applied sciences

MDPI

Article

Object/Scene Recognition Based on a Directional Pixel Voting Descriptor

Abiel Aguilar-González [1,*], Alejandro Medina Santiago [1] and J. A. de Jesús Osuna-Coutiño [2]

[1] Computer Science Department, Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Santa Maria Tonantzintla 72840 , Mexico
[2] Mechatronics Engineering Department, Tecnológico Nacional de México, Instituto Tecnológico de Tuxtla Gutiérrez (ITTG) , Tuxtla Gutiérrez 29050, Mexico
* Correspondence: abiel@inaoep.mx; Tel.: +52 (222) 2663100 (ext. 8302/8308)

Abstract: Detecting objects in images is crucial for several applications, including surveillance, autonomous navigation, augmented reality, and so on. Although AI-based approaches such as Convolutional Neural Networks (CNNs) have proven highly effective in object detection, in scenarios where the objects being recognized are unknow, it is difficult to generalize an AI model for such tasks. In another trend, feature-based approaches like SIFT, SURF, and ORB offer the capability to search any object but have limitations under complex visual variations. In this work, we introduce a novel edge-based object/scene recognition method. We propose that utilizing feature edges, instead of feature points, offers high performance under complex visual variations. Our primary contribution is a directional pixel voting descriptor based on image segments. Experimental results are promising; compared to previous approaches, ours demonstrates superior performance under complex visual variations and high processing speed.

Keywords: object recognition; scene recognition; image processing; feature extraction; feature matching

Citation: Aguilar-González, A.; Medina Santiago, A.; Osuna-Coutiño, J.A.d.J. Object/Scene Recognition Based on a Directional Pixel Voting Descriptor. *Appl. Sci.* **2024**, *14*, 8187. https://doi.org/10.3390/app14188187

Academic Editor: Andrés Márquez

Received: 3 July 2024
Revised: 28 August 2024
Accepted: 5 September 2024
Published: 11 September 2024

1. Introduction

Detecting objects in images is crucial in multiple applications, including surveillance, autonomous navigation, augmented reality, and so on [1–4]. While AI-based approaches such as Convolutional Neural Networks (CNNs) have demonstrated high performance in object detection [5,6], in scenarios where object/scene being recognized is unknown it is difficult to achieve accurate recognition. i.e., without prior training data on that specific object, traditional AI methods reach poor performance.

On the other hand, feature-based approaches like SIFT [7], SURF [8], and ORB [9] offer capability to search any template but have limitations under complex visual variations. These methods extract distinctive features from an image and use them to detect objects [10,11]. By carefully selecting and combining these feature-based techniques, developers can achieve efficient object detection solutions without compromising performance.

Another approach to object detection involves high-level abstraction descriptors such as those based on edges and shapes. These methods combine the advantages of both CNNs and feature-based techniques [1,12]. They aim to be more robust than feature-based approaches, akin to CNNs, and at the same time they can maintain relatively simple complexity allowing to search any template, akin to feature-based methods.

In this work, we present an edge-based object/scene recognition method. We propose that the utilization of shapes obtained from edge points should achieve high performance given any type of template. Our contribution lies in the introduction of a novel directional pixel voting descriptor using image segments. The experimental results demonstrated high performance under complex visual variations. Furthermore, our methodology enables both recognition of both objects and scenes. This dual capability opens up promising avenues for future exploration and practical real-world applications.

1.1. The Object Recognition Problem

The object recognition problem is a key challenge in computer vision, with applications in several domains, such as robotics, surveillance, and augmented reality. Object recognition involves identifying and categorizing objects within digital images or video frames. This task is particularly challenging due to the variability in object appearance, including differences in viewpoint, scale, lighting conditions, occlusions, and so on.

Addressing the object recognition problem typically involves developing algorithms and techniques capable of extracting discriminative features from visual data, followed by classification or matching processes to determine the identity or category of detected objects. Traditional approaches, such as SIFT, SURF, or ORB, were designed to capture distinctive characteristics of objects, as shown in Figure 1. However, these methods offer poor accuracy under complex visual variations and required extensive tuning for optimal performance across different scenarios [13,14].

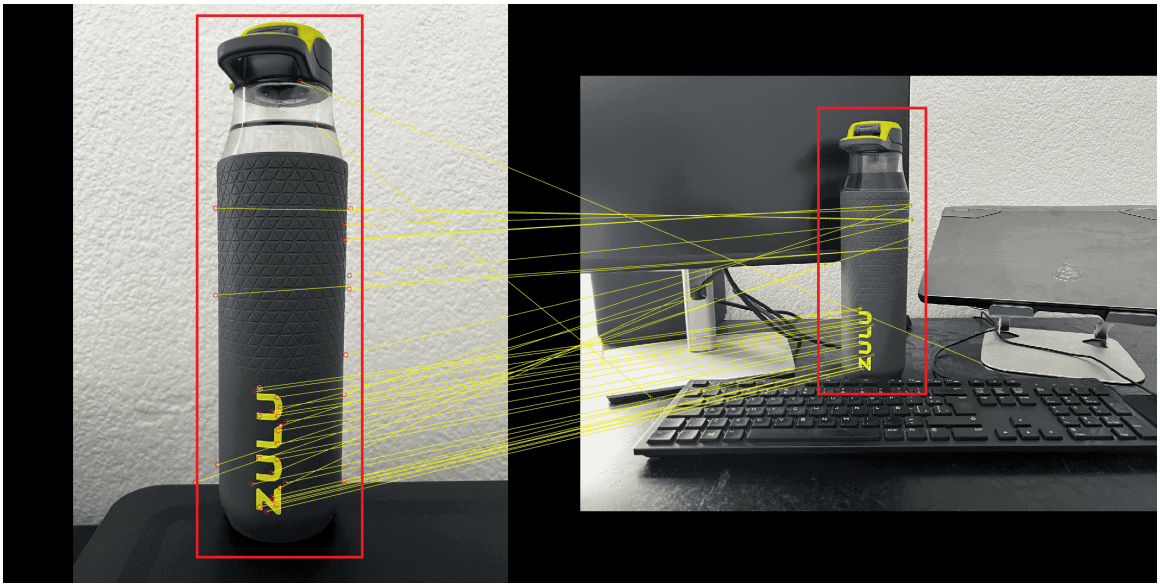


Figure 1. The object recognition problem: features (such as blobs and corners) are extracted and matched from a template to a query. Then, depending on the percentage of "successfully" matched features, the object can be recognized within the scene. By interpolating the matched features, it is possible to determine the corresponding bounding box.

