Received: November 1, 2022. Revised: December 3, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Discriminant Feature Trace Transform for Predictive Object Rotation

Nattapong Jundang[1]* **Suchada Sitjongsataporn[2]**

[1]*The Electrical Engineering Graduate Program, Faculty of Engineering and Technology, Mahanakorn University of Technology*
[2]*Department of Electronic Engineering, Mahanakorn Institute of Innovation (MII), Faculty of Engineering and Technology, Mahanakorn University of Technology*
* Corresponding author's Email: jnattapo@mut.ac.th

Abstract: This paper presents how to modify the proposed Discriminant Feature Trace Transform (DFTF) algorithm for object rotation. This proposed DFTF algorithm based on the Trace Transform (TF) domain can collect data for the formation of distinctive features and is suitable for predicting the rotational direction of an object with the supervised machine learning. TF domain is used after removing the background image that transformation properties can be generated a feature referred to the rotation features of objects inside. After that, the proposed DFTF algorithm transforms the two-dimensional data of each image from the TF domain into the one-dimensional data that can indicate the direction of object rotation within the image. The results of the DFTF algorithm, which is a 1D data vector from each image, are then generated into the labeled datasets for machine learning algorithms including with the Naïve Bayes (NB) for predicting the rotation direction and Random Forest (RF) to reinforce the predicted values from the NB in the form of quadrant in which the interested object is rotated. The simulation experiments are conducted with two types of databases. The first experiment combines with the three different datasets which the results are very effective and can provide the average accuracy rate for all databases up to 99.8%. The second experiment is derived from the visualization of the water bottle production line. It measures the accuracy of the bottled water transfer to the second station. The proposed approach is an accuracy rate up to 92.2%.

Keywords: Trace transform, Machine learning, Computer vision, Machine vision.

1. Introduction

Nowadays, image processing is currently utilized in a vast array of applications [1]. The fundamentals of image processing, such as techniques for image smoothing, noise reduction, and object detection. As fundamental image processing grew in popularity, it evolved into the functional paradigm known as computer vision. Computer vision has essential objectives, such as attempting to process camera data, which method is analogous to a simulation of the human visual system. And there is the second method that considers images by reconstructing the output format, such as translating the image into the trace transform (TF) domain based on the Radon transform [2]. Another well-known approach is the TF algorithm, which encodes picture data by taking the intensity values from images. The parameters that govern the direction of rotation and the separation from the point of interest are computed to acquire this information. New image domain can be used to create the context categorization [3]. Another feature of this TF algorithm is its rotation invariance [4]. TF algorithm can still be included into other algorithms that use artificial intelligence even with all the advantages listed above.

Artificial intelligence (AI) is currently the most widely used and approved algorithm [5]. AI is able to create knowledge that can be utilized to find and predict the best answer. A noteworthy part of a major AI project is machine learning (ML), which can predict continuous and discrete data in many ways. Supervised learning [6] can predict the answer in the classification format. The followings are used as the Logistic Regression (LR), Naïve Bayes (NB), k-Nearest Neighbors (k-NN), Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM) [7, 8]. These methods listed above have a wide range of applications, especially in the field of robotics engineering. This was initially necessitated by the need to boost the robots' visibility with image data in order to enable robotic arms to do work effectively for individuals. The nature of the interaction between camera images, ML and robots are often mentioned in machine vision research [9]. An example of how the camera and robot work together, such as locating objects in a monocular vision environment using ML algorithms [10], considering visual tools to find defects on leather in automated machine vision systems [11]. From the following, it can be concluded that the quantity of data is crucial for working with supervised ML, whose working model utilizes historical data. The produced weights are only useful for analyzing historical data. When input data arrives, the system must reintroduce the new data into the supervised ML process in order to develop new knowledge.

As previously stated, Trace Transform (TF) is a well-known method for determining the rotational orientation of an object. This is a strong point that makes it beneficial for this paper. But the limitation of TF is that the complexity of both the computational algorithm and the internal functions makes it difficult to implement in real life. This paper will focus on how supervised ML classifies the discrete predictions.

Therefore, the main objective of this paper is to propose a DFTF (Discriminant Feature Trace Transform) algorithm with the help of TF technique in order to collect data for the formation of distinctive features by converting the two-dimensional image data into a modeling process with the supervised ML. The benefit of this new feature is that the aggregation of data from the TF domain is presented in a one-dimension format, which helps the prediction of ML while preserving the object's rotation measurement features. The mentioned methods will be explained in the following section.

2. Related works

The primary algorithms at the heart of the experiment are TF and classification ML algorithms, where TF tries to generate data features and perform ML on a targeted basis. The objective is to predict the response using DFTF data. Before entering the main section of the text, this part will explain how to implement the aforementioned method in several ways.

2.1 Trace transform and implementation

Due to TF algorithm's many advantages of an image rotational tolerance is one of them. In this section, we show some researches that have utilized the TF algorithms. An image rotational tolerance is one of many advantages from TF algorithm shown in many researches. For example, the analyzed inverse synthetic aperture radar (ISAR) image using TF algorithm are very prone to spatial variation [12]. Feature extraction with circular trace transform (CTT) for image texture analysis [13] and analysis of the recorded tracks of high-velocity emission with TF [14] are presented. The aforementioned research demonstrates that the trace transform technique can be utilized in a variety of ways dependent on the user-generated data selection.

2.2 Machine learning: classification

Due to the prediction of responses in the classification ML, it is very accurate when we can determine the exact prediction answer. Therefore, these researches will demonstrate a wide range of ML implementation methods. For example, there are Naïve Bayes to perform real time classification accelerator on FPGA [15], polarimetric synthetic aperture radar (POLSAR) image classification by developing Decision Tree algorithm [16], Predicting solar energy from satellite images with Support Vector Machine [17], and classification hyperspectral image based on Random Forest algorithm [18]. All of the aforementioned classifications are applied in a large diversity of businesses based on algorithms that predict with high accuracy for specific activities. Consequently, these strategies for predicting the response are utilized to determine the rotation of an item.

3. Proposed system

In this section, an overview of the proposed system is comprising with the three main functions as shown in Fig. 1. Preliminary is the preparation of data for modeling, the training process adoption of machine learning algorithms is used to learn data through the previous image manipulation steps, and then to implement the model obtained from the second step to predict the rotation direction of the object from the real data. The principle of operation relies on the machine learning algorithms to model each part, which consists of two algorithms as the Naive Bayes for result of orientation prediction

Received: November 1, 2022. Revised: December 3, 2022.

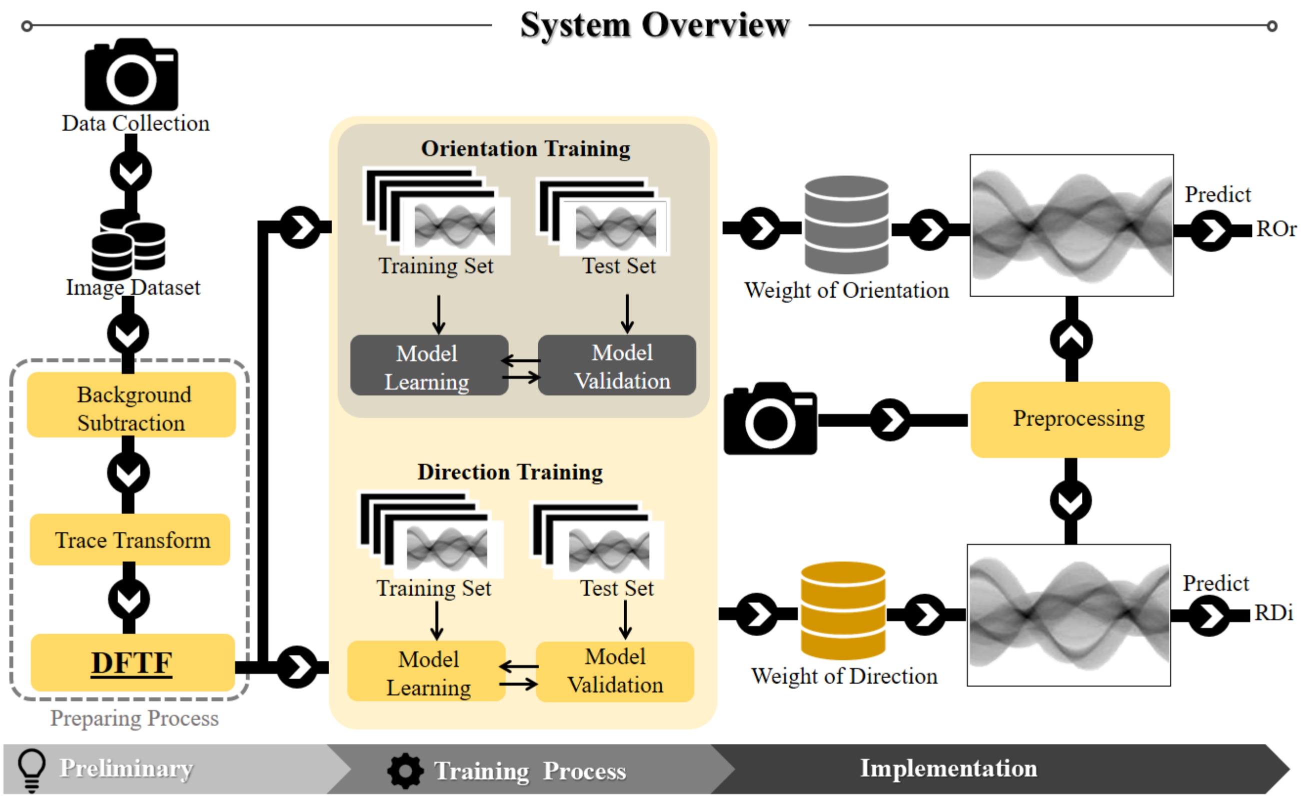


Figure. 1 Overview of the proposed system

(ROr) and random forest calculation for result of direction prediction (RDi). The highlight of this paper is to transform the data into the trace transform domain first to create new data characteristics. It is hoped that it can be used to find the direction of rotation of an object without the need for complex calculations. The above can be explained in the next section.

