solar panels with related RGB and thermal images and maps.

As shown in the upper image of Figure (1), the solar panel consists of black cells and a white border. We use two features, the rectangular shape and the white borders, to find the solar panels in the RGB image. To figure out the solar panel in the RGB image, the preprocessing of the proposed system highlights the border of the solar panels and the border of the hot spot with an RGB to grayscale and an OTSU threshold (Otsu, 1979), a canny edge detection (Canny, 1987), a morphological transformation (Suzuki and Be, 1985). The color of the border is unique in solar power plants. The reason is that most of the solar power plants are placed on the plains where the white color is hard to find. Figure (4) shows the result of the RGB to grayscale conversion. Then, the grayscale image changes to a binary image. We want to highlight only the solar panel border in the binary image. So, the OTSU threshold algorithm is a good fit for this. Because the white border of the solar panel is also remarkable in the grayscale image, the OTSU threshold algorithm finds the most optimal value for the threshold to divide the background and foreground. In the binary image of the solar panel array, the foreground is the border, and the background is the border. As shown in Figure (5), we get the binary image that can figure out the solar panel from the image. So, we separate the border from the binary image with the OTSU threshold. However, the binary image has some white noise on the solar panels. The black cell composed of the solar panel has a border, too. We use morphological transformations such as erosion and dilation to reduce white noises. The erosion makes the border thinner and removes the small objects, such as the white noise in the binary image. The dilation works the opposite of the erosion. It makes the white objects bigger and connects the border, which is not closed. The erosion and dilation sequence, called an opening, makes the reduction of the small white object and the border more evident. The erosion and dilation follow Algorithm 1.

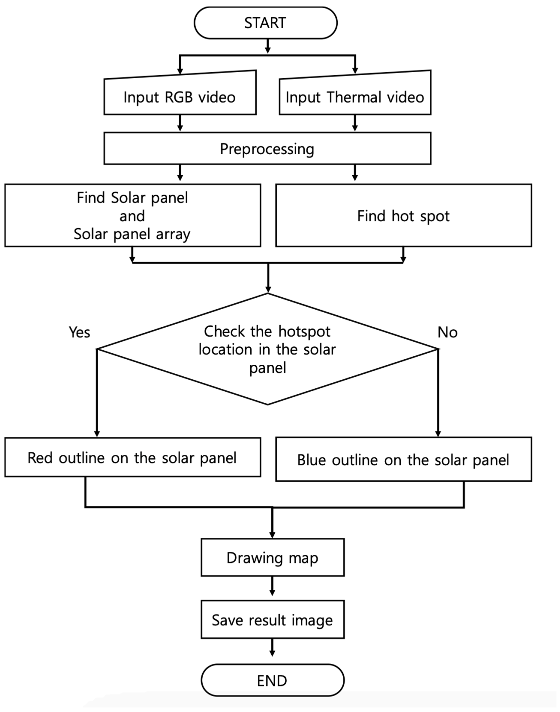

START
Input RGB video
Input Thermal video
Preprocessing
Find Solar panel and Solar panel array
Find hot spot
Check the hotspot location in the solar panel
Yes
No
Red outline on the solar panel
Blue outline on the solar panel
Drawing map
Save result image
END

Fig. 3: The flow chart of the proposed system

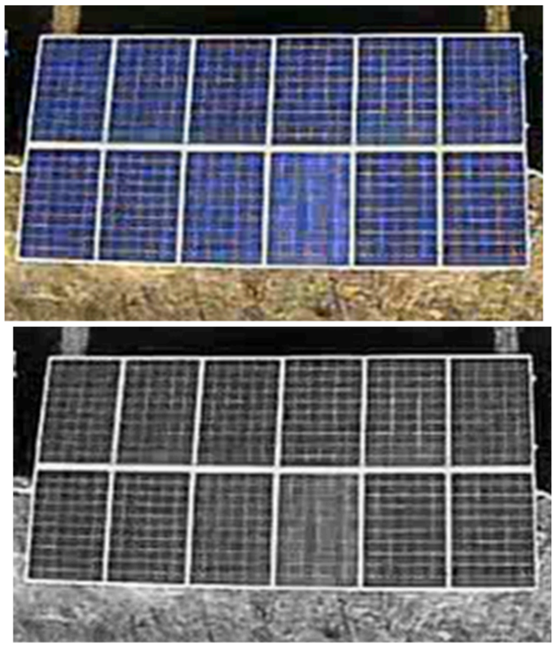


Fig. 4: The solar panel by the RGB (Upper) and the solar panel by the grayscale (Lower)

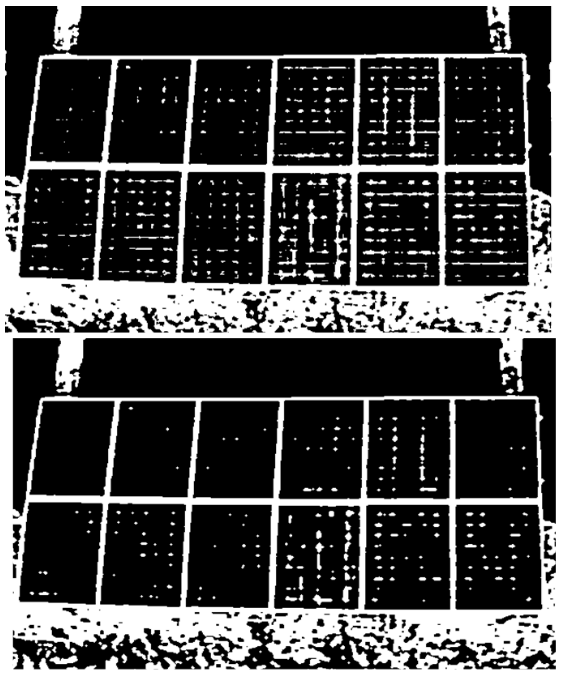


Fig. 5: The binary image by the OTSU threshold (Upper), and the result of the morphological transformation (Lower)

In Algorithm 1, the *img* denotes the input binary image, the dst denotes the output binary image that results from the morphological transform, and the *option* indicates the flag that chooses the erosion or dilation. We use 3 x 3 kernels that are composed of 1. The erosion process occurs when the kernel slides the input image. Then, if all values are the same as the kernel, the center of the kernel location value is stored in the output image.

```
Algorithm 1: Morphological Transform

function morph(img, dst, option);

Input: The input image img and the result image dst,
the option to select erosion or dilation

# erosion
    if option is erosion then
        # sliding kernel in the img
        for x ← 0 to img.width
            for y ← 0 to img.height
                if img[x-1 to x+1][y-1 to y+1] is 1 then
                do dst[x][y] = 1
# dilation
    else if option is dilation then
        # sliding kernel in the img
        for x ← 0 to img.width
            for y ← 0 to img.height
                if img[x][y] is 1 then
                do dst[x-1 to x+1][y-1 to y+1] = 1
end
```

Figure (6) shows the result of the morphological transformation. The top is the original binary image. Solar panels have lots of white noise. Then, the Middle is the result of the erosion. It removes the white noises of the left solar panel and reduces the white noises of the right solar panel. Then, the Bottom is the result of the opening. Panel images have thicker borders and less noise than those of the Middle. Furthermore, the borders of the Bottom are more precise than the Top. The right solar panel on Top cannot quickly determine the shape of the solar panel because the white noise is connected to the border. Therefore, the result of the opening shows that it typically removes the white noise and detaches the white noise from the border to make it clear.

