Identification of Magnetic Islands in Optimized Configuration

Sora YABUMOTO[1)], Shinsuke SATAKE[1,2)] and Hiroyuki YAMAGUCHI[1,2)]

[1)]*The Graduate University for Advanced Studies, SOKENDAI, 322-6 Oroshi-cho, Toki, Gifu 509-5292, Japan*
[2)]*National Institute for Fusion Science, 322-6 Oroshi-cho, Toki 509-5292, Japan*

(Received 6 September 2022 / Accepted 23 December 2022)

To aim at the realization of a helical fusion reactor, we study multi-objective optimization of coil shapes, which satisfy various requirements. In the magnetic field configuration created by these coils, several unfavorable examples are found: some of them have magnetic islands or doublet configurations. In order to automatically and quickly exclude such cases that hinder the optimization, we have developed a new method to detect unfavorable magnetic surfaces by using image recognition. Binarization and erosion are performed as preprocessing, and then blanks of magnetic islands and doublets are extracted as recognition targets. Consequently, we have developed a classifier with high performance. Using this trained classifier, we have shown that almost all cases with unfavorable magnetic surfaces in various magnetic configurations can be excluded in a short time and with high precision.

Keywords: Haar-like feature, image recognition, image processing, machine learning, magnetic island

DOI: 10.1585/pfr.18.1403018

1. Introduction

In order to aim at the realization of a helical fusion reactor, it is necessary that the reactor simultaneously satisfies various requirements, which is treated as a multi-objective optimization problem [1–9]. In the optimization of plasma confinement properties, improvement of the symmetry of magnetic field in Boozer coordinates has often been used as an objective function. This is because it is theoretically known that neoclassical transport is kept low in magnetic field configurations with a certain symmetry [10, 11, and references therein]. Therefore, in traditional optimization studies, the optimization of the magnetic field, or the shape of the last closed flux surface (LCFS) is initially carried out, and then a coil shape is obtained to create a configuration with good symmetry, as another optimization problem. On the other hand, in our optimization of machine leaning, we use the winding law of helical coils as parameters to create various magnetic field configurations and to optimize the objective functions. This method allows one to search for the optimal configuration in a wide range of possible magnetic field configurations under engineering constraints (*e.g.*, maximum coil curvature and minimum distance between coils). However, in our test calculations, although this method allows a large degree of freedom in generated magnetic field configuration, it sometimes results in the formation of inappropriate magnetic field configurations as confinement ones: some of them have magnetic islands or doublet configurations, which have two split magnetic axes. They must be automatically excluded during machine learning because they would hinder optimization. For magnetic islands, a part of them can be detected by tracing magnetic field lines and checking the radial distribution of iota, or the rotation transformation. The iota profile can be calculated by using a variation of the poloidal phase $\Delta\theta$ and one of the toroidal phase $\Delta\phi$ along a magnetic field line. If some of the field lines are contained in an island, then a flat iota profile appears there. However, this calculation sometimes fails near the magnetic axis. Moreover, a magnetic island cannot be found by this method, unless there are several starting points of field lines located inside it. Therefore, for the purpose of detecting magnetic islands efficiently and avoiding this problem, we have developed a new method to detect unfavorable magnetic configurations by image recognition. With regard to the application of image recognition in the research of fusion plasma, several studies have used image processing and machine learning for observed plasma image data [12, 13]. On the other hand, graphical image has not been used as training data for the detection of unfavorable magnetic configurations in the optimization study so far.

The paper is structured as follows: in Sec. 2, we describe the method, which contains preprocessing of image data, extraction of their features, evaluation of performance of classifiers, and parameters. In Sec. 3, we show the results of detection of unfavorable configurations and the performance of classifiers. Section 4 summarizes the results.