3.1 Background subtraction

The background estimation to extract only object information from the image is based on the finding the foreground mask position as shown in Fig. 2. By starting with the use of reference images and definition, this reference image is intended to be an image that does not contain objects in the image. The image is then extracted with the reference image, and the image data is filtered again after the threshold is removed, resulting in a foreground mask. Finding the foreground mask is shown as

$$J(x,y)=I(x,y)-B(x,y)\ ;\forall x\forall y \qquad (1)$$

where I(x,y) is the original image, B(x,y) is the reference image, and J(x,y) is the result of deleting the image and to do until every pixel in the x- and y- axes.

The result of this image deletion is compared with a threshold value to select the data to be assigned as the foreground mask, which can be calculated as

$$Fg(x,y)=\begin{cases}0, & J(x,y)<\text{threshold}\\ 1, & \text{Otherwise}\end{cases}\ ;\forall x\forall y \qquad (2)$$

where Fg(x,y) is the result of filtering the image data for every pixel position of the x- and y- axes.

And the last step is that after obtaining the foreground mask, the object must be extracted from the original image again in order to obtain specific information, this can be done as

$$H(x,y)=\begin{cases}I(x,y), & Fg(x,y)I(x,y)\geq 0\\ 0, & \text{Otherwise}\end{cases}\ ;\forall x\forall y \qquad (3)$$

where we assume every pixel position of the x and y axes to H(x,y) as a response from foreground mask filtering, which is calculated by logically comparing the original image.

Therefore, an overview of this work for cutting the background of the image can be described as Algorithm 1.

3.2 Trace transform

In the previous section, we present how to extract the objects from the background, which can take the resulting image data to transform data in the trace transform (TF) domain. Transformation using

Received: November 1, 2022. Revised: December 3, 2022.

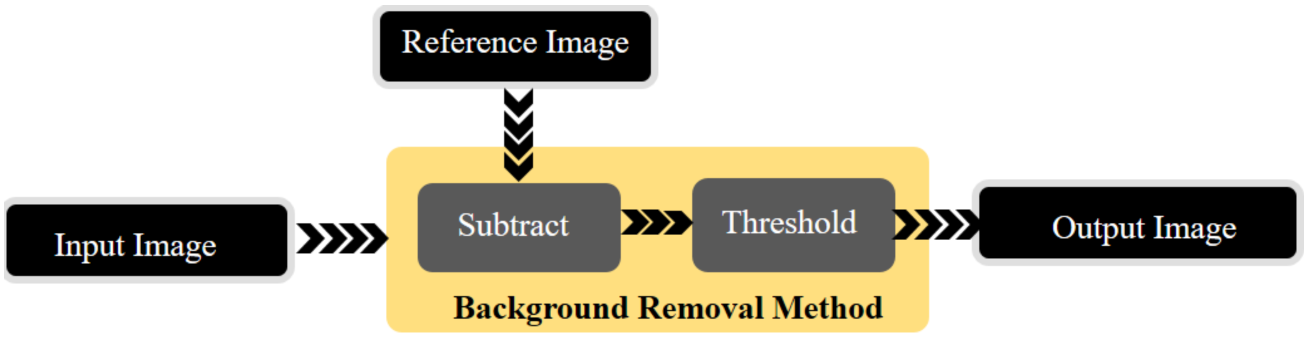


Figure. 2 Structure for background removal method

Algorithm 1 background estimation with image to select an object

Input: *original image (I), reference image (B)*
Output: *result image (H)*
Initial of variables: *assign image width and height to variable* ***w*** *and* ***h****; assign threshold to variable* ***t***

```
1   for $y = 1, 2, \ldots, h$ do
2       for $x = 1, 2, \ldots, w$ do
3           $J(x,y)=I(x,y)-B(x,y)$
4           if $J(x,y)<t$ then
5               $Fg(x,y)=0$
6           else then
7               $Fg(x,y)=1$
8           end if
9           if $(Fg(x,y)\&\&I(x,y))\geq 0$ then
10              $H(x,y)=I(x,y)$
11          else then
12              $H(x,y)=0$
13          end if
14      end for
15  end for
```

Algorithm 2 Trace transform

Input: *input image (H)*
Output: *result image (g)*
Initial of variables: *create zero array to variable* ***g*** *with width is 360 and y is image's height; : assign image width to variable* ***w***

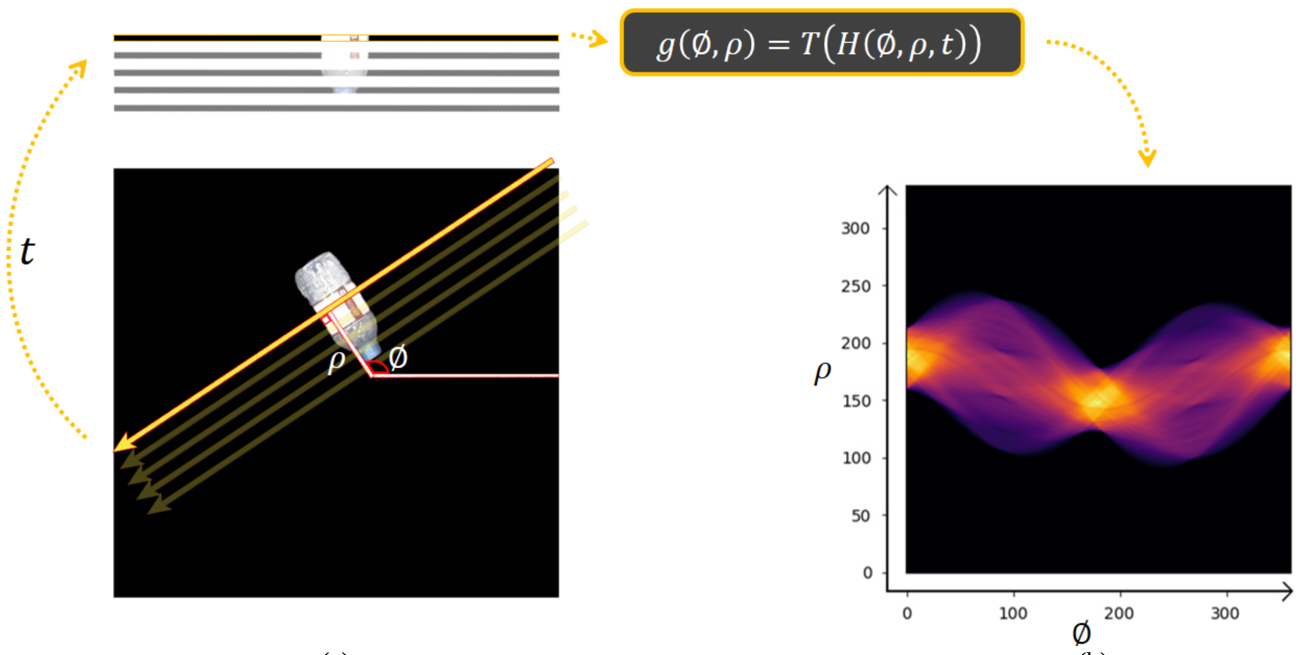
```
1   for $\phi = 0, 1, 2, \ldots, 359$ do
2       for $p = 0, 1, 2, \ldots, w-1$ do
3           $t$ = fetch tracing line at $\rho$ and $\emptyset$
4           if sum($t$) is not 0 then
5               $g(\emptyset,\rho)=T(H(\emptyset,\rho,t))$
6           else then
7               continue
8           end if
9       end for
10  end for
```

TF algorithm has the following steps as shown in Fig. 3. The main content of this TF algorithm is based on the intensity value of the image to be considered in order to create new data. This information is referred to as the tracing line (t). The t data obtained will be characterized as a long data line depending on the width and height of the image. Therefore, the size of the data affects the amount of data in t. The acquisition of t data is based on parameters $\emptyset$ and ρ that control the values as shown in Fig. 3 (a) where $\emptyset$ controls the direction of rotation from the centre axis of the reference point to retrieve the data. For each operation, a degree $\emptyset$ is adjusted, which is rotated from 0 to 360 degrees. Another parameter, ρ is the distance (pixel) measured from the reference point to perpendicular to the data line t, where the size of ρ is more or less depending on the size of the image. Each acquisition of the data string t is taken through the equation shown in Fig. 3 (b), which is a two-dimensional nature of the new data comprising the y-axis ranging from 0 to ρ and the x-axis range from 0 to 360 degrees.

