Applying OCR in the Processing of Thermal Images with Temperature Scales

W.T. Chan, T.Y. Lo, C.P. Tso and K.S. Sim

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/51475

1. Introduction

Present-day thermal image processing is dependent on the use of metadata. This metadata, such as colour-to-temperatures indices that help the conversion of the colour value of every pixel in the image into a temperature reading, may be stored within the image files as supplementary information that is not immediately apparent from a glance of the image. Instead, they are kept within the image bytes, to be fetched by metadata-dependent programs that had been designed to examine to work with said metadata.

However, reasons such as company policies on the protection of information or circumstances such as data corruption may cause the loss of metadata. The user may also derive thermal images from sources that do not have metadata, such scans from thermal imaging magazines. In the absence of metadata, the aforementioned programs do not work. Therefore, other methods have to be used to derive the needed information. One of these is the use of optical character recognition.

Optical character recognition (OCR) is the machine equivalent of human reading [1]. Research and development of OCR-based applications has mainly focused on having programs read text that have been written in languages that do not use the Latin alphabet, such as Japanese and Arabic [2]. However, there have been endeavours in more industrious applications, such as having cameras read license plates [3]. This book chapter is dedicated to an application of the latter sort.

The use of OCR scripts is meant to complement the usual method of processing thermal images, which is using metadata, instead of substituting it. This is because fetching and reading metadata for processing parameters is still significantly faster than having the program read temperature scales and derive processing parameters from them. In other words, it is more efficient to use metadata when it is available, and resort to OCR when there is not any. However, OCR has the benefit of being usable on any thermal image as long as it has a temperature scale, and generally, thermal images do have this.

Implementing OCR has challenges, most of which stem from weaknesses of the methods used to have the program capture shapes from the thermal image and recognize these as characters. The quality of the thermal image itself also presents challenges of its own.

This book chapter will elaborate on a method, which has its inspirations from a MatLab project [4]. This method uses a library of pre-existing alphabetical and numerical characters as references to be compared with the shapes of objects, including those of text, captured from a thermal image. The disadvantages of this method and the solutions to overcome it will be mentioned later, along with the results of the comparisons between programs that use this method, manual input and the checking of metadata.

2. Body

2.1. Problem statement

The first hurdle in implementing OCR is determining the kinds of information that could be derived from the thermal image, if it lacks the metadata that conveniently provides the parameters needed to process the image with. The solution is to have the program examine the temperature scale that is included with the image and derive the temperature range from it. This temperature range is needed for association with the colour range, so that conversion from colour values to temperature readings can be made.

The next problem is to have the program search for the temperature scale without resorting to hard-coding that points that program at a specific region of the image. Hard-coding may be desirable if efficiency is valued over versatility, but the program can only process thermal images of specific formats as a result. The idea of the solution for this can be obtained from examining the layout of the thermal image first. If the temperature scale is placed separately from the image subject, i.e. not imposed onto the image subject itself so as to obscure it, then a separation of the two suffices. Such a layout is usually the case for the majority of thermal images; an example is shown in **Figure 1**.

The next complication is figuring out what aspects of the scale would give the necessary information for the processing. The temperature labels on the scale would give this, but there may be several numbers on it depicting both the intervals in the scale and its limits. Picking the highest and lowest numbers would give the maximum and minimum temperatures in the scale, respectively. The physical distances between the interval labels on the scale can also denote whether the scale is linear or logarithmic.

Figuring out which shapes belong to the characters that make up the numbers in the temperature labels is also a challenge. When designing the program to recognize characters as part of a number, the developer should keep in mind how a human recognizes numbers. The key characteristic of coherently displayed numbers is that the characters in numbers are adjacent to each other and are usually aligned in the same orientation. Therefore, he/she can include these considerations in the program as logical processes.