2. Method

2.1 Generation of magnetic field

Machine learning for the detection of unfavorable cases by using image recognition proceeded as follows.

author's e-mail: yabumoto.sora@nifs.ac.jp

First, various shapes of helical coils were generated by randomly changing parameters of the coil shape, and the magnetic fields produced by the coils were calculated by using a magnetic field line tracking code. Here, the winding law of a helical coil is represented as follows:

$$\begin{cases} R(t) \\ \quad = R_{ax}(t) + r_{00}\epsilon_r \cos(\phi + \phi_{0r} + \alpha_r \sin\phi)[1 + \epsilon_{2r}\cos(\phi + \phi_{2r})], \\ Z(t) \\ \quad = Z_{ax}(t) + r_{00}\epsilon_z \sin(\phi + \phi_{0z} + \alpha_z \sin\phi)[1 + \epsilon_{2z}\cos(\phi + \phi_{2z})], \\ \phi(t) = Nt, \\ R_{ax}(t) = R_{00}(1 + \epsilon_{aR}\cos(\phi + \phi_{aR})), \\ Z_{ax}(t) = R_{00}\epsilon_{aZ}\sin(\phi + \phi_{aZ}). \end{cases} \tag{1}$$

In the abovementioned equation, the parameters are defined as follows: ϵ_r and ϵ_z are the magnifications of the minor radius of the coil; ϵ_{2r} and ϵ_{2z} are the scalar values to provide the bumpiness of coils; α_r and α_z are the helical pitch modulation parameters, and ϕ_{0r}, ϕ_{0z}, ϕ_{2r}, and ϕ_{2z} are the initial phases. Another coil is given by the law $(R(t), -Z(t), -\phi(t))$, which makes the produced magnetic field stellarator symmetric. For simplicity, we set $\epsilon_r = \epsilon_z$ and $\epsilon_{2r} = \epsilon_{2z}$ in the present study. The Poincaré mapping of the magnetic field lines in the cross-section at the toroidal angle ϕ = const. was generated by tracking each of the lines starting from the points on the $Z = 0$ plane, which divides the distance between the magnetic axis and LCFS into 30 equal parts. The 500 points at an intersection of each line with the ϕ = const. plane were recorded. Second, preprocessing was performed for image data. This method is described in detail in Secs. 2.2 to 2.3. Third, some operations were executed to extract indispensable information for machine learning. The details of the processes are explained in Sec. 2.4. Finally, machine learning was conducted, and the performance of the classifier was tested, which are described in Secs. 2.5 and 2.6. At this time, classifiers were created by changing the parameters and learning settings, which are described in Sec. 2.7, and their performances were compared. In this study, we used OpenCV library on Python for programming.

The images that correspond to positive and negative examples, which are classification classes, are shown at the top of Fig. 1. The images in the middle of Fig. 1 are the preprocessed top ones, and the bottom ones are the coil shapes corresponding to them.

2.2 Erosion

Binary erosion is a process for reducing the white area of a binary image, which can be executed by using the built-in function, cv2.erode. In this process, an array for conversion, namely, a kernel, is determined in advance. This array can be a rectangle, ellipse, and cross of any size. In this study, the kernel is a square matrix. Then, if the kernel is superimposed on a focused pixel in an image, and if there are one or more black points in the area, then the pixel is converted to black, and if not, it is left as it is. When this process is applied to Poincaré plots of magnetic field lines, where the kernel is a 5×5 square matrix, it works to fill the blanks between the plot points, and rational surfaces (Fig. 1). The magnetic islands and blanks of unfavorable cases become clearer.

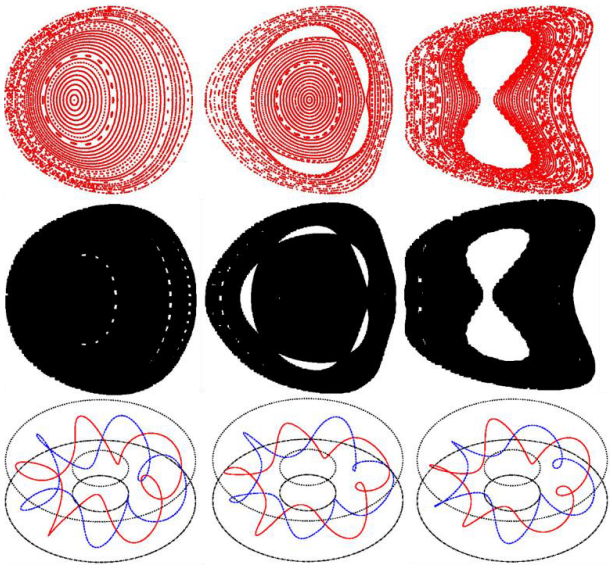

Fig. 1 These images are original Poincaré plot data, after preprocessing ones, and coil shapes. The top left shows normal magnetic surfaces, which correspond to negative cases in the present classification. The top center is an image containing magnetic islands, and the top right is a doublet example. These two images correspond to positive cases. Each of the middle and bottom images is a preprocessed one just above it and the coil shapes generating the magnetic fields, respectively. In the images of coil shapes, the four circles are two pairs of upper and lower vertical magnetic field coils. Details of preprocessing are explained in Sec. 2.2.

