Liquid Level Detection via Node-red and Automatic Image Processing System

Tai Huu LE[1], *Phu Thai NGO*[2], *Van Ngu TRAN*[3], *Viet Hung TRAN*[4,*]

[1]Department for Facility Management, Ton Duc Thang University, Ho Chi Minh City, Vietnam.
[2]Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam.
[3]Faculty of Applied Sciences, Ton Duc Thang University, Ho Chi Minh City, Vietnam.
[4]Modeling Evolutionary Algorithms Simulation and Artificial Intelligence, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Vietnam.

*Corresponding Author: Viet Hung TRAN (email: tranviethung@tdtu.edu.vn)
(Received: 10-Oct-2021; accepted: 6-Dec-2021; published: 31-Mar-2022)
DOI: http://dx.doi.org/10.55579/jaec.202261.351

Abstract. *Internet of things (IoT) is increasingly useful in connecting network of devices on the internet. A popular need of IoT in the industry is to measure the level of the liquid in a faraway tank of interest. The traditional measurement method uses float, pressure or ultrasonic sensor and provides reliable results. Nonetheless, the disadvantages of traditional methods are that their range of measurement is rather short and their maintenance on the field is also difficult. To solve these problems, we propose a novel method that combines the image processing algorithm with Node-Red IoT systems. The original idea of this paper is to use the image of the ruler on the tank for measuring the physical height of liquid level in standard units. Our experimental results on real-time collected images via IoT live-streaming show that our proposed method is very fast and robust with different size and shape of different rulers. Our method is also accurate enough to facilitate the application of liquid level measurement in practice.*

Keywords

Node-Red, image processing, ruler, edge detection, IoT.

1. Introduction

Liquid level measurement is essential to the process control in industry. In practice, we may use the physical sensors (e.g. float, pressure or ultrasonic sensors) for liquid level measurement. Yet it will be difficult to adjust the physical sensor's position and handle the error when a manufacturing problem occurs. In this case, a low-cost camera combined with computer vision would solve these disadvantages of physical sensors and give us a wider range of uses when it comes to different types of liquids or objects when determining their height.

1.1. Literature review

There are many papers that apply algorithms in image processing to measure the distance of different objects, as briefly summarized in Tab. 1.

In [1], experiments are conducted in order to establish the relationship between pixels to region-of-interest (ROI) of the ruler's image area. It used edge-detection algorithm to determine the river's water level with an error of

Algorithms	Advantages	Disadvantages
Select the image's region of interest (ROI) and detect edges in the image [1]	Accurately measure 99.6% of river water level in good conditions.	The scale is detected only on a specific ruler of water river.
Fourier transform [2]	Measure the exact size of the fish	Inaccurate measurement results if the fish size exceeds the ruler distance
Pattern matching algorithm for character recognition [3–5]	Recognize characters on the ruler to determine the water level in the river. The error is within the allowable limit.	It limits the number of recognizable characters and is affected by noise.
Dictionary Learning [6]	Train the images received from the camera to measure the water level in the river	It is hard to process untrained images
Regression [7, 8]	Collecting images from social networks as data to accurately determine the flood level in the area	The method requires manual image labeling with large data sets. It will cause errors for unlabelled or untrained data.

Tab. 1: Comparison of image processing methods for distance measurement.

0.4% based on the distance of the lines of the ruler.

The paper [2] applied the Fourier transform to estimation of the proportional distances on the ruler to measure the fish size within the limit of 30 cm. The limitation of this method is that it is rather difficult to process images with noise, blur, and oversize fish.

The papers [3–5] used pattern algorithms to recognize characters on rulers to measure river water levels. The paper [3] used a multi-pattern sequential matching algorithm to check and compare the height of the character on the ruler. This method also accurately measures the water level but only recognizes 63% of the characters on the ruler and the error is 0.9 cm. Nonetheless, the measuring environment is not always favorable. When the ruler is placed in a noisy environment, there are many impurities and poor lighting conditions. The paper [4] used a method based on image processing and sparse representation to recognize the character of the ruler in the condition that many factors affect the measurement results. The method's maximum error is less than 0.9 cm within a tolerance of 1 cm. The paper [5], again, applied water level measurement to find out the correlation of the up and down velocity of the water level in the river. The author set the threshold value of the image via the Otsu method, then removed the small object and used Morphology to process the binary image and recognize the characters on the ruler. The recorded water level results correspond well with the measurement data.

The paper [6] used a Dictionary Learning model to train the camera images (15 minutes per image in average) and measure the water level in the river. The received images have different types of rulers, so the model must be trained for each specific case. Experiments give good results with an average error of 3 cm and a maximum error of 5 cm.

The papers [7, 8] used a regression algorithm [9] to estimate water level by collecting images for training from social networks to determine the flood level in the area. The method requires manual labeling of large data sets. If the label is wrong or the training image is different from the data, it will cause errors. For flood warning, the paper [7] combined regression method with image-pairs ranking to calculate depth of water level. The water level is calculated by finding the objects shown in the image and determining the height of those objects. The minimum average

error calculated is 11.8 cm with the processing data of 1 million images. Nonetheless, when the processing data is larger, the error is not reduced and the system is slower. Similar to [7], the prediction in [8] is made via deep learning framework of scanned images. The average error of this method is about 7 cm with the processed data of 7000 images taken from the internet.