Finally, the data is collected at the different locations referring to the variables ρ and $\emptyset$ as two-dimensional data that the behavior is shown in Fig. 3. The combined equation for the trace transform can be represented as

$$g(\emptyset,\rho)=T(H(\emptyset,\rho,t)) \tag{4}$$

where $H(\emptyset,\rho,t)$ means the values of the image function along the chosen line. Taking this function, we can eliminate variable t. The result is a two-dimensional function of the variables $\emptyset$ and ρ can be interpreted as another image defined on $g(\emptyset,\rho)$ which was calculated with the T function as shown in Table 1. All equations in this table have parameters set and the different computational models result in different results from each function. Therefore, the precise selection of functions is unknown in this paper, TF-domain results are



(a) (b)

Figure. 3 Trace transform algorithm: (a) Tracing line data retrieval and (b) Calculation result from trace function

Table 1. Trace transform functions [19]

No.	Trace Functionals	Details
1	$T(f(t)) = \int_0^\infty f(t)dt$	*The line integral transformation*
2	$T(f(t)) = \left[\int_0^\infty \lvert f(t)\rvert^p dt\right]^q$	$p = 0.5, q = 1/p$
3	$T(f(t)) = \left[\int_0^\infty \lvert f(t)\rvert^p dt\right]^q$	$p = 4, q = 1/p$
4	$T(f(t)) = \int_0^\infty \lvert f(t)\rvert' dt$	*The one-dimensional numerical gradient of* t $f(t)' = [(t_2 - t_1), (t_3 - t_2), \ldots, (t_n - t_{n-1})]$
5	$T(f(t)) = median_t\{f(t), \lvert f(t)\rvert\}$	-
6	$T(f(t)) = median_t\{f(t), \lvert f(t)\rvert'\}$	-
7	$T(f(t)) = \left[\int_0^x \lvert F\{f(t)\}\rvert^p\, dt\right]^q$	F *denote the discrete Fourier transform computed with the FFT algorithm,* $p = 0.5, q = 1/p, x = n/2$
8	$T(f(t)) = \int_0^\infty \left\lvert \frac{d}{dt} M\{f(t)\}\right\rvert\, dt$	M *is a median filter operator, using a local window of length 3, and differentiation means taking the difference of successive samples.*

chosen from all functions to summarize the best functions among these eight functions. The workflow of TF algorithm can be shown as Algorithm 2.

The results of the TF domain for the image sample data are shown in Fig. 4, we found that in the case of object rotation within the original image in the upper of Fig. 4(a) to 4(d) would present in an outcome in such a way. The reference point of rotation direction can be found in the lower of Fig. 4(a) to 4(d). The advantage of the results from TF domains are used to determine machine learning models to optimize the ability and to predict the rotational direction of objects within the image, which will be described in the next section.

3.3 Discriminant feature trace transform

In this section, we propose the Discriminant Feature Trace Transform (DFTF) that is data transformation from 2-D to 1-D array. This idea is to convert the two-dimensional image data into a modeling process by extracting data in each row of an image that consists of every column in that row. The total length of data is concatenated to count as a group of data, known as a feature for modeling. The size of all features can be calculated as the width of

Received: November 1, 2022. Revised: December 3, 2022.

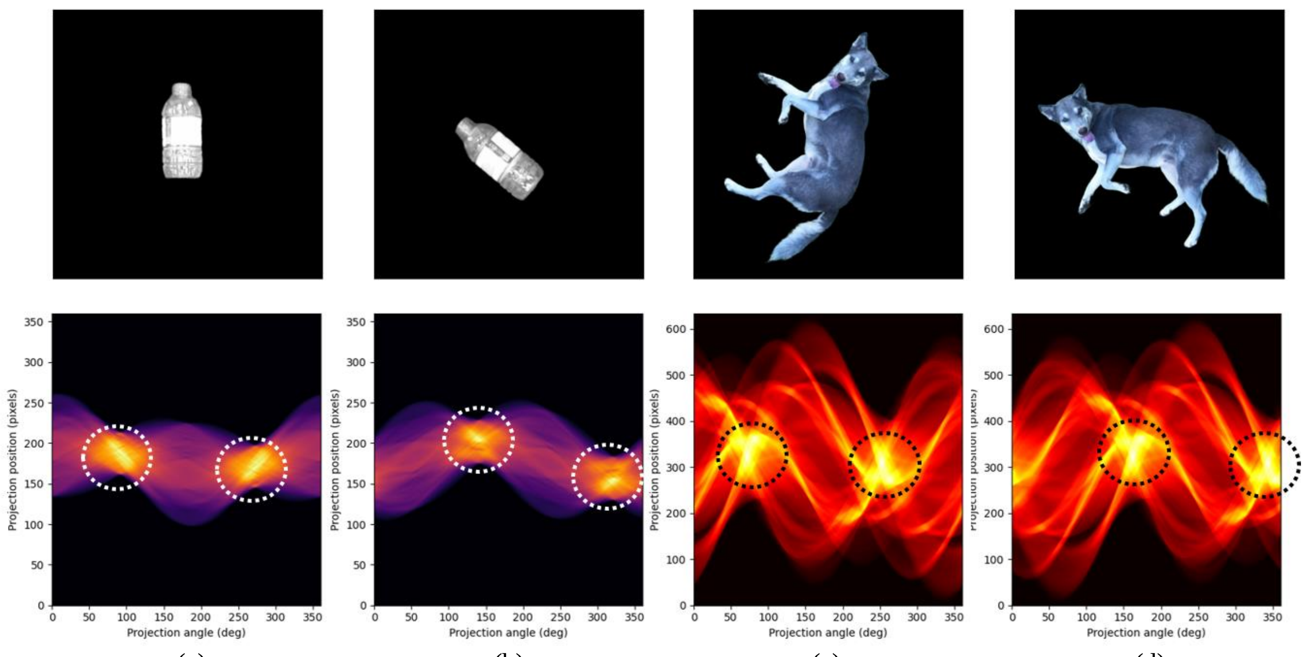


Figure. 4 Original image and Trace transform's result: (a) Bottle at 90 degree, (b) Bottle at 130 degree, (c) Dog at 70 degree, and (d) Dog at 150 degree

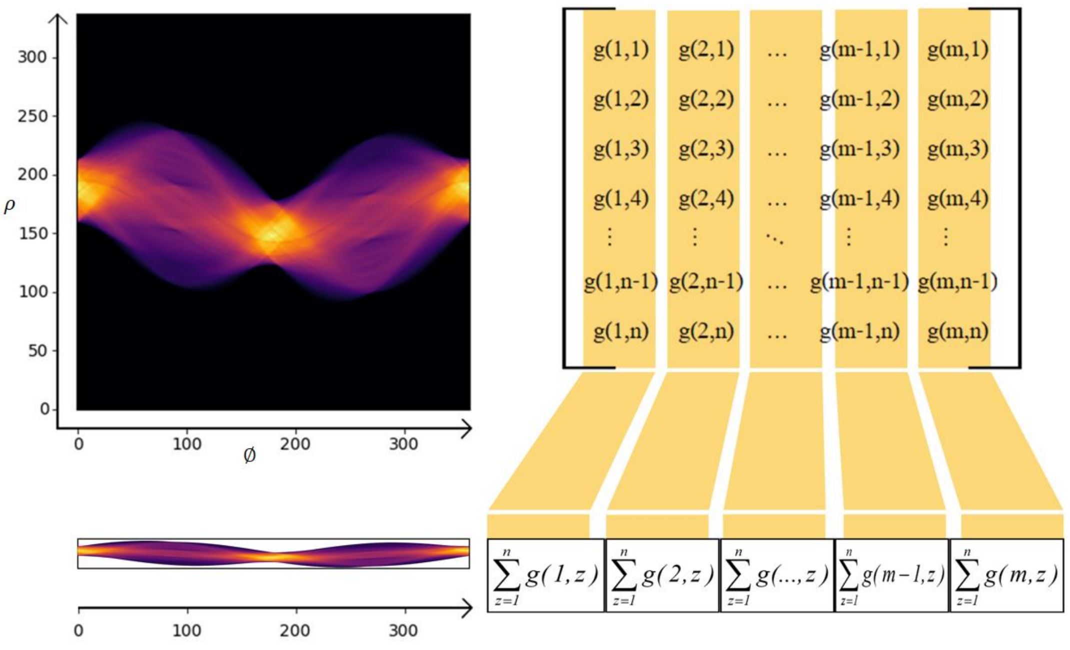

ρ
Ø
g(1,1) g(2,1) … g(m-1,1) g(m,1)
g(1,2) g(2,2) … g(m-1,2) g(m,2)
g(1,3) g(2,3) … g(m-1,3) g(m,3)
g(1,4) g(2,4) … g(m-1,4) g(m,4)
g(1,n-1) g(2,n-1) … g(m-1,n-1) g(m,n-1)
g(1,n) g(2,n) … g(m-1,n) g(m,n)
$\sum_{z=1}^{n} g(1,z)$ $\sum_{z=1}^{n} g(2,z)$ $\sum_{z=1}^{n} g(\ldots,z)$ $\sum_{z=1}^{n} g(m-1,z)$ $\sum_{z=1}^{n} g(m,z)$