2.3 Contour extraction

In image recognition with OpenCV, it is necessary to give information about the positions of all detection targets as rectangles for learning. Therefore, all doublet configurations and magnetic islands of the training data are extracted, and they must be converted into rectangles encircling them. They are executed with built-in functions in two steps. First, the algorithm of contour extraction function, cv2.findContours, devised by S. Suzuki and K. Abe [14], is applied. The pixels are checked from the upper left to the right to search for a white one, and if not, the next line is scanned: this method is known as a raster scan. If found, then the pixels around the white one are checked clockwise, and the white pixel found at first becomes the next focused point. Repeating this operation, when an edge or the first point is found, a raster scan starts again. Looking for a white pixel, which is not included in the contours already found, a contour is extracted again. Repeating this sequence, all pixels that compose the con-

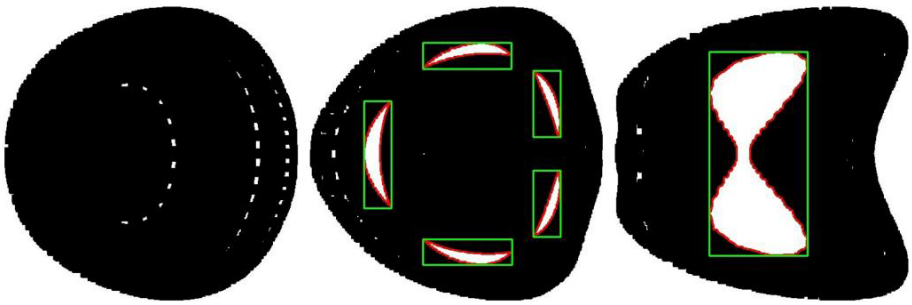

Fig. 2 Images of the processes explained in Secs. 2.3 and 2.4 are executed for the preprocessed images at the bottom of Fig. 1, without adjusting the aspect ratio. The red lines are the detected contours, and the green lines are the rectangles enclosing them. By setting the width and height thresholds above 20, some small blanks are ignored.

tours are searched for. The outermost contour, which corresponds to the LCFS, can be removed by using the hierarchy of the contours obtained by cv2.findContours.

2.4 Specifying the rectangles

In the second step, the minimum and maximum values in horizontal and vertical directions from the positions of the pixels composing each contour are extracted, and they are converted into rectangular information encircling the contour. The "opencv_traincascade" command requires rectangular information about target objects. Figure 2 shows the contours extracted from the preprocessed images of Fig. 1 by using the method described in Sec. 2.3 and the rectangles encircling them. In this operation, whether or not the positions, widths and heights of the rectangles should be adjusted to make all of them a uniform aspect ratio remains a problem. As shown in Sec. 3, training results are better when the aspect ratio of the rectangle regions is unified. However, whether the operation was executed was treated as a parameter in training because the sizes of the rectangles must be changed to unify the ratios. In addition, by setting thresholds for the widths and heights of the rectangles, leftover small spot-like holes (Fig. 2) are excluded from the training targets as much as possible.

Training was executed using the information about rectangles, and the test data were classified. Finally, the performance indexes of the classifier were calculated.