1.2. The Scene Recognition Problem

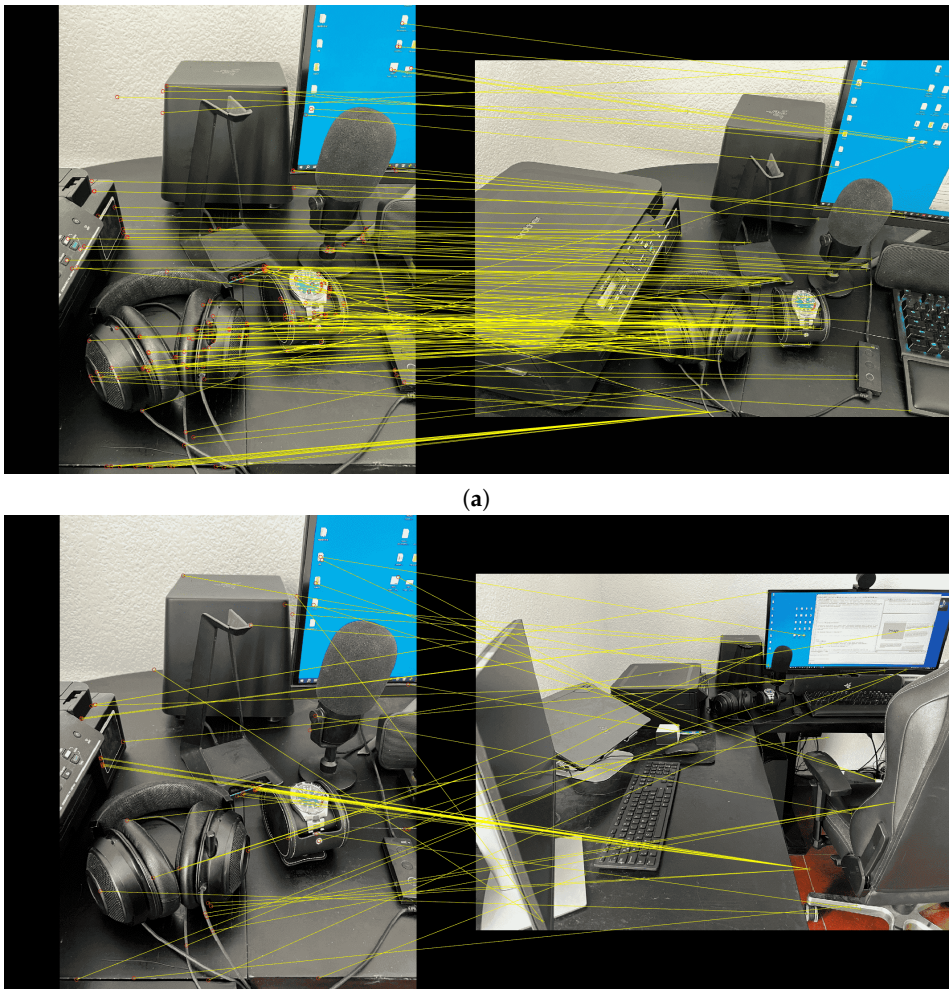
Scene recognition is a key issue in several computer vision applications such as robotics and autonomous driving. In recent years, deep learning-based approaches have demonstrated high performance for scene recognition. In particular, convolutional Neural Networks (CNNs) demonstrated remarkable capabilities in learning representations of scenes directly from raw pixel data, enabling accurate scene categorization across diverse datasets and scenarios [15–18].

Unlike object recognition, which focuses on identifying individual objects within images, scene recognition involves understanding and categorizing entire scenes or environments depicted in images or video frames. This task presents its own set of challenges, primarily regarding the interaction on numerous objects, textures, structures, and spatial arrangements within a scene, see Figure 2.

Scene recognition algorithms must consider a wide range of factors, including variations in scene composition, lighting conditions, viewpoints, occlusions, and clutter. In addition, scenes often exhibit semantic context and spatial relationships between objects, further complicating the recognition process. Overcoming these challenges involves the development of complex computational models capable of extracting robust visual features

from scene representations and effectively capturing the contextual information suitable for accurate scene categorization.

In the current literature, scene recognition approaches consist of techniques such as bag-of-visual-words, spatial pyramid matching, and deep learning architectures like Convolutional Neural Networks (CNNs). These methods aim to encode both local and global spatial information, enabling robust recognition across diverse scene categories. However, achieving reliable scene recognition performance is not possible, with opportunities for innovation in feature representation, model design, and training strategies [19–22].



(**a**)

(**b**)

Figure 2. The scene recognition problem: consist of extracting and matching features from a template to a query. Then, depending on the percentage of "successfully" matched features, the scene can be recognized or not. This approach works with relatively high performance under simple scenarios (low changes in texture, illumination, etc.) (**a**), but presents poor performance under complex scenarios (**b**). In those scenarios, deep learning-based approaches have demonstrated a proper trade-off between robustness and computational resources usage.

2. Related Works

Several works which addressed both feature-based and CNN-based approaches for object and scene recognition have been presented. Researchers have explored several feature extraction techniques, such as SIFT, SURF, ORB, and their variants, to capture discriminative information from visual data. Furthermore, multiple works have focused on the utilization of deep learning architectures, particularly CNNs. These works have contributed to advancing the state-of-the-art in object and scene recognition, offering insights into effective feature representation, model architecture design, and optimization strategies.

2.1. State of the Art for Object Recognition

In current state of the art in object recognition, deep-learning-based approaches, particularly Convolutional Neural Networks (CNNs), have demonstrated high performance and flexibility in terms of application domain. Models like ResNet, EfficientNet, and DenseNet have laid down the boundaries of object recognition accuracy and efficiency. In [23], the authors combine both object detection and image dehazing methods for real-time applications such as remote sensing, video surveillance, driverless automatic vehicles, etc. They presented an effective and efficient image dehazing method using transfer learning, which helps recognize objects in real time with more clarity and can automatically detect objects with a high recognition rate and lower probability of error. The trained model AOD-net + YOLO v3 consistently outperforms non-joint and naive YOLO v3 techniques. Ref. [24] proposed a recurrent CNN (RCNN) for object recognition by incorporating recurrent connections into each convolutional layer. Though the input is static, the activities of RCNN units evolve over time so that the activity of each unit is modulated by the activities of its neighboring units. This property enhances the model's ability to integrate context information, which is important for object recognition. Like other recurrent neural networks, unfolding the RCNN through time can result in an arbitrarily deep network with a fixed number of parameters. Although both approaches allow high robustness in terms of detection, both algorithms were based on machine learning, and the accuracy under unknown objects, different than those in the training, limits the scope and performance.

Furthermore, transfer learning techniques demonstrated high performance with pre-trained CNN models on large-scale datasets such as ImageNet for downstream object recognition tasks with limited labeled data. This approach allows for rapid development and deployment of object recognition systems in various domains. In [25], the use of transfer learning, via deep Convolutional Neural Networks (CNN), for the image classification problem within the context of X-ray baggage security screening was presented. The proposed algorithm achieved 98.92% detection accuracy, outperforming previous work, and further extended their evaluation to a multiple object classification task within this context. Additionally, in [26], multiple CNN architectures were proposed to improve the system performance. In this context, AlexNet, VGG16, and VGG19 are popular CNNs which allow transfer learning to fine-tune the pre-trained network parameters (VGG19) for image classification. Performance analysis demonstrated that the fine-tuned VGG19 architecture outperforms other CNN architectures and hybrid learning approaches for image classification tasks.