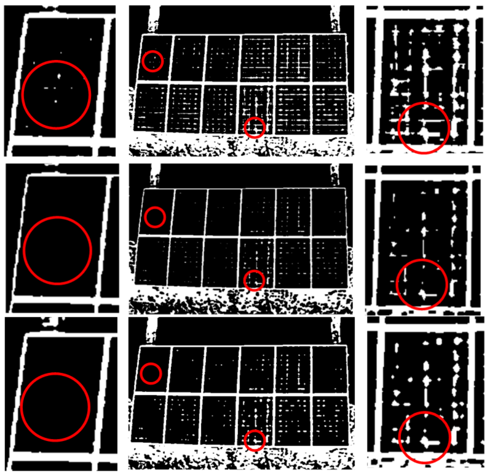


Fig. 6: The result of the morphological transformation (Top is the original binary image, Middle is the result of result of the erosion of Top, and Bottom is the result of the dilate of Middle

We want to separate the hotspot area from solar panels. The thermal image is the same sequence as the preprocessing of the RGB image. The thermal image has a more unique feature than the RGB image. In the RGB preprocessing, the OTSU threshold finds the proper value for making the binary image. It works because the border has a unique color in the solar power plants. However, finding the appropriate threshold value in the thermal images is challenging. The thermal image makes it easy to change the feature. Most of the thermal cameras have automatic normalizing. Then, it makes the thermal image look relatively. This means that the thermal camera is sensitive to outdoor conditions such as weather and captured positions. However, the hot spot is a unique feature that wasn't affected by the outdoor conditions. In Figure (2), the hot spot has static values; the red and green colors are almost 255. The threshold algorithm for the thermal image follows Formula (1). For these reasons, we set the threshold values that the red and green are 220 and the blue is 150. Then, if each channel value is higher than the threshold, set 1 or not set 0. Figure (7) shows the result of the binary. In Figure (7) (a), the three hot spots are highlighted, and in (b), the binary image is highlighted too:

$$Y = \begin{cases} 1, & if\ R\ and\ G > 220, B > 150 \\ 0, & otherwise \end{cases} \tag{1}$$

Our system preprocesses RGB and thermal images to highlight features. We highlight the border of the solar panel in the RGB images and the hot spots in the thermal images. To find the solar panel in the binary image, we use the shape of the solar panel with a rectangular border. The Canny edge detection is based on a Gaussian filter (Deng and Cahill, 2005). It detects all of the edges from the image with four steps. First, it removes the noise with the Gaussian filter. Then, finding the gradients for detecting the edge where the pixel value changes rapidly. Then, all edges are scanned to leave the local maxima edge where the pixel value is higher than others. Then, the edges are filtered to find a substantial edge and after the above steps, the noise can occur. Then, we applied the double threshold to remove the noise edge. So, the canny edge detection finds a firm edge and makes the minimum edge.

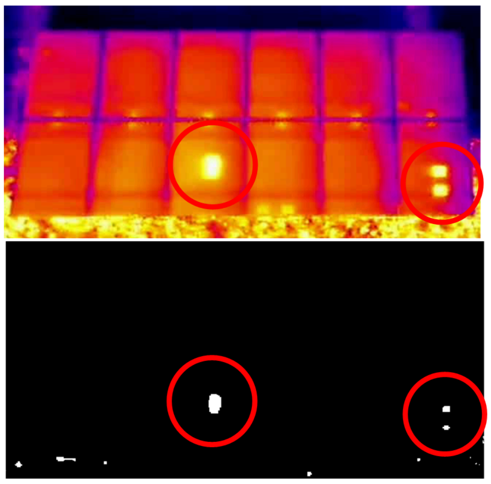


Fig. 7: The thermal image (Upper) and the binary image (Lower) of the thermal image

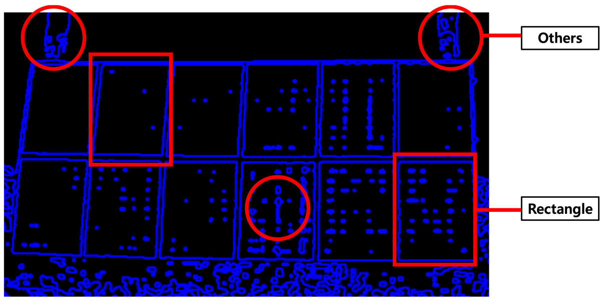


Fig. 8: The result of the canny edge detection in the binary image

Figure (8) shows the result of the canny edge detection in the binary image. It shows all of the edges, including the solar panel. The solar panel has a rectangular shape; we find the rectangle with a Ramer-Douglas-Peucker algorithm for clearing edges (Douglas and Peucker, 1973). Moreover, a Harris corner detection (Harris and Stephens, 1988) is used to find corners. As shown in Figure (7), only the solar panel is rectangular. So, if the edge has four corners, it means the solar panel. So, the algorithm for detecting the solar panel follows Algorithm 2. The image denotes the binary image that is the result of the preprocessing. Then, Figure (8) shows the result of the detection of the solar panels.

```
Algorithm 2: Detect the Solar panel
function detectSolarPanel(img);

Input: The binary image img

# list of the solar panel
SolarPanel[ ] = { }

# list of the solar panel array
SolarPanelArray[ ] = { }

# Canny edge detection
resultGaussian = gaussianFiltering(img)
edges = findingEdge(resultGaussian)
maximanEdges = localMaxima(edges)
contours = filteringWeakEdge(maximanEdges)

# searching the rectangle contours
    for i ← 0 to numbers of contours

        # make the contour clearly
        dp = douglasPeucker(contour[i])

        # find the corner of the contour
        corner = harrisCorner(dp)

        # when the contour is rectangle
        if corner is 4 then
            if contour.hierarchy is outer then
                SolarPanelArray.add(contours)
            else if contour.hierarchy is inner then
                SolarPanel.add(contours)
end
```

Figure (9a) is the result of filtering that only left the rectangle contour. Figure (9b) shows whether the rectangle contour is correct. Blue contours are the rectangles, and White contours are not. The contours resulting from the finding solar panel algorithm have a hierarchy. Figure (10) shows an example of the hierarchy. The outer contour has inner contours, which are in the outer contour. We classify the solar panel and solar panel array according to the hierarchy. Because the border of the solar panel array is the same shape as the shape of the solar panel. So, the result of the detecting algorithm is the list of the solar panels and the list of the solar panel array.