1.2. Paper's contribution

In summary, many methods in literature have been proposed in the literature to measure distances, but these methods are only correct when the distance of the divisions on the ruler is known. In this paper, we will design a simple algorithm capable of measuring liquid level by image processing and independent of the scale of the ruler. Instead of slow manual calibration, our ruler detection algorithm will help camera system automatically calibrate with physical measurement in real time.

Also, a rather new approach to achieve better manufacturing and enhance controlling quality is to combine Internet-of-things (IoT) and aided-monitoring computer vision. It is cost-effective and achieves simplification in vision installation and handling system. Our image processing algorithm for Node-Red IoT program in this paper can continuously analyze and process the images obtained from the system operation in practice.

In comparison with other methods in Tab. 1, our ruler detection has similar accuracy (with virtually perfect detection) for simple scenarios like clear image of single ruler. Despite the simplicity, this scenario is a popular case in industrial systems like water level measurements, etc.

The virtue of our method is more about the processing speed. Our method has much lower computational cost, with linear complexity $O(n)$, in which n is the number of pixel of ROI area. Indeed, our algorithm is very fast since it only involves basic operators (namely addition, multiplication and comparison) directly on pixels, while other methods in Tab. 1 require more complex transformation in general. To our knowledge, our algorithm is the first ruler detection algorithm fast enough for low-cost IoT systems.

1.3. Paper's structure

The structure of this paper is as: The image of ruler, liquid level and the image processing algorithm via Node-Red will be presented in the section 2. . The section 3. is the evaluation of our experiment result and the section 4. is the conclusion about the merits of the paper.

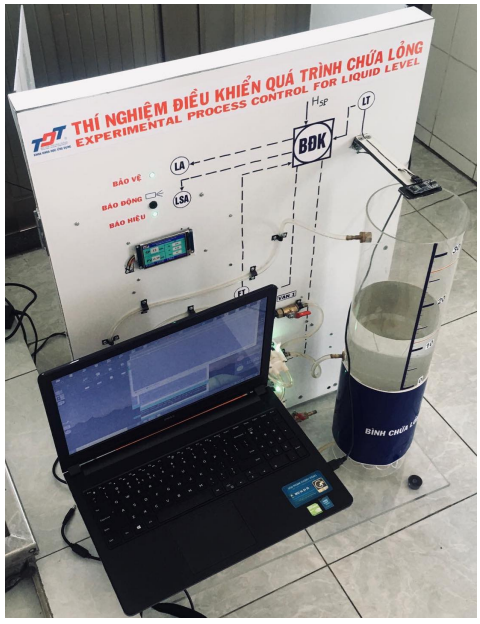

Fig. 1: Experimental process control for liquid level.

2. Methodology

In this section, we will design a Node-Red program for supervisory control and data acquisition the system. This is also a tool to test the liquid level measurement algorithm proposed in this paper.

2.1. Experimental tank of liquid level

Figure 1 shows our experimental model of liquid level control at the laboratory of the Faculty

Fig. 2: Liquid level measurement algorithm in Node-red.

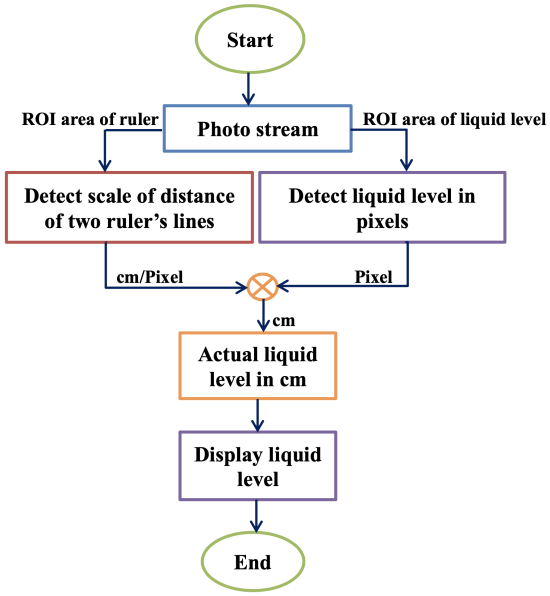

Fig. 3: Flowchart of Node-Red algorithm in Fig. 2.

of Applied Sciences, Ton Duc Thang University, Vietnam. In general scenario, the scale of liquid tank may be different from our custom ruler in Fig. 1. The physical value of the distance between two ruler's lines may depend on the shape, width and height of the tank. This will make it difficult to determine the actual volume of liquid in the tank. To solve this problem, we will use Node-Red program to take the image of the tank and calculate the distance between two ruler's lines. This distance serves as a reference frame for measuring the liquid level in Fig. 1.