While deep learning models achieved high performance in several domains, feature-based approaches like SIFT and SURF still find applications in scenarios with limited computational resources or when there are specific challenges such as small object detection or texture-rich environments. Feature-based methods such as Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) are still relevant in object recognition tasks. These feature-based approaches offer distinct advantages, including robustness to variations in scale, rotation, and illumination. In [27], a new tactile-SIFT descriptor is proposed to extract features based on gradients in tactile images to represent objects, allowing the features to be invariant to object translation and rotation. The tactile-SIFT segments a tactile image into overlapping subpatches, each represented using a dn-dimensional gradient vector, similar to the classic SIFT descriptor. Tactile-SIFT descriptors obtained from multiple touches form a dictionary of k words, and the bag-of-words method is then used to identify objects. The proposed method was validated by classifying 18 real objects with data from an off-the-shelf tactile sensor, achieving a recognition rate of 91.33% with a dictionary size of 50 clusters using only 15 touches. Additionally, in [28], the authors have compared the performance of the Shi-Tomasi corner detector with SIFT and SURF feature descriptors and evaluated the performance of Shi-Tomasi in combination with SIFT and SURF feature descriptors. In both cases, relatively high accuracy in terms of detection was achieved, motivating us to continue this trend with an edge-based approach which increases the robustness of traditional approaches (SIFT/SURF/ORB).

Finally, hybrid approaches that combine deep learning with traditional feature-based methods have gained traction, aiming to maintain the strengths of both paradigms. For instance, deep learning models can be augmented with features extracted using SIFT or SURF, providing complementary information and improving overall recognition accuracy. For example, in [29], a technique was proposed using deep convolutional neural network (DCNN) and scale invariant features transform (SIFT). First, an improved saliency method is implemented, and the features points are extracted. Then, DCNN features are extracted from two deep CNN models like VGG and AlexNet. The proposed method is evaluated on three public datasets including Caltech101, Barkley 3D, and Pascal 3D and reached classification accuracy of 93.8%, 99%, and 88.6%—it demonstrated high performance compared to previous works.

2.2. State of the Art for Scene Recognition

For scene recognition, deep learning-based approaches remain at the forefront of the current state of the art, offering significant advancements in accuracy and robustness. Convolutional Neural Networks (CNNs) have emerged as the backbone of several recognition systems, capable of automatically learning visual features from images. One notable trend in scene recognition is the integration of contextual information and spatial relationships between objects within a scene [30,31].

Finally, the fusion of multi-modal data sources, such as images, textual descriptions, and semantic annotations, have demonstrated high performance for scene recognition. Techniques such as multi-modal fusion networks and cross-modal retrieval provide complementary information from different modalities in order to improve the scene understanding [32].

3. The Proposed Algorithm

In Figure 3, an overview of our algorithm is shown. Our approach consists of an image preprocessing step, in which input images are filtered and some primitive information such as image gradients (g_x, g_y, d) are extracted. Then, in a second step, relevant patterns such as edge segments s are extracted and described using our directional pixel voting descriptor. Finally, the input image is compared with a target (template), which can be an object or a scene, to determine whether it can be recognized or not.

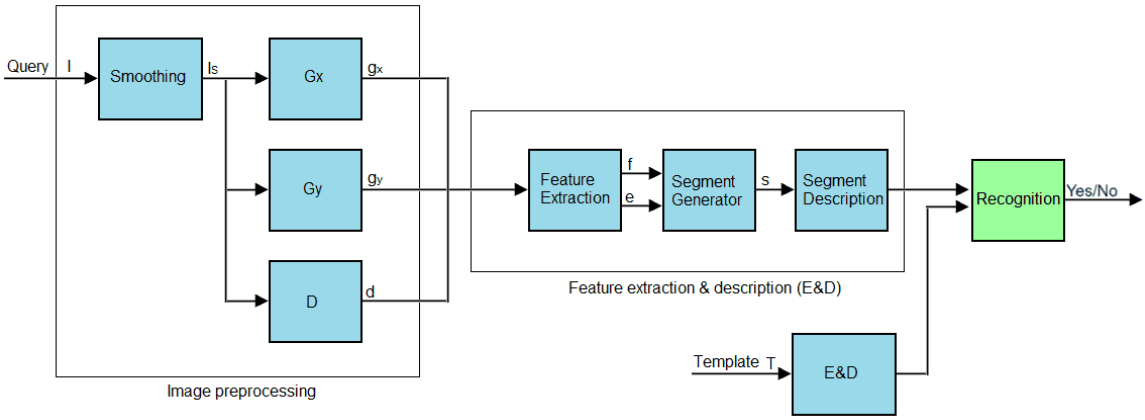


Figure 3. Block diagram of the proposed algorithm.

3.1. Image Preprocessing

Given a gray scale image $I(i, j)$, a Gaussian filtering is applied on I image on order to reduce noise and to remove fine-scale structures that affect the performance of the feature extraction step. The Gaussian filter is defined by Equation (1) as follows:

$$G(m, n) = \frac{1}{2\pi\sigma^2} e^{-\frac{m^2+n^2}{2\sigma^2}} \quad (1)$$

where $G(m,n)$ is the Gaussian function and σ is the standard deviation, which controls the amount of smoothing applied to the image.

To perform Gaussian filtering on the image I, a convolution operation is applied between the image I and a Gaussian kernel G. The Gaussian kernel is usually represented as a odd size 2D matrix or mask. For a generic 5×5 ($N \times N$) Gaussian mask, it can be represented as:

$$\frac{1}{273}\begin{pmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{pmatrix}$$

The above matrix represents the coefficients of the Gaussian mask, which are computed based on the Gaussian function with an appropriate standard deviation. This mask is then applied to the image using convolution, resulting in a smoothed version of the original image.

$$I_s(i,j) = \sum_{m=-\frac{N-1}{2}}^{\frac{N-1}{2}} \sum_{n=-\frac{N-1}{2}}^{\frac{N-1}{2}} I(i+m, j+n) \cdot G(m + \frac{N-1}{2}, n + \frac{N-1}{2})$$

Given the smoothed image $I_s(i,j)$, the gradients g_x and g_y are calculated via Prewitt operator [33] as follows:

$$g_x(i,j) = (-I_s(i-1,j-1) - I_s(i,j-1) - I_s(i+1,j-1) + I_s(i-1,j+1) + I_s(i,j+1) + I_s(i+1,j+1))$$

$$g_y(i,j) = (-I_s(i-1,j-1) - I_s(i-1,j) - I_s(i-1,j+1) + I_s(i+1,j-1) + I_s(i+1,j) + I_s(i+1,j+1))$$

Finally, the direction $d(i,j)$ is determined as follows:

$$d(i,j) = \begin{cases} 1 & \text{if } (g_x(i,j) > g_y(i,j)) \,\&\, (g_x(i,j) + g_y(i,j)) \geq \text{th} \\ 2 & \text{if } (g_x(i,j) \leq g_y(i,j)) \,\&\, (g_x(i,j) + g_y(i,j)) \geq \text{th} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where th is a threshold for a "valid" gradient value. We define the possible values of th inspired by the image bit-depth standard format, i.e., 8, 16, 32, 64, and so on. We propose th $= 32$ as a threshold because, even after the smoothing process using the Gaussian filter, certain levels of noise and fine details may persist in the image. These artifacts can negatively affect the performance of subsequent steps, such as feature extraction or object recognition. By setting a threshold equal to 32, we aim to identify and suppress gradients that are not significant enough to represent relevant features in the image. Gradient values below this threshold may be considered as unwanted noise or fine details that we aim to eliminate. Note that the threshold value th does not explicitly depend on the bit depth of the image. However, given that the gradients g_x and g_x are computed from the absolute values of the image, which inherently depend on its bit depth, it is reasonable to argue that the sum g_{xy} should be compared to a threshold th that scales with the maximum gray level of the image. Specifically, a fixed threshold value of 32 might be appropriate for images with 8-bit depth (gray values ranging from 0 to 255), but could be inadequate for images with higher bit depths, such as 16-bit images with gray values ranging from 0 to 65,535.