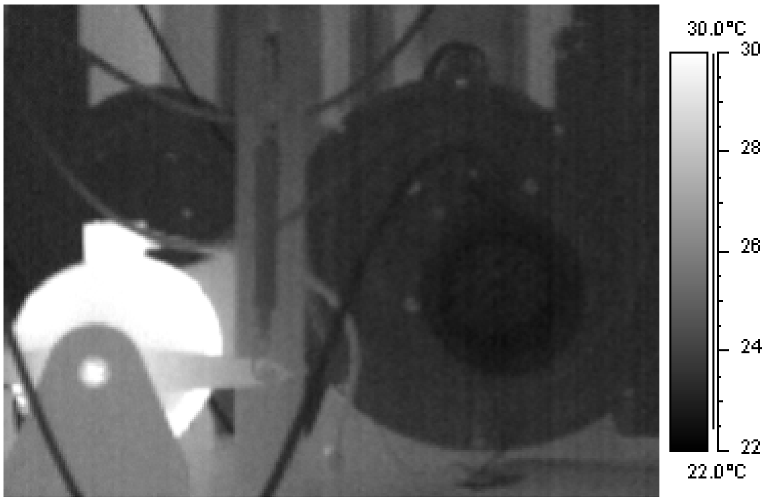

Figure 1. Thermal image of a TE84 Centrifugal Fan Test Set

Recognizing which shapes are characters and which are not is important because the processing parameters have to be either coherent numbers or words. Having strings of characters infiltrated by shapes that are mistakenly recognized as characters can result in errors. The use of a correlation coefficient offers a solution for this through the implementation of a threshold value that the shape must surpass if it is to be considered a character.

In summary, most of the problems that would be faced in the development program concerns the processes used to implement OCR. Most of them can be overcome as long as the definition of OCR as the machine equivalent of human reading is kept in mind, i.e. the processes in the OCR scripts are digital versions of the logic processes that humans use to recognize and identify characters in written languages.

2.2. Application area

The area that the output of this research is immediately useful in is research in thermal image processing itself. Such research requires the analysis of many thermal images, and samples of these may not be sourced from original thermal images but rather from other media such as science magazines. Such other media obviously do not support the inclusion of metadata in these thermal images, so using an OCR-assisted program on scans or crop-outs of thermal images from these media can be useful.

Preventive maintenance in the industrial sectors may use thermal imaging as one of the tools for examining systems and machines, but company policies on information may prevent the inclusion of metadata in thermal images that are to be used in day-to-day operations or for archiving. Therefore, an OCR-assisted program can provide a solution if already generated thermal images that have to be analyzed again happen not to have metadata.

2.3. Research course

The course of this research is to develop a program that can utilize OCR to automatically process batches of thermal images and obtain the information that is usually wanted from a thermal image, such as a chart of temperature readings, one for each pixel on the image. The program must produce the same results as a program that uses manual user input, for the program to be considered a viable upgrade over the latter at processing batches of thermal images without metadata.

The techniques used in implementing OCR can be swapped out for others to achieve better processing times. This is so that the gap in processing time between programs that use OCR and those that rely on metadata can be reduced, such that the former can be more competitive and thus reduce the reliance on metadata.

2.4. Methodology

The work-flow of the program is shown in **Figure 2**, **Figure 3** and **Figure 4**. Firstly, the program looks into a directory where thermal images are stored, fetches and makes copies of these. The copies are analyzed and manipulated on throughout the flow of the program, in order to preserve the originals.

Next, the program determines the range of colours that represent the range of temperatures in the image. This is best done by examining the temperature scale that came with the image, so the scale has to be isolated from the image subject. This also helps the capture of characters that make up the temperature labels, if the image subject happens to have paint or labelling that emit infrared radiation on its surface; see **Figure 5** for illustration. Preventing the image subject from being subjected to OCR scripts prevents confusion.

To separate the image subject from the temperature scale, converting the entire image to a binary image changes the objects into discrete blobs that can be identified using connected-component labelling. The image subject should be reduced to a mass of pixels that is the largest object in the entire image, so scripts that check the sizes of objects can identify and isolate it. Scripts can also be introduced to differentiate the colour strip of the scale from its temperature labels, such as through the fact that the strip is generally the larger object.

Any remaining shapes in the region of the image that the temperature scale is located in should belong to characters that compose the temperature labels for the scale, but there may also shapes for blemishes that may have been captured as well, especially if the thermal image came from a scan of a printed image. Therefore, the OCR scripts are performed on these to identify whether they resemble a character or not. Those that are not are discarded.