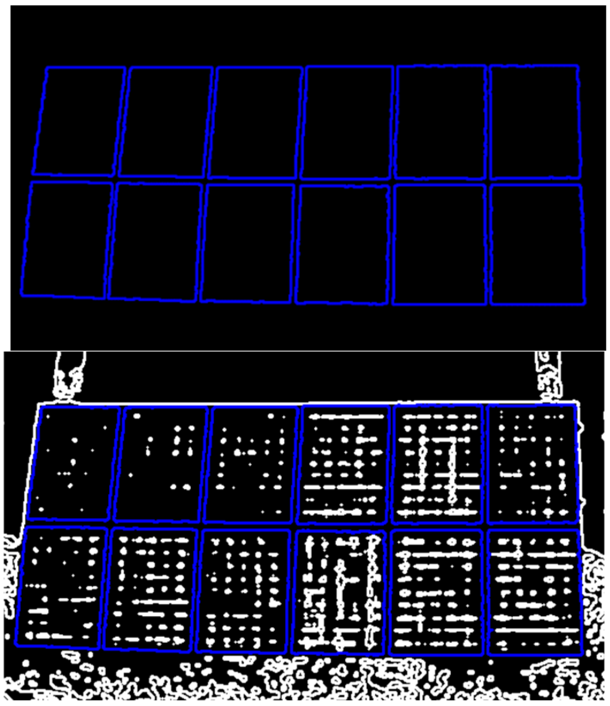


Fig. 9: The result of the find solar panel with the feature

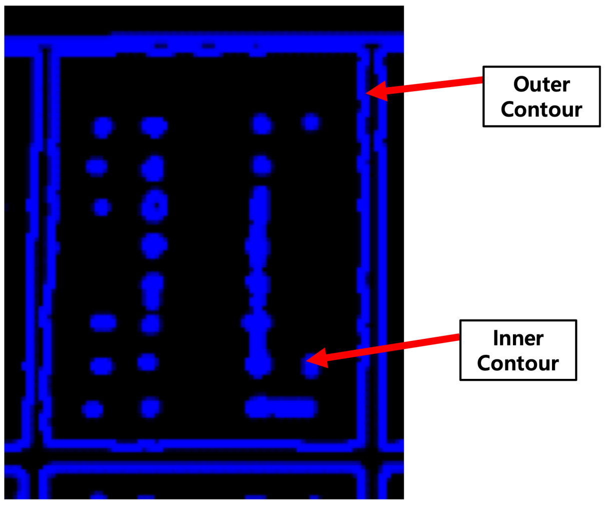


Fig. 10: The example of the hierarchy

To specify the faulty solar panel, we check the location of the hot spot. The checking of the hot spot follows Algorithm 3. SolarPanelArray denotes the list of solar panels, and hotSpotArray denotes the list of hot spots. If the hot spot is placed on the solar panel, we draw the red rectangle on the border of the solar panel. If not, then the solar panel is typical. So, we draw the blue rectangle on the border of the solar panel, as shown in Figure (11). The system draws and stores solar power plants and locations of faulty solar panels.

```
Algorithm 3: Find the Faulty Solar Panel
function findFaulty(SolarPanelArray[ ], hotSpotArray[
]);

Input: the list of the solar panel SolarPanelArray[ ]
the list of the hot spots hotSpotArray[ ]

# checking all of the solar panels
    for i ⟵ 0 to the number of Solar panels
        for j ⟵ 0 to the number of hot spots
            if solar panel inside hot spot then
            do draw red rectangle on the solar panel
            else
            do draw green rectangle on the solar panel
```

Figure (12) shows a flowchart of the proposed system, and Figure (13) shows the hardware of the proposed system. The user can control the drone with a smartphone or wireless controller. The drone has four wings and can fly for almost 15 minutes with a fully charged battery. The RGB and thermal cameras are mounted at the lower part of the drone with a gimbal, which can minimize the shaking of the drone by the wind. We applied a sender to the drone and a receiver to a laptop computer to collect videos using the drone. The sender sends the RGB and thermal videos to the receiver placed on the drone. The receiver gets the videos that the sender sent. This means that we can show the videos from the laptop and save them to the laptop computer. Then, we can get the result of the proposed system with saved video.

The application of the proposed system works on Windows 10 with Java runtime environment eight and OpenCV 3.2 dynamic library to process the image processing. The main UI of the system is shown in Figure (14). The user can input the video sequentially as the RGB and thermal with the open dropdown menu.

When the RGB video and the thermal video can be input, the application shows the video sequence by slicing the RGB video in 10 sec. It shows the summary of the RGB video in the white area on the left. Also, we can select the starting and ending points of the RGB video because the video did not only show solar systems. It can record when the drone is going up to the proper height or going down to the end of the flight. So, if the user cannot consider editing the videos, it is easy to check the section that the user wanted. If the RGB and thermal video are set typically, the start button is activated and is placed on the bottom right of the main UI. When the start button is pressed, the algorithm is processed. The main UI's black area on the left side plays the RGB video. The bottom text area on the left side of the main UI shows the system manual and a system log for debugging. The white area on the right side of the main UI shows a map of the solar power plant system drawn by the system. With the map, users can easily find faulty panels where they are placed. The bottom black area on the right of the main UI results from the proposed processing and the thermal image.

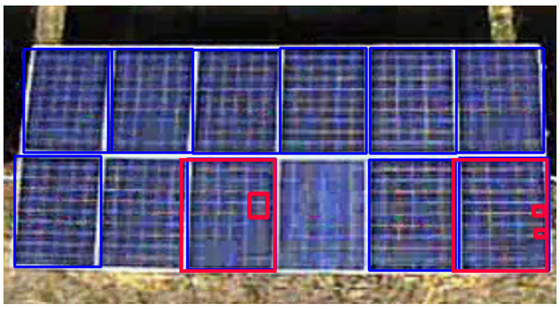


Fig. 11: The result of the detecting faulty solar panel

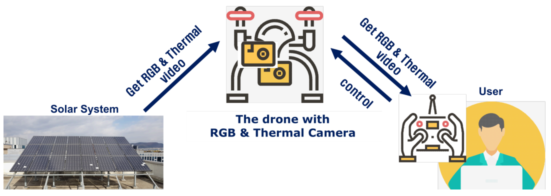


Fig. 12: The flowchart of the proposed system

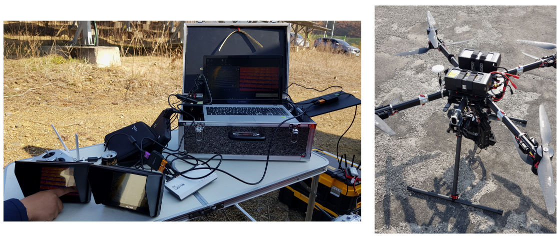
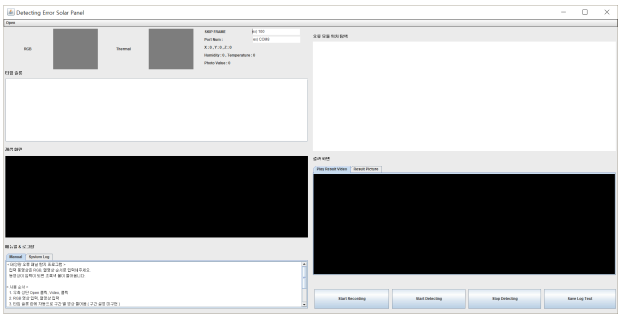


Fig. 13: The proposed system



SKIP FRAME ex) 100
Port Num : ex) COM8
X : 0 , Y : 0 , Z : 0
Humidity : 0 , Temperature : 0
Photo Value : 0

< 태양광 오류 패널 탐지 프로그램 >
입력 동영상은 RGB, 열영상 순서로 입력해주세요.
동영상이 입력이 되면 초록색 불이 들어옵니다.