3.2. Feature Extraction and Description

Our main contributions, on the one hand, consist of introducing a novel approach for feature (segments) extraction and, on the other hand, presenting a robust description criterion.

3.2.1. Feature Extraction

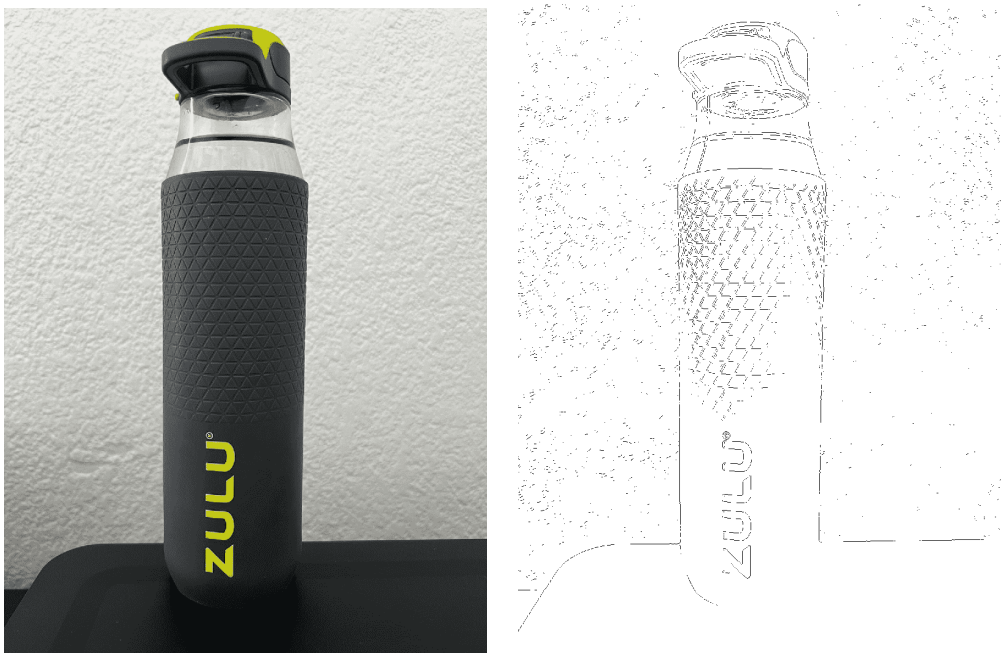
As mentioned before, our approach is based on edge primitives. Although there are existing algorithms, such as Sobel and Canny, for that purpose, they are sensitive to noise or difficult to segment properly. To address those issues, we propose a new extraction method as follows, where the edge map $e(i,j)$ (see Figure 4) is defined as follows:

$$e(i,j) = \begin{cases} 1 & \text{if } ((\text{dif}_1 \geq \text{th } \& \ \text{dif}_2 \geq \text{th})) \ \& \ d(i,j) == 1 \\ 1 & \text{if } ((\text{dif}_3 \geq \text{th } \& \ \text{dif}_4 \geq \text{th})) \ \& \ d(i,j) == 2 \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where dif_1, dif_2, dif_3, dif_4, and $g_{xy}(i,j)$ are defined as follows: $g_{xy}(i,j) = g_x(i,j) + g_y(i,j)$

$\text{dif}_1 = g_{xy}(i,j) - g_{xy}(i,j-1)$,
$\text{dif}_2 = g_{xy}(i,j) - g_{xy}(i,j+1)$,
$\text{dif}_3 = g_{xy}(i,j) - g_{xy}(i-1,j)$,
$\text{dif}_4 = g_{xy}(i,j) - g_{xy}(i+1,j)$.

Then, the set of anchor points $f(k)$ is computed via Algorithm 1.



(**a**) (**b**)

Figure 4. Let (**a**) be the input image template an edge map is computed via Equation (3) given the resulting image presented in (**b**).

Algorithm 1: Constructing anchor points vector $f(k)$

1. Initialize vector f as empty.
2. Initialize $k = 0$.
3. **For** each pair of coordinates (i,j) in the edge image $e(i,j)$:
   (a) **If** $e(i,j) == 1$:
      i. $k \leftarrow k + 1$
      ii. Add coordinate $\{i,j\}$ to vector f at position k.
4. **Return** vector f.

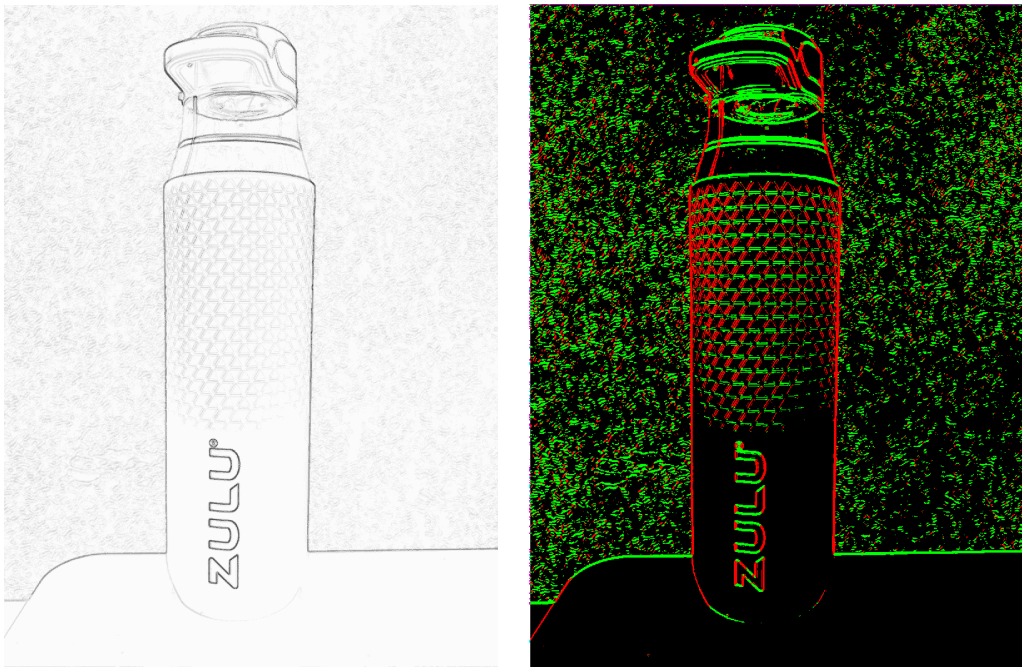
3.2.2. Segment Generator (Feature Extraction)

To generate "robust" edge segments, we propose an approach inspired on the canonical seed-segmentation method [34,35]. In this method, a seed is selected to initiate the segmentation process. This seed serves as the starting point for segment expansion. The segmentation algorithm iteratively grows the segment by incorporating neighboring pixels that satisfy certain criteria, such as similarity in color, texture, or intensity. This process continues until the segment reaches a predefined stopping criterion, ensuring that it encompasses a coherent region of interest. By leveraging the canonical seed-segmentation approach, we aim to achieve accurate and reliable edge segments that capture meaningful structures in the image data. This approach offers robustness against noise, variations in lighting conditions, and other sources of interference, making it well-suited for a wide range of image segmentation tasks.