Figure. 5 Transforming image data from trace transform model to 1-D array

the image multiplied by the length of the image. As observing the information, it is found that the larger image will have the more feature number. Therefore, it is not a good idea to directly model the entire image as it would lead to excessive calculations that need to be taken into account. It also results in computational delays due to the increased number of features. Therefore, the converting images is different from the usual method. This is because, if observed from the results of the trace transform domain, it is in the perspective that the axis intersects the distance from the center with the distance varying with the image size. And the larger the value in the y-axis and the direction of rotation of the tracing line ranges from 0 to 360 degrees. For the x-axis, it is observed that no matter how large the image is, the size of the x-axis will remain the same because the output in TF domain continues to rotate in the same direction as

$$x_i = \sum_{z=1}^{n} g(i,z) \tag{5}$$

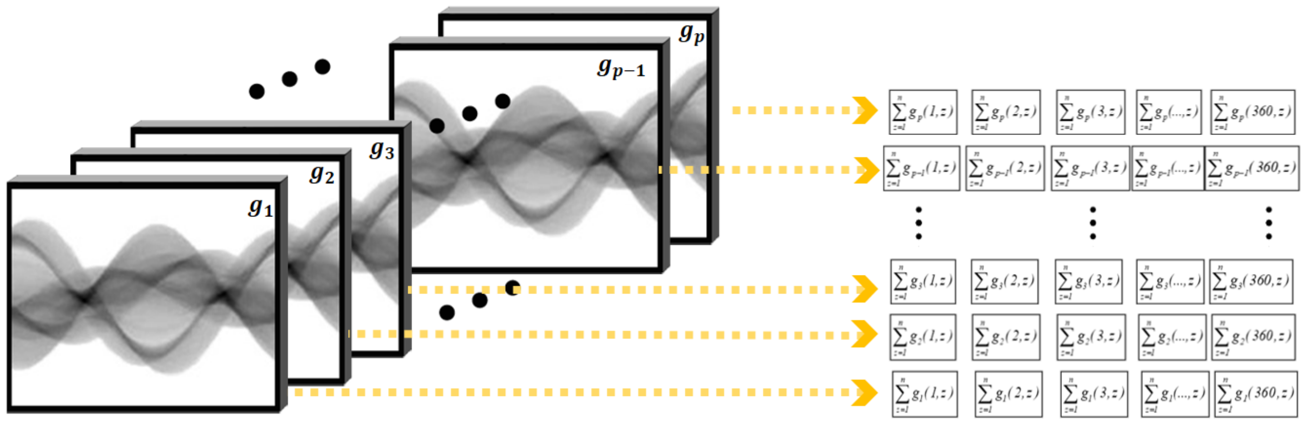

g_1
g_2
g_3
g_{p-1}
g_p
$\sum_{z=1}^{n} g_p(1,z)$ $\sum_{z=1}^{n} g_p(2,z)$ $\sum_{z=1}^{n} g_p(3,z)$ $\sum_{z=1}^{n} g_p(\dots,z)$ $\sum_{z=1}^{n} g_p(360,z)$
$\sum_{z=1}^{n} g_{p-1}(1,z)$ $\sum_{z=1}^{n} g_{p-1}(2,z)$ $\sum_{z=1}^{n} g_{p-1}(3,z)$ $\sum_{z=1}^{n} g_{p-1}(\dots,z)$ $\sum_{z=1}^{n} g_{p-1}(360,z)$
$\sum_{z=1}^{n} g_3(1,z)$ $\sum_{z=1}^{n} g_3(2,z)$ $\sum_{z=1}^{n} g_3(3,z)$ $\sum_{z=1}^{n} g_3(\dots,z)$ $\sum_{z=1}^{n} g_3(360,z)$
$\sum_{z=1}^{n} g_2(1,z)$ $\sum_{z=1}^{n} g_2(2,z)$ $\sum_{z=1}^{n} g_2(3,z)$ $\sum_{z=1}^{n} g_2(\dots,z)$ $\sum_{z=1}^{n} g_2(360,z)$
$\sum_{z=1}^{n} g_1(1,z)$ $\sum_{z=1}^{n} g_1(2,z)$ $\sum_{z=1}^{n} g_1(3,z)$ $\sum_{z=1}^{n} g_1(\dots,z)$ $\sum_{z=1}^{n} g_1(360,z)$

Figure. 6 Transforming multiple images to data table

X_1	X_2	X_3	$\dots$	X_{360}	C
$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$\dots$	$x_{m,1}$	c_1
$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$\dots$	$x_{m,2}$	c_2
$\vdots$	$\vdots$	$\vdots$	$\dots$	$\vdots$	$\vdots$
$x_{n,k}$	$x_{2,k}$	$x_{3,k}$	$\dots$	$x_{m,k}$	c_k

Figure. 7 Data table's example

$$x_{i,j} = \sum_{z=1}^{n} g_j(i,z) \qquad (6)$$

Therefore, the way is to deal with the information in this domain is defined as shown in Fig. 5. The left-side of Fig. 5 present the image in the digital image in the TF domain, and the right-side of Fig. 5, show the image as a matrix data. Let n be a value from 1 to ρ and m is from 1 to ∅. For reducing the number of features to be modeled, a fixed number of m is used to convert the image from 2D to 1D by the method shown in Fig. 6. We found that the data in each digit will be summed up for creating a 1-dimensional feature. Therefore, for processing one image from the trace transform domain, the transform equation can be substituted as shown in Eqs. (5) and (6), where data$_i$ is the sum of data from row positions, z is equal to 1 until z is equal to m, and the same data is obtained for all positions i when i is integer value from 1 to m.

For creating a table of data from all images of the trace transform domain imported into the modeling, we can substitute Eq. (6), which adds a sequencer of the jth image data to determine where the data from $g_j(i,z)$ comes from which column and image by completing all data transformations. The characteristics of the data before modeling are shown in Fig. 7, which is all data that has been transformed to be ready for further modeling. Next, let X_i represent the feature data in each column which contains a total of 360 feature data. Each row of this data collection defines which row represents a classification answer, where each group is replaced by c. The proposed DFTF algorithm is shown in Algorithm 3.

Algorithm 3 Create Discriminant Feature Trace Transform

__Input:__ trace transform result (g)
__Output:__ DFTF result (DFTF)
__Initial of variables:__ assign width and height of __g__ to variable __w__ and __h__; create zero array to variable __DFTF__ with width is 360 and y is 1;

```
1  for x =0,1,2,…,w then
2      for y=0,1,2,…,h then
3          | DFTF[x][0] += g[x][y]
4      end for
5  emd for
6  DFTF = DFTF/max(DFTF)
7  return DFTF
```

3.4 Training process

In this section, images in the trace transform domain are used to model them with a machine learning algorithm based on the data obtained as a group of the feature and the target part defined by the user. Model system is built with the machine learning as presented in the next section.

3.4.1. Orientation training

Each class's answer assignment in Naïve Bayes demands that the answer be predicted as the direction of the object in degrees, and since the prediction specification requires that the answer prediction error is not exceeded 10 degrees. Thus, 36 classes of rotation direction can be divided into a total of 36 classes, which are Class 1 with rotation values from 0 to 9 degrees, Class 2 with rotation values from 10 to 19 degrees, Class 3 having The rotation value of the image ranges from 20 to 29

Received: November 1, 2022. Revised: December 3, 2022.

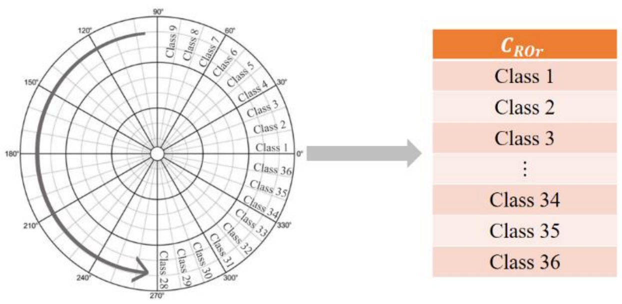


Figure. 8 Classification for modeling with NB

degrees and continues in the same way until the 36th class, which has the rotation value of the image from 350 to 359 degrees, which is shown in Fig. 8, where the nature of the answer will be in the range of 10 degrees respectively.