< 사용 순서 >
1. 우측 상단 Open 클릭, Video, 클릭
2. RGB 영상 입력, 열영상 입력
3. 타임 슬롯 란에 자동으로 구간 별 영상 들어옴.(구간 설정 미구현)

Fig. 14: The proposed application of detecting faulty solar panel system

Results and Discussion

We tested the proposed system with two metrics. First, the proposed system could find the solar panel or faulty solar panel in a variable height. Finding the optimal height of the drone gives the system high precision. Second, we tested the system on how much tolerance there is in the image of noise. Because the height of the drone makes the image noise. Most of the height of the fly is almost 70m ~ 90m. Note that it can occur quickly in other environments. Figure (15) shows an example of the noise of the image. The red rectangle shows blocking noises, and the dotted rectangle shows line noises. So, we need to test the system using the image's noise to test the proposed system's accuracy. Then, we tested the proposed system in two real working solar power plant systems to find the accuracy of detecting faulty solar panels.

Experimental Environment

We tested the proposed system in Windows 10 with i7-4550u, DDR3L 8GB, Intel HD 5000, Java runtime environment 8 and OpenCV 3.2. The RGB and thermal cameras have 1280×720 resolutions.

We tested the accuracy of the proposed system at various heights. The drone can fly at the fixed height that the user wants. So, we made the different videos we already had. We flew the drone to almost 90m and 40 m. We used two methods to make the various heights. First, we set a Region of Interests (ROI) size of 320×180, then rescaling the origin image progressively from 1280×720 to ROI. Then, we can see the two types of the test set as shown in Figure (16a) is the rescaling of the 41 m of the origin video and (b) is the rescaling of the 90m of the origin video. The other method is to scale up with an ROI size of 1280×720. The scale-down method has no blocking noise but a low resolution. The scale-up method has a high resolution but has blocking noises. So, we got the three test sets, as shown in Table (1). The first test set has one solar panel array of 12 solar panels. Then, the second test set has two arrays; each solar panel array consists of 16 and 32 solar panels. Then, the other test set has three solar panel arrays that consist of 32, 48 and 48 solar panels. We evaluate the accuracy of the solar panel by the number of solar panels divided by the number of detected solar panels.

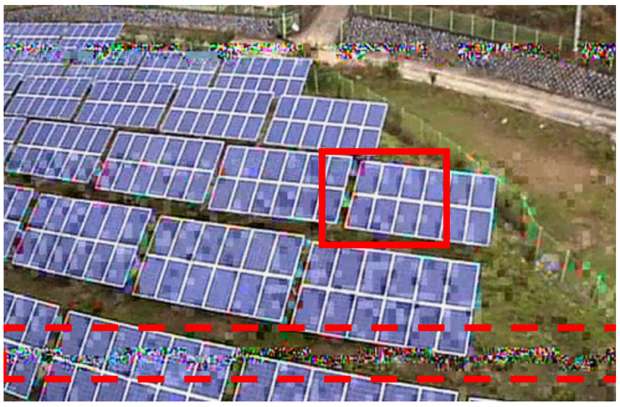


Fig. 15: The example image of the noised image

Table 1: Test sec for accuracy in the various height

Test set	No. of Solar Panel Array	No. of Solar Panel in each Array
A	1	12
B	2	16, 32
C	3	32, 48, 48

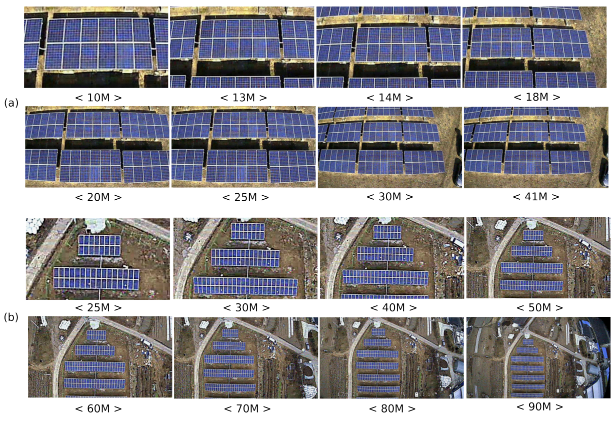


Fig. 16: Solar panel images from different height

Because long-range transmission can cause noise, we tested the proposed system in the noise. Therefore, we generate the noise with a Gaussian distribution. The Gaussian distribution is usually used to make a noise based on the noise that occurs in a general situation. We make the noise with mean and standard deviation. We use the signal-noise ratio (SNR) and SNR decibel to check how much noise is in the image. The SNR follows Formula (2). The *f* denotes the original image, and the ^*f* represents the noise image that we created. Following Formula (2), first, calculate the difference between the original image and the noise image, then square the pixel of the noise image and sum all pixels, then divide by the differences between the original image and the noise image that was calculated before. We selected the noise image generated by Formula (2). We used the SNR_{db} by Formula (3) to choose the noise image. So, if SNR_{dB} has a high decibel, the noise image is not entirely different from the origin image. Then, if SNR_{db}has a low decibel, it means that the noise image has much noise. Figure (17) shows examples of the first test set 41 m image. With the Gaussian noise:

$$SNR = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\hat{f}(x,y)^2}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x,y)-\hat{f}(x,y)\right]^2} \tag{2}$$

$$SNR_{dB} = 10\log_{10}(SNR) \tag{3}$$

Table (2) and Figure (18) show that the proposed system was tested in two real working solar power plants. We made a factitious spot on the solar panel with sticky papers. The fictitious spot was the same as the actual hot. spot. Then, we flew the drones 3 times in each place. We got six RGB and six thermal videos from the testing flight. Then, the drone flew at different heights at test time. However, the testing video had lots of noise to test. Also, the RGB and thermal cameras had different zoom factors and a frame rate. We resized the RGB and thermal videos for the experiment to see the same region with the same zoom factor and frame rate. So, we tested the proposed system with the two videos from each testing place.