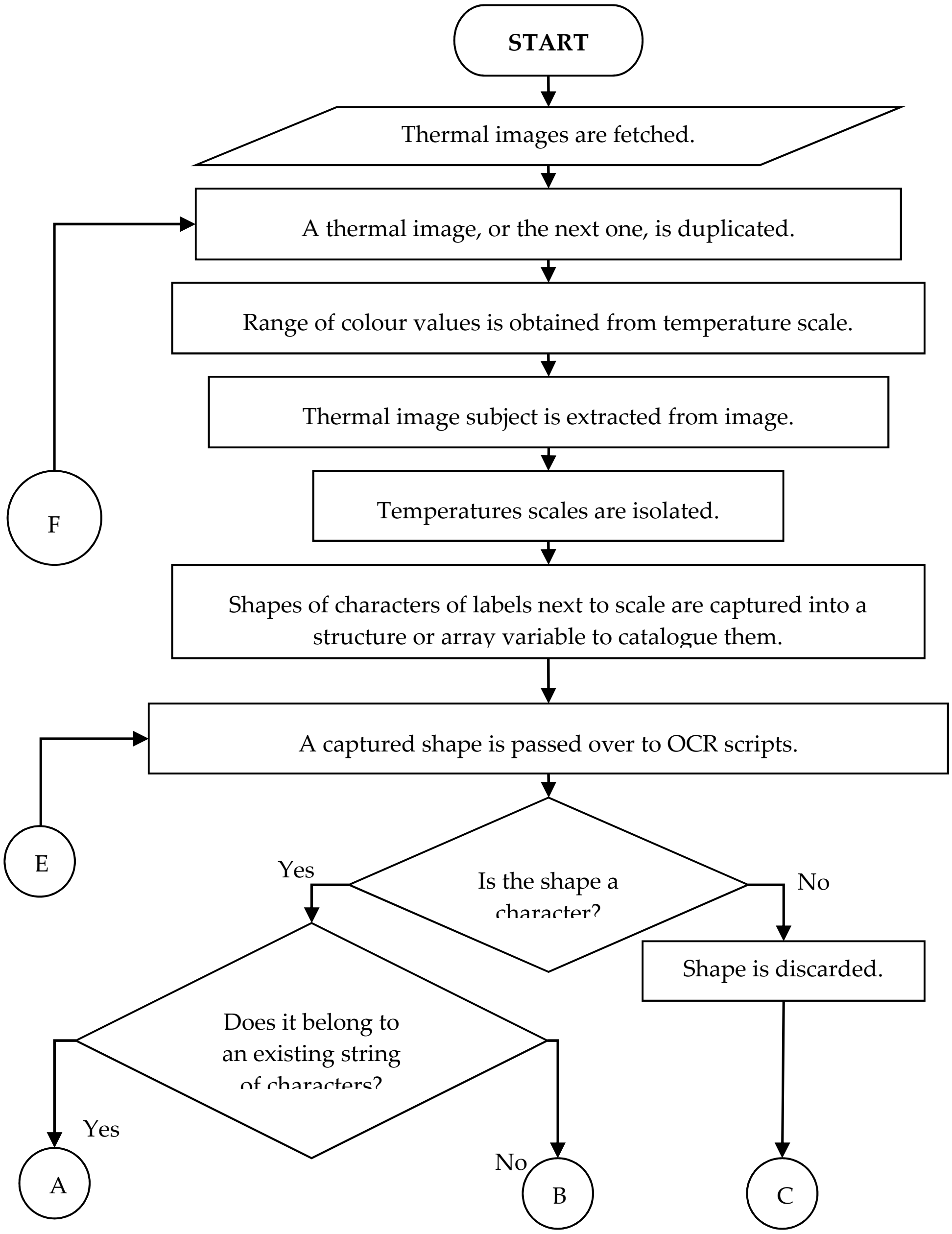
START
Thermal images are fetched.
A thermal image, or the next one, is duplicated.
Range of colour values is obtained from temperature scale.
Thermal image subject is extracted from image.
Temperatures scales are isolated.
Shapes of characters of labels next to scale are captured into a structure or array variable to catalogue them.
A captured shape is passed over to OCR scripts.
Is the shape a character?
Yes
No
Shape is discarded.
Does it belong to an existing string of characters?
Yes
No
A
B
C
E
F

Figure 2. Workflow of program (Part 1)

