



An Approach to the Problem of Calculating Target Scores Using Camera to Support Shooting Training

Minh H. Nguyen ^{*}, Long Q. Trinh , and Tuan D. Nguyen 

Institute of Control Engineering, Le Quy Don Technical University, Hanoi, Vietnam
Email: nguyenhongminh@lqdtu.edu.vn (M.H.N.); trinquanglong@lqdtu.edu.vn (L.Q.T.);
nguyenductuan@lqdtu.edu.vn (T.D.N.)

^{*}Corresponding author

Abstract—Image processing methods are now widely used in the world in civilian fields such as handwriting recognition, face recognition, gender classification, self-driving cars, disease diagnosis, etc. The paper introduces a target reporting method using machine learning algorithms and image processing methods to identify bullet marks on targets, supporting shooting training. Bullet mark images taken by camera will be digitized and the K-Nearest Neighbor machine learning classification algorithm will be applied. The algorithm models after being trained give quite accurate results in identifying bullet mark locations in the target mark circle, thereby calculating the total score achieved by the shooter after shooting, contributing to improving shooting training capacity. The overall average accuracy rate achieved is over 98.5%, which is comparable to other techniques such as Histogram of Oriented Gradients + Support Vector Machines (HOG + SVM) and Faster Recurrent Convolutional Neural Network (Faster R-CNN). However, the processing time when applying the K-Nearest Neighbor machine learning classification algorithm is much faster, meeting the requirements for fast processing in actual shooting training.

Keywords—machine learning classification, image processing, target score reporting

I. INTRODUCTION

Currently, there are many different ways to report the target's score, such as the traditional way of sending people to the target to report the score. This method is potentially unsafe, causing danger to people when they are in the shooting range. Or using sound sensors installed on the target to measure the sound when the bullet hits the target [1]. From the sound wave spectrum to analyze, compare with sample sounds when the bullet hits different areas on the target, thereby knowing the position of the bullets on the target [2, 3]. However, this method also has many limitations such as noise interference, sound interference from the surrounding environment, and the material used to make the target is not uniform, leading to the recorded sound not being

accurate to the sample set. The sound sensors must be placed around the target surface, while at a distance of about 50–100 m, the bullet has quite a large kinetic energy, so protecting the sensors is difficult, the sound sensors installed on the target can be destroyed if hit by a bullet and the target system will be heavy, making it difficult to hide and reveal the target. When the bullet hole density is dense or the target surface is punctured, the impact sound quality will be poor or absent, then the determination of the bullet hole coordinates will have a large error or cannot be determined.

Nowadays, technology is increasingly developing, cameras have increasingly perfect hardware, long observation distance while still ensuring compact size and high definition. At the same time, today's computers have high processing speed, algorithms, and image processing methods are more diverse and optimal than before [4, 5]. Machine learning applications are widely used in many fields such as education, healthcare, etc. Iqbal *et al.* [6] performed automatic identification of human gastrointestinal abnormalities based on deep convolutional neural network with endoscopic images. This study illustrates the pipeline of using deep Convolutional Neural Networks (CNNs) for detecting abnormalities in endoscopic images. The methodological design, especially the data preprocessing and pixel-based classification framework, shares similarities with bullet holes and classifying them based on spatial features. There have also been a number of research papers on automatic shooting score using image processing methods or applying machine learning algorithms. However, these papers only mention simple target shooting objects. The boundaries of the target score areas are concentric circles so calculating the number of bullets on the target is quite easy [7]. There are research papers using machine learning algorithms such as robust Support Vector Machine (SVM) for the classification of bullet hole images [8]. Although these methods have high accuracy of about 95% to 98%, they also have many limitations such as requiring large sample data and long processing time. Additionally, bullet holes may be incorrectly

Manuscript received March 14, 2025; revised April 8, 2025; accepted June 9, 2025; published September 17, 2025.

recognized in cases where bullet holes are smaller in size or have different shapes than the training sample data. This paper introduces another approach to solving the problem of calculating target scores for shooting training. The requirement is to be applicable to targets with complex shapes, high accuracy and fast processing speed. That is, using image processing methods to identify bullet holes, then applying K-Nearest Neighbor machine learning algorithm to classify the bullet position according to the score areas on the target [9].

II. LITERATURE REVIEW

In recent years, Machine Learning has developed very strongly. The classification algorithm applies artificial intelligence technology using the most popular and effective classification algorithms today. The requirement in classifications to choose the algorithm and classification parameters that are suitable for the corresponding dataset. Each algorithm is programmed with options for algorithm parameters to help users change the structure, parameters and model of the classification system to suit the dataset and test to find the most effective model.

Song *et al.* [8] proposed a robust SVM for pattern classification, which aims at solving the over-fitting problem when outliers exist in the training data set. Experiments for the bullet hole classification problem show that the number of the support vectors is reduced, and the generalization performance is improved significantly compared to that of the standard SVM training.

Ali and Mansoor [10] proposed a computer vision based automatic scoring method for the shooting targets, performed morphological processing of the target image to thicken the boundaries of the bullet hits and then segment the target area by hysteresis thresholding. The bullet hits are labelled after the segmentation of the target area and the overlapping bullets are also scored by defining a threshold pixel area for the bullet hits.

Zhu *et al.* [11] proposed the support vector machine algorithm and convolutional neural network to extract and recognize Bullet Holes in the digital video and compares their performance. It extracts Histogram of Oriented Gradients (HOG) characteristics of bullet holes and train Support Vector Machines (SVM) classifier quickly, though the target is under outdoor environment.

Iqbal *et al.* [12] thoroughly evaluated machine learning classifiers for small image datasets—a situation directly analogous to scoring systems, where local scoring areas are used. The findings can provide valuable insights into the limitations and strengths of machine learning algorithms.

In this paper the authors propose an alternative approach to solve the problem of calculating target scores in shooting training. For the problem of determining which area of scores on the target a bullet mark belongs to, the set of characteristic parameters will be the X and Y coordinates of the pixels in the image areas. With those characteristics, we can choose K-Nearest Neighbors (KNN) Algorithm [13]. Because this

is an object classification method based on the closest distance between the object to be classified and all objects in the training dataset, where the distance between objects is usually calculated by the Euclidean Distance which is the distance determined between two points on the OXY coordinate plane. This method is suitable for the characteristic parameter set of the problem. Moreover, it is suitable for datasets that are not too large, the processing speed is fast but still ensures accuracy, meeting the requirements. In addition, there are many other machine learning classification methods such as Decision Tree Algorithm, Random Forests Algorithm, Kernel Support Vector Machine, etc. However, these algorithms are very complex, slow processing speed, require high server configuration, so they are not suitable for the given problem.

III. MATERIALS AND METHODS

Fig. 1 describes a target reporting system.

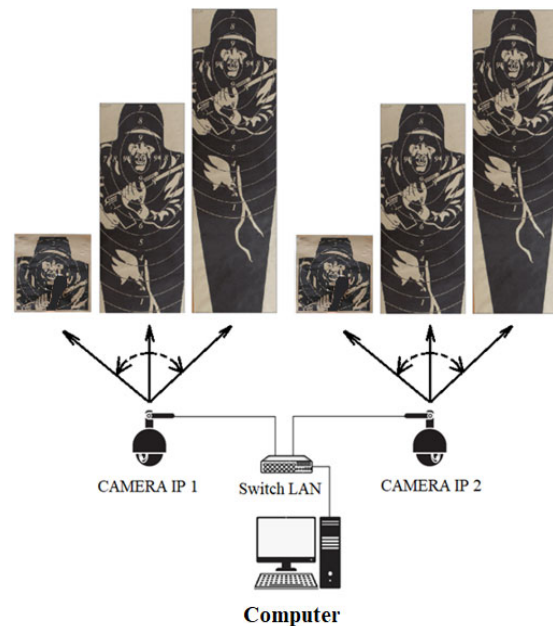


Fig. 1. Model of target scoring system.