2.5 Haar-like features and cascade classifier

In this study, some classifiers were created by using Haar-like features. The Haar-like feature was devised by P. Viola and M. Jones [15]. It is less susceptible to noise, compared with using pixel values. Harr-like features are obtained as follows. First, some patterns with blue and red regions, shown on the right in Fig. 3, are prepared to calculate the features. Next, as shown on the left side of Fig. 3, a detection window is set at an arbitrary position in an image, and a rectangular area is set at an arbitrary position inside it. Then the prepared pattern is arranged in this area, and the average of pixels in the monochrome image,

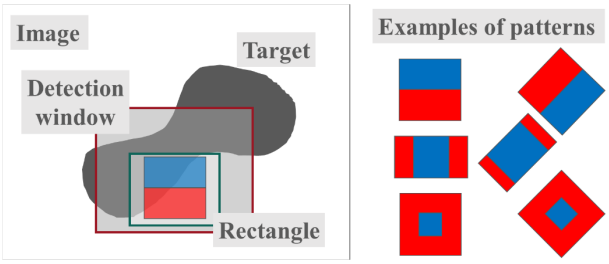

Fig. 3 Method of obtaining Haar-like features. In the left figure, a detection window is the area encircled by the reddish-brown line and a rectangular region is the one encircled by the deep-green line. Many different patterns, as shown at the right are used to evaluate the Haar-like feature.

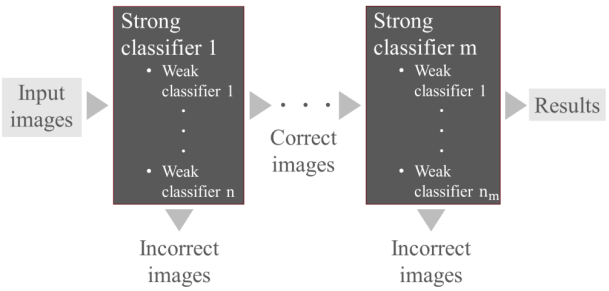

Fig. 4 Cascade classifier is composed of many strong classifiers and it also contains several weak ones.

which is covered by the blue region (A), and red region (B) of the pattern, are evaluated. The Haar-like feature H is the difference between the two regions:

$$H = A - B. \tag{2}$$

A sequence of operations is repeated by changing the position of the detection window, the position and size of the rectangular area, and a prepared pattern.

A cascade classifier is created from the features collected using the aforementioned method. It consists of several weak and strong classifiers (Fig. 4). A weak classifier is a Haar-like feature. One weak classifier cannot

determine a recognition target as a correct image, and a combination of many of them is used as a strong classifier. Strong classifiers have different detection accuracies, and by arranging them in order from the one with the lowest accuracy to the highest, a cascade classifier can quickly remove the images that are clearly different from the correct ones and speed up the recognition. Finally, only the images determined as correct by all strong classifiers remain.

2.6 Performance indexes

Indexes such as accuracy, precision, and recall are used as criteria for performance evaluation of classification by machine learning. Accuracy is the correct answer rate for all predictions, and precision is the rate at which data are determined to be positive in the prediction. In addition, recall is the rate at which positive data are determined to be so in the prediction. In general, a trade-off relationship is observed between precision and recall. Using the symbols of the confusion matrix shown in Fig. 5, the above-mentioned indexes are described by using the following equations:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}, \tag{3}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{5}$$

The higher the precision and recall are, the better the classifier is, but, as previously mentioned, a trade-off relationship is observed. Therefore, in this study, a scalar value , namely, the F-score, was used for performance evaluation, and the others were also used complementarily. It comprehensively evaluates two indexes, namely, precision and recall. This scalar value is presented by as follows:

$$\text{F}_\beta = \frac{(1+\beta^2) \times (\text{precision}) \times (\text{recall})}{(\beta^2 \cdot \text{precision}) + (\text{recall})}, \tag{6}$$

where β is the weight of recall against precision. In this study, $\beta = 1$.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

Fig. 5 Confusion matrix for a binary classification. T, F, P, and N indicate true, false, positive, and negative, respectively. TP and TN are correctly classified. The larger these are, the better the classifier is.

2.7 Parameters

During erosion, the size of a kernel and iterations are used as parameters. In the contour extraction phase, two kinds of parameters are identified. First, they are the thresholds of width and height to exclude the white spaces which are smaller than them. Such blanks appear near the rational surfaces or are merely tiny islands which can be accepted. The other parameter is whether the aspect ratio of rectangles and the size of targets at training are unified. If executed, then two cases are considered: the aspect ratio is square, or the same ratio as the original image (10:7). In the machine learning phase, two parameters are identified, namely, the size and view of the object. The classifier module used in this study requires that the training targets should have the same size. The size $\text{Y} \times \text{Z}$ is also treated as a parameter. The view is whether the contour extraction is executed for training data or only eroded images are used.