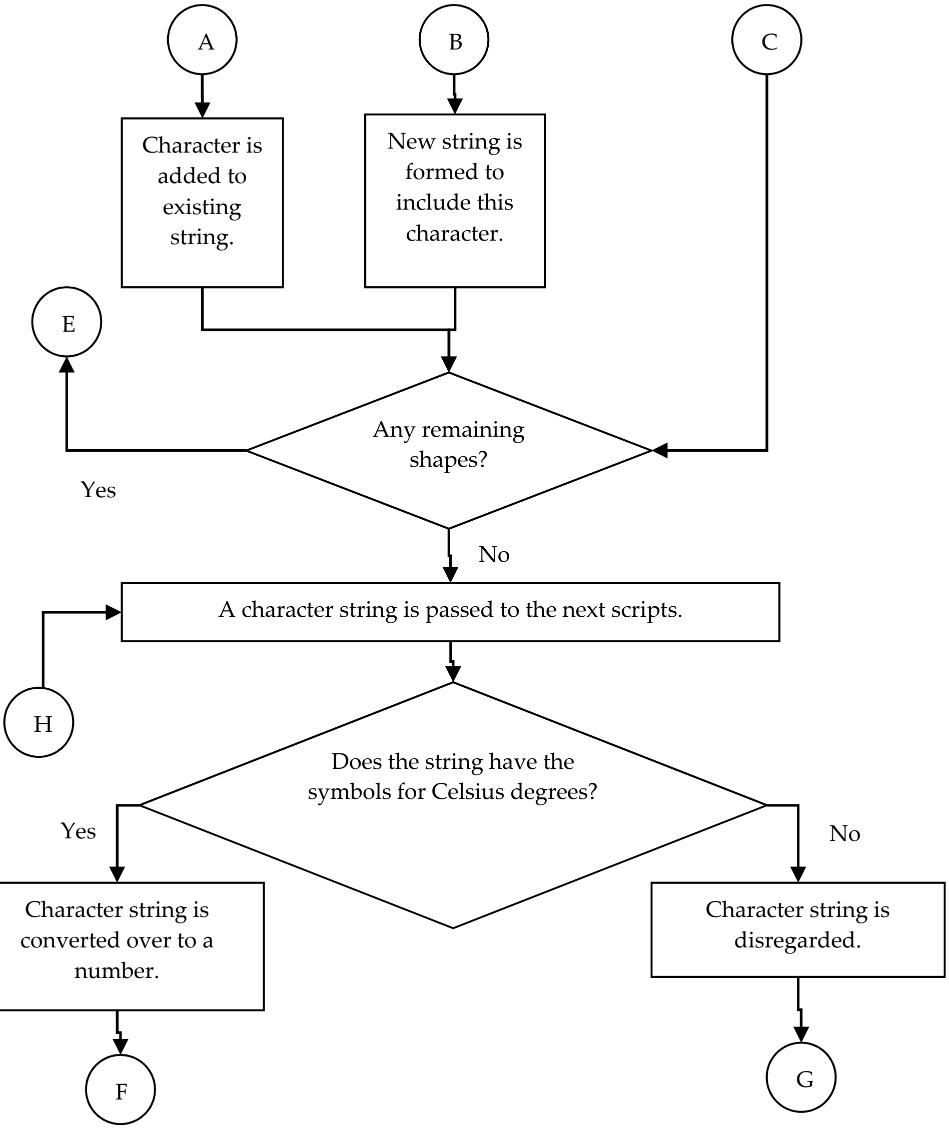
A
B
C
Character is added to existing string.
New string is formed to include this character.
E
Any remaining shapes?
Yes
No
A character string is passed to the next scripts.
H
Does the string have the symbols for Celsius degrees?
Yes
No
Character string is converted over to a number.
Character string is disregarded.
F
G

Figure 3. Workflow of program (Part 2)

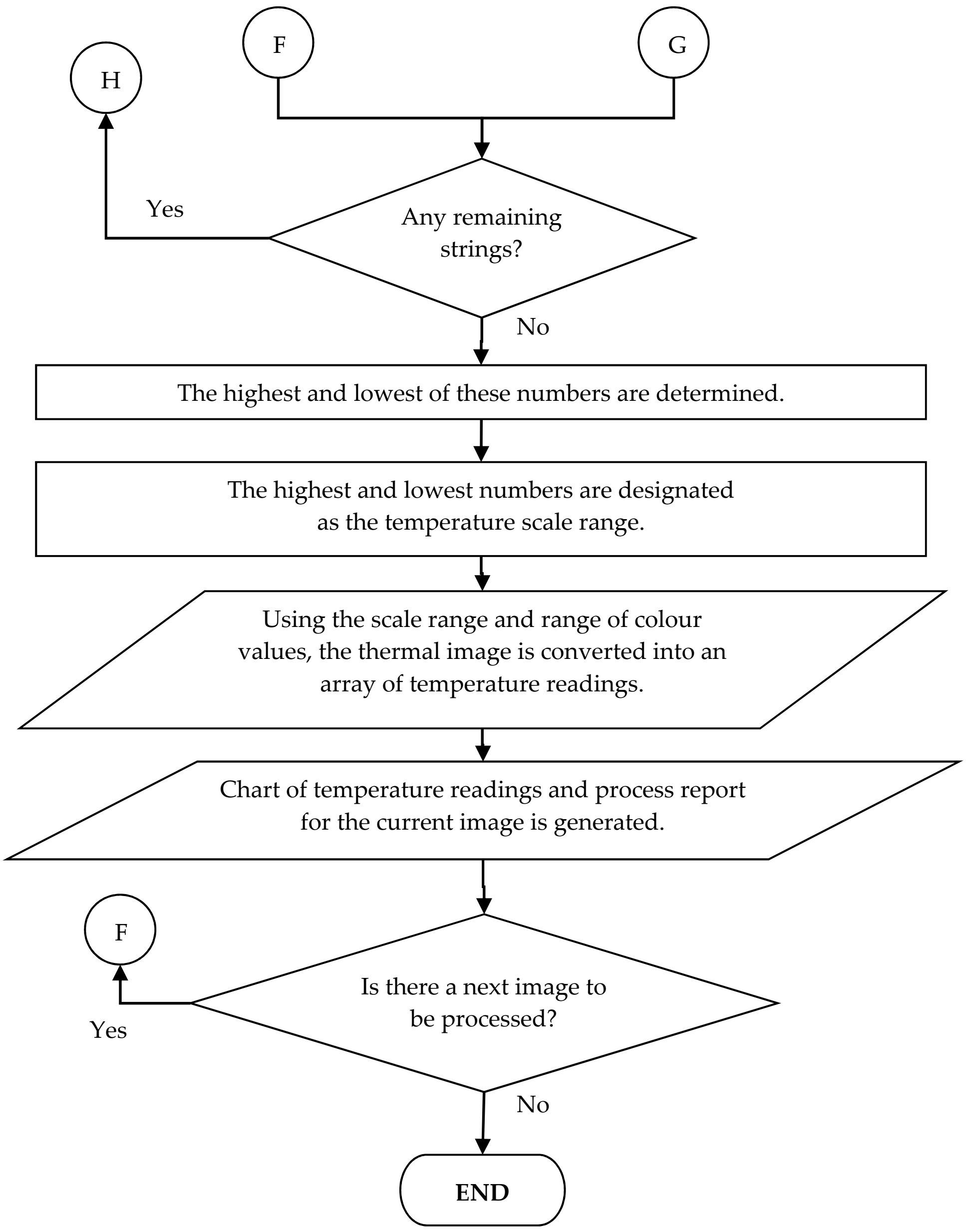
H
F
G
Yes
Any remaining strings?
No
The highest and lowest of these numbers are determined.
The highest and lowest numbers are designated as the temperature scale range.
Using the scale range and range of colour values, the thermal image is converted into an array of temperature readings.
Chart of temperature readings and process report for the current image is generated.
F
Is there a next image to be processed?
Yes
No
END