In this paper, we use Bayesian learning to model. Naïve Bayes [20] is a supervised learning algorithm in machine learning that can be used to model and predict the answer classification based on probability. Based on Bayes' theorem, the initial equation can be written as Eq. (7), where X is an attribute and C is a class.

$$P(C|X)=\frac{P(X|C)P(C)}{P(X)} \tag{7}$$

where P(C|X) is a posterior probability, which is the probability that the data has an attribute of X, it has a class of C. P(C|X) is a likelihood, the probability that the training data has class C and has attribute X, where $X=(x_1,x_2,x_3,\ldots,x_n)$ and n is the number of attributes in the training data P(C) is a prior probability, which is the probability of class C. Then, to consider that makes the naive assumption of independence among every pair of predictor variables given the value of the target class. Given class variable c_k when $k=\{1,2,3,\ldots,36\}$ and dependent feature vector $X=(x_1,x_2,x_3,\ldots,x_n)$ Bayes' theorem states the following relationship.

$$P(c_k|x_1,x_2,x_3,\ldots,x_n)=\frac{P(x_1,x_2,x_3,\ldots,x_n|c)P(c_k)}{P(x_1,x_2,x_3,\ldots,x_n)} \tag{8}$$

Assuming the naive conditional that

$$P(x_i|C)=P(x_i|c,x_1,x_2,x_3,\ldots,x_n) \tag{9}$$

For all i, this relationship is simplified to

$$P(c_k|x_1,x_2,x_3,\ldots,x_n)=\frac{P(c_k)\prod_{i=1}^{n}P(x_i|c_k)}{P(x_1,x_2,x_3,\ldots,x_n)} \tag{10}$$

Since $P(x_1,x_2,x_3,\ldots,x_n)$ is constant given the input, the following classification rule can be used as

$$P(C|x_1,x_2,x_3,\ldots,x_n)=P(c_k)\prod_{i=1}^{n}P(x_i|C) \tag{11}$$

Then substituting the equation and considering the solution of the algorithm, only the answer with the highest probability is chosen as the answer from all attributes. Therefore, it can be written as

$$V_{NB}=\text{argmax}\{P(c_k)\prod_{i=1}^{n}P(x_i|c_k)\} \tag{12}$$

where, x_i represents the i^{th} predictor variable. We can obtain estimates of the maximum a posteriori probability for P(c) and $P(x_i|c)$ from the training data. Learning algorithms that are based on Bayes' theorem differ mainly by the assumption, they make regarding the distribution of the likelihood $P(x_i|c)$. If having data sets with numerical features, a common assumption is that the likelihood of the features is assumed to be Gaussian function as

$$P(x_i|c_k)=\frac{1}{\sqrt{2\pi\sigma_c^2}}\exp\left(-\frac{(x_i-\mu_c)^2}{2\sigma_c^2}\right) \tag{13}$$

Maximum likelihood estimation is used to compute σ_c^2 and μ_c. Apart from its naive assumption of conditional independence, Naive Bayes classification has shown good performances in many complex real-world problems [21, 22].

3.4.2. Quadrant training

As this section of the modeling is used to refer to the direction of the water bottle to confirm the direction of the predict results from the Naïve Bayes model. Therefore, the class is divided into 4 classes as shown in Fig. 9.

By y assigns Class 0 instead of the 1st quadrant at the upper right position (0-89 degree), Class 1 presents in the 2nd quadrant at the upper left position (90-179 degree), Class 2 represents the 3rd quadrant at the bottom left (180-269 degree), and class 3 represents the 4th quadrant at the lower right position (270-359 degree), which can be substituted for the equation for defining classes as Eq. (14)

$$C_{RDi}=\{Q_0,Q_1,Q_2,Q_3\} \tag{14}$$

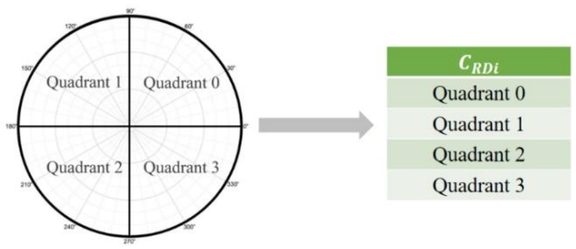


Figure. 9 Classification for modeling with RF

By predicting the object's quadrant values, the Random Forest (RF) [23] algorithm is used to solve the problem. The RF algorithm works based on the principle of creating a tree from the Decision Tree (DT) algorithm [24]. RF is to create multiple and distributed DT tree structures to find answers. Then every response received from every DT goes through a majority vote to find the most predicted answer. Unlike other supervised learning machine learning algorithms, the DT algorithm is a knowledge discovery model to find the underlining rule of the data used to model. Another highlight of doing DT is that it can find its feature importance. The DT algorithm considers a loss function that measures randomness with entropy. The default equation for constructing a tree is a measure of information gain, which can be calculated from

$$\text{Gain}(X)=\text{Entropy}(X)-\text{Entropy}_A(X) \quad (15)$$

where the Entropy is computed, it can be calculated from

$$\text{Entropy}(X)=-\sum_{i=1}^{\omega} p_i \log_2(p_i) \quad (16)$$

where p_i is the probability of class i having values from class 1 to class ω when ω calculates the total number of classes based on the number of classes in C_{RDi}, calculated from

$$p_i=\frac{|c_{i,x}|}{|X|} \quad (17)$$

where X represents the data attribute and $c_{i,x}$ represents the dataset within X belonging to class c_i, and $|X|$ and $|c_{i,x}|$ represent the number of data in groups X and $c_{i,x}$ respectively, the assignment of c_i has the configuration as

$$c_i \in C_{RDi} \quad (18)$$

where each class of Quadrant 0 through 3 represents the following data: Q_0 represents the rotation data range from 0-89 degrees, Q_1 represents the rotation data range from 90-179 degrees, Q_2 represents the rotation data range from 180-269 degrees, and Q_3 represent the rotation data range from 270-359 degrees respectively. For $\text{Entropy}_A(X)$, it can be found from Eq. (19), where $\frac{|X_i|}{X}$ represents the number of data in X with class A of a_j and divides by the total number of data in X.

$$Entropy_A(X)=\sum_{j=1}^{9}\frac{|X_i|}{X}\times Entropy(X_j) \quad (19)$$

The final step is to manage the random forest algorithm, i.e. a majority vote to take the results of all trees to determine which class the majority of the responses are as

$$\text{RDi}=\text{majority\_vote}(\varphi) \quad (20)$$

where φ represents the answer from the tree from every DT. A collaborative algorithm from Naïve Bayes and RF to make decisions for predicting the object rotation direction. As shown in Algorithm 4, we found that the value of the answer from NB was mainly used to predict the direction of rotation, and then RF was used to confirm. In other words, when found in a wrong quadrant, the system will adjust the predicted answer in the opposite direction by 180 degrees.

Algorithm 4 Orientation detection

```
Input: orientation weight (WOr), quadrant weight(WDi), image test (I)
Output: orientation class (ROr)
Initial of variables: initial ROr and RDi to zero

1   H = Algorithm1(I)
2   G = Algorithm2(H)
3   DFTF = Algorithm3(G)
4   ROr = Predict DFTF with WOr
5   RDi = Predict DFTF with WDi
6   ROr = ROr * 1/9
7   if (RDi == 3 and ROr == 1) or (RDi == 2 and ROr == 0) then
8       ROr +=18
9   elif (RDi == 0 and ROr == 2) or (RDi == 1 and ROr == 3) then
10      ROr -=18
11  end if
12  return ROr
```

4. Dataset and evaluation criteria

To measure the performance of the DFTF, we test the proposed DFTF algorithm with a commonly used image standard database and a specialized database. Therefore, the performance is measured both from the test data groups and individually in all classes for prediction.

4.1 Image dataset characteristics

Experiments were presented in the first experimental group. The experiments were

Received: November 1, 2022. Revised: December 3, 2022.

Figure. 10 Sample images in: (a) MNIST, (b) GTSRB, and (c) Caltech-256 datasets

performed on three data groups MNIST (Modified National Institute of Standards and Technology database) [25], GTSRB (German Traffic Sign Recognition Benchmark) [26], and Caltech-256 [27]. MNIST is a numerical handwriting data group divided into 10 groups from 0-9. In the experiments, we compare the rotational predictions of all numeric classes from the database. GTSRB is a collection of images of various traffic symbols in Germany, and Caltech-256 is a generic image database of various object classes. The GTSRB and Caltech-256 databases contain a large number of data groups as shown in Fig. 10

The second experimental group was tested with images from a small drinking water filling station. The resulting image is of a drinking bottle on the production line. The goal of working with the proposed DFTF algorithm is to find the orientation of water bottles on the production line.

4.2 Performance evaluation criteria

To illustrate the efficiency of proposed DFTF algorithm, we also adopt the similar measurement to other frameworks using the following criteria:

$$\text{Accuracy}=\frac{\text{TP+TN}}{\text{TP+FN+TN+FP}} \tag{21}$$

$$\text{Recall}=\frac{\text{TP}}{\text{TP+FN}} \tag{22}$$

$$\text{Precision}=\frac{\text{TP}}{\text{TP+FP}} \tag{23}$$

$$\text{F1}=\frac{2\times\text{Precision}\times\text{Reacll}}{\text{Precision+Reacll}} \tag{24}$$

where TP is the number of correctly detected object, FN is the number of missed detected object, and FP is the number of false object detection. Recall is the rate of missed detections. Precision, on the other hand, represents the false detection rate. The overall performance can be described by the F1-measure, which is a measure considering both recall and precision value.

5. Experimental results and performance

These experiments were tested based on Python version 3 on a computer running Windows 10 64-bit with a 3.40 GHz intel core i7-6700 CPU and 16 GB of RAM. The machine learning algorithms are based on the Scikit-learn library. The parameters for every experiment are set: Linear Regression (LGR) [random_state=0 , solver='lbfgs', multi_class='ovr',max_iter=1 0 0 0], Support Vecter Machine (SVN) [kernel="linear . ", max_iter=1000], Decision Tree (Tree) [max_depth=1 8], k-Nearest Neighbor (kNN) [n_neighbors=5], Naïve Bayes (NB) [var_smoothing = 1 e-9], and Random Forest (RF) [n_estimators=1 0 , max_depth=1 4 , random_state=0].

5.1 General datasets

The experiments for general datasets in these three databases were predicted. Each experiment in these databases will be defined with some different conditions. These conditions will be described in each experiment.