1) Node-Red IoT program

Figure 2 is a diagram describing the image processing algorithm to measure the liquid level in the tank in Fig. 1. Node-Red is a web-browser programming tool on flows for connecting hardware devices, APIs (Application Programming Interface) and online services together as part of IoT [10, 11]. In Node-Red, there are many built-in libraries that are drag-and-drop and allow to build a system or build a function using JavaScript programming feasibly. Node-Red is built upon the Node.JS platform and stores JSON files with a small size that can be shared across the network feasibly. For this reason, Node-Red is one of the most popular IoT systems in both industry and literature.

2) Block "Photo stream"

The input data to be used in our simulation is a streaming video showing the liquid being poured into the tank in Fig. 1. We are going to measure this liquid level by combining function blocks in Node-Red. All parameters will be saved as Messages (msg) and Flow in Node-Red in order to facilitate programming. The "Photo stream" block in Fig. 2 can capture the content of this streaming video and split it into a sequence of frames of size m*n pixels sequentially in time in Fig. 4. These transmitted frames will be fed to the processors of the "ROI ruler" block and the "ROI liquid level" block to calculate the liquid level in the image.

2.2. Block "ROI ruler"

In this section, we will present our ruler's detection algorithm being used in "ROI ruler" block of Fig. 2, as shown in Fig. 5

1) Block "Ruler's image pre-processing"

Similar to the image pre-processing stage of the ruler in [1, 5], the block "Ruler's image pre-processing" of Fig. 5 with color images as input will be processed for high contrast and clarity. The transformation is performed according to

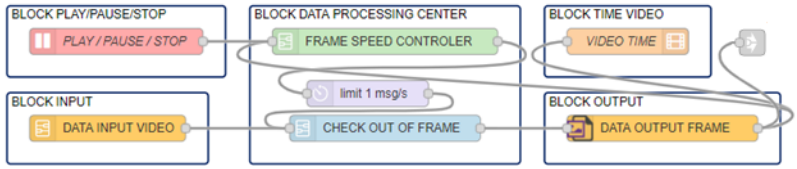

Fig. 4: Block extracting image from video in block "Photo stream" in Fig. 2.

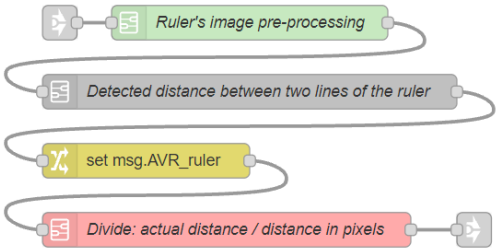

Fig. 5: Ruler's detection algorithm in block "ROI ruler" in Fig. 2.

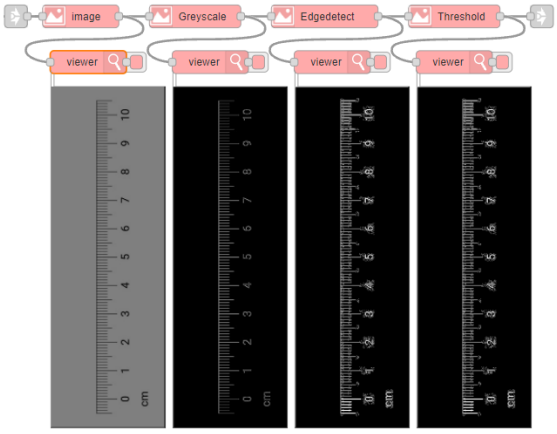

Fig. 6: Ruler's image pre-processing in block "Ruler image processing" in Fig. 5.

Eq. (1), as follows:

$$g(x, y) = f(x, y) + \beta, \tag{1}$$

in which:

β: self selected constant.

$f(x, y)$: brightness level of pixels in the original image.

$g(x, y)$: brightness level of pixels in the image after transformation.

After this process, the color image will be converted to gray image for easy processing, as shown in Fig. 6. In Node-Red dashboard, we will create a toolbar with the adjustment range $\beta \in [-1, 1]$ in order for user to adjust the visual contrast of gray image to make the image clearer. Note that, this brightness adjustment is not a part of our detection algorithm, since it is similar to our threshold step below. This brightness step is simply a traditional step in industrial camera system, which merely helps user calibrate the light for camera system on the spot.

The image is then put into the gradient edge detection in Fig. 6 via the standard 3×3 convoluted matrix: $[[0, 1, 0], [1, -4, 1], [0, 1, 0]]$. For threshold block in Fig. 6, the goal is to distinguish image regions on an image, as follows:

$$h(x, y) = \begin{cases} 255 & \text{if } g(x, y) > T \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

It is relatively feasible to determine the threshold value T in 8-bit greyscale, i.e. $T \in [0, 255]$, in Eq. 2 because there is a clear distinction between the pixel values in the ruler.

The Ruler's image pre-processing step is to adjust the image for better display and readout of the image parameters, which is combined with the brightness $\beta \in [-1, 1]$ and threshold $T \in [0, 255]$ parameters. In our experiment, the setting $\beta = 0$ and $T = 5$ is good enough for most cases, as shown in Fig. 6.