Let $g_{xy}(i,j)$ be the gradient map ($g_{xy}(i,j) = g_x(i,j) + g_y(i,j)$), $d(i,j)$ be the direction map computed via Equation (2), as shown in Figure 5b, and $f(k)$ denote an anchor point (i,j); these anchor points are grown horizontally or vertically depending on the direction image at anchor point (i,j), as illustrated in Figure 6, where for each anchor $(f(k))$, there are 5 (n) possible growth directions. Each direction (marked by a red square) is evaluated using the gradient map $(g_{xy}(i,j))$ as an edge criterion, where $\mathrm{Crl}(n)$ defines the correlation between the anchor point and the $n-$ th growth direction. In practice, the growth direction could change from vertical to horizontal in the same segment, but this is supported by the diagonal element of the grown pattern, $\{p_2, p_4\}$, $\{p_4, p_2\}$ Figure 6a. Therefore, when a change in direction occurs, it is addressed by the current growth pattern, and when the direction changes (in the directions image), the growth pattern is changed. All possible directions are ordered clockwise. This facilitates the software implementation by implementing shift vectors. Notice that all evaluated directions are disabled by resetting their corresponding value to zero. Finally, the segment grows in the direction with the highest edge correlation compared to the anchor, and only if this correlation is always greater than a threshold th. The whole process is lay down in Algorithm 2. For practical purposes, we recommend a threshold value of 32. The whole process is illustrated in Figure 7; when the current pixel does not allow growth, the algorithm returns to the original anchor and starts to grow in the opposite direction. Finally, in Figure 8, the extracted features, 'edge segments', are shown.

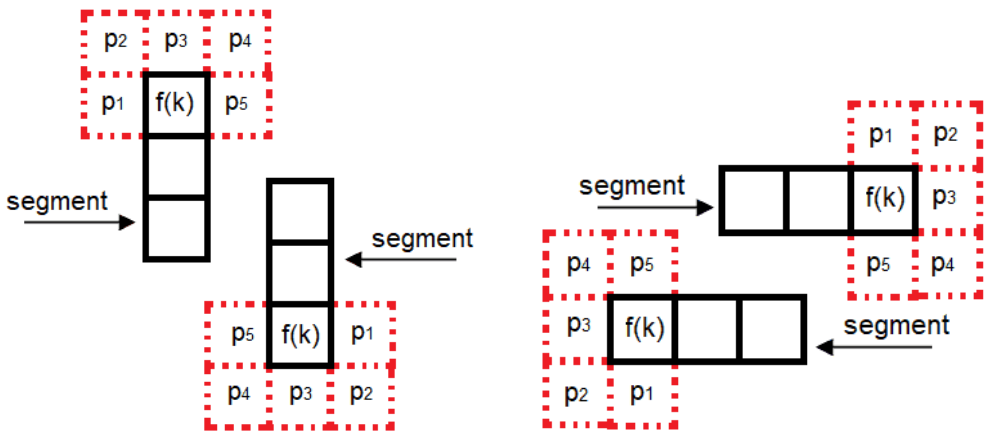
Algorithm 2: Determining the segment image $s(i,j)$ and segment vectors s_k

1. Initialize $s(i,j) = 0$ for all (i,j).
2. **For** each pair of coordinates (i,j) in the image:
   - (a) **If** $s(f_k) == 0$:
     - i. Initialize vector s_k as empty.
     - ii. Set $s(i,j) = 1$, add tuple (i,j) to vector s_k.
   - (b) **Else if** $s(p_n) == 0$:
     - i. **While** segment can grow:
       - A. **If** $s(p_n) == 0$ (where n is the grown direction being evaluated):
       - B. Find $n = \arg\min_n \mathrm{Crl}(n)$.
       - C. **If** $\mathrm{Crl}(n) \geq$ th:
       - D. Set $s(i,j) = 1$, add grown point (i,j) to vector s_k.
       - E. **Else**:
       - F. Break the loop.
3. **Return** the segment image $s(i,j)$ and the segment vectors $s_k\{i,j\}$.



(**a**) (**b**)

Figure 5. Image patterns for the feature extraction. On the one hand, a gradient map determines each potential growth direction based on a threshold th (**a**). On the other hand, vertical (red) and horizontal (green) gradient directions are utilized to define the growth direction (**b**).



(**a**) (**b**)

Figure 6. For each anchor f_k, there are five possible growth directions $p_1 \dots p_5$ and two direction patterns, i.e., vertical (**a**) and horizontal (**b**). Each direction is evaluated disabling all evaluated pixels, and if the most similar candidate meets a threshold value, it is considered a valid growth direction; for more details, please see Algorithm 2.

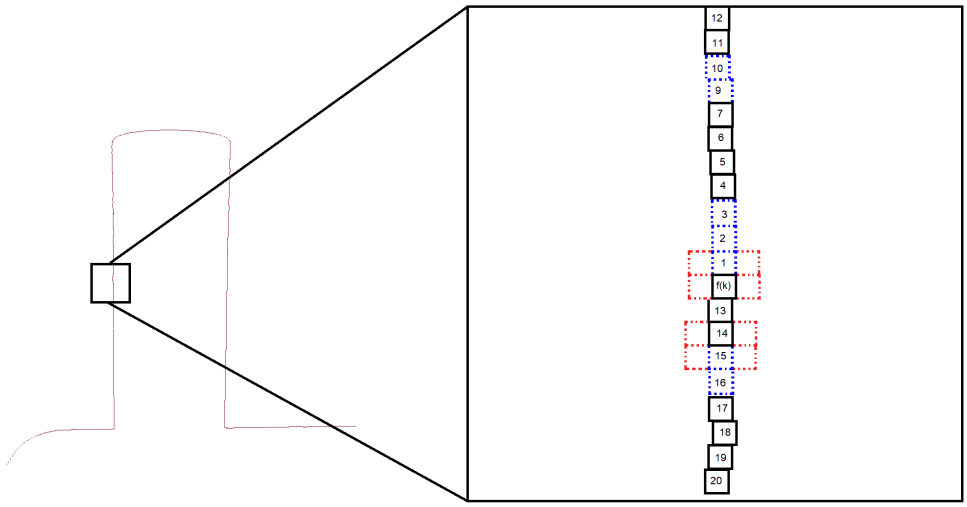


Figure 7. The feature extraction process let f_k be an anchor point. This point grows vertically first from f_k to the top $1, 2, \ldots, 12$ and then from f_k to the bottom $13, 14, \ldots, 20$.