The MNIST database experiment was determined by classifying all data images to 10 groups of class. Each group represents handwritten numbers from 0 to 9 and have a total of 8 rotations: 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. This experiment will measure the predictive accuracy of our model. According to the experiment, there are two groups of rotation images that have the highest accuracy from experiments that are in the 0 and 180 degree rotation groups and both of them has the best accuracy in this experiment at 99.8%. The group with the worst accuracy from this experiment was in the 135° rotated image group with an average accuracy of 89.45%. The summary of the results from the MNIST database is shown in Table 2.

Table 2. Results of estimated angles on MNIST

	0	1	2	3	4	5	6	7	8	9	**Avg**
0°	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	**99.8**
45°	93.3	96.4	96.7	96.7	96.7	96.7	96.7	96.7	96.7	96.7	**96.33**
90°	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	96.7	99.8	**99.49**
135°	87.8	90.1	88.3	88.8	92.1	88.5	95.8	87.3	87.3	88.5	**89.45**
180°	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	99.8	**99.8**
225°	96.7	96.7	96.7	96.7	96.7	96.7	96.7	96.7	96.7	96.7	**96.7**
270°	99.8	99.8	99.8	99.8	95.8	99.8	99.8	99.8	99.8	99.8	**99.4**
315°	95.1	92.9	90.5	91.4	92.9	94.3	95.5	90.5	90.1	91.4	**92.46**

Table 3. Results of estimated angles on GTSRB

	Class1	Class2	Class3	Class4	Class5	**Avg**
0°	99.8	99.8	99.8	99.8	99.8	**99.8**
45°	99.8	99.8	96.7	96.7	99.8	**98.56**
90°	99.8	99.8	99.8	99.8	99.8	**99.8**
135°	99.8	95.8	96.7	96.7	95.8	**96.96**
180°	95.5	99.8	99.8	99.8	99.8	**98.94**
225°	96.7	96.7	96.7	96.7	99.8	**97.32**
270°	99.8	99.8	99.8	99.8	95.8	**99**
315°	99.8	95.5	99.8	96.4	99.8	**98.26**

Table 4. Results of estimated angles on Caltech-256

	Buddha	coffee-mug	bowling-pin	cactus	giraffe	**Avg**
0°	99.8	99.8	99.8	99.8	99.8	**99.8**
45°	99.8	96.7	96.7	96.7	96.7	**97.32**
90°	99.8	99.8	99.8	99.8	99.8	**99.8**
135°	92.5	88.3	99.8	93.3	95.8	**93.94**
180°	99.8	99.8	99.8	99.8	99.8	**99.8**
225°	99.8	99.8	96.7	96.7	99.8	**98.56**
270°	99.8	99.8	99.8	99.8	99.8	**99.8**
315°	95.5	99.8	95.5	99.8	95.5	**97.22**

The next experiment was the GTSRB database. We selected five groups of images from the database, which consisted of Buddha, coffee-mug, bowling-pin, and giraffe. All groups were rotated with eight groups of degrees of rotation: 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°, as in previous database experiments. From this experiment, we found that two groups of rotation images with the best accuracy from the experiments within the article were the experimental images in the 0 and 90-degree rotation groups. Both of which had the best accuracy at 99.8%. The worst average accuracy from these experiments from the GTSRB database was in the 135 degree rotated image group with an average accuracy of 96.96%. The summary of the results from the GTSRB database is shown in Table 3.

The final experiment in this general datasets group was Caltech-256, and five groups of image data were selected from the database, and all groups were subjected to a rotational image test with a total of eight groups of rotation angles: 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. From this experiment, two groups of rotation images with the highest accuracy from the experiments were found, in the rotation group 0, 90, 180, and 270 degrees, both of them had average accuracy. The best in this experiment was 99.8%. The group with the worst accuracy from the Caltech-256 database was in the 135° rotated image group, with an average accuracy of 93.94%. The summary of the results from the Caltech-256 database is shown in Table 4.

5.2 Our dataset

The second part is an experiment with a specialized database which is to examine the image and to predict the direction of rotation of the objects. The goal is to visualize experiments from a small drinking water bottling plant that would consist of two stations as shown in Fig. 11(a) that is, the first station is the bottle picking station and the second station is the drinking water filling station. The operation of this system relies on the picking of the water bottles that will be placed on the picking base in the direction of the water supply in different rotation. Then, the system commands the robotic arm to pick up the object with the suction head as shown in Fig. 11(b) lower move it to the object and suck it up.

The direction of the water bottles must be rotated to match the conveying channel, which has a rotation error of less than 10 degrees in such a way that the water bottles are arranged to the position of water bottle conveying as shown in Fig. 11(b) upper.

The sequence of steps in the program consists of steps as shown in Fig. 12 This application works as follows: the first step of the program Fig. 12(a) is to receive the image from the camera installed at the top of the first station and then receive the input for processing. The resulting image is then applied to the background removal as shown in Fig. 12(b), this step is done to remove unnecessary information from the image by relying on the reference image to remove the background. In Fig. 12(c), this step is

Received: November 1, 2022. Revised: December 3, 2022.

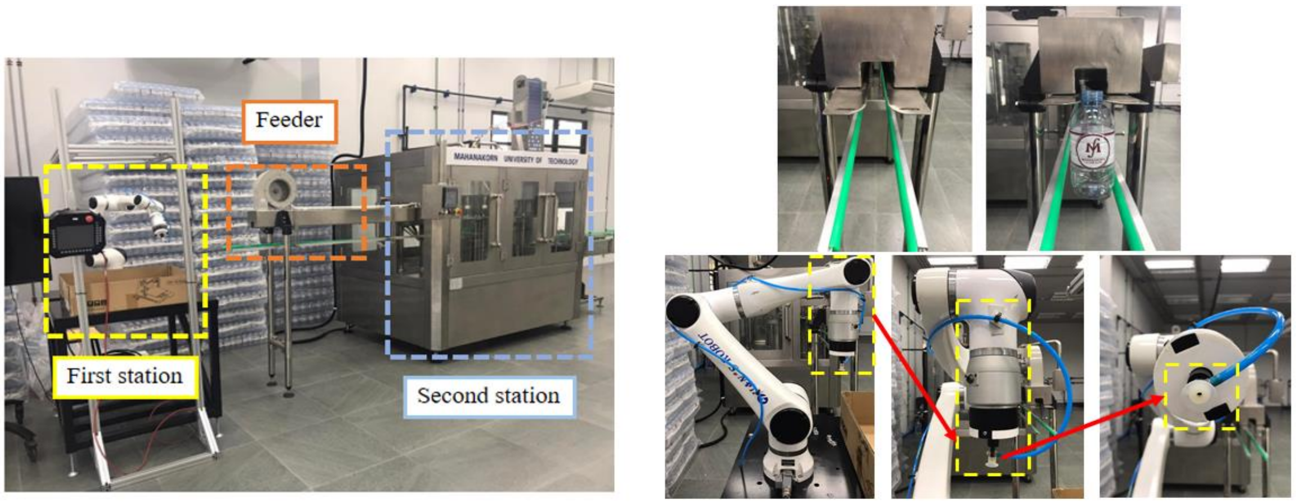


Figure. 11 All stations in the drinking water packing plant at Mahanakorn Institute of Innovation (MII), Mahanakorn University of Technology

processed by TF algorithm. Taking the output from TF to step in Fig. 12 (d) the operations are divided into two. The first step is computation with the DFTF algorithm to reduce the number and encode the data. The second step is to use the DFTF results to predict the rotation direction of the objects within the image by using a combination of prediction ROr and RDi. The answer from this last step is sent to the robotic arm to further adjust the suction position.

Considering the trace transform images can be used in a variety of ways when comparing the transformation results from Fig. 13, this will display the result of computation with a trace transform from images with different trace function shown in Table 1.

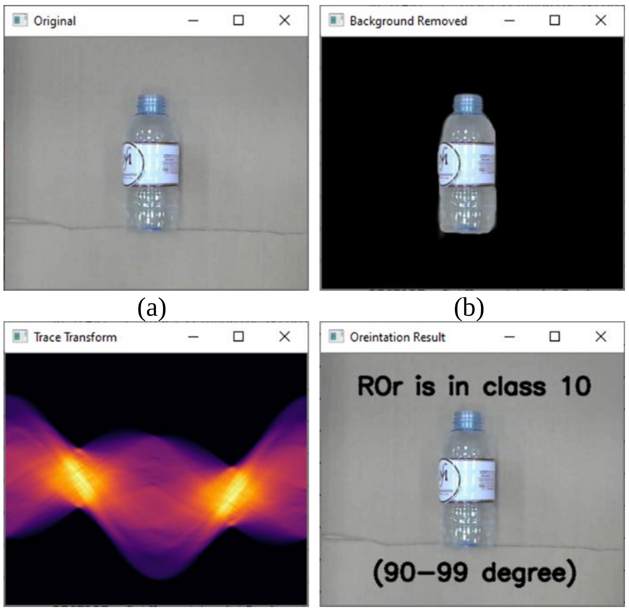


Figure. 12 Working on software side: (a) 1st step, (b) 2nd step, (c) 3rd step, and (d) 4th step

The experiment develops the DFTF algorithm to manipulate the data, it was used with the machine learning algorithm. Therefore, the test used to compare the results from the prediction of answers using the following algorithms: Logistic Regression (LGR), Support Vector Machine (SVM), Decision Tree (Tree), k-Nearest Neighbor (k- NN), Naïve Bay (NB) and Random Forest (RF) compared the results to determine the predictive accuracy of the object's rotational orientation.