2) Block "Detected distance between two ruler's lines"

To simplify the calculation of the distance between two ruler's lines in Fig. 7, we chose ROI on the ruler's image with the similar characteristics for processing in [12], as shown in Fig. 8. Hence, the aim of Ruler's image pre-processing step above is to turn the color image into a binary image and create a mask for ROI in Fig. 7.

Fig. 7: Algorithm for calculating the scale in block "Detected distance between two lines of the ruler" in Fig. 5.

This is very useful for image information recognition and extraction algorithms below.

3) Block "ROI area"

Before calculating the distance of the tank's ruler in Fig. 1, let us demonstrate our algorithm in Fig. 7 via an image of a normal ruler used in daily life, as shown in Fig. 8. This allows us to have an overview of the proposed algorithm in this paper.

The length of the standard ruler is 10 cm, corresponding to 21 lines longer than the remaining lines, with the distance between two lines is 0.5 cm in Fig. 8. There are 11 longest lines with the distance between the two lines being 1 cm in Fig. 8. The ROI marked on the ruler's image is the area of blue rectangle. Note that, by definition of ROI, we only consider ROI area in our detection algorithm. This ROI area, which is possibly as large as the whole image (c.f. Fig. 11), can be freely chosen by users on the spot. Nonetheless, the ROI area should not be too large. The larger the ROI area, the higher computational cost.

4) Block "Histogram of pixel's distance"

Next, we scan the white pixels (i.e. the pixel's maximum value of 255 in 8-bit greyscale) in ROI area in Fig. 8 and take the Y-axis values of

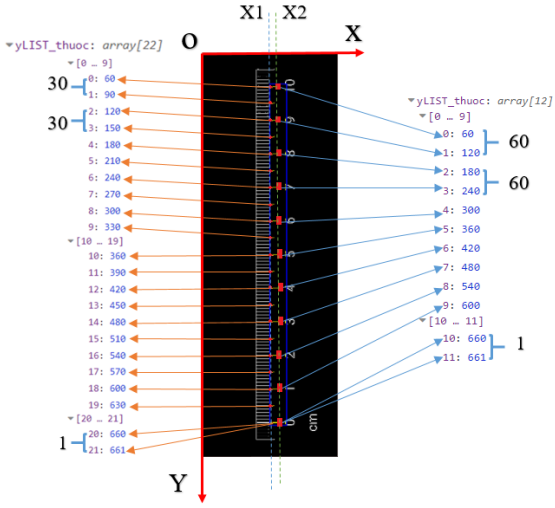

Fig. 8: ROI area of the ruler, as chosen in block "ROI area" in Fig. 7.

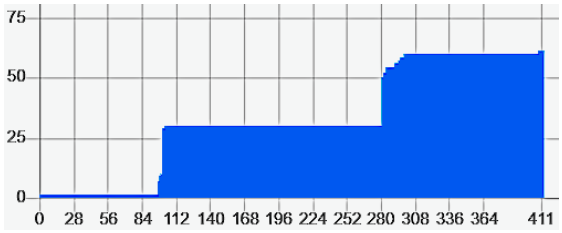

Fig. 9: Sorted distances ΔY in pixels of the ruler's lines, as calculated in Eq. (9) in block "Histogram of pixel's distance" in Fig. 7.

Fig. 10: Histogram of the distances ΔY in pixels of the ruler's lines, as calculated from Fig. 9.

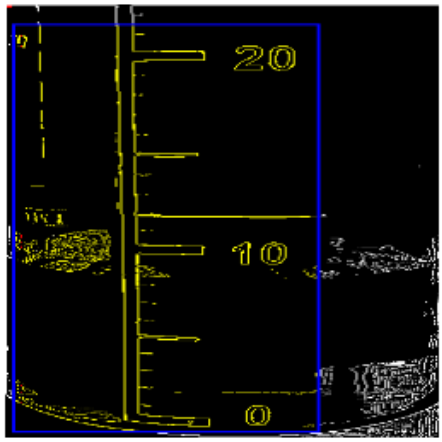

Fig. 11: ROI of the ruler on the tank.

these white pixels and put them into Ylist arrays. Here, let us denote "yLIST_thuoc" array as Ylist_i for i-th columnX_i, as illustrated in Fig. 8.

In practice, the pixels of ruler's lines in ROI may be noisy and, as consequence, there may be noise elements in Ylist arrays. To solve this problem, we will calculate the distance between white pixels in Ylist array and create ΔY arrays in the block "Histogram of pixel's distance" in Fig. 6, as follows:

$$\Delta Y_i[j] \triangleq \text{Ylist}_i[j] - \text{Ylist}_i[j-1], \ \forall i, j, \quad (3)$$

As shown in Fig. 9, the distance in pixels between adjacent elements in Ylist arrays has the value in range $[1, 61]$, in which the value 30 is the most occurring. From the histogram in Fig. 10, we see that the value 30 appears again with the highest frequency of 178 times and then the value 60 appears with the second highest frequency of 111 times. We believe that the value 30 is the distance between two lines of the ruler in pixels, corresponding to a physical unit of 0.5 cm in Fig. 8.