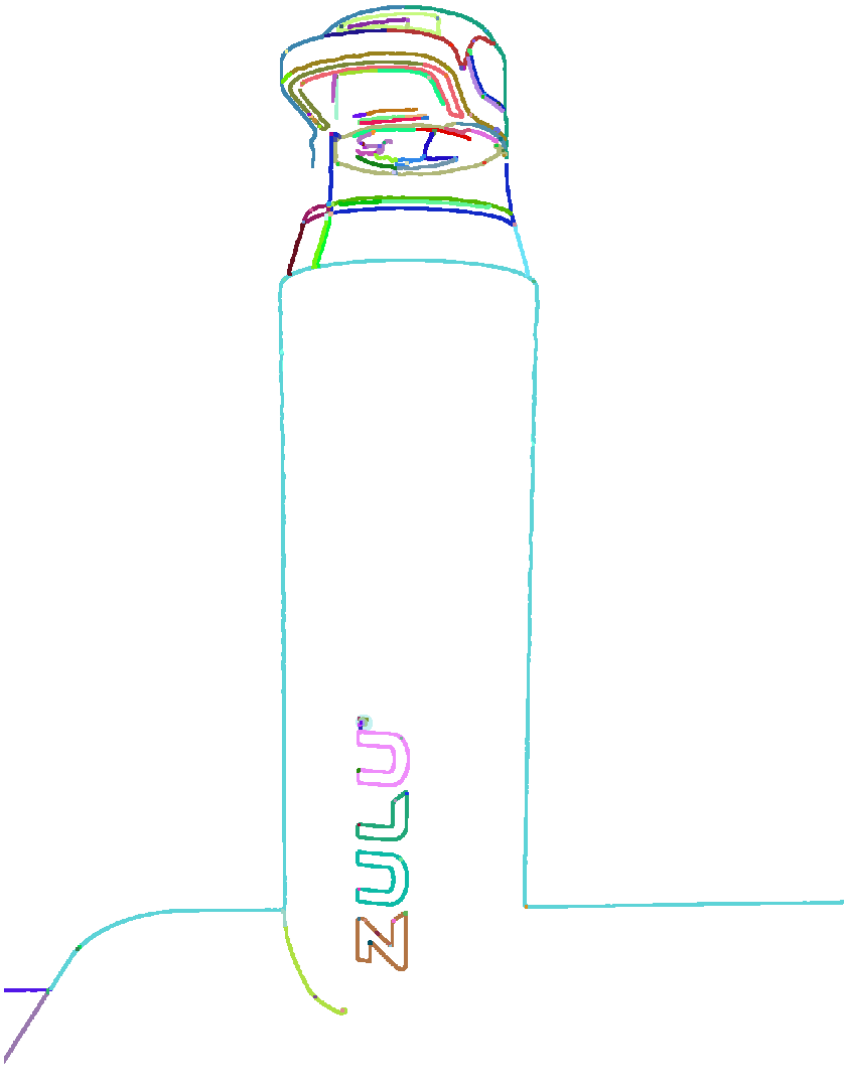
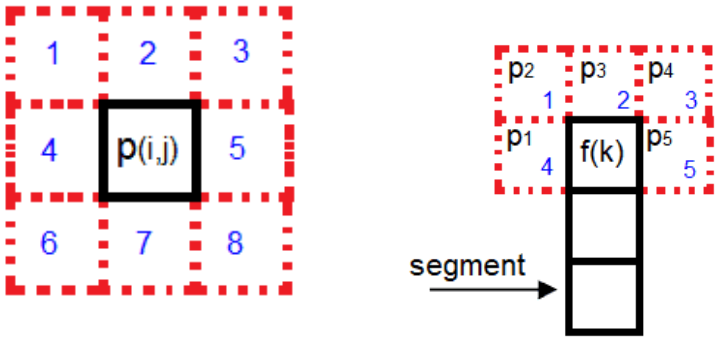


Figure 8. The extracted segments via the proposed algorithm. Each segment is represented by a different color.

3.2.3. The Directional Pixel Voting Descriptor (Feature Description)

Let $p(i,j)$ represent a pixel inside a segment (it could be an anchor f_k or any pixel added after the growing process, see Section 3.2.2). Each added pixel involves a global image direction, as illustrated in Figure 9a. There are eight possible image directions. Here, it is not possible to maintain our clockwise order, since at the global level the objective is to describe the shape of the segment. However, the global directions order can be superposed on the growth patterns presented in the previous section, as shown in Figure 9b. Let $d_1, d_2, \ldots, d_8$ represent the global image directions. A segment descriptor k (for the k-th segment) is defined as follows: $S_k\{d_1, d_2, \ldots, d_8, g_1, g_2, \ldots, g_8\}$, where d_n represents the count of pixels inside the segment grown in the i-th direction and is computed via Equations (4)–(6) and is initialized to zero, p_n is the global grown direction of the i-th pixel $s_k\{p(i,j)_1,\ p(i,j)_2,\ \ldots p(i,j)_i\}$ inside the k segment, see Algorithm 2, and N is the size of the segment vector. Moreover, $g_1, g_2, \ldots, g_8$ represent the normalized mean of the gradient values at each global grown direction. For this, we define Equations (7)–(9), where g_{xy} is the gradient value of the i-th pixel $p(i,j)$ inside the segment, $g_{xy}(i,j) = g_x(i,j) + g_y(i,j)$, see Section 3.2.1, $\frac{1}{|\mathrm{Max}(g_k)|}$ is the normalization factor, and M is the operator for the mean value. This process is illustrated in Algorithm 3.



(**a**) (**b**)

Figure 9. Global grown direction (**a**). There are eight possible image directions. Here, it is not possible to maintain our clockwise order, since at the global level, the objective is to describe the shape of the segment under different object rotations. However, for the implementation process, the global directions order can be superposed on the growth patterns presented in the previous section, as shown in (**b**).

$$S_k\{d_1\} = \sum_{i=1}^{N} d_1 = d_1 + 1 \text{ if } s_k(i) \text{ has } p_1 \text{ grown direction} \tag{4}$$

$$S_k\{d_2\} = \sum_{i=1}^{N} d_2 = d_2 + 1 \text{ if } s_k(i) \text{ has } p_2 \text{ grown direction} \tag{5}$$

$$\vdots$$

$$S_k\{d_8\} = \sum_{i=1}^{N} d_8 = d_8 + 1 \text{ if } s_k(i) \text{ has } p_3 \text{ grown direction} \tag{6}$$

$$S_k\{g_1\} = \frac{1}{|\text{Max}(g)|} \times \text{M}_{g_k>0}\left(g(i) = \sum_{i=1}^{N} |g_{xy}(s_k(i))| \text{ if } s_k(i) \text{ has } p_1 \text{ grown direction 0 otherwise}\right) \quad (7)$$

$$S_k\{g_2\} = \frac{1}{|\text{Max}(g)|} \times \text{M}_{g_k>0}\left(g(i) = \sum_{i=1}^{N} |g_{xy}(s_k(i))| \text{ if } s_k(i) \text{ has } p_2 \text{ grown direction 0 otherwise}\right) \quad (8)$$

$$\vdots$$

$$S_k\{g_8\} = \frac{1}{|\text{Max}(g)|} \times \text{M}_{g_k>0}\left(g(i) = \sum_{i=1}^{N} |g_{xy}(s_k(i))| \text{ if } s_k(i) \text{ has } p_8 \text{ grown direction 0 otherwise}\right) \quad (9)$$

Algorithm 3: Creation of the directional pixel voting descriptor

1. Initialize $g(i)$ as empty for $k = 1$ to 8.
2. **For** each tuple (i, j) in s_k:
   (a) **For** $n = 1$ to 8 *(where n are the grown directions, see Figure 9a)*:
      i. **If** $s_k(i, j)$ has p_n grown direction:
         A. $g(i) \leftarrow g(i) + |g_{xy}(s_k(i, j))|$.
      ii. **Else**:
         A. $g(i) \leftarrow g(i) + 0$.
3. **For** $n = 1$ to 8 *(where n are the grown directions, see Figure 9a)*:
   (a) Compute $S_k\{g_n\} = \frac{1}{|\text{Max}(g)|} \times$ Mean of $g(i)$ where $g(i) > 0$.
4. **Return** $S_k\{g_n\}$ for $n = 1$ to 8 *(where n are the grown directions, see Figure 9a)*.