3. Result

AMD Ryzen 5 3600 6-core CPU was used as the execution environment. The training data consisted of 100 or 1200 positive images and 800 negative ones, and the test data consisted of 1000 images including 300 positive cases. The common points among the three best classifiers are to execute erosion processing using a 5×5 square matrix as the kernel and one iteration, to set the thresholds of rectangle width or height to more than 20, to unify the aspect ratios, and to extract contours. We tested several sizes of the final unified images in the training phase, which are denoted as "training size = $\text{Y} \times \text{Z}$".

The recognition results are shown in Fig. 6. The recognized blank spaces are enclosed with rectangles.

Figure 7 shows the F-score of three classifiers of which 0.9 or higher and one or more best indexes in ones created under various training conditions. The third number on the left is the time for learning, and "h" and "m" indicate hours and minutes, respectively. The average time required to classify the verification data is about 20 seconds per 1000 images.

In the image recognition of unfavorable cases, when

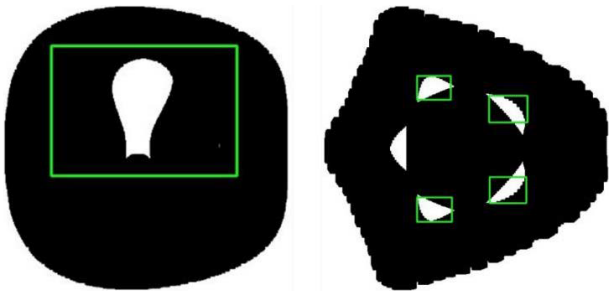

Fig. 6 Recognition results. The left image is a doublet example and the right one is an example containing several islands. The detected objects are enclosed with green lines.

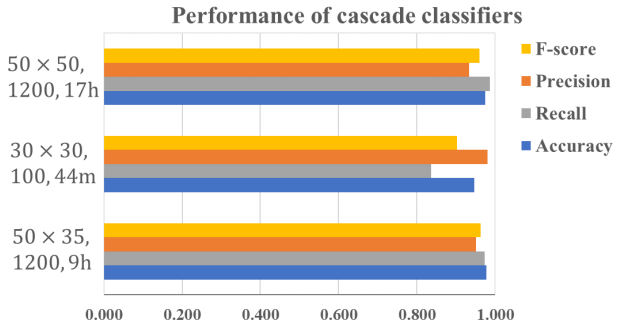

Fig. 7 Comparison between the performance indexes of cascade classifiers for anomalous cases. In classifiers with $F_1 \geq 0.9$, those with the best indexes are listed. The numbers on the left are, in order, the training size, the number of positive data, and the training time.

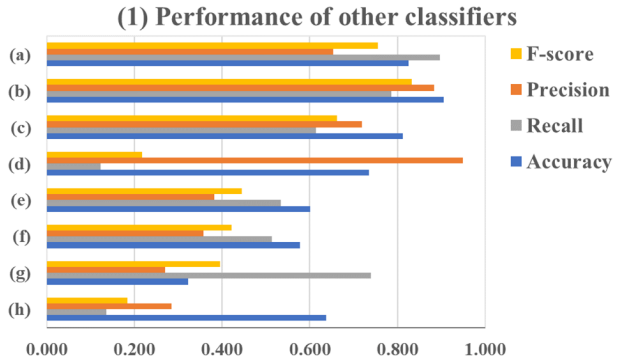

Fig. 8. 1 Performance of some cascade classifiers created by using the whole of each image is compared. In cases (a), (b), (c), and (d), the images were preprocessed, which is contrary to cases (e), (f), (g), and (h). The former had better performance than the latter. However, in two cases, the recall got worse.
(a) With erosion, training size = 50 × 35, and positive data = 1200
(b) With erosion, training size = 50 × 35, and positive data = 100
(c) With erosion, training size = 30 × 21, and positive data = 1200
(d) With erosion, training size = 30 × 21, and positive data = 100
(e) Without erosion, training size = 50 × 35, and positive data = 1200
(f) Without erosion, training size = 50×35, and positive data = 100
(g) Without erosion, training size = 30 × 21, and positive data = 1200
(h) Without erosion, training size = 30 × 21, and positive data = 100

the training size was set to 30 × 30 with 100 positive data, the precision was the best, but the other indicators were smaller than the others, and the F-score was slightly over 0.9. When the training size was set to 50 × 50 with 1200 positive data, the recall was the best, and the precision was low. The F-score exceeded 0.95. Then, in the case of 50 ×