The target reporting system consists of the following main components:

- A set of targets in a firing line. Behind each target, a red screen is installed so that when the bullet passes through the target, the bullet mark on the target displayed by the camera will be red. Then the displayed bullet mark will not be the same color as the target's background color. The targets are all equipped with a remote target replacement system after each shot.
- An IP camera is responsible for taking pictures of the target, connecting to a Local Area Network (LAN) and transmitting the images to the central computer via a Switch LAN network port splitter. This is a high-resolution camera, capable of rotating, installed in the middle, in front of the targets at a suitable distance and

- placed on the ground in a technical box.
- A set of Switch LAN network port splitters is responsible for connecting the IP cameras of the firing lines to the central computer.
- A central computer is installed with target reporting software.

The flowchart of the proposed method is clearly shown in Fig. 2.

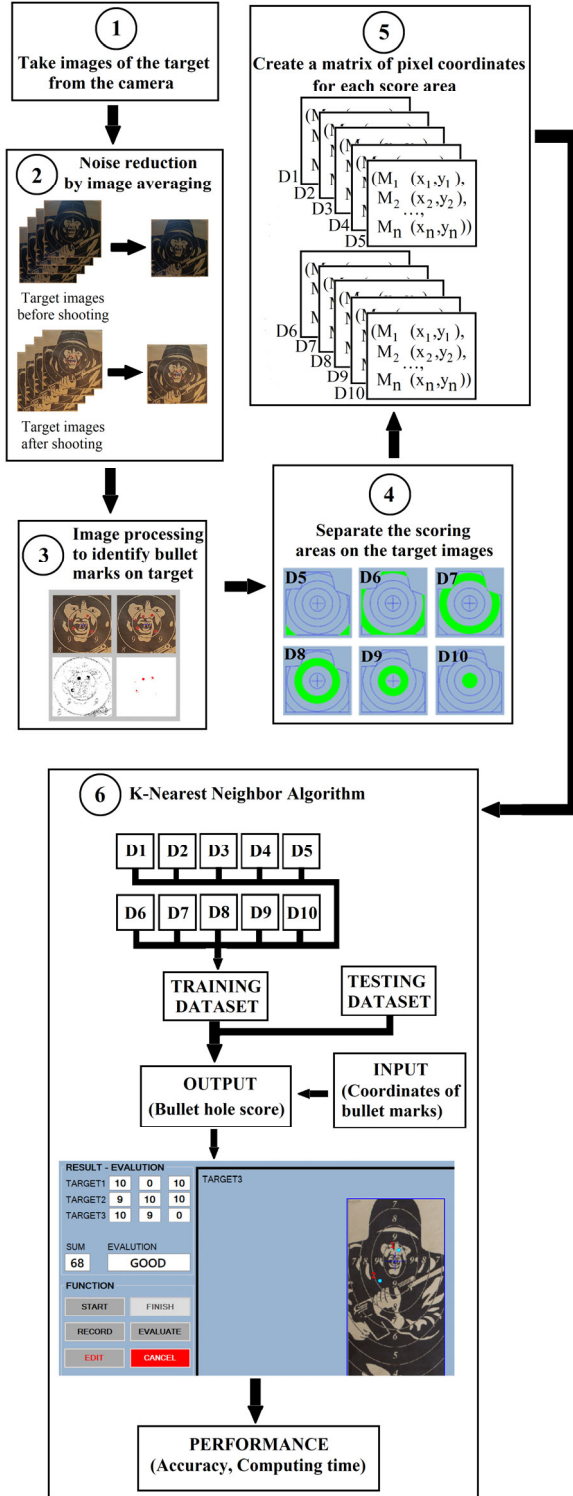


Fig. 2. Flowchart indicating the procedures carried out in automatic shooting scoring system.

The process of identifying bullet marks on the target image includes the following steps:

A. Step 1

Take some photos of the target before shooting and take some photos of the target after shooting. Photos taken will be filtered for noise by averaging the images to improve image quality [14–17]. Fig. 3 explains how to average images to reduce noise.

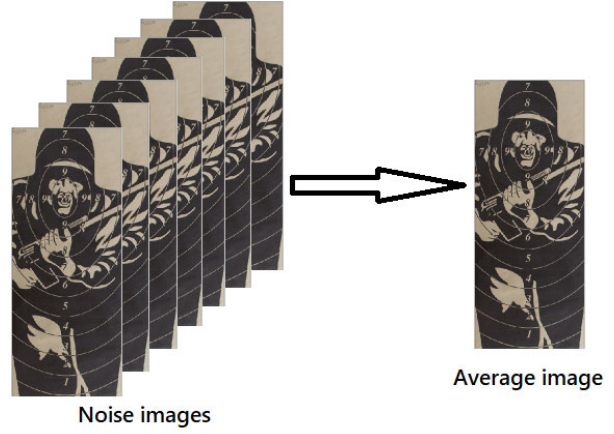


Fig. 3. Noise reduction by image averaging.

A noisy image $h(x, y)$ will consist of noise $\eta(x, y)$ and an original image $m(x, y)$, expressed as follows:

$$h(x, y) = m(x, y) + \eta(x, y) \quad (1)$$

where it is assumed that at all coordinate pairs (x, y) the noise has zero mean and is uncorrelated. The goal of the following procedure is to add a set of noisy images to reduce the noise content, $\{h_i(x, y)\}$.

When the noise meets all the above conditions, we proceed to average K different noise images to create an image $\bar{h}(x, y)$,

$$\bar{h}(x, y) = \frac{1}{K} \sum_{i=1}^K h_i(x, y) \quad (2)$$

Therefore,

$$E\{\bar{h}(x, y)\} = m(x, y) \quad (3)$$

And

$$\sigma_{\bar{h}(x, y)}^2 = \frac{1}{K} \sigma_{\eta(x, y)}^2 \quad (4)$$

where the expected value of \bar{h} is $E\{\bar{h}(x, y)\}$ and the variances of \bar{h} and η are $\sigma_{\bar{h}(x, y)}^2$ and $\sigma_{\eta(x, y)}^2$ all at coordinates (x, y) . At any point, the standard deviation in the average image as follow:

$$\sigma_{h(x,y)} = \frac{1}{K} \sigma_{\eta(x,y)} \quad (5)$$

B. Step 2

Use image processing methods to compare each pixel on the 2 photos of the target before and after shooting. Filter out the pixels that have been changed in the photo of the target after shooting with the photo of the target before shooting, with colors similar to the color of the background behind the target.

In Fig. 4, we can see that the bullet hole on the target shown in the image will have the color of the object behind the target (for example a wall, cliff or sandbags). If this color is similar to the color of an area of the target image, it is difficult to determine the bullet hole on the target image. So, authors installed a red screen behind the targets, the bullet hole on the target shown in the image will be red.

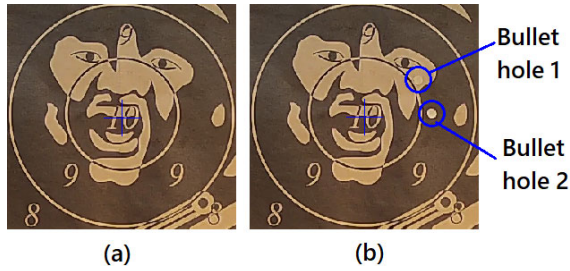


Fig. 4. Target images without a red screen behind the target. (a) Target image before shooting; (b) Target image after shooting.

By computing the difference between all corresponding pixel pairs from $m(x, y)$ and $n(x, y)$, we find the difference between two images m and n is expressed by the following expression [18]:

$$g(x, y) = m(x, y) - n(x, y) \quad (6)$$

We have the color difference between 2 pixels as follows [18]:

$$Dif = \sqrt{(A_1 - A_2)^2 + (B_1 - B_2)^2 + (C_1 - C_2)^2} \quad (7)$$

where A_1, B_1, C_1 are the red, green, blue values of the pixel before firing and A_2, B_2, C_2 are the red, green, blue values of it after firing.

If the Dif value is greater than the threshold value D_{T1} set in the software, these pixels will be considered as changed pixels.