3.3. Recognition

Let S_k be the descriptors for the template while S_q be the descriptors on the query (the real-world scene in which the object/place is being looked at). Comparisons between the S_k segment from template and the S_q segment in query are carried out by the Hamming distances of the descriptor components as follows:

$$\text{hammingDistance} = \sum_{i=1}^{n} |S_k - S_q| \quad (10)$$

where n is the number of components in the descriptor.

Of course, an all vs. all approach could be inconvenient in terms of processing speed and resources usage; to address this problem, we propose a look-up table approach based on the segment length, where each block represents a range of segment lengths. Thus, let S_k be the segment being matched and S_k inside the block 2; this segment is compared with each segment on the query table for the same block size. Of course, smaller length partitions will allow fast searches, since there will be only a few candidates for each size, but this could limit proper matching, since lengths between template and query could be a little different. On the other hand, huge length partitions will allow a proper match even with strong changes between template and query, but the search process will require high computational resources since a lot of candidates could be available. In practice, we recommend a partition as follows : $th_1 = 8, th_2 = 16, th_3 = 32, \cdots, th_{n-1} = 128$. The whole process is shown in Algorithm 4.

$$\text{Crl}(i) = \sum_{i=1}^{i=n} \text{hammingDistance}(S_k, S_i) \quad (11)$$

$$\text{match}(i) = \begin{cases} 1 & \text{if } \text{argMin}_i \text{Crl}(i) \leq \text{th}_{\text{match}} \\ 0 & \text{if otherwise} \end{cases} \quad (12)$$

where $\text{match}(i)$ represents the match status, being zero if the segment i was not found in the query or 1 otherwise. th_{match} is a similarity threshold which defines if a valid matching was found. We recommend values between 0.7 and 0.8. Finally, an object or scene is recognized only if a proper number of the i segments which define the object or scene were found. A proper number of matches could be greater than 70%. Of course, greater values could guarantee less recognition accuracy, but detection could not allow important changes (in terms of illumination, rotation, or scale) between template and query. On the other hand, lower values will allow easy detection with high changes between template and query but can generate false positives, i.e., wrong object or scene detections.

Algorithm 4: Object recognition

1. Initialize hammingDistance $= 0$.
2. Group all segments S_q into blocks based on their length:
   - (a) **For** each segment S_q:
     - i. **If** $S_{\text{length}}(S_q) \leq \text{th}_1$:
       - A. Assign S_q to block 1.
     - ii. **Else if** $S_{\text{length}}(S_q) \leq \text{th}_2$:
       - A. Assign S_q to block 2.
     - iii. **Else if** $\cdots$:
       - A. $\cdots$
     - iv. **Else**:
       - A. Assign S_q to block M.
3. **For** each segment S_k:
   - (a) Identify block b to which S_k belongs.
   - (b) **For** each segment S_q in block b:
     - i. hammingDistance $\leftarrow$ hammingDistance $+ |S_k - S_q|$.
4. **Return** hammingDistance.

4. Results

In this section, we present experimental results for the proposed algorithm compared with previous approaches. Since the use of standard datasets has become common in the computer vision area (for learning-based approaches), we looked for some proper datasets. However, since there is a lack of standardized datasets for image-based algorithms, previous works such as [13,14,27] used their own images, and this is the approach that we will follow here. Following this idea, we compare our algorithms with other canonical algorithms in the current literature, which have served as the basis for multiple works. These are SIFT [7], SURF [36], FAST [37], and ORB [9]. Multiple tests for object/scene detection and processing speed were carried out. For practical purposes, only some examples will be presented in this manuscript. In all cases, MatLab 2023b and a PC with an Intel(R) Core(TM) i9-9900K CPU 3.60 GHz, 32 GB RAM DDR3 were used (Intel, Santa Clara, CA, USA).

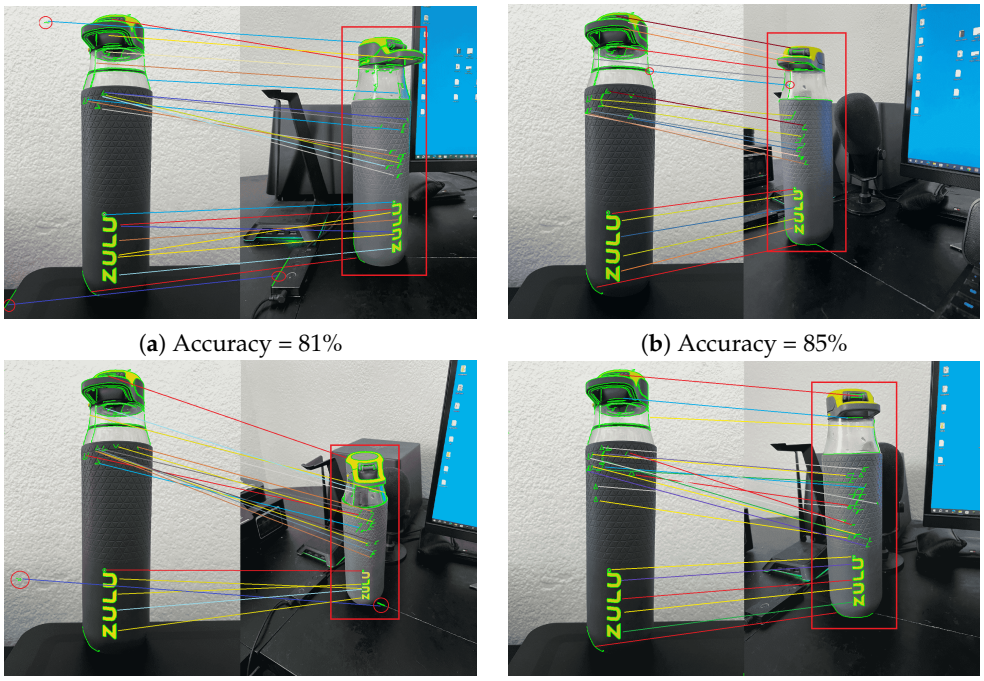
In Figure 10, results for object detections are shown. In all cases, accuracy is defined as the percentage of template features successfully detected. Since accuracy is higher than 70%, the object being recognized is found in all cases. Different rotations and scale variations between the template and query were tested. In all cases, accurate detection can be reached. In Figure 10a, two mismatches (outliers) were generated: in one case, the texture on the wall matched with a small pattern on the bottle. On the other hand, a line in the template was