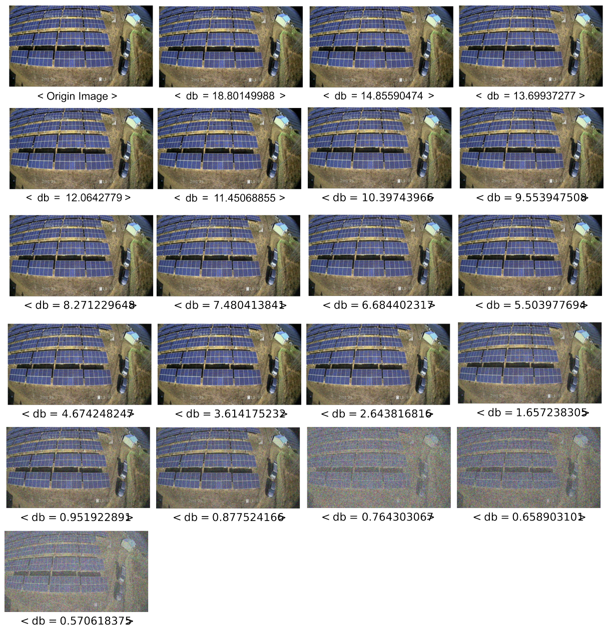


Fig. 17: The example of the first test set 41 m image with the Gaussian noise

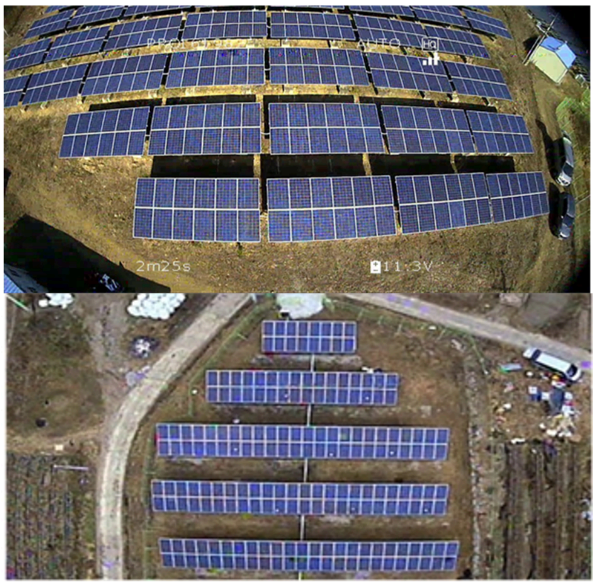


Fig. 18: The image the testing place. (Upper) Site A. (Lower) Site B

Table 2: Experimental environment

	Place	No. of Panel Array	No. of Panel	No. of Faulty Panel
Site A	Geochang	5	192	8
Site B	Changnyeong	4	48	15

Experimental Result and Discussion

Figures (19-21) show the accuracy at various heights with 320×180 resolution. The origin resolution of the image in Figures (19-20) are recorded at 41 and 90 m in height. The lower height is made by scaling down. They have no additional blocking noise.