From Figs. 5 and 6, we can see that when there is a change in the light source, the larger the threshold value D_{T1} , the more noise pixels are eliminated, making the bullet marks on the target more clearly visible. However, the D_{T1} value must not exceed the smallest value between D_{T2} and D_{T3} . D_{T2} and D_{T3} are determined by the following equations:

$$D_{T2} = \sqrt{(A_D - A_S)^2 + (B_D - B_S)^2 + (C_D - C_S)^2} \quad (8)$$

$$D_{T3} = \sqrt{(A_L - A_S)^2 + (B_L - B_S)^2 + (C_L - C_S)^2} \quad (9)$$

where A_D, B_D, C_D are the red, green, blue values of the pixel in the dark area of the target. A_L, B_L, C_L are the red, green, blue values of the pixel in the light area of the target. A_S, B_S, C_S are the red, green, blue values of the pixel in the red screen of the target.

Because if $D_{T1} > D_{T2}$ then bullet holes that on the dark area of target will not be shown in the image of the changed pixels or if $D_{T1} > D_{T3}$ then bullet holes that on the light area of target will not be shown in the image of the changed pixels.

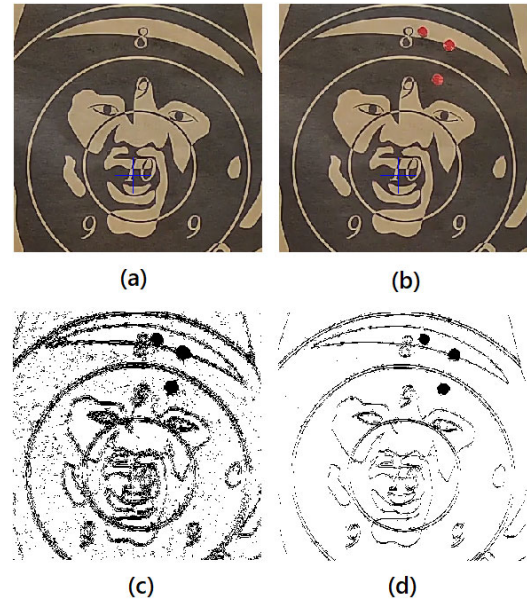


Fig. 5. Target images before and after shooting with steady light source. (a) Target image before shooting; (b) Target image after shooting; (c) The changed pixels when the threshold value $D_{T1} = 15$; (d) The changed pixels when the threshold value $D_{T1} = 35$.

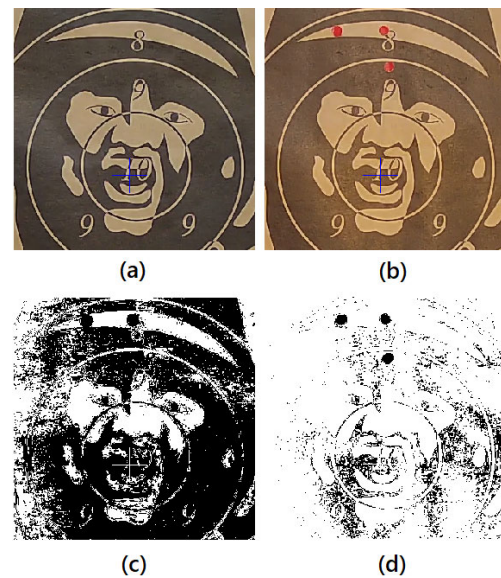


Fig. 6. Target images before and after shooting with unstable light source. (a) Target image before shooting; (b) Target image after shooting; (c) The changed pixels when the threshold value $D_{T1} = 35$; (d) The changed pixels when the threshold value $D_{T1} = 60$.

To evaluate the effect of lighting conditions, the authors used a total of 200 images, covering 4 types of illumination conditions (indoors with steady light, outdoors with clouds, outdoors with sunshine, outdoors with shade). Let D_{T4} be the color difference of pixels in the dark areas between images, and D_{T5} be the color difference of pixels in the bright areas between images. Through calculations, the author finds that D_{T4} and D_{T5} are almost smaller than D_{T2} and D_{T3} . Thus, when choosing a suitable D_{T1} , the bullet marks are clearly displayed between the different changing pixels. Fig. 7 shows how to get the average color of image areas on the target.

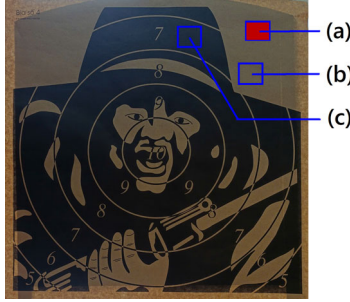


Fig. 7. Get average color of image areas on target. (a) A piece of red screen. (b) A dark area on the target. (c) A light area on the target.

These changed pixels will be further compared with the color of the red screen to remove noisy pixels of the target image background and determine the bullet hole on the target. This is clearly shown in Fig. 8.

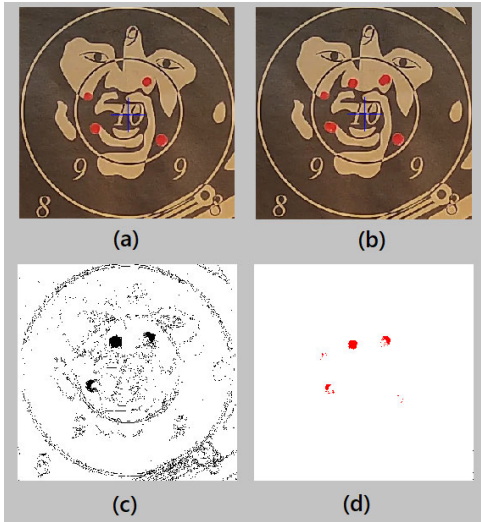


Fig. 8. (a) Target image before shooting; (b) Target image after shooting; (c) The changed pixels when the threshold value $D_{T1} = 60$; (d) The changed pixels have a color that closely matches the color of red screen behind the target.

In Fig. 9, we can see the shape of bullet holes are very diverse. They can be circular, semicircular, or any other shape. Moreover, the size of the bullet hole image can be large or small depending on the distance between the target and the camera. Therefore, the authors do not use image recognition algorithms such as deep learning, convolutional neural networks, etc. to determine the

bullet holes on the targets. They are quite complex, require large sample data, making the processing speed slow and ineffective in this case. In practice, the scoring process from the time the shot is finished to the time the results are available requires a fast enough time, about 1 to 3 Minutes (depending on the supervisor's requirements). The solution to identify bullet holes will be presented in the next step.

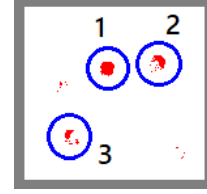


Fig. 9. Bullet holes shown on target image. (1) A perfect bullet hole. (2) New bullet hole covers a small part of old bullet hole. (3) New bullet hole covers a large part of old bullet hole.

C. Step 3

Then from the above pixel set, use the image processing algorithm to remove discrete, unconnected pixels.

The coordinates of the pixels are taken according to the coordinate axes as shown in Fig. 10.

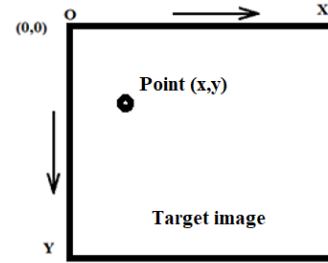


Fig. 10. Coordinates of a point on the target image.

Euclidean distance formula is often used to calculate the distance between 2 pixels (x_1, y_1) and (x_2, y_2) [19].

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (10)$$

If the distance D is greater than the threshold value set in the software, the 2 pixels will be considered discrete and not connected.

After that remove pixels that form an image area larger than the largest possible area of a bullet hole on the target and remove pixels that form an image area smaller than the smallest possible area of a bullet hole on the target. The remaining image areas that satisfy the above conditions will be considered bullet holes. This process is shown in Fig. 11. The bullet holes will be labeled in order from small to large according to the distance between the bullet holes and the center of the target.

$$S_{Min} \leq S \leq S_{Max} \quad (11)$$

where S is size of bullet hole image, S_{Min} is minimum size

of bullet hole image, S_{Max} is maximum size of bullet hole image. S_{Min} and S_{Max} are adjusted based on the distance between the target and the camera.