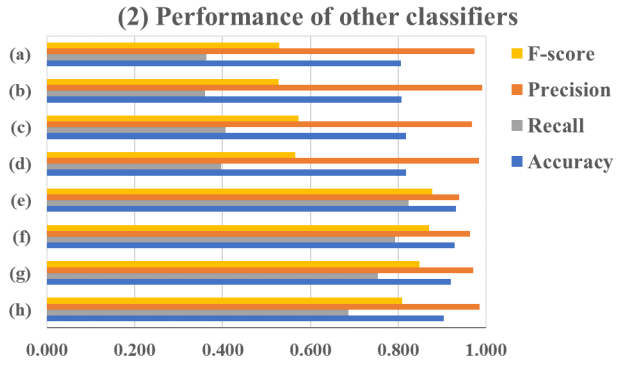

Fig. 8. 2 Performance of some cascade classifiers created by using each cavity in the images is compared. The images were preprocessed. In cases (a), (b), (c), and (d), the rectangle just surrounded each contour, and in cases (e), (f), (g), and (h), one was slightly larger. The former is called a "tight-fit rectangle," and the latter is a called a "loose-fit rectangle." The latter had better performance than the former.
(a) Tight-fit rectangle, training size = 50 × 35, positive data = 1200
(b) Tight-fit rectangle, training size = 50 × 35, positive data = 100
(c) Tight-fit rectangle, training size = 30 × 21, positive data = 1200
(d) Tight-fit rectangle, training size = 30 × 21, positive data = 100
(e) Loose-fit rectangle, training size = 50 × 35, positive data = 1200
(f) Loose-fit rectangle, training size = 50 × 35, positive data = 100
(g) Loose-fit rectangle, training size = 30 × 21, positive data = 1200
(h) Loose-fit rectangle, training size = 30 × 21, positive data = 100

35, which had a slightly smaller training size, a classifier with the best F-score and accuracy was created. In detail, compared with the case of 50×50, precision improved, and recall decreased. Given the similarity in their variations, the F-score and accuracy were almost the same. However, the training time was reduced to about half.

As a result of trials with various parameter changes, several tendencies of learning were observed. First, during preprocessing, that is, erosion, the performance indexes were improved (Fig. 8.1). Next, a larger size of targets at training and the number of images caused higher recall but lower precision. These two factors strongly affected learning time, and appropriately setting the former was particularly important for reducing the time and maintaining high performance. Third, when contour extraction was executed rather than using the images as they were, the precision was improved, but the recall tended to decrease. In this case, the recall became better by slightly expanding the size of the rectangles (20 pixels added to the width and height, Fig. 8.2). In addition, in preventing the deformation of the targets caused by enlargement or reduction,