5) Block "Estimated distance of two ruler's lines"

From the above remarks, our block "estimated distance of two ruler's lines" in Fig. 7 simply returns the value with highest frequency in histogram, as illustrated in Fig. 10.

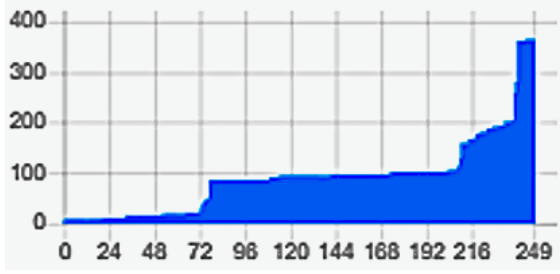

Fig. 12: Sorted distances ΔY in pixels of the ruler's lines on the tank.

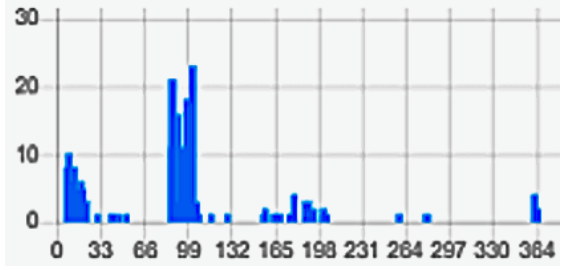

Fig. 13: Histogram of the distances ΔY in pixels of the ruler's lines on the tank, as calculated from Fig. 12

6) Case study - Tank's ruler

From the tank's image in Fig. 1, let us apply the ROI ruler's algorithm described above to calculate the distance between two ruler's lines on the tank.

The length of the ruler is 30 liters, corresponding to 7 lines in ROI with the distance between two lines is 5 liters in physical units, as illustrated in Fig. 11. From this ROI (blue area in Fig. 11), let us find the histogram and the distance between two adjacent lines of the ruler in the tank.

As shown in Fig. 12, the distance in pixels between adjacent elements in Ylist arrays has the value in range $[8, 364]$. In which the value 102 is the most occurring.

7) Ruler's detection algorithm via histogram

Table 2 is the comparison of histograms in Fig. 10 and Fig. 13. These two histograms have independent peak distributions with the highest

Histogram	The ruler in Fig.8	The ruler in Fig.11
Number of ruler's lines	21	7
Range of distance	[1,61]	[8,364]
The value that appears the most	30	102
The highest frequency	178	23
Second highest frequency	111	21

Tab. 2: Comparison of the histogram of two types of ruler.

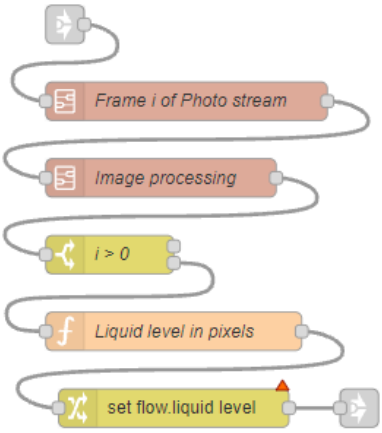

Fig. 14: Image processing algorithm for liquid level in block "ROI liquid level" in Fig.2.

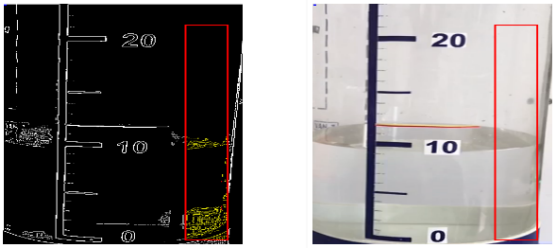

Fig. 15: ROI of the liquid level.

value dominating the adjacent values in the histograms.

Note that, our detection algorithm has exploited a very common feature of all rulers, i.e. there are many identical distances between pixels of ruler's lines. Thus, our method in this paper can feasibly determine the distance between two ruler's lines according to the histogram of pixel's distances for many different types of rulers.

2.3. Block "ROI liquid level"

Visual control of liquid level plays an important role in automated monitoring systems [13, 14]. The "Photo stream" block in Fig. 2 can capture the streaming video and split it into a sequence of image frames sequentially in time. For each image frame of the liquid level, the estimation of the liquid level in pixels is then solved via the algorithm in Fig. 14.

In block "Image processing" in Fig. 14, the system determines the liquid level by the number of pixels in the tank. The image pre-processing step and ROI selection of the liquid level from the image is similar to those of the ruler image above.