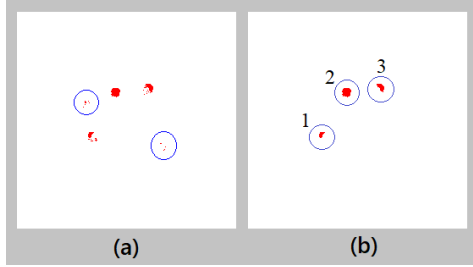


Fig. 11. (a) The changing pixels have a color that closely matches the color of the red screen behind the target. (b) After removing the discrete and unconnected pixels, bullet holes will be labeled.

Under normal conditions, bullet marks can be easily identified automatically through image processing algorithms as above. To improve accuracy, users can use semi-automatic methods to identify bullet marks. The algorithms perform a first pass of pre-annotation then users correct and refine the results. However, in some special cases such as bullet marks partially overlapping, the software will mistake them for a bullet hole or eliminate them due to excessive size, manual methods will be applied to identify bullet marks. Users will label the bullet marks on the target themselves.

D. Step 4

Use the algorithm to find the centroid for the remaining image areas [20, 21]. The remaining image areas that satisfy the conditions set for the actual image area are the bullet holes that need to be found on the target image. The centroid G of the image areas is calculated by the formula of the average of all pixels that make up that image area.

$$x_G = \frac{x_1 + x_2 + \dots + x_n}{n} \quad (12)$$

$$y_G = \frac{y_1 + y_2 + \dots + y_n}{n} \quad (13)$$

where: x_G, y_G , are the coordinates of the centroid G of an image area. n is the number of pixels in an image area. x_i, y_i (with $i = 1, \dots, n$) are the coordinates of the pixels in an image area.

From there, calculate the coordinates of the bullet image corresponding to each circle of points on the target, as a basis for determining the total score achieved.

IV. APPLY K-NEAREST NEIGHBORS ALGORITHM TO CALCULATE BULLET MARKS ON TARGET

If the boundaries of the score areas on the target are concentric circles, so calculating the scores of bullet marks on the target is quite easy. However, the boundaries of the score areas on the target are concentric ellipses and the scoring areas on the target have complex,

dissimilar shapes. To calculate the score of a bullet mark, it is necessary to determine which scores area on the target the bullet mark is located in.

In Fig. 12, from the images of the targets, the layers are separated, obtain the dividing contours on the target to determine the areas for calculating the scores on the target [21–25].

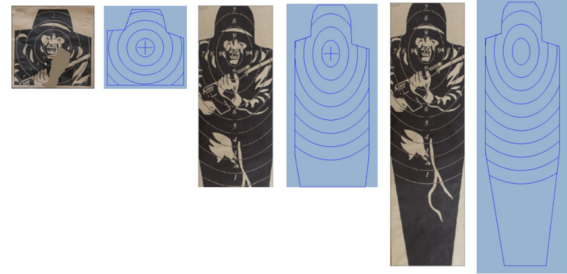


Fig. 12. Separate the scoring areas on the target image.

From the borders we continue to divide the specific score areas on the target.

The actual score areas, as defined by the above markers, are bounded by specific straight lines and curves. The contours of the score areas are straight lines and circles or straight lines and ellipses. The geometric shapes of the score areas are quite diverse, different and complex, not just consisting of circles and ellipses. If geometric methods are applied to determine the location of the bullet hole at which score area on the target, it will be quite complicated. Fig. 13 shows scoring zones on the targets.

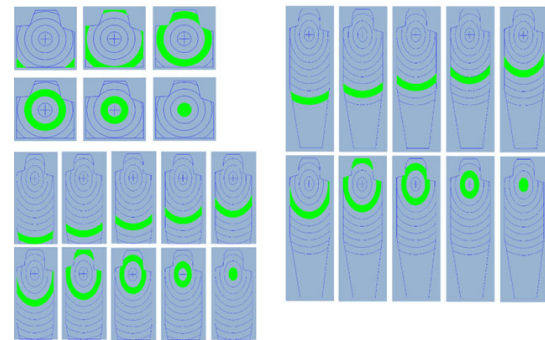


Fig. 13. Scoring zones on the targets.

To solve this problem, after taking a photo of the target and separating the score areas, image processing steps are used to find all pixels that have the same color as the dark and light areas in the target score area. Next, convert these pixels in the score area into an array of pixel coordinates. However, these pixels will not be seamless but will be patchy and uneven due to image noise and non-uniform color on the target background image. The array of coordinates of these pixels does not represent the entire score area in the target. Fig. 14 shows score level 6 area in Target 2 after being separated and image processed.

Therefore, this paper proposes a method of applying KNN algorithm to classify and calculate target scores according to the location of bullet holes [26]. The KNN algorithm is widely used in Machine Learning. This is

an object classification method based on the closest distance between the Query Point (the object to be classified) and the Training Data (all the objects in the training dataset). Based on K neighbors, an object is classified where K is a positive integer determined before implementing the algorithm. the distance between objects is usually calculated by Euclidean Distance [27].



Fig. 14. Example of a score level 6 area in Target 2 after being separated and image processed.

The Euclidean distance is the distance defined between two points in the OXY coordinate plane. Euclidean distance (D) for 2 points $P1(x_1, y_1)$ and $P2(x_2, y_2)$ is calculated according to the formula below:

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (14)$$

The K-Nearest Neighbor algorithm is defined as follows:

- Based on the problem requirements to determine the parameter value K (number of nearest neighbors).
- Calculate the Euclidean distance between the object to be classified (called "Query Point") and all objects in the training data set (called "Training Data").
- Based on increasing distance values to sort order. Determine K-Nearest Neighbors to the object to be classified (Query Point).
- Get all the classes of the determined K-Nearest Neighbor.
- Class for Query Point determined based on majority of classes of nearest neighbors.

In the figure below, the training data set is described by "+" sign (representing pixels in the score area 7) and "-" sign (representing pixels in the score area 8), the object that needs to be classified (Query point) is the red circle (representing the coordinates of the center of the shot mark on the target). Based on the number of nearest neighbors to predict the class of the Query Point. From there we consider whether the Query Point will be classified into "+" class (score 7) or "-" class (score 8).

In Fig. 15:

1-Nearest neighbor: The result is the bullet mark in the area of score 7 (because the "+" class is the class into which the Query Point is classified)

2-Nearest neighbors: the class for Query Point is not determined because the number of nearest neighbors to it is 2, in which 1 is the "+" class and 1 is the "-" class (no class has more objects than the other class). In this case, for the target score problem, the higher score will be prioritized.

5-Nearest neighbors: The result is the bullet mark in the area of score 8 (in this case, the "-" class is the class

into which the Query Point is classified because in the 5 nearest neighbors to it, there are 3 objects in the "-" class, more than the "+" class, which has only 2 objects).

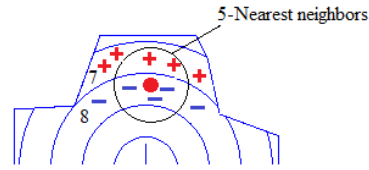


Fig. 15. Example of using KNN algorithm to classify bullet marks as belonging to score 7 or 8.

Applied to the problem at hand, the target score areas will be converted into the corresponding pixel coordinate matrices. For example, in Fig. 14 above, in the image area of score level 6, the black and white pixels will be converted into a pixel coordinate matrix $(M1(x_1, y_1), M2(x_2, y_2), \dots, Mn(x_n, y_n))$. This coordinate matrix will be used as a sample dataset of the score level 6 value on the target. Similarly, determine the pixel coordinate matrix with the image areas of the remaining score levels on the target and use it as a sample dataset. The pixel coordinates of each score area will be introduced in the training dataset. For Target 1, the dataset consists of 5 classes corresponding to scores from 5 to 10. As for Target 2 and Target 3, the dataset includes 11 classes corresponding to scores from 0 to 10. This is shown in the Table I below.

TABLE I. SAMPLE DATASET CORRESPONDING TO THE PIXEL COORDINATE MATRIX OF THE "0" SCORE AREA ON THE TARGET

X coordinate	Y coordinate	¹ Class
3009	538	0
3020	540	0
3102	545	0
3194	544	0
3083	546	0
2922	539	0
3001	542	0

¹ Class represents that the target score values from 0 to 10.

The pixels in each target score area can be labelled manually, semi-automatically or automatically. Because the number of pixels of each target score area is very large, semi-automatic annotation and manual annotation are usually only used to calibrate and refine the results.