Figure 4. Workflow of program (Part 3)

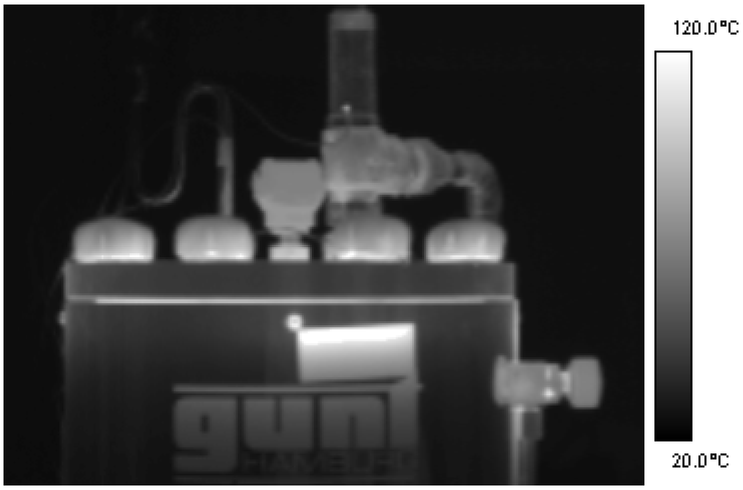

Figure 5. Marcet Boiler with logo that emits infrared radiation.

The OCR scripts use the algorithm for correlation coefficient to determine the character that the captured shapes are most similar to. The program compares a captured shape to each entry of an archive of pre-existing images of characters, generating a correlation coefficient from each comparison. If one of the correlation coefficients surpasses a threshold value (which is implemented to prevent illegible shapes from being identified as text characters) and is the highest, the captured shape is identified as this character.

$$r = \frac{\sum_m \sum_n \left(A_{mn} - \overline{A}\right)\left(B_{mn} - \overline{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \overline{A}\right)^2\right)\left(\sum_m \sum_n \left(B_{mn} - \overline{B}\right)^2\right)}} \tag{1}$$

The correlation coefficient algorithm cannot be used on simply-shaped characters, such as the period symbol, as this may lead to division-by-zero errors. For these characters, special scripts can be devised to identify captured shapes as such by examining their dimensions.

After the filtering of captured shapes with the OCR scripts, the remaining ones should be those for characters that make up the temperature labels. However, before they can be examined further, they need to be grouped into strings of characters, i.e. words and numbers, as noted in the workflow diagrams above.

The following steps are used to group characters together into strings:

1. A character is picked from the structure or array variable.
2. It is checked for adjacency to any other character. The character and the other one are grouped together into a string.
3. The string is entered into a list of strings, with tags to associate the string with these characters.