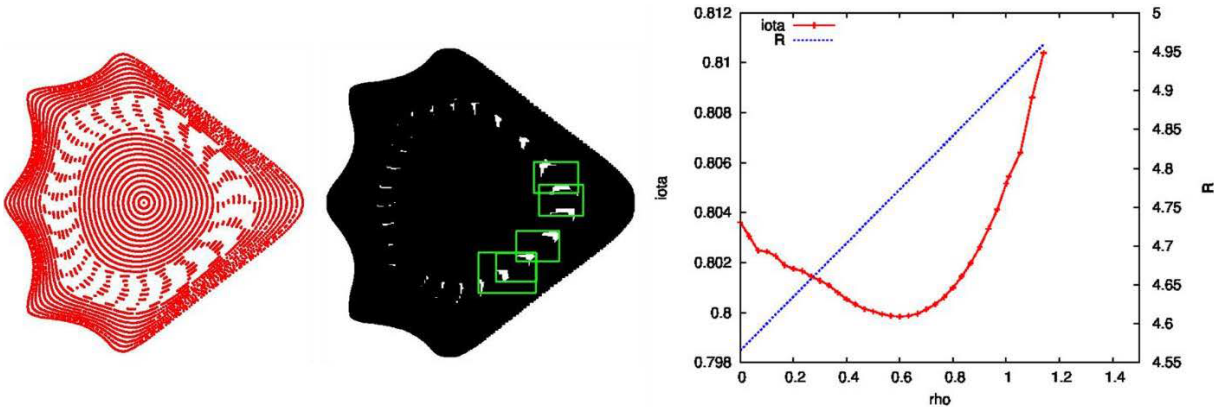

Fig. 9 Left image is the original one in which rational surfaces are erroneously detected as islands, and the center one is the result of the detection. The red line in the right graph is its iota profile. In this case, it does not monotonically increase, and it has the local minimum near a rational value.

the aspect ratio of the rectangles in the contour extraction phase was set the same as that of the size in the training phase. By unifying the aspect ratios in the extraction and training phases, the recall was improved (Fig. 7). However, the rectangle to be cut out was enlarged to match the ratio in two phases. Thus, whether the effect was on the size of targets, the matched ratio, or both of them remained unclear.

By selecting a proper set of parameters for image recognition, we demonstrated that the blank region of islands and doublet configurations can be detected with high precision. Checking for the flat-iota region in combination with the image recognition, we can automatically detect unfavorable magnetic configurations. However, we observed several false positive cases in which blank spaces around a rational surface were misjudged as islands. The detail of one sample was checked. When an iota profile has a local minimum or a weak magnetic shear region near a low-order rational value, the Poincaré plot of magnetic field lines is similar to the one on the left of Fig. 9. Despite explicit magnetic islands appearing in such cases, they are considered as inappropriate candidates for optimized configurations in the view of MHD equilibrium. This is because such a wide region with iota value close to a low-order rational value tends to be a wide island region by a small change in magnetic field. Therefore, such images should be excluded.

4. Summary

Doublet cases and magnetic islands could be recognized in the image of magnetic field lines with image recognition. Although the learning time was long, it took less than 1 day, which was sufficiently acceptable. However, in the detection of magnetic islands, some blank spaces near rational surfaces are often erroneously detected. They will be treated as positive images because their features are inappropriate as candidates of optimized configurations. When dealing with a variety of magnetic field configurations in the optimization study, unfavorable cases such as large magnetic islands could be excluded in a short time, with high precision, from the candidates for the optimized configuration survey by using the classifier.

Acknowledgments

This work was partly supported by JSPS KAKENHI Grant Number 21K03517.

[1] H. Yamaguchi *et al*., Nucl. Fusion **61**, 106004 (2021).
[2] M. Drevlak, Fusion Technol. **33**, 106 (1998).
[3] M. Drevlak *et al.*, Nucl. Fusion **59**, 016010 (2019).
[4] M. Landreman, Nucl. Fusion **57**, 046003 (2017).
[5] D.J. Sticker *et al*., Fusion Sci. Technol. **41**, 107 (2001).
[6] C. Zhu *et al.*, Nucl. Fusion **58**, 016008 (2018).
[7] C. Zhu *et al.*, Plasma Phys. Control. Fusion **60**, 065008 (2018).
[8] W.H. Miner *et al.*, Nucl. Fusion **41**, 1185 (2001).
[9] D.A. Gates *et al*., Nucl. Fusion **57**, 126064 (2017).
[10] M. Landreman *et al*., Plasma Phys. **85**, 905850103 (2019).
[11] M. Landreman *et al.*, Plasma Phys. **85**, 905850602 (2019).
[12] E. Grelier *et al*., Plasma Phys. Control. Fusion **64**, 104010 (2022).
[13] X.D. Du *et al*., Phys. Plasmas **26**, 42505 (2019).
[14] S. Suzuki and K. Abe, Computer Vision, Graphics, and Image Processing **30**, 32-46 (1985).
[15] P. Viola and M. Jones, In: CVPR, IEEE Computer Society, 511-518 (2001).