Next, the K-Nearest Neighbor Algorithm model is built to apply to this problem, illustrated in Fig. 16.

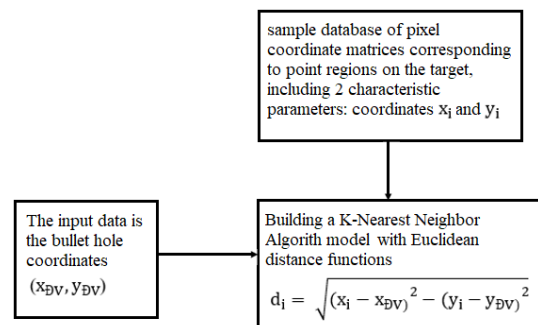


Fig. 16. K-Nearest Neighbor Algorithm model.

V. RESULT AND DISCUSSION

Set the training parameter for the KNN algorithm as parameter K (the number of nearest neighbors considered). If the parameter K is chosen too high or too low, it will affect the processing speed of the computer and reduce the accuracy of the algorithm. With too small a value of K can make the model very sensitive to noise and may overfit the training data. It will focus on the nearest neighbors, which may not always represent the global distribution. With large K values can cause the model to smooth the decision boundary leading to mismatch. The classifier becomes less sensitive to local patterns and more influenced by the overall data distribution. The K value can be found using the elbow method by plotting the performance of KNN against different K values. The elbow point on the plot represents the optimal K , at which the error rate starts to flatten out. Next, train the feature parameter set, which is the X , Y coordinates, with the obtained sample dataset. When the camera is set to Full HD (1920×1080 pixel), the image of targets will be adjusted to fit the screen, then they will have a resolution of about 242×242 pixels (Target 1), 259×536 pixels (Target 2) and 259×789 pixels (Target 3) in Fig. 17.

For testing, the authors programmed a software using Visual Studio and Accord. Machine Learning package. This package is a part of Accord.Net Framework. It contains machine learning models such as K-means model, Decision Trees model, Random Forest model, ... for machine learning applications. The software runs on a computer with the following configuration: Windows 10 (64 bit), Intel(R) Core (TM) i7-4910MQ, CPU 2.90GHz, 16.0 GB RAM.

First, test with an image of Target 1 of size 242×242 pixels, the number of pixels of all classes given in Fig. 18 is 61,116. From there we have a training dataset which will contain 61,116 labeled pixels where each pixel is characterized by its x and y . This dataset consists of 5 classes corresponding to scores from 5 to 10.

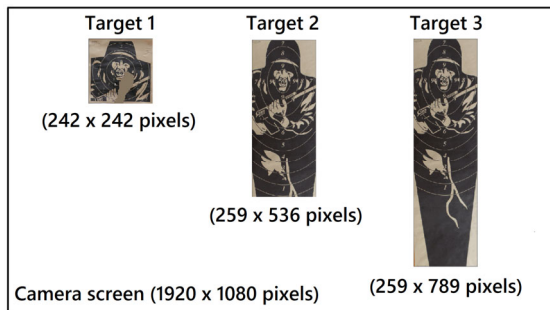


Fig. 17. Size of target images on screen.

	Score	Numer of pixels
0	0	21268
5	5	1235
6	6	6713
7	7	13014
8	8	10643
9	9	6234
10	10	2009

Fig. 18. The number of pixels of each class (Target 1).

For Target 1, the K value can be found using the elbow method with K values ranging from 1 to 300.

From Figs. 19 and 20, for Target 1, the optimal K can be between 8 and 25, where the accuracy is consistently high indicating the best balance between overfitting and underfitting, processing time is also guaranteed to be fast enough as required. According to the elbow method, the authors choose the value K as 20 when applying the KNN algorithm to Target 1. The confusion matrix with $K = 20$ for Target 1 is shown in Fig. 21. Table II presents the relationship between K value and processing time and accuracy for Target 1.

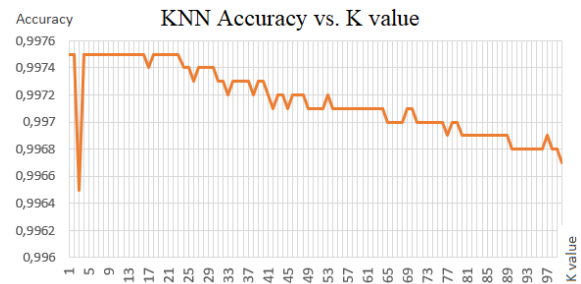


Fig. 19. KNN Accuracy vs. K value (Target 1).

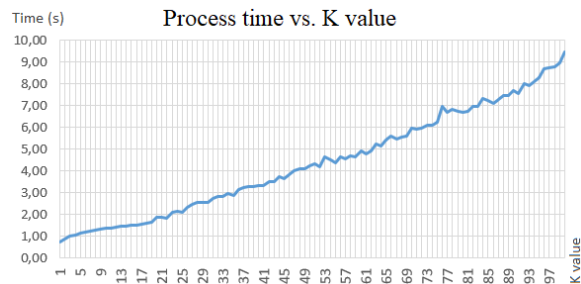


Fig. 20. Process time vs. K value (Target 1).

		Predicted class						
		0	5	6	7	8	9	10
Actual class	0	21268 34.8%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
	5	0 0%	1211 1.98%	24 0.04%	0 0%	0 0%	0 0%	24 0.04%
	6	1 0%	16 0.03%	6675 10.92%	21 0.03%	0 0%	0 0%	38 0.06%
	7	0 0%	0 0%	33 0.05%	12975 21.23%	6 0.01%	0 0%	39 0.06%
	8	0 0%	0 0%	0 0%	8 0.01%	10628 17.39%	7 0.01%	15 0.02%
	9	0 0%	0 0%	0 0%	0 0%	11 0.02%	6222 10.18%	1 0%
Actual class	10	0 0%	0 0%	0 0%	0 0%	0 0%	11 0.02%	1998 3.27%
		1 0%	16 0.03%	57 0.09%	29 0.04%	17 0.03%	18 0.03%	1 0%
		60977 99.77%						

Fig. 21. Confusion matrix with $K = 20$ (Target 1).

Continue testing with an image of Target 2 of size 259×536 pixels. The number of pixels of all classes given in Fig. 22 is 139,408. From there we have a training dataset which will contain 139,408 labeled pixels where each pixel is characterized by its x and y . This dataset

consists of 11 classes corresponding to scores from 0 to 10.

TABLE II. TABLE OF RELATIONSHIP BETWEEN K AND PROCESSING TIME AND ACCURACY (TARGET 1)

K value	Processing time (s)	Accuracy
20	1.6	0.998
30	1.8	0.998
100	9.2	0.997
200	13.9	0.994
300	22.1	0.991

Note: Test with an image of Target 1 of size 242×242 pixels.

Score	Number of pixels
0	40960
1	8048
2	8575
3	8761
4	8974
5	9410
6	10994
7	16189
8	14966
9	9414
10	3117

Fig. 22. The number of pixels of each class (Target 2).

For Target 2, the K value can be found using the elbow method with K values ranging from 1 to 300.

From Figs. 23 and 24, for Target 2, the optimal K can be between 8 and 25, where the accuracy is consistently high indicating the best balance between overfitting and underfitting, processing time is also guaranteed to be fast enough as required. According to the elbow method, the authors choose the value K as 20 when applying the KNN algorithm to Target 2. The confusion matrix with $K = 20$ for Target 2 is shown in Fig. 25. Table III presents the relationship between K value and processing time and accuracy for Target 2.

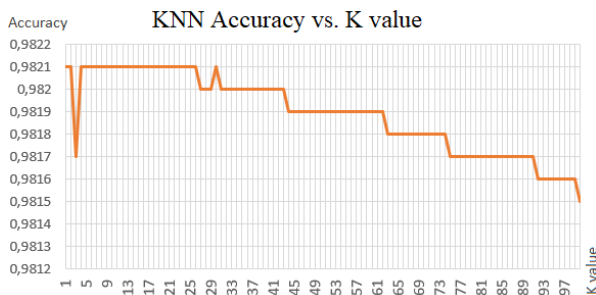


Fig. 23. KNN Accuracy vs. K value (Target 1).