The ROI of the liquid area before and after processing step is then shown in Fig. 15. Via image threshold processing, the liquid surface can be detected feasibly via the standard gradient edge detection in Fig. 15. To calculate the liquid level in pixels, we find the YLIST array in the ROI region (i.e. the red area in Fig. 15) of the liquid in the function node "Liquid level in pixels" in Fig. 14. It is similar to how the Ylist array of the ruler is calculated and this value is calculated by the formula:

$$c = \mathrm{YLIST}_n[end] - \mathrm{YLIST}_n[0] \qquad (4)$$

in which $[end]$ denotes the last element of Ylist_n and n is an arbitrary n-th column X_n in the ROI area of the liquid level.

Via Eq. 4, the system will calculate the distance from the bottom of the tank to the surface level of the liquid in the tank. This process will be run continuously until no more image frames arrive.

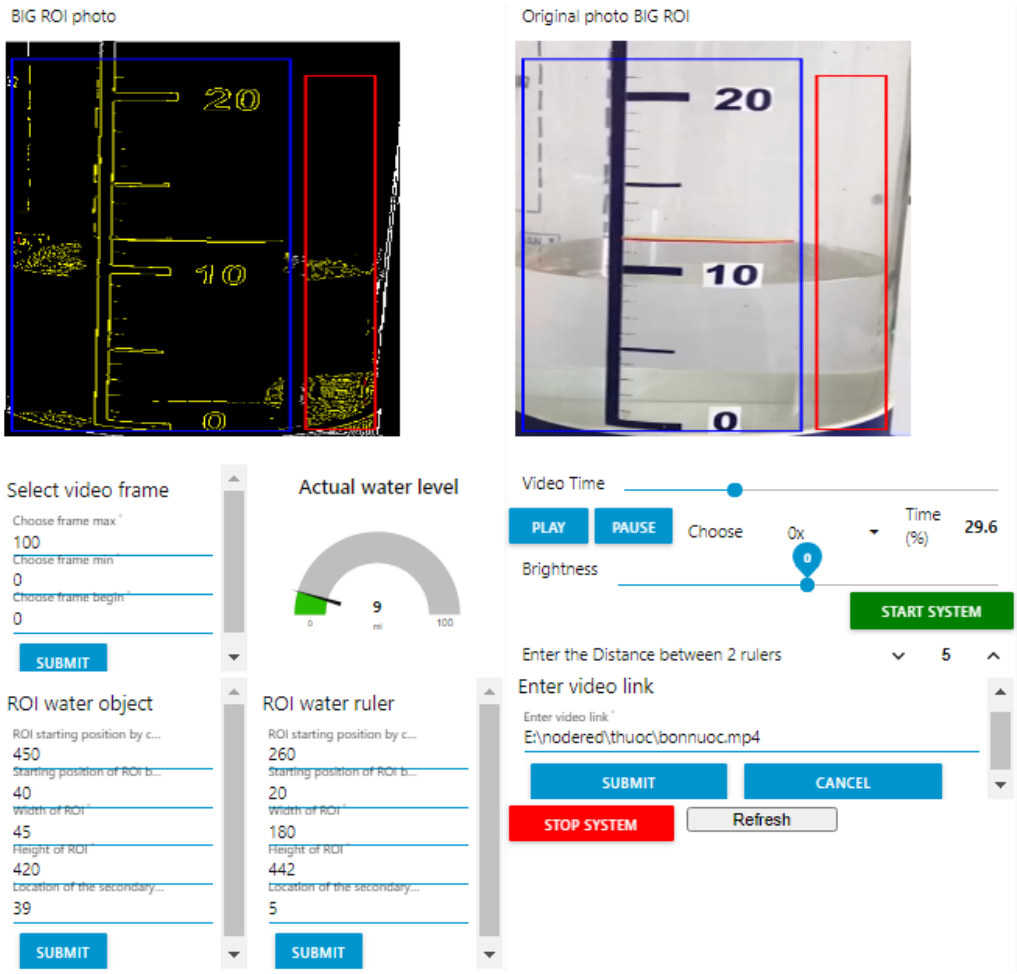

Fig. 16: Monitor interface for measuring liquid level on the tank. The physical unit of distance between two ruler's lines is $a = 5$ liters.

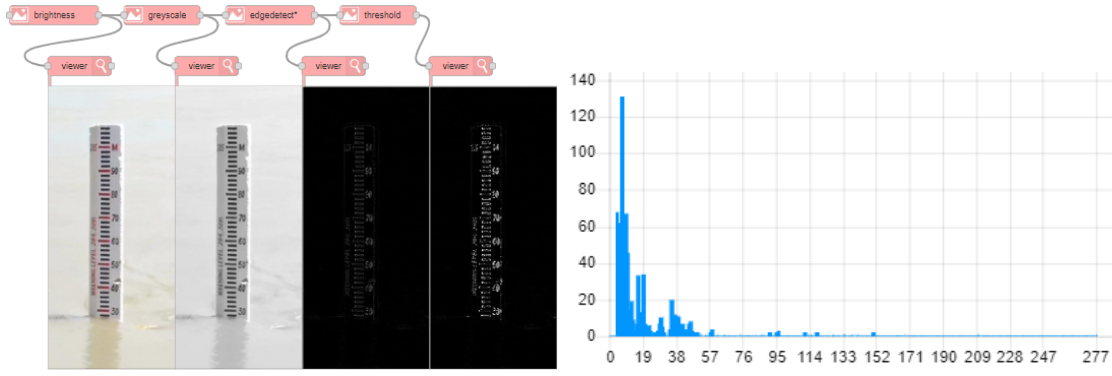

Fig. 17: Histogram result of our algorithm in Fig. 8-10 for a water level ruler at Yamuna river in India [15]. The distance of two ruler's lines corresponds to the highest peak at 7 pixels of histogram. This result has demonstrated the universality of our ruler detection algorithm in practice.