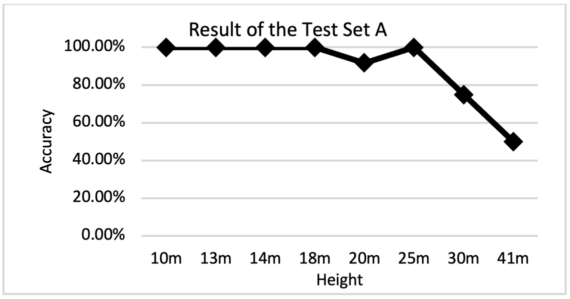


Fig. 19: Accuracy in the test set A

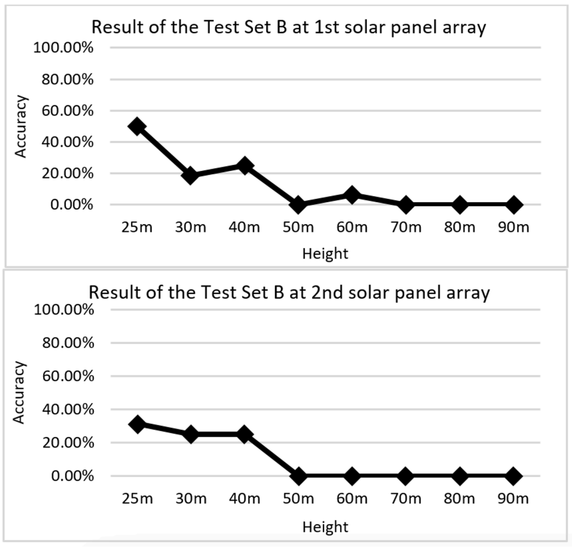


Fig. 20: Accuracy in the test set B

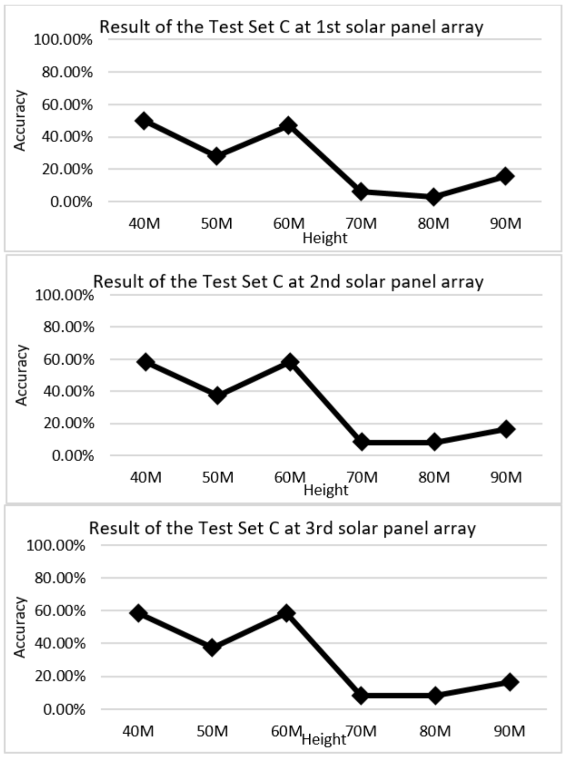


Fig. 21: Accuracy in the test set C

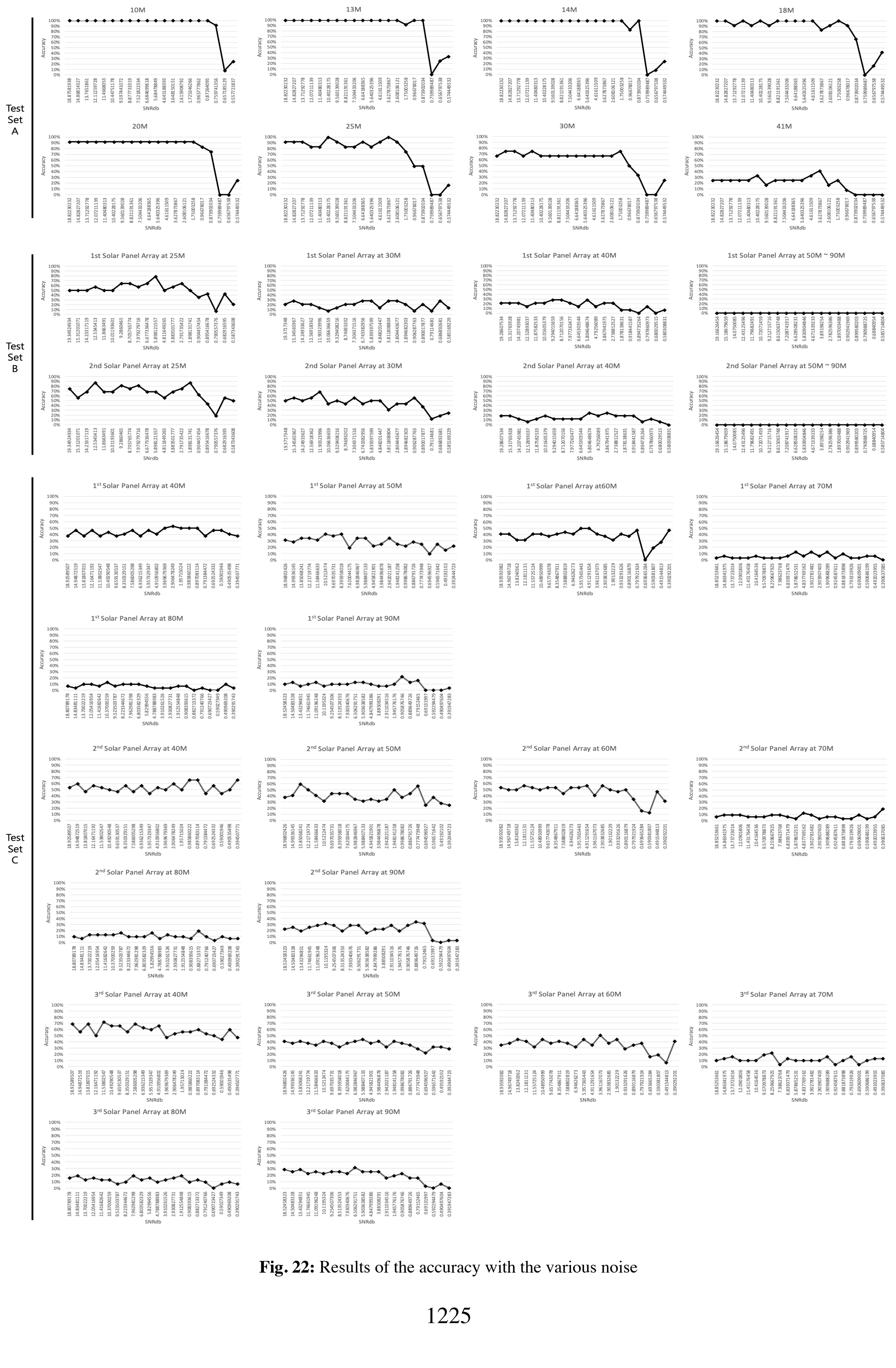
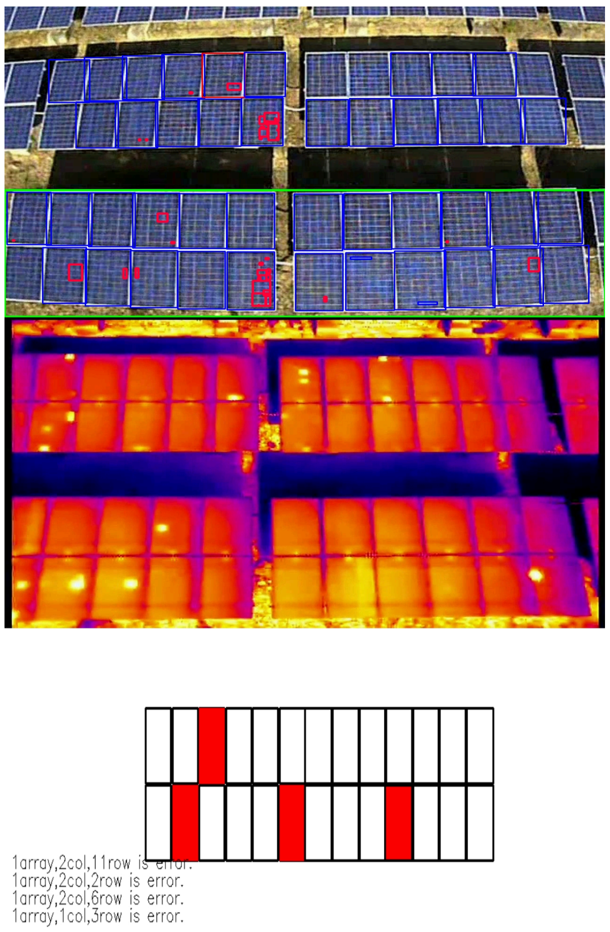


Fig. 22: Results of the accuracy with the various noise

Figure (21) shows the accuracy result at the various heights with 1280×720. The origin image is recorded at 90 m in height. The test set C has three solar panel arrays. Each accuracy decreases when height is increases. If a drone flies under 30 m, the proposed system can capture most of the solar panel.

We tested our proposed system to determine how much tolerance there is in the noise with test sets. Figure (22) shows the accuracy of the various noises. The result with test set A shows that the proposed system had a meaningful result before 0.76 dB and 30 m. At 10, 13, 14, 20 and 30 m in height, our system has consistency accuracy before 0.76 dB. However, the accuracy of 25 m shows that the proposed system is sensitive to noise. Then, the 41 m still has low accuracy.

The result with test set B shows that the proposed system had a meaningful result before 0.76 dB and 30 m. The first solar panel array shows anomaly results when the SNR_{db} is 5.0, which is the highest accuracy. At 30 m height, the noise is not affected by the proposed system. Then, the 40 m shows that the noise decreased the accuracy. The second solar panel array shows that the noise decreases at 25, 30, and 40 m. Both arrays have 0% accuracy from 50m to 90 m. Because of the noise, it was still 0% accuracy.

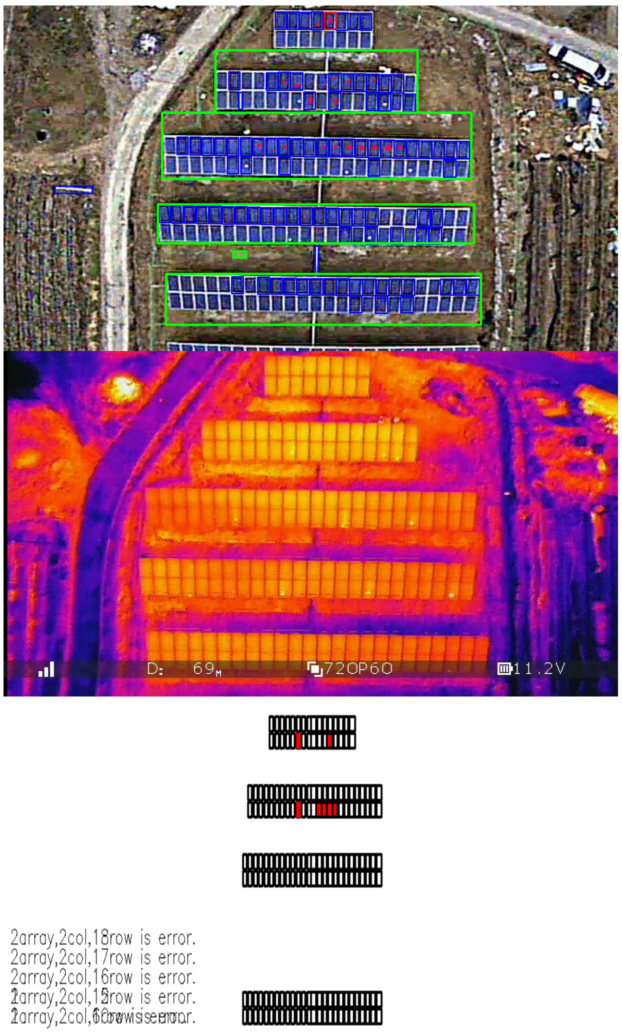


```
1array,2col,11row is error.
1array,2col,2row is error.
1array,2col,6row is error.
1array,1col,3row is error.
```