4. The current character is marked as having been examined, and the next one is fetched.
5. The next character is checked for any existing tags; these tags will be passed to any character found to be adjacent to this one.
6. Afterwards, steps 2 to 5 are repeated until all characters are accounted for.

To decide whether the strings formed are temperature values or not, they can be first examined to see if they contain digits; if they do not, they are most certainly not temperature labels. To confirm whether they are temperature values or not, they are either examined for any adjacency to symbols for Celsius degrees; these are definite indications that they are temperature labels.

If they are not adjacent to the symbols, the thermal image is checked for the presence of these anyway; a thermal image should have text that denotes the unit of measurement used. If the temperature-to-colour index is presented in a manner similar to how axes are presented on charts, then the strings are examined to determine if they are wholly composed of digits and/or period symbols, if any.

Once all characters have been accounted for, there should be a list of character strings that are all temperature labels. The limits for the temperature scale can be obtained from these by searching for the highest and lowest values. These are then associated with the highest and lowest colour values, respectively.

The program should then examine the thermal image for any indication that the temperature scale is logarithmic or linear. Most thermal images use linear temperature scales, so simple interpolation can be used to convert the colour value of every pixel over to a temperature reading. Otherwise, the program may have to examine the distances of every temperature label to the others; if the distance of separation between them changes exponentially, this is an indication that the scale is logarithmic, and thus the conversion has to be designed accordingly.

The temperature readings can be grouped into a chart of the same dimensions as the image subject. It is preferable to use universal file formats for these charts, such as Comma-Separated Value, to avoid compatibility issues or the technical limitations of more specific file formats. Any other output, such as a report on the peak and lowest temperature readings can be derived from the chart.

To measure the expected benefit of using OCR to automate the analysis of the temperature scale over human reading, another version of the program has been created. It is similar to the one that uses OCR scripts, with the exception that the OCR scripts have been replaced with a user interface that requests for manual input of the temperature scale limits; to aid this, the thermal image being processed is displayed for the user to examine. The processing times for the two versions of the program are compared over a range of numbers of thermal images to be processed.

The processing time is defined as the time from the launch of a program to the generation of the results. Therefore, the time taken for the user to examine the thermal image is considered too for the manual-input version. To reduce the uncertainties in the measurements for the manual-input version, the user practices to achieve the shortest possible times.

Another version of the program that checks for metadata instead of using OCR has also been created and subjected to the tests above as a control test. To this end, the header bytes of the thermal image files are embedded with metadata, which this version of the program checks for.

2.5. Status

The program is complete and working in two formats, MatLAB m-file and Visual C#. Currently, the program is made for thermal images in grayscale, though it can also be used for broader colour palettes by simply converting them over to grayscale first. This is justified if the colour scale organizes colour values in the same arrangement as the temperature scale. The program also works for thermal images of any size.

The methods used in this program do not work for thermal images with text that is so small as to be illegible for human reading. As OCR is the machine equivalent of human reading, a program that uses OCR is incapable of reading what a human cannot possibly read. Even if they are legible, small, low-resolution text may give rise to problems such as mistaking one character for another, as a human would if he/she did not further scrutinize said character. **Figure 6** shows an example of such an occurrence.

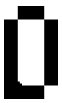

Figure 6. Mistaking a low-resolution zero digit as the letter 'D'.

Currently, the problem overcomes such a problem by examining any character that is next to the dubious character. If the other character is a digit, then the logic that since the characters are expected to compose the numbers that make up temperature labels, the dubious character is likely a digit as well.

As the average resolution of thermal images increases as the resolution of thermal imagers improves due to technological advances, more legible text can be used for the labelling of temperature scales. It is expected that this thermal image processing program and others would not have to encounter such problems in the future.

Any flaws incurred during the generation of the thermal image in the first place may be carried over to the processing of the thermal image. For example, failure to present the digits in the temperature labels as distinctly separate characters may cause problems for certain pattern-recognition techniques like connected-component labelling. **Figure 7** shows such a case for the digits "40", as capture by a FLIR A20M, which would be identified by connected-component labelling techniques as a single object.

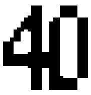

Figure 7. The digits for number "40" conjoined.

The solution for this example of a flaw is to examine the dimensions of the conjoined characters. If the composite object is too big to be a single character, it is likely a pair or more of conjoined characters, and can be split according to its dimensions into its constituents, which can then be analyzed separately.

2.6. Results

The measurements of processing times for the OCR-implemented and manual-input versions of the program are as shown in **Table 1**. A graphical presentation of the measurements is shown in **Figure 7**. The measurements were obtained with the MatLab version of the program, ran with MatLAB R2008b IDE, Windows XP Service Pack 2 and Intel Core2Duo E4700 CPU.