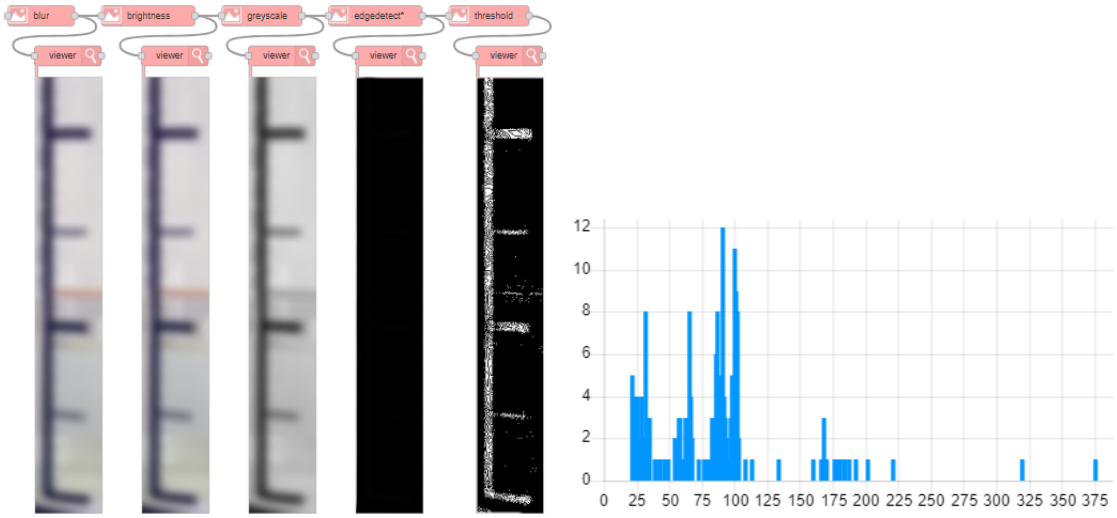

Fig. 18: Histogram result of our algorithm in Fig. 8-10 for a blurry image of the tank ruler in Fig. 1. With blur's pixel radius = 5, the distance of two ruler's lines still corresponds to the highest peak at 91 pixels of histogram. This result has demonstrated the robustness of our ruler detection algorithm.

2.4. Calculation of the liquid level in physical unit

The function node "Multiplication" in Fig. 2 is programmed to calculate the actual liquid level on the tank according to the formula:

$$d = \frac{a \times c}{b} \tag{5}$$

in which:

a: distance in physical unit between two ruler's lines

b: distance in pixels between two ruler's lines

c: liquid level in pixels on the tank

d: liquid level in physical unit on the tank

Since b and c can be estimated via the above block of ROI ruler and ROI liquid level in Fig. 2, respectively, the value of a is simply an arbitrary input from users, as shown in Fig. 16.

3. Result of experiment

In this section, we will provide the result of our experiment and some discussions on computational complexity.

3.1. Block "Display on dashboard"

Owing to the block "Display on dashboard" in Fig. 2, Node-Red will display the experiment result in Fig. 16. We can see that, after calculation step of the liquid level in physical unit via Eq. 5, the measured liquid level value of 9 liters is visually identical to the value of the liquid level in the tank in Fig. 16.

3.2. Discussion on computational complexity

Figure 16 is an interface designed for supervisory control and data acquisition system. In this paper we have proposed a method to process the sequence of acquired images to measure the liquid level in the tank. The calculation process consists of 3 key steps:

(i) extraction of image frames from streaming video via Node-Red framework,

(ii) image pre-processing in order to create a binary image,

(iii) calculation of the histogram of distance in pixels in ROI areas of the ruler image and the image of the liquid level.

Our detection algorithm only involves very basic and low-complexity operators, yet it yields satisfied result with high accuracy in Fig. 16. Owing to the low-cost Node-Red IoT framework, our experiment shows that the distance in pixels of the ruler's lines and the liquid level in the tank can be computed fast enough for a live-streaming scenario.

4. Conclusion and further work

This paper examined a very basic computer vision method for the recognition of liquid surfaces. With the popular Node-Red IoT framework, it is feasible to connect our designed system to industrial control devices and cloud-computing system via internet.

Our method was shown to be capable of accurately detecting the distance in pixels between two lines of a tank's ruler in real time. Furthermore, since there is no assumption on ruler's property, our histogram method can be applied feasibly to arbitrary ruler in practice.