Fig. 23: The result image of the site B video

The test set C results show that the proposed system has a tolerance for noise. We can see that the accuracy of the 40, 50, and 60 m are decreasing, but before 0.629 SNR_{db}, they had similar accuracy. At 70, 80 and 90 m height, our system has low accuracy and cannot figure out meaningful data. At 60m height, the anomaly that the maximum noise isn't easy to detect the solar panel array with the human eyes. However, the result shows the same accuracy with the minimum noise. The result of the second solar panel array is similar to the first solar panel array. In the result of 40 m, the accuracy is consistent with comparable accuracy. The accuracy is decreased at 50, 60, and 90 m in height. At 70 and 80 m height, there are no meaningful results. As with the same result as the first noise, at 60 m height, an anomaly in the maximum noise was found. The result of the last array shows that the accuracy is decreasing at all heights. At 60 m in height, the result is the same as at 40 m.

The proposed system has a tolerance before almost 0.7 SNR_{db}. Some results show the anomaly. However, the proposed system is consistent and accurate in the noise image generated by the Gaussian noise.



```
2array,2col,18row is error.
2array,2col,17row is error.
2array,2col,16row is error.
2array,2col,15row is error.
2array,2col,60rowisiserror.
```

Fig. 24: The result image of the site A video

Table 3 shows the result of the experiment. Six faulty solar panels of site A were found. The accuracy of site A is 62.5%. Twenty-two faulty solar panels of site B were found. The accuracy of site B is 93%. An RGB video of site A had low quality. Moreover, the video was captured at a high altitude. This problem made the features of solar panels ambiguous. The features of solar panels on site B were more distinct than those on site A. This difference made different accuracies in the same method.

Table 3: Experimental results

	No. of Faulty Panel	No. of Detected Faulty panel	Accuracy
Site A	8	5	62.5%
Site B	15	14	93%

Figure (23) shows one of the resulting images tested in site A. The top image detects solar panels and arrays. The solar panel is shown in blue rectangles. The solar panel array is shown in a green rectangle. The bottom image is a generated map of the solar plant. Detected panels draw the map. When a faulty solar panel is detected, it is filled with red color with text. Figure (24) is the resulting image at site B. The total process is shown in Figure (25). The application process is shown in Figure (26). All of the detected faulty solar panels are confirmed by a solar panel expert.

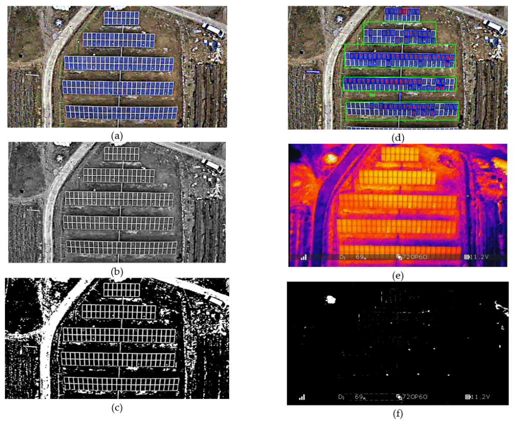
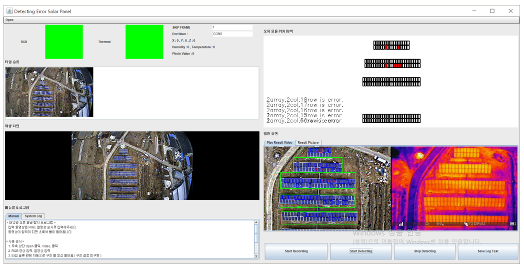


Fig. 25: (a) The RGB image which get from the RGB video sequence. (b) The result of converting the RGB image to the grayscale image. (c) The result of getting the binary image from the grayscale using otsu threshold. (d) The result of getting the solar panel and the solar panel array. (e) The thermal image that captured by the thermal camera on the drone. (f) The result of getting the binary image from the thermal image using static threshold


Detecting Error Solar Panel

SKIP FRAME: 1
Port Num: COM8
X:0, Y:0, Z:0
Humidity: 0, Temperature: 0
Photo Value: 0

```
2array,2col,18row is error.
2array,2col,17row is error.
2array,2col,16row is error.
2array,2col,13row is error.
2array,2col,6row is error.
```

Fig. 26: The proposed application of detecting faulty solar panel system during detecting of faulty solar panel

Conclusion

We proposed a real-time system for detecting faulty solar modules with a drone, an RGB camera and a thermal camera. To find the faulty solar panel, we used features of hot spots, the shape of solar panels and the surface of solar panels. The thermal camera was used to capture the hot spot areas. The RGB camera was used to capture the solar panels and arrays. We used the canny edge algorithm and Harris corner detection to find the solar panels and arrays' locations. The system was tested in two real working solar power plants. We proved that our system worked and got meaningful results.

In the future, we will change the RGB and the thermal cameras to get the same frame rate and zoom factor. The map of the solar power plant system is not exactly the same as the real environment. We need more research on image stitching with a continuous image using similar patterns and shapes and a deep learning-based approach.

Acknowledgement

The authors did not receive support from any organization for the submitted work.

Funding Information

The authors have not received any financial support or funding to report.

Author's Contributions

All authors equally contributed to this study.

Ethics

This manuscript is an original work. The corresponding author declares that no ethical concerns are associated with this submission.

References

Ancuta, F., & Cepisca, C. (2011). Failure Analysis Capabilities for PV Systems. *Recent Researches in Energy, Environment, Entrepreneurship, Innovation*, 109-115.