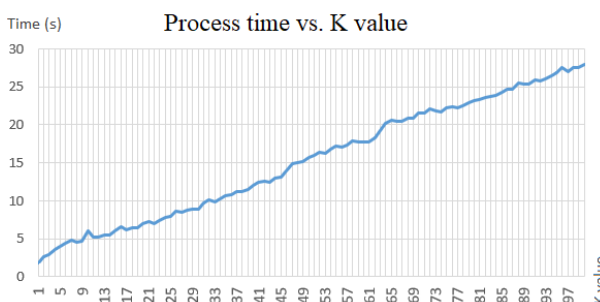


Fig. 24. Process time vs. K value (Target 2).

Actual class	Predicted class										
	0	1	2	3	4	5	6	7	8	9	10
0	40957 29.38%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	2 0%	1 0%	0 0%	3 0%
1	1 0%	7954 5.71%	93 0.07%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	94 0.07%
2	1 0%	116 0.08%	8363 6%	95 0.07%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	212 0.15%
3	0 0%	0 0%	114 0.08%	8555 6.14%	92 0.07%	0 0%	0 0%	0 0%	0 0%	0 0%	206 0.15%
4	1 0%	0 0%	0 0%	120 0.09%	8757 6.28%	96 0.07%	0 0%	0 0%	0 0%	0 0%	217 0.16%
5	0 0%	0 0%	0 0%	0 0%	124 0.09%	9166 6.57%	120 0.09%	0 0%	0 0%	0 0%	244 0.18%
6	1 0%	0 0%	0 0%	0 0%	0 0%	127 0.09%	10700 7.68%	166 0.12%	0 0%	0 0%	294 0.21%
7	6 0%	0 0%	0 0%	0 0%	0 0%	0 0%	211 0.15%	15773 11.31%	199 0.14%	0 0%	416 0.29%
8	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	263 0.19%	14534 10.43%	169 0.12%	0 0%	432 0.31%
9	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	197 0.14%	9151 6.56%	66 0.05%	263 0.19%
10	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	118 0.08%	2999 2.15%	118 0.08%
	10 0%	116 0.08%	207 0.15%	215 0.16%	216 0.16%	223 0.16%	331 0.24%	431 0.31%	397 0.28%	287 0.2%	136909 98.21%

Fig. 25. Confusion matrix with $K = 20$ (Target 2).

TABLE III. TABLE OF RELATIONSHIP BETWEEN K AND PROCESSING TIME AND ACCURACY (TARGET 2)

K value	Processing Time (s)	Accuracy
20	6.7	0.982
30	8.5	0.982
100	25.3	0.982
200	63.1	0.981
300	90.9	0.980

Note: Test with an image of Target 2 of size 259×536 pixels.

Final testing with an image of Target 3 of size 259×789 pixels. The number of pixels of all classes given in Fig. 26 is 169,485. From there we have a training dataset which will contain 169,485 labeled pixels where each pixel is characterized by its x and y . This dataset consists of 11 classes corresponding to scores from 0 to 10.

Score	Number of pixels
0	71033
1	8048
2	8575
3	8761
4	8978
5	9410
6	10994
7	16189
8	14966
9	9414
10	3117

Fig. 26. The number of pixels of each class (Target 3).

From Figs. 27 and 28, for Target 3, the optimal K can be between 8 and 25, where the accuracy is consistently high indicating the best balance between overfitting and underfitting, processing time is also guaranteed to be fast enough as required. According to the elbow method, the authors choose the value K as 20 when applying the KNN algorithm to Target 3. The confusion matrix with $K = 20$ for Target 3 is shown in Fig. 29. Table IV presents the relationship between K value and processing time and

accuracy for Target 3.

For Target 3, the K value can be found using the elbow method with K values ranging from 1 to 300.

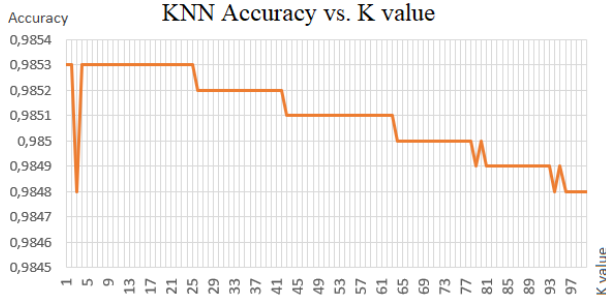


Fig. 27. KNN Accuracy vs. K value (Target 3).

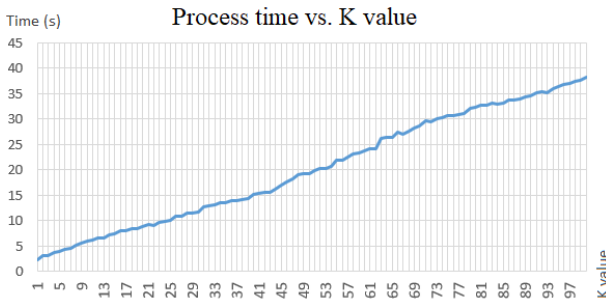


Fig. 28. Process time vs. K value (Target 3).

		Predicted class										
		0	1	2	3	4	5	6	7	8	9	10
Actual class	0	71033 41.91%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
	1	0 0%	7949 4.69%	98 0.06%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	99 0.06%
	2	0 0%	0 0%	8371 4.94%	93 0.05%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	204 0.12%
	3	0 0%	0 0%	116 0.07%	8547 5.04%	98 0.06%	0 0%	0 0%	0 0%	0 0%	0 0%	214 0.13%
	4	0 0%	0 0%	0 0%	114 0.07%	8764 5.17%	100 0.06%	0 0%	0 0%	0 0%	0 0%	214 0.13%
	5	0 0%	0 0%	0 0%	0 0%	125 0.07%	9172 5.41%	113 0.07%	0 0%	0 0%	0 0%	238 0.14%
	6	2 0%	0 0%	0 0%	0 0%	0 0%	134 0.08%	10687 6.31%	171 0.1%	0 0%	0 0%	307 0.18%
	7	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	206 0.12%	15780 9.31%	196 0.12%	0 0%	409 0.24%
	8	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	266 0.16%	14538 8.58%	162 0.1%	0 0%	428 0.26%
	9	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	204 0.12%	9121 5.38%	89 0.05%	293 0.17%
	10	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	3022 1.78%	95 0.06%	106904 60.53%

Fig. 29. Confusion matrix with $K = 20$ (Target 3).

TABLE IV. TABLE OF RELATIONSHIP BETWEEN K AND PROCESSING TIME AND ACCURACY (TARGET 3)

K value	Processing time (s)	Accuracy
20	8.0	0.985
30	11.1	0.985
100	34.3	0.985
200	76.9	0.984
300	126.1	0.983

Note: Test with an image of Target 3 of size 259×789 pixels.

In Figs. 21, 25 and 29, from the confusion matrices we can see that errors occur when some pixels are divided

into 2 or 3 classes because these pixels are located on the dividing lines between 2 score areas. With the size of the target images on the screen as shown in Fig. 15, the K value is chosen in the range of 8–25 by the elbow method. In the experiments, the authors chose the value K as 20. With that K value, the processing time is fast enough and meets the highest accuracy.

After having the training algorithm model, we check the classification by entering the coordinate parameters of the bullet mark (x_{BH} , y_{BH}) into the input data table. The result will be returned as the score value along with the accuracy percentage table corresponding to each score level on the target. The accuracy percentage table is determined by the following equations:

$$P_S = \frac{K_S}{K} \times 100\% \quad (15)$$

$$K = \sum_{i=0}^{10} K_i \quad (16)$$

where S is the score level from 0 to 10, K is the number of nearest neighbors, P_S is accuracy percentage of score level S , K_S is the number of pixels belonging to the score level S , K_i is the number of pixels belonging to the score level from 0 to 10. To test the accuracy of the algorithm when calculating the target score, the authors tested with 100 bullet holes. The coordinate arrays of 100 bullet holes were entered into the software for calculation. After comparing the software's calculation results with the observation of bullet holes displayed on the target interface in the software, the authors found that the correct scoring rate was about 90 to 95 out of 100 points.