The result of our experiment shows that even a basic histogram method can be a formidable method for liquid level recognition, if the repeating property of objects like rulers can be exploited properly. Besides, this study has merits in the application of measuring object dimensions by low-cost image processing in future works.

References

[1] Ariawan, A., Pebrianti, D., Ronny, Akbar, Y.M., Margatama, L., & Bayuaji, L. (2019). Image Processing-Based Flood Detection. In *Proceedings of the 10th National Technical Seminar on Underwater System Technology 2018*, Springer Singapore, 371–380.

[2] Konovalov, D.A., Domingos, J.A., Bajema, C., White, R.D., & Jerry, D.R. (2017). Ruler Detection for Automatic Scaling of Fish Images. In *Proceedings of the International Conference on Advances in Image Processing*, ACM.

[3] Chen, G., Bai, K., Lin, Z., Liao, X., Liu, S., Lin, Z., Zhang, Q., & Jia, X. (2020). Method on water level ruler reading recognition based on image processing. *Signal, Image and Video Processing*, *15*(1), 33–41.

[4] Guo, S., Zhang, Y., & Liu, Y. (2020). A Water-Level Measurement Method Using Sparse Representation. *Automatic Control and Computer Sciences*, *54*(4), 302–312.

[5] Lin, F., Chang, W.Y., Lee, L.C., Hsiao, H.T., Tsai, W.F., & Lai, J.S. (2013). Applications of Image Recognition for Real-Time Water Level and Surface Velocity. In *2013 IEEE International Symposium on Multimedia*, IEEE.

[6] Pan, J., Fan, Y., Dong, H., Fan, S., Xiong, J., & Gui, G. (2019). Image-Based Detecting the Level of Water Using Dictionary Learning. In *Lecture Notes in Electrical Engineering*, Springer Singapore, 20–27.

[7] Chaudhary, P., D'Aronco, S., Leitão, J., Schindler, K., & Wegner, J. (2020). Water level prediction from social media images with a multi-task ranking approach. *ISPRS Journal of Photogrammetry and Remote Sensing*, *167*, 252–262.

[8] Chaudhary, P., D'Aronco, S., de Vitry, M.M., Leitão, J.P., & Wegner, J.D. (2019). Flood-water level estimation from social media images. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *IV-2/W5*, 5–12.

[9] Tran, V.H. (2014). *Variational Bayes inference in digital receivers*. Ph.D. thesis, Trinity College Dublin, https://arxiv.org/pdf/1811.02506.pdf.

[10] Rajalakshmi, A. & Shahnasser, H. (2017). Internet of Things using Node-Red and alexa. In *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*, IEEE.

[11] Torres, D., Dias, J.P., Restivo, A., & Ferreira, H.S. (2020). Real-time Feedback in Node-RED for IoT Development: An Empirical Study. In *2020 IEEE/ACM 24th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, IEEE.

[12] Sonka, M. (2015). *Image processing, analysis, and machine vision*. Stamford, CT, USA: Cengage Learning.

[13] Ramirez, M.M.G., Rincon, J.C.V., & Parada, J.F.L. (2014). Liquid level control of Coca-Cola bottles using an automated system. In *2014 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, IEEE.

[14] Feng, F., Wang, L., Zhang, Q., Lin, X., & Tan, M. (2012). Liquid Surface Location of Milk Bottle Based on Digital Image Processing. In *Multimedia and Signal Processing*, Springer Berlin Heidelberg, 232–239.

[15] ANI, Water level rises in Yamuna river following rains over past several days. http://www.businessworld.in/article/Delhi-Water-level-rises-in-Yamuna-river-following-rains-over-past-several-days/22-08-2020-311796/.

About Authors

Tai Huu LE received his MSc degree from University of Transport and Communications, Vietnam in 2015. His research interests include process control and image processing.

Phu Thai NGO is an undergraduate student in Automation and Control Engineering at Ton Duc Thang University, Vietnam. His research interest is image processing for aquaculture. He plans to pursue a PhD and bring technology back to his hometown in Ca Mau province, Vietnam.

Van Ngu TRAN received his Ph.D. degree from University of Sophia, Bulgaria in 1983. He is currently a senior lecturer at Faculty of Applied Science in Ton Duc Thang university, Vietnam. His research interests are process control and industrial heat technology.

Viet Hung TRAN received the B.Eng. degree from Hochiminh City University of Technology, Vietnam, in 2008, the MSc. degree from ENS Paris-Saclay, France, in 2009, and the Ph.D. degree from the Trinity College Dublin, Ireland, in 2014. From 2014 to 2016, he held a post-doctoral position with Telecom ParisTech, France. From 2017 to 2018, he was a Research Fellow at University of Surrey, U.K. He is currently a researcher at Ton Duc Thang University, Vietnam. His research interest is artificial intelligence and information theory. He was awarded the best mathematical paper prize at Irish Signals and Systems Conference, 2011.