matched with a line in the query that is different from the line in the template. There is no global spatial information, which could be expected, since, following our descriptor, both lines' matches present a similar orientation, length, and gradient (as both present similar contrasts between background and object, of which the feature owns). Furthermore, we can see that the "U" character was found two times. This is because we have not applied any non-maximal matched, and therefore, one feature in the template can be matched multiple times in the query. In this case, each "U" character in the template can be matched with each "U" character in the query, as expected. In 10b,c, only one outlier was generated each. Finally, no outliers are presented in 10d. In all cases, the square/mesh pattern on the bottle generated mismatches, but again, given our algorithm undertakes feature matching based on shape and color contrast against the features (gradients), it is expected that repetitive patterns can generate wrong matches. These problems can be addressed by adding global information so that each feature can be the same in terms of gradient or shape but different, for example, with respect to the Euclidean distance to the origin. Of course, this is only a hypothesis and will be explored in future work. In any case, these results demonstrate, at a high level, relatively accurate detection for object recognition. Regarding performance under complex visual variations in In Figure 11, results for object detection are shown. In all cases, accuracy is defined as the percentage of template features that are successfully detected. Accuracy higher than 70% is reached, and therefore, accurate/robust detection is reached for objects. In Figure 11a, changes in rotation and objects with similar shapes were tested. In all cases, accurate detection can be reached. Even with rotations such as the one presented in the black cylinder and the wood box, our descriptor can properly match these objects. This demonstrates the rotational invariance of our descriptor. In Figure 11b, the complex texture scenario was tested, which demonstrated the robustness of the proposed method; shapes such as circles, cuboids, and so on and complex segments such as a book cover were correctly detected. Again, rotational invariance is demonstrated by the algorithm's ability to properly match segments inside the book cover.

In Figure 12, the performance for other methods in the literature, in particular, the original SIFT [7] with non-maxima suppression (i.e., one feature on the template (left side), which can have multiple matches in the query (right side)) and original SURF [36], is shown. As can be seen, by comparing only feature points or blobs, low performance is achieved. In both cases, several mismatches or multiple matches for the same feature are generated. This demonstrates the low performance that point information can generate. On the other hand, our approach in Figures 10 and 11, which is based on shape and gradient, involves higher robustness for the matching process. This is shown in Table 1, in which average accuracy values by considering multiple test for several templates are presented.

Table 1. Mean accuracy for the proposed algorithm compared with previous approaches in the literature.

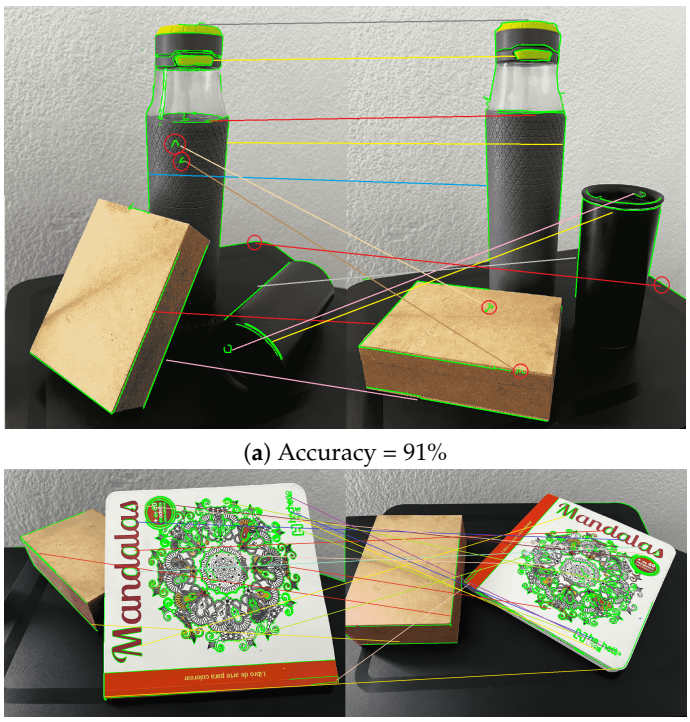
Method	Accuracy
SIFT [7]	43%
SURF [36]	69%
FAST [37]	60%
ORB [9]	67%
This work	81%



(**a**) Accuracy = 81% (**b**) Accuracy = 85%

(**c**) Accuracy = 85% (**d**) Accuracy = 91%

Figure 10. Object detection by the proposed method. Different rotations between template and query (**a**,**b**,**d**) and scale variations (**c**) were tested. In all cases, accurate detection can be reached. Mismatches are marked as red circles.



(**a**) Accuracy = 91%

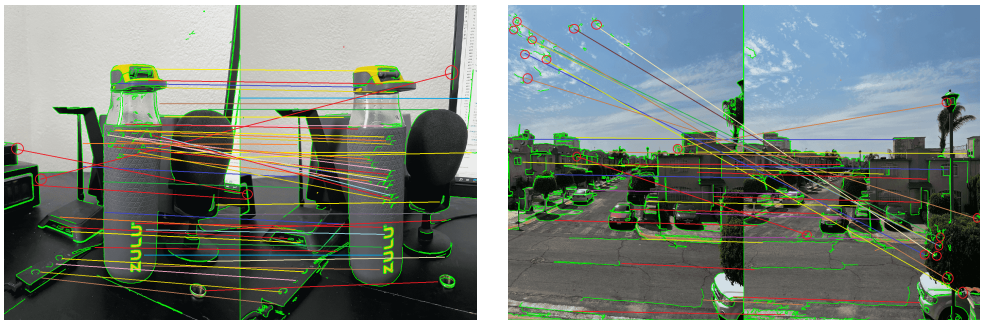
(**b**) Accuracy = 71%

Figure 11. Object detection by the proposed method under complex visual variations/rotations (**a**) and complex texture objects (**b**). In both cases, high accuracy is reached. Mismatches are marked as red circles.

(**a**) Accuracy = 37% (**b**) Accuracy = 71%

Figure 12. Object detection by other algoritmms in the literature. (**a**) original SIFT [7], (**b**) original SURF. [36].

In Figures 13 and 14, results for scene recognition are shown. Again, when compared to our approach (Figure 13) with the original SIFT (Figure 14), our approach clearly outperforms previous work. In our work, most important mismatches are generated for some cloud patterns in the template which match the tree patterns in the query. Again, this is as expected, since matched patterns have the same shape and gradient. On the other hand, SIFT generated several mismatches, and in this case, it could be difficult to determine if the scene was recognized or not.



(**a**) Accuracy = 87% (**b**) Accuracy = 73%

Figure 13. Results for scene recognition: (**a**) Indoor scenario: only two mismatches are generated. In both cases, features that were matched present similarities in terms of shape and gradient. (**b**) Outdoor scenario: the most important mismatches are generated for some cloud patterns in the template, which match the tree patterns in the query. Again, this is as expected, since matched patterns have the same shape and gradient.

Finally, in Tables 2 and 3, processing speed comparisons are presented. Our approach, which consists of complex features (compared with points which are the features used by previous approaches), requires more processing time. However, even with this complexity, our algorithms outperform the original ORB. Considering that algorithms like ORB have been successfully used in real-time applications such as SLAM [38], robotics [39], and so on, we consider our approach can be successfully used. Note that our algorithm was implemented in MATLAB, without hardware acceleration or advanced programming patterns, while the other algorithms have been highly improved in terms of implementation in order to reach optimal processing speed. We believe our approach can be improved in terms of implementation, and this should increase the processing speed. This will be addressed in future work.