The implementation of the OCR scripts is found to have decreased the time taken for the program to process the same thermal images from that it would have if manual-input was used instead. Therefore, the automation of the examination and processing of thermal images without metadata is feasible with the utilization of OCR.

However, for thermal images with metadata, the version of the program with OCR scripts is not as competitive as the program that checks the metadata immediately. The differences should be apparent in **Table 2** and **Figure 8**.

No.of Images	OCR-Implemented Program				Manual User Input Program			
	Time Measured for Run no. (s)				Time Measured for Run no. (s)			
	1	2	3	Average	1	2	3	Average
1	0.2779	0.2764	0.2743	**0.2762**	3.9782	2.6486	3.0375	**3.2214**
2	0.5558	0.557	0.5574	**0.5567**	8.5646	9.2787	9.1721	**9.0051**
3	0.8411	0.8325	0.8428	**0.8388**	11.7308	11.2316	11.7690	**11.5771**
4	1.1037	1.1135	1.1138	**1.1103**	12.7000	12.8020	11.5818	**12.3613**
5	1.3929	1.371	1.3781	**1.3807**	14.3850	14.8614	14.5479	**14.5981**
10	2.6240	2.6541	2.6741	**2.6507**	29.0014	29.0213	30.0042	**29.3423**
15	4.1154	4.0098	4.016	**4.0471**	45.9954	46.0872	46.1534	**46.0787**
20	5.5412	5.4001	5.4003	**5.4472**	62.9987	63.5599	63.5401.	**63.2793**
25	7.0141	7.0098	6.9421	**6.9887**	77.1524	77.0016	78.0003	**77.3848**
30	8.4386	8.445	8.4825	**8.4554**	97.0983	97.3609	95.5554	**96.6715**

Table 1. Processing Times of OCR-Implemented versus Manual-Input

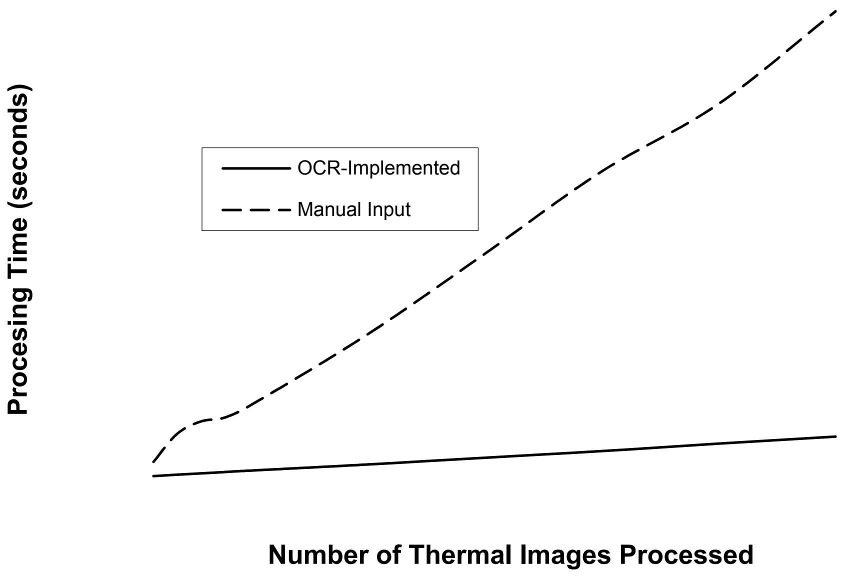

Figure 8. Processing Times of OCR-Implemented versus Manual-Input

No. of Images	OCR-Implemented Program				Metadata-Checking Program			
	Time Measured for Run no. (s)				Time Measured for Run no. (s)			
	1	2	3	Average	1	2	3	Average
1	0.2779	0.2764	0.2743	**0.2762**	0.0554	0.0560	0.0562	**0.0559**
2	0.5558	0.557	0.5574	**0.5567**	0.1543	0.1464	0.1472	**0.1493**
3	0.8411	0.8325	0.8428	**0.8388**	0.2541	0.2498	0.2513	**0.2517**
4	1.1037	1.1135	1.1138	**1.1103**	0.3498	0.3501	0.3478	**0.3492**
5	1.3929	1.371	1.3781	**1.3807**	0.4453	0.4532	0.4501	**0.4495**
10	2.6240	2.6541	2.6741	**2.6507**	0.5399	0.5468	0.5513	**0.5460**
15	4.1154	4.0098	4.016	**4.0471**	0.6534	0.6501	0.6521	**0.6519**
20	5.5412	5.4001	5.4003	**5.4472**	0.7521	0.7498	0.7501	**0.7507**
25	7.0141	7.0098	6.9421	**6.9887**	0.8600	0.8512	0.8532	**0.8548**
30	8.4386	8.445	8.4825	**8.4554**	0.9564	0.9513	0.9498	**0.9525**