Instead of using the original image to learn to generate weight values for the direction of rotation of the object in Fig. 14 (a), the data set in the LGR, SVM, Tree, k-NN and RF training set algorithms were best accurate at 100%, only NB was 89% in the training set. However, we found that the best experimental accuracy was only 69% with the k-NN algorithm and the lowest experimental accuracy was only 31% with the Tree algorithm.

Considering in Fig. 14 (b), which is to generate results from the TF algorithm and then use those results as training set. We found that for all the algorithms, the accuracy of the data in the training set is 100%. We observed that measuring the accuracy of the test set data, the algorithm achieved the best experimental accuracy of 82% with the SVM algorithm. And we found that the lowest accuracy in the experiment was 48% with the NB algorithm.

Considering in Fig. 14 (c). This is an experiment with the DFTF algorithm, to predict results in orientation class (ROr). We found that in the training set, only two algorithms, Tree and RF, reached at 100% accuracy. We considered the accuracy from the test set and found that the algorithm with the best accuracy was 79% with the

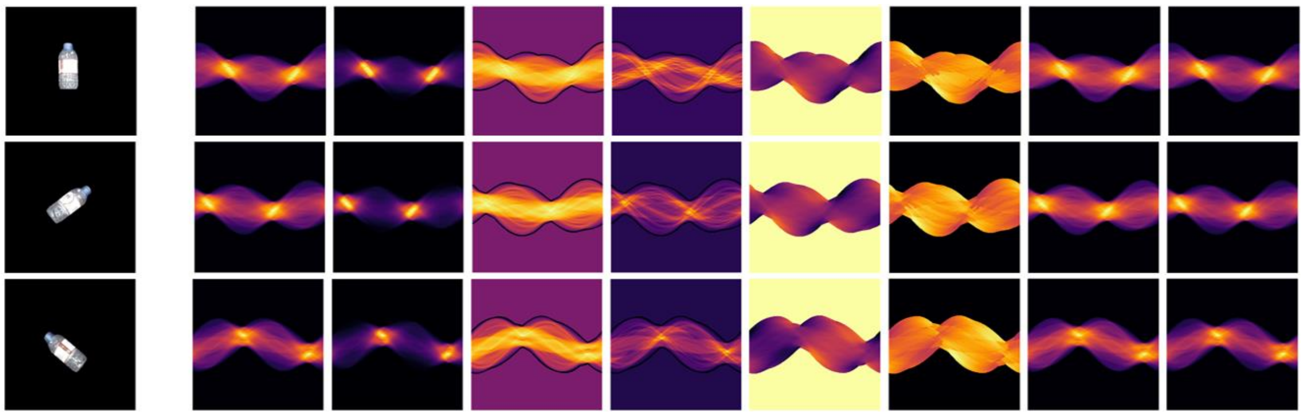


Figure. 13 Trace transform results with different function

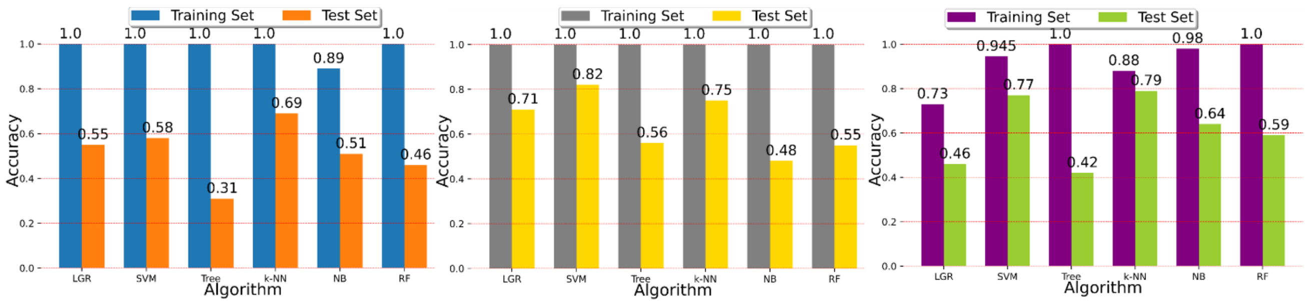


Figure. 14 Accuracy score for prediction rotation

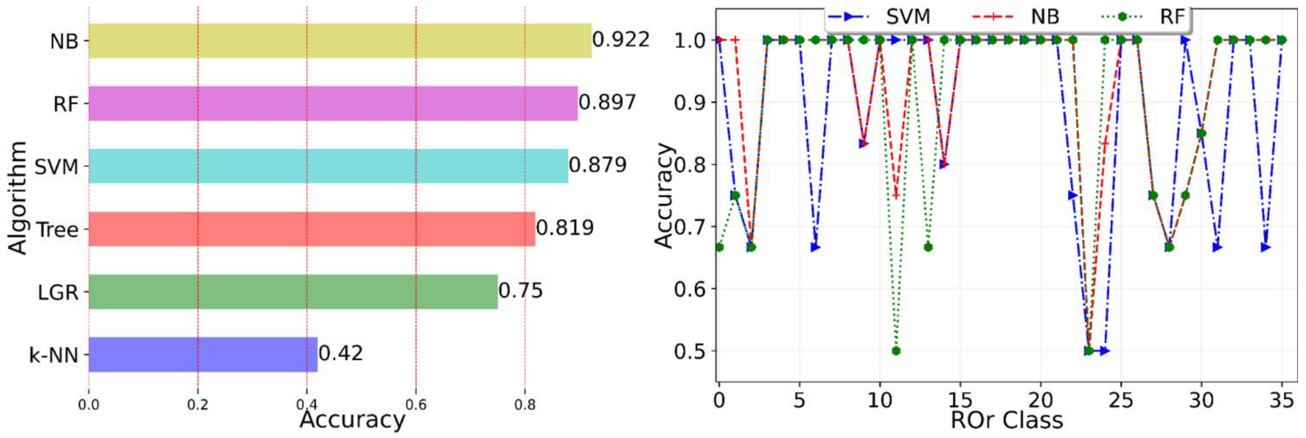


Figure. 15 The best accuracy score for prediction rotation with DFTF with quadrant prediction

k-NN algorithm. The worst accuracy was 42% with the Tree algorithm.

From the accuracy comparison results from Fig. 14 as above. We found that the use of images from the TF algorithm to generate predictive weights produced the best results from this experimental group.

However, when considering the quantity of features used for the dataset, it was found that the quantity was too large because data from both axes (∅ and ρ) which were required to predict the answer. The amount of features deployed to affect both training and prediction, with the higher the number of features and the greater the processing time.

Therefore, the contribution of another model algorithm is proposed for reinforcing the prediction of the object orientation from the first model, which is assigned to the direction prediction class (RDi).

Considering the reinforcement results with the second model for quadrant prediction, we found that the accuracy of object orientation prediction of the object increases as shown in Fig. 15.

Fig. 15 (a) demonstrates the predicted accuracy of responses within the test set of each algorithm was increased in the order of highest accuracy as follows: NB, RF, SVM, Tree, LGR, and k-NN. Each algorithm had an accuracy of 92.2%, 89.7%, 87.9%, 81.9%, 75% and 42%, respectively.

Fig. 15(b) shows the accuracy for predictions from Class 1 to 36 with the reinforcement algorithm. The top-three experimentally obtained best results are SVM, NB, and RF. From this experiment, it was

Received: November 1, 2022. Revised: December 3, 2022.

Table 5. Accuracy, precision, recall, F1-score and ROC score

Data Train	TF Function	Method	Accuracy	Precision	Recall	F1-Score	ROC
DFTF	2	LGR	0.75	0.75	0.828	0.734	0.871
	2	SVM	0.879	0.879	0.919	0.885	0.938
	2	Tree	0.819	0.819	0.887	0.818	0.907
	6	k-NN	0.422	0.422	0.458	0.41	0.703
	8	**NB**	**0.922**	**0.922**	**0.945**	**0.926**	**0.96**
	1	RF	0.914	0.914	0.939	0.915	0.956
TF	2	LGR	0.767	0.778	0.767	0.738	0.88
	2	**SVM**	**0.819**	**0.806**	**0.819**	**0.796**	**0.907**
	8	Tree	0.655	0.679	0.655	0.635	0.823
	4	k-NN	0.543	0.606	0.543	0.539	0.765
	7	NB	0.629	0.712	0.629	0.631	0.81
	2	RF	0.672	0.677	0.672	0.654	0.832
Original Image	-	LGR	0.483	0.453	0.483	0.399	0.734
	-	SVM	0.784	0.786	0.784	0.762	0.889
	-	Tree	0.664	0.683	0.664	0.645	0.827
	-	k-NN	0.422	0.441	0.422	0.407	0.703
	-	NB	0.6*81*	0.701	0.716	0.676	0.845
	-	**RF**	**0.784**	**0.829**	**0.784**	**0.765**	**0.889**

found that when using the NB algorithm for predicting ROr and using the reinforced with RF algorithm for predicting RDi, the accuracy was the best for this experiment. The average of all classes in the direction of rotation prediction is greater than 80%.