Canny, J. (1987). A Computational Approach to Edge Detection. *Readings in Computer Vision*, 184-203. https://doi.org/10.1016/b978-0-08-051581-6.50024-6

Carletti, V., Greco, A., Saggese, A., & Vento, M. (2020). An intelligent flying system for automatic detection of faults in photovoltaic plants. *Journal of Ambient Intelligence and Humanized Computing*, *11*(5), 2027-2040. https://doi.org/10.1007/s12652-019-01212-6

Chamberlin, C. E., Rocheleau, M. A., Marshall, M. W., Reis, A. M., Coleman, N. T., & Lehman, P. A. (2011). Comparison of PV module performance before and after 11 and 20 years of field exposure. *2011 37th IEEE Photovoltaic Specialists Conference*, 000101-000105. https://doi.org/10.1109/pvsc.2011.6185854

Chao, K.-H., Tsai, J.-H., & Chen, Y.-H. (2019). Development of a Low-Cost Fault Detector for Photovoltaic Module Array. *Electronics*, *8*(2), 255. https://doi.org/10.3390/electronics8020255

Deng, G., & Cahill, L. W. (2005). An adaptive Gaussian filter for noise reduction and edge detection. *1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference*, 1615-1619. https://doi.org/10.1109/nssmic.1993.373563

Dhanraj, J. A., Mostafaeipour, A., Velmurugan, K., Techato, K., Chaurasiya, P. K., Solomon, J. M., Gopalan, A., & Phoungthong, K. (2021). An Effective Evaluation on Fault Detection in Solar Panels. *Energies*, *14*(22), 7770. https://doi.org/10.3390/en14227770

Dotenco, S., Dalsass, M., Winkler, L., Wurzner, T., Brabec, C., Maier, A., & Gallwitz, F. (2016). Automatic detection and analysis of photovoltaic modules in aerial infrared imagery. *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1-9. https://doi.org/10.1109/wacv.2016.7477658

Douglas, D. H., & Peucker, T. K. (1973). Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature. *Cartographica*, *10*(2), 112-122. https://doi.org/10.3138/fm57-6770-u75u-7727

Gokmen, N., Karatepe, E., Silvestre, S., Celik, B., & Ortega, P. (2013). An efficient fault diagnosis method for PV systems based on operating voltage-window. *Energy Conversion and Management*, *73*, 350-360. https://doi.org/10.1016/j.enconman.2013.05.015

Guasch, D., Silvestre, S., & Calatayud, R. (2003). Automatic failure detection in photovoltaic systems. *3rd World Conference OnPhotovoltaic Energy Conversion, 2003. Proceedings Of*, 2269-2271.

Harris, C., & Stephens, M. (1988). A Combined Corner and Edge Detector. *Procedings of the Alvey Vision Conference 1988*, 23.1-23.6. https://doi.org/10.5244/c.2.23

Harrou, F., Sun, Y., & Saidi, A. (2017). Model-based fault detection algorithm for photovoltaic system monitoring. *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1-5. https://doi.org/10.1109/ssci.2017.8285435

Hu, Y., Gao, B., Song, X., Tian, G. Y., Li, K., & He, X. (2013). Photovoltaic fault detection using a parameter based model. *Solar Energy*, *96*, 96-102. https://doi.org/10.1016/j.solener.2013.07.004

Ji, D., Zhang, C., Lv, M., Ma, Y., & Guan, N. (2017). Photovoltaic Array Fault Detection by Automatic Reconfiguration. *Energies*, *10*(5), 699. https://doi.org/10.3390/en10050699

Kim, K. A., Seo, G.-S., Cho, B.-H., & Krein, P. T. (2016). Photovoltaic Hot-Spot Detection for Solar Panel Substrings Using AC Parameter Characterization. *IEEE Transactions on Power Electronics*, *31*(2), 1121-1130. https://doi.org/10.1109/tpel.2015.2417548

Liu, S., Dong, L., Liao, X., Cao, X., & Wang, X. (2019). Photovoltaic Array Fault Diagnosis Based on Gaussian Kernel Fuzzy C-Means Clustering Algorithm. *Sensors*, *19*(7), 1520. https://doi.org/10.3390/s19071520

Liu, Y., Li, B., & Cheng, Z. (2010). Research on PV module structure based on fault detection. *2010 Chinese Control and Decision Conference*, 3891-3895. https://doi.org/10.1109/ccdc.2010.5498470

Lu, S., Phung, B. T., & Zhang, D. (2018). A comprehensive review on DC arc faults and their diagnosis methods in photovoltaic systems. *Renewable and Sustainable Energy Reviews*, *89*, 88-98. https://doi.org/10.1016/j.rser.2018.03.010

Madeti, S. R., & Singh, S. N. (2017). Online fault detection and the economic analysis of grid-connected photovoltaic systems. *Energy*, *134*, 121-135. https://doi.org/10.1016/j.energy.2017.06.005

Malof, J. M., Hou, R., Collins, L. M., Bradbury, K., & Newell, R. (2015). Automatic solar photovoltaic panel detection in satellite imagery. *2015 International Conference on Renewable Energy Research and Applications (ICRERA)*, 1428-1431. https://doi.org/10.1109/icrera.2015.7418643

Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), 62-66. https://doi.org/10.1109/tsmc.1979.4310076

Patil, T. G., & Asokan, A. (2016). A proficient solar panel efficiency measurement system: Using current measurements. *2016 International Conference on Communication and Electronics Systems (ICCES)*, 1-6. https://doi.org/10.1109/cesys.2016.7889927

Pei, T., & Hao, X. (2019). A Fault Detection Method for Photovoltaic Systems Based on Voltage and Current Observation and Evaluation. *Energies*, *12*(9), 1712. https://doi.org/10.3390/en12091712

Renewable Energy Focus. (2009). IRENA greeted as milestone for renewables. *Renewable Energy Focus*, *9*(7), 19. https://doi.org/10.1016/s1755-0084(09)70029-8

Salamanca, S., Merchan, P., & Garcia, I. (2017). On the detection of solar panels by image processing techniques. *2017 25th Mediterranean Conference on Control and Automation (MED)*, 478-483. https://doi.org/10.1109/med.2017.7984163

Selvaraj, T., Rengaraj, R., Venkatakrishnan, G., Soundararajan, S., Natarajan, K., Balachandran, P., David, P., & Selvarajan, S. (2022). Environmental Fault Diagnosis of Solar Panels Using Solar Thermal Images in Multiple Convolutional Neural Networks. *International Transactions on Electrical Energy Systems*, *2022*, 1-16. https://doi.org/10.1155/2022/2872925

Suzuki, S., & be, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, *30*(1), 32-46. https://doi.org/10.1016/0734-189x(85)90016-7

Tao, Y., Zhang, M., & Parsons, M. (2017). Deep learning in photovoltaic penetration classification. *2017 IEEE Power & Energy Society General Meeting*, 1-5. https://doi.org/10.1109/pesgm.2017.8274395

Yao, Y., & Hu, Y. (2017). Recognition and location of solar panels based on machine vision. *2017 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, 7-12. https://doi.org/10.1109/acirs.2017.7986055

Yuan, J., Yang, H.-H. L., Omitaomu, O. A., & Bhaduri, B. L. (2016). Large-scale solar panel mapping from aerial images using deep convolutional networks. *2016 IEEE International Conference on Big Data (Big Data)*, 2703-2708. https://doi.org/10.1109/bigdata.2016.7840915

Zhang, X., Sun, H., Zhou, Y., Xi, J., & Li, M. (2013). A Novel Method for Surface Defect Detection of Photovoltaic Module Based on Independent Component Analysis. *Mathematical Problems in Engineering*, *2013*, 1-8. https://doi.org/10.1155/2013/520568