Errors occur when bullet holes are located near the dividing lines between 2 score areas. For special case such as bullet hole coordinates with equal P_S values at the score levels. In that case, the Majority vote technique (whichever class is in the majority, predict the result as that class) will not be applicable. Then we will assign different weights to these K-Nearest Neighbors. The way to assign weights must satisfy the condition that the closer a point is to the test data point, the higher the weight (more trust). The simplest way is to take the inverse of this distance. In case the test data coincides with a data point in the training data, i.e. the distance is 0, we always take the label of the training data point.

$$w_i = \exp\left(\frac{-\|x - x_i\|^2}{2\sigma^2}\right) \quad (17)$$

where x is test data, x_i is a neighbor in K-Nearest Neighbors of x , w_i is the weight of that point (corresponding to the data point under consideration), σ is a positive integer.

Eq. (17) is another popular weighting method in Machine Learning [28]. We see that this function also satisfies the condition: the closer the point is to x , the

higher the weight (the highest is 1). When σ is small, the weights decrease very quickly, meaning that only very close neighbors have significant influence. This makes the model very “local” and can be sensitive to noise or outliers. When σ is large, the weights decrease slowly, meaning that more distant neighbors still have significant influence. This makes the model “smoother” but can blur the boundaries between classes if σ is too large. For choosing σ in weighted KNN, the most robust and practically cited method is cross-validation (e.g., using Grid Search or Random Search) over a predefined range of σ values, often combined with the tuning of K (the number of neighbors) [29]. In the experiments, the authors optimize σ on the validation set through grid search and select the σ value ($\sigma = 10$) that gives the highest classification accuracy.

In Fig. 30, there are 2 objects in the “-” class, equal to the “+” class, which has 2 objects. According to Eq. (16), the result is the bullet mark in the area of score 8.

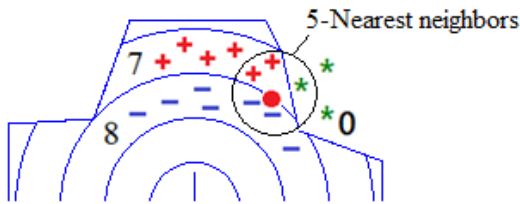


Fig. 30. A special case when 2 classes have the same number of objects.

Evaluation of the results of bullet holes on targets during the test at the shooting range is given in Figs. 31–33.

Bullet hole coordinate	Score	The score levels				
		0	1	2	3	4
		Percentage (%)				
(46,29)	0	100	0	0	0	0
(212,317)	5	0	0	0	0	0
(104,223)	9	0	0	0	0	0
(195,53)	0	100	0	0	0	0
(66,303)	6	0	0	0	0	0
(127,441)	3	0	0	0	100	0
(188,382)	4	0	0	0	0	100

Fig. 31. Evaluation of the results of bullet holes on targets (0–4).

Bullet hole coordinate	Score	The score levels					
		5	6	7	8	9	10
		Percentage (%)					
(46,29)	0	0	0	0	0	0	0
(212,317)	5	100	0	0	0	0	0
(104,223)	9	0	0	0	44	56	0
(195,53)	0	0	0	0	0	0	0
(66,303)	6	0	100	0	0	0	0
(127,441)	3	0	0	0	0	0	0
(188,382)	4	0	0	0	0	0	0

Fig. 32. Evaluation of the results of bullet holes on targets (5–10).

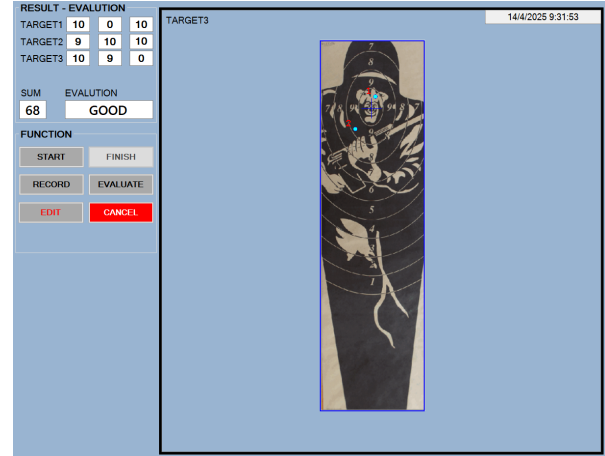


Fig. 33. Bullet holes identified on target image.

VI. EXPERIMENT WITH SOME OTHER CLASSIFICATION ALGORITHMS

To evaluate objectively, the authors conducted experiments with several classification algorithms such as Decision Trees and Random Forest. For testing, the author used Decision Trees model, Random Forest model in Accord. Machine Learning package. For the Decision Trees algorithm, the author choose C4.5 learning algorithm with the parameters set to default. For the Random Forest algorithm, the author set number of trees to 10 and the remaining parameters are set to default values. The evaluation results are shown in the comparison table below.

From Tables V–VII, we can see that the larger the amount of data, the longer the processing time of the Decision Trees and Random Forest algorithm. Although the accuracy is similar, the processing time of the Decision tree and Random Forest algorithm is longer than that of the KNN algorithm for the same target image. For some more advanced classification algorithms the processing time is even longer. This is not suitable for the fast-processing requirements of the scoring problem. Zhu *et al.* [10] in his research has shown that for the recall, the deep learning method has shown a good performance, comparing the SVM. But for the precision, performance of the shallow machine learning method is much better than it that Faster Recurrent-Convolutional Neural Network (R-CNN) shows. It means that features of bullet holes Faster R-CNN learned is not robust enough. The reasons for this result may be that bullet holes are too small to study the inherent features, and the size of the training sample is inappropriate. To the samples of small resolutions, convolutional neural network cannot get enough negative samples from whole frame images, while negative samples for SVM is different. For Faster R-CNN, to the small target such as bullet holes in this experiment, target image block can be shrunk to a point after several convolution operations. So, the last layer of convolution layer has lost some of the features of bullet holes. This shows the limitations of deep learning in this bullet hole scoring problem. Table VIII gives the accuracy of the representative studies.

TABLE V. TABLE OF RELATIONSHIP BETWEEN CLASSIFICATION ALGORITHMS AND PROCESSING TIME AND ACCURACY (TARGET 1)

Algorithms	Processing time (s)	Accuracy
Decision Trees	9.2	0.998
Random Forest	8.7	0.993

Note: Test with an image of Target 1 of size 242×242 pixels.

TABLE VI. TABLE OF RELATIONSHIP BETWEEN CLASSIFICATION ALGORITHMS AND PROCESSING TIME AND ACCURACY (TARGET 2)

Algorithms	Processing time (s)	Accuracy
Decision Trees	70.7	0.982
Random Forest	57.5	0.979

Note: Test with an image of Target 2 of size 259×536 pixels.

TABLE VII. TABLE OF RELATIONSHIP BETWEEN CLASSIFICATION ALGORITHMS AND PROCESSING TIME AND ACCURACY (TARGET 3)

Algorithms	Processing time (s)	Accuracy
Decision Trees	75.2	0.985
Random Forest	61.7	0.983

Note: Test with an image of Target 3 of size 259×789 pixels.

TABLE VIII. REPRESENTATIVE STUDIES AND ACCURACY

Experiment	Method	Accuracy
Dena Hendriana <i>et al.</i> [7]	Python script and OpenCV with HoughCircles	0.97
Faizan Ali <i>et al.</i> [9]	Computer vision and image process	0.983
Zhu Ruolin <i>et al.</i> [10]	HOG+SVM and Faster R-CNN	0.85–0.98

VII. CONCLUSION

The method of reporting target scores using image processing methods and machine learning algorithms to support shooting training has achieved quite positive results. After many self-tests, the bullet marks on the target were identified and scored accurately. K-Nearest Neighbors Algorithm applied to the problem has the advantage of fast processing speed and high accuracy when choosing appropriate parameters and configuration. However, it also has the disadvantage that when the display size of the target is large, the number of pixels is large, the large K value will make the processing speed slow down significantly. The solution is to position the target and camera appropriately. Then from the target image size on the screen, select the appropriate K parameter. During the testing phase, the author team tested the target scoring software in an indoor shooting range and in a shooting pit. When calibrated and improved in accuracy, this method will be very useful to support shooting training, increasing safety and reducing manpower.