Table 2. Processing Times of OCR-Implemented versus Metadata-Checking

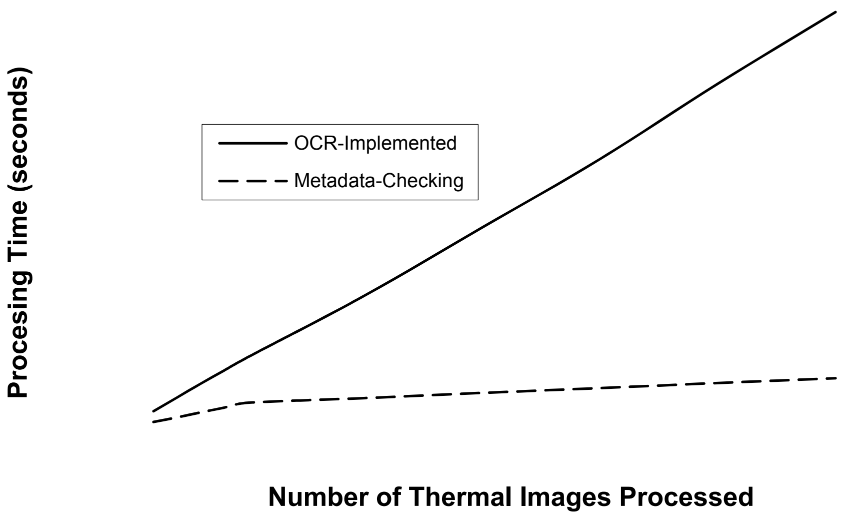

Figure 9. Processing Times of OCR-Implemented versus Metadata-Checking

Much of the time taken by the version that uses OCR is spent on running the OCR scripts. The more labels there are on the temperature scale, the longer it takes. Some reduction could be achieved by hard-coding into the program some short-cuts, such as where it looks in the thermal image for the temperature scale limits, but this reduces the versatility of the program. However, such a finding also shows where there is room for improvement.

2.7. Future research

Currently, the program is most suitable for the processing of thermal images without metadata. It can process thermal images with metadata as well, as long as it has a temperature scale. However, as shown earlier, it takes a longer time to produce results than checking for metadata does.

While hard-coding is not a desired solution to increase their competitiveness, the OCR scripts can be improved with more efficient techniques. Of particular interest are the ways used in the program to recognize and capture shapes from the image, and the ways to filter and identify shapes as characters. It is the hope of this future research that the OCR-utilizing method can become as competitive as the methods that relies on metadata.

3. Conclusion

Optical character recognition is feasible for use as a method in the processing of thermal images for information. It forgoes the need for metadata and can be used on any thermal image as long as it has a legible temperature scale, but the OCR scripts used may consume time that can be otherwise saved if metadata had been available instead.

Therefore, it is currently practical only for thermal images that happen to have no metadata or has damaged metadata. However, it has been shown to be useful in automating the processing of large numbers of thermal images without metadata, which would otherwise be a daunting task if the user has to manually input the processing parameters used.

Considering that there are more methods of optical character recognition than the one shown in this book chapter that utilizes correlation coefficients, processing thermal images with OCR can be developed to be as close to competitive as the use of metadata.

Author details

W. T. Chan, T. Y. Lo and K. S. Sim
Faculty of Engineering and Technology, Multimedia University, Melaka, Malaysia

C. P. Tso
School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore

4. Acknowledgement

The authors would like to thank W.K. Wong of Multimedia University for allowing the use of the FLIR A20M thermal imager under his purview, and D.O.B. Guerrero for having published his MatLab work on OCR.

5. References

[1] V.K Govindan , A.P Shivaprasad (1990), Abstract: Character recognition — A review, Pattern Recognition, Volume 23, Issue 7, 671-683.

[2] Hiromichi Fujisawa (2008), Forty years of research in character and document recognition—an industrial perspective, Pattern Recognition, Volume 41, Issue 8, 2435-2446.

[3] Amir Sedighi, Mansur Vafadust (2011), A new and robust method for character segmentation and recognition in license plate images, Expert Systems with Applications, Volume 38, Issue 11, 13497–13504.

[4] Diego Orlando Barragan Guerrero (2007), Optical Character Recognition (OCR), MatLab Central, Files Exchange. Available: http://www.mathworks.com/matlabcentral/fileexchange/18169-optical-character-recognition-ocr. Accessed: November 2010.