The summary results for the drinking water bottle database are shown in Table 5. It shows the various measurement methods used to measure the performance of all algorithms were tested for comparison and determine the best test value for each machine learning algorithms through the following measurements accuracy, precision, recall, F1-Score and ROC. Considering the weight measurements generated with the original image data, we found that the RF algorithm gave the best experimental results of 78.4%, 82.9%, 78.4%, 76.5%, and 88.9% respectively. Subsequently, considering the measurement values of the weighted constructs with TF results, we found that out of algorithms 1–8 that were used to generate results in the TF domain and algorithm 8 gave the best results. The model used to generate the best weight values was SVM, with measurement values of 81.9%, 80.6%, 81.9%, 79.6%, and 90.7%, respectively. Although it was found that the measurement values from visually generated weights from domain TFs exceeded the mean by more than 80%. We found that the domain TF data that was used to construct the DFTF, the appropriate TF function was also the 8th function by NB algorithm with measurements of 92.2%, 92.2%, 94.5%, 92.6%, and 96%, respectively.

6. Conclusion

In this paper, we presented the process to use the advantage of Trace Transform (TF) algorithm using supervised machine learning. The goal of this paper is to generate a weight value for predicting the object rotation. We have found that images from the TF domain are generated weights to predict responses exceeds the number of features. Therefore, our proposed is to reduce the number of features from the resulting TF domain by the Discriminant Feature Trace Transform (DFTF) algorithm. This algorithm accumulates the resulting values in a one-dimensional format and increases processing speed. The reduced data for this feature is then used to generate weights for the primary data predict, Orientation (ROr) by the Naïve Bayes (NB) algorithm, and quadrant predict generation (RDi), which complements the first prediction with the Random Forest (RF) algorithm. We found that accuracy was measured against the MNIST, Caltech-256, and GTSRB databases, measuring the direction of rotation of eight object groups: 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. We noticed that all three databases using the proposed algorithms which had the lowest average accuracy at 89.45% and the highest at 99.8%.

In part of experiment with image data from water bottle databases, we found that the best accuracy is at 92.2% whereas the accuracy from learning from TF domain images only has the accuracy is the best at 78% . From all the experiments, we can conclude that the proposed methods can predict with more accuracy rate of the orientation and reduce the amount of information needed to determine the rotation. This affects the processing time, which takes less time than predicting with the TF image.

Future work that can be used for industrial work with the need to arrange objects on the production line. Within the production line there is a need to arrange objects passed through another station. Another example is the task of grabbing and arranging objects into containers. Another work that needs to find the direction of object rotation within the image used in conjunction with a robotic arm.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Nattapong Jundang and Suchada Sitjongsataporn; methodology, Nattapong Jundang and Suchada Sitjongsataporn; software; Nattapong Jundang; formal analysis, Nattapong Jundang and Suchada Sitjongsataporn; writing—original draft preparation, Nattapong Jundang and Suchada Sitjongsataporn; writing—review and editing, Nattapong Jundang and Suchada Sitjongsataporn; supervision, Suchada Sitjongsataporn.

References

[1] S. Prongnuch, T. Wiangtong, and S. Sitjongsataporn, "Qualitative Precipitation Estimation from Satellite Data Based on Distributed Domain-Specific Architecture", *Modelling and Simulation in Engineering (MSE)*, Vol. 2021, 2021.

[2] Y. Yao, C. Sun, X. Liu, G. Jiang, A. Kadyrov, and M. Petrou, "Robust Range Estimation Based on the Slope of the Two-Dimensional Fourier Transform Ridge Using Radon Transform", *IEEE Access*, Vol. 9, pp. 24093-24104, 2021.

[3] J. Duan, G. Zhang, G. Hu, C. Yu, Y. Yue, B. Li, and Y. Li, "High-Precision Iterative VSP Wavefield Separation", *IEEE Geoscience and Remote Sensing Letters*, Vol. 19, pp. 1-5, 2022.

[4] S. Dua, J. Singh, and H. Parthasarathy, "Rotation angle estimation of an image using least square method", In: *Proc. of International Conf. on Signal Processing and Integrated Networks*, pp. 296-300, 2016.

[5] J. Liu, X. Kong, F. Xia, X. Bai, L. Wang, Q. Qing, and I. Lee, "Artificial Intelligence in the 21st Century", *IEEE Access*, Vol. 6, pp. 34403-34421, 2018.

[6] J. C. S. Reis, A. Correia, F. Murai, A. Veloso, and F. Benevenuto, "Supervised Learning for Fake News Detection", *IEEE Intelligent Systems*, Vol. 34, No. 2, pp. 76-81, 2019.

[7] A. S. Alatrany, A. J. Hussain, J. Mustafina, and D. A. Jumeily, "Machine Learning Approaches and Applications in Genome Wide Association Study for Alzheimer's Disease: A Systematic Review", *IEEE Access*, Vol. 10, pp. 62831-62847, 2022.

[8] M. P. Hosseini, A. Hosseini, and K. Ahi, "A Review on Machine Learning for EEG Signal Processing in Bioengineering", *IEEE Reviews in Biomedical Engineering*, Vol. 14, pp. 204-218, 2021.

[9] G. J. Horng and C. X. Wu, "Building an Adaptive Machine Learning Object-Positioning System in a Monocular Vision Environment", *IEEE Sensors Journal*, Vol. 21, No. 7, pp. 9553-9566, 2021.

[10] Y. Luo, K. Dong, L. Zhao, Z. Sun, E. Cheng, H. Kan, C. Zhou, and B. Song, "Calibration-Free Monocular Vision-Based Robot Manipulations With Occlusion Awareness", *IEEE Access*, Vol. 9, pp. 85265-85276, 2021.

[11] M. Aslam, T. M. Khan, S. S. Naqvi, G. Holmes, and R. Naffa, "On the Application of Automated Machine Vision for Leather Defect Inspection and Grading: A Survey", *IEEE Access*, Vol. 7, pp. 176065-176086, 2019.

[12] S. J. Lee, S. H. Park, and L. T. Kim, "Improved Classification Performance Using ISAR Images and Trace Transform", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 53, No. 2, pp. 950-965, 2017.

[13] W. Y. Ling and L. Ming, "Circular Trace Transform and Its Applications to Image Texture Analysis", *Acta Electronica Sinica*, Vol. 10, pp. 2351-2358, 2018.

[14] P. Gulyaev, V. Jordan, I. Gulyaev, and A Dolmatov, "Trace-transform invariants of tracks of high-velocity jets from the surface of tungsten droplets in the plasma flow", *Journal of Physics: Conference Series*, Vol. 830, 2016.

[15] Z. Xue, J. Wei, and W. Guo, "A Real-Time Naive Bayes Classifier Accelerator on FPGA", *IEEE Access*, Vol. 8, pp. 40755-40766, 2020.

[16] Q. Yin, J. Cheng, F. Zhang, Y. Zhou, L. Shao, and W. Hong, "Interpretable POLSAR Image Classification Based on Adaptive-Dimension Feature Space Decision Tree", *IEEE Access*, Vol. 8, pp. 173826-173837, 2020.

[17] H. S. Jang, K. Y. Bae, H. S. Park, and D. K. Sung, "Solar Power Prediction Based on Satellite Images and Support Vector Machine", *IEEE Transactions on Sustainable Energy*, Vol. 7, No. 3, pp. 1255-1263, 2016.

[18] B. Liu, W. Guo, X. Chen, K. Gao, X. Zuo, R. Wang, and A. Yu, "Morphological Attribute Profile Cube and Deep Random Forest for Small Sample Classification of Hyperspectral Image", *IEEE Access*, Vol. 8, pp. 117096-117108, 2020.

[19] S. Srisuk, M. Petrou, W. Kurutach, and A. Kadyrov, "Face authentication using the trace transform", In: *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. I-I, 2003.

[20] C. K. Aridas, S. Karlos, V. G. Kanas, N. Fazakis, and S. B. Kotsiantis, "Uncertainty Based Under-Sampling for Learning Naive Bayes Classifiers Under Imbalanced Data Sets", *IEEE Access*, Vol. 8, pp. 2122-2133, 2020.

[21] T. Kim and J. S. Lee, "Exponential Loss Minimization for Learning Weighted Naive Bayes Classifiers", *IEEE Access*, Vol. 10, pp. 22724-22736, 2022.

[22] Z. Xue, J. Wei, and W. Guo, "A Real-Time Naive Bayes Classifier Accelerator on FPGA", *IEEE Access*, Vol. 8, pp. 40755-40766, 2020.

[23] M. Jeong, J. Nam, and B. C. Ko, "Lightweight Multilayer Random Forests for Monitoring Driver Emotional Status", *IEEE Access*, Vol. 8, pp. 60344-60354, 2020.

[24] M. Söchting, S. Allegretti, F. Bolelli, and C. Grana, "A Heuristic-Based Decision Tree for Connected Components Labeling of 3D Volumes", In: *Proc. of International Conference on Pattern Recognition*, pp. 7751-7758, 2021.

[25] L. Deng, "The mnist database of handwritten digit images for machine learning research", *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp. 141-142, 2022.

[26] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German Traffic Sign Recognition Benchmark: A multi-class classification competition", In: *Proc. of International Joint Conference on Neural Networks, San Jose, CA, USA*, pp. 1453-1460, 2011.

[27] G. Griffin, A. Holub, and P. Perona, "Caltech-256 Object Category Dataset", *Technical Report 7694*, 2007.