From this problem, the authors' next research direction is to test the problem with other machine learning algorithms or combine additional image processing methods to improve accuracy and reduce processing speed. The products and results of the research are presented in Fig. 34.

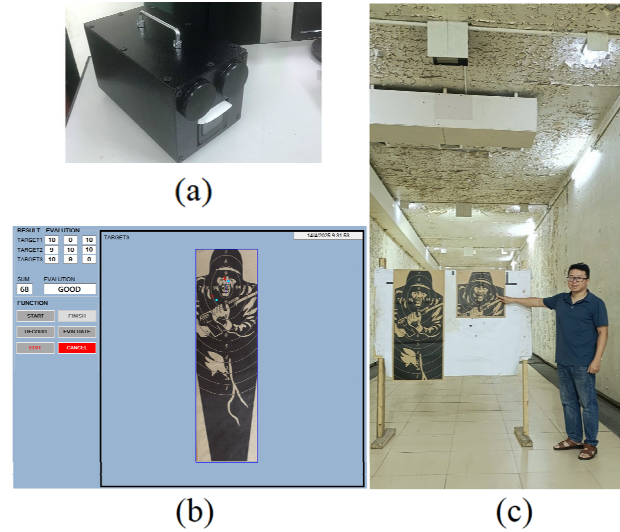


Fig. 34. (a) Camera box. (b) Testing software. (c) Test in a shooting pit.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Long. Quang Trinh conducted the research; Tuan. Nguyen Duc analyzed the data; Minh. Nguyen Hong wrote the paper; all authors had approved the final version.

ACKNOWLEDGMENT

Grateful acknowledgement is made to all the researchers mentioned in the references. And we would like to thank the anonymous reviewers and editors for the constructive and insightful comments for improving the quality of this paper.

The authors wish to thank Institute of Control Engineering and Le Quy Don Technical University for providing funding and creating favorable conditions for the authors to conduct experiments and complete the paper. The paper is the result of the Academy-level project (code: 24.1.71) of Le Quy Don Technical University.

REFERENCES

- [1] A. Shulgin, O. Tymoschuk, D. Khaustov *et al.*, "Analysis of the ballistic sound wave of the bullet to determine the point of hitting the target," *Military Technical Collection*, no. 27, pp. 45–52, Nov. 2022.
- [2] A. Hujatulatif, J. Jumadi, H. Kuswanto *et al.*, "Analyzing and comparing frequency of the birds sound spectrum using Audacity software in practicum activity," *Journal Penelitian Pendidikan IPA*, vol. 8, no. 6, pp. 2586–2592, Dec. 2022.
- [3] A. Azalia, D. Ramadhanti, H. Hestiana *et al.*, "Audacity software analysis in analyzing the frequency and character of the sound spectrum," *Journal Penelitian Pendidikan IPA*, vol. 8, no. 1, pp. 177–182, Jan. 2022.
- [4] D. S. Bilvesh, "Image processing techniques: A review," *International Journal of Innovative Science and Research Technology*, vol. 8, no. 2, Feb. 2017.

- [5] R. M. Parthe, "The importance of centroid in image processing," *International Journal of Scientific Research in Engineering and Management (IJSREM)*, vol. 8, no. 4, Apr. 2024.
- [6] I. Iqbal, K. Walayat, M. U. Kakar *et al.*, "Automated identification of human gastrointestinal tract abnormalities based on deep convolutional neural network with endoscopic images," *Intelligent Systems with Applications*, vol. 16, Nov. 2022.
- [7] D. Hendriana, Y. Umniyati, E. F. Soonggalon *et al.*, "Advanced shooting target with bullet collector, semi-automatic bulls-eye paper positioning and automatic shooting score," *SINERGI*, vol. 29, no. 1, pp. 51–58, Feb. 2025.
- [8] Q. Song, W. J. Hu, and W. F. Xie, "Robust support vector machine with bullet hole image classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 4, pp. 440–448, Nov. 2002.
- [9] X. Vasques, *Machine Learning Theory and Applications: Hands-on Use Cases with Python on Classical and Quantum Machines*, John Wiley & Sons, USA, 2024.
- [10] F. Ali and A. B. Mansoor, "Computer vision based automatic scoring of shooting targets," in *Proc. 2008 IEEE International Multitopic Conference*, 2008, pp. 515–519.
- [11] R. L. Zhu, J. B. Liu, Y. Zhang *et al.*, "Recognition of bullet holes based on video image analysis," *IOP Conference Series: Materials Science and Engineering*, vol. 261, 2017.
- [12] I. Iqbal, G. A. Odesanmi, J. Wang *et al.*, "Comparative investigation of learning algorithms for image classification with small dataset," *Applied Artificial Intelligence*, vol. 35, no. 10, pp. 697–716, 2021.
- [13] S. Ben-David and S. Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge, ENG.: Cambridge University Press, 2014, pp. 258–267.
- [14] K. Riyazuddin, S. Bajidvali, B. A. Raheem *et al.*, "An enhanced woelfel image noise filter," *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough, Studies in Computational Intelligence*, vol. 1117, pp. 71–80, 2024.
- [15] T. Srinivasulu and J. J. Sheela, "Enhancing the quality of fog/mist images by comparing the effectiveness of kalman filter and adaptive filter for noise reduction," *SPAST Reports*, vol. 1, no. 3, Mar. 2024.
- [16] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. New Jersey, U.S.: Prentice Hall, 2001, pp. 111–120.
- [17] C. Lin, C. Qiu, C. Wu *et al.*, "Adaptive noise detector and partition filter for image restoration," *Computers, Materials & Continua*, vol. 75, no. 2, pp. 4317–4340, Mar. 2023.
- [18] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Prentice Hall, 2001, pp. 110–111.
- [19] J. R. Parker, *Algorithms for Image Processing and Computer Vision*, 2nd ed. New Jersey, U.S.: Wiley, 2010, pp. 300–302.
- [20] R. Gohane, S. Adatrao, and M. Mittal, "Comparison of various centroiding algorithms to determine the centroids of circular marks in images," *IET Image Processing*, vol. 19, Jan. 2025.
- [21] R. Szeliski, *Computer Vision Algorithms and Applications*, 2nd ed. New York, U.S.: Springer, 2022, pp. 141–465.
- [22] P. Cai, Z. Cai, Y. Fan *et al.*, "Image contour detection based on visual pathway information transfer mechanism," *Neural Process Letters*, vol. 56, no. 6, Feb. 2024.
- [23] F. Li, C. Lin, Q. Zhang, and R. Wang, "A biologically inspired contour detection model based on multiple visual channels and multi-hierarchical visual information," *IEEE Access*, vol. 8, pp. 15410–15422, Jan. 2020.
- [24] L. Yan, X. Zhang, K. Wang *et al.*, "Image segmentation refinement based on region expansion and minor contour adjustments," *IET Image Processing*, vol. 19, no. 1, p. e70017, Jan. 2025.
- [25] N. Mamatov, M. Jalelova, B. Samijonov *et al.*, "Algorithms for contour detection in agricultural images," *E3S Web of Conferences*, vol. 486, p. 03017, Feb. 2024.
- [26] P. Barceló, A. Kozachinskiy, M. Romero *et al.*, "Explaining k-nearest neighbors: Abductive and counterfactual explanations," *Proceedings of the ACM on Management of Data*, vol. 3, no. 2, pp. 1–26, Jan. 2025.
- [27] S. Ben-David and S. Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge, ENG.: Cambridge University Press, 2014, pp. 258–267.
- [28] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, U.S.: Springer, 2009, pp. 34–200.
- [29] B. K. Khotimah, A. Y. R. Wulandari, A. D. Cahyani *et al.*, "A novel weight k-nearest support vector machine for livestock imbalance data classification," *Mathematical Modelling of Engineering Problems*, vol. 12, no. 2, pp. 425–433, Feb. 2025.

Copyright © 2025 by the authors. This is an open access article distributed under the Creative Commons Attribution License ([CC-BY-4.0](https://creativecommons.org/licenses/by/4.0/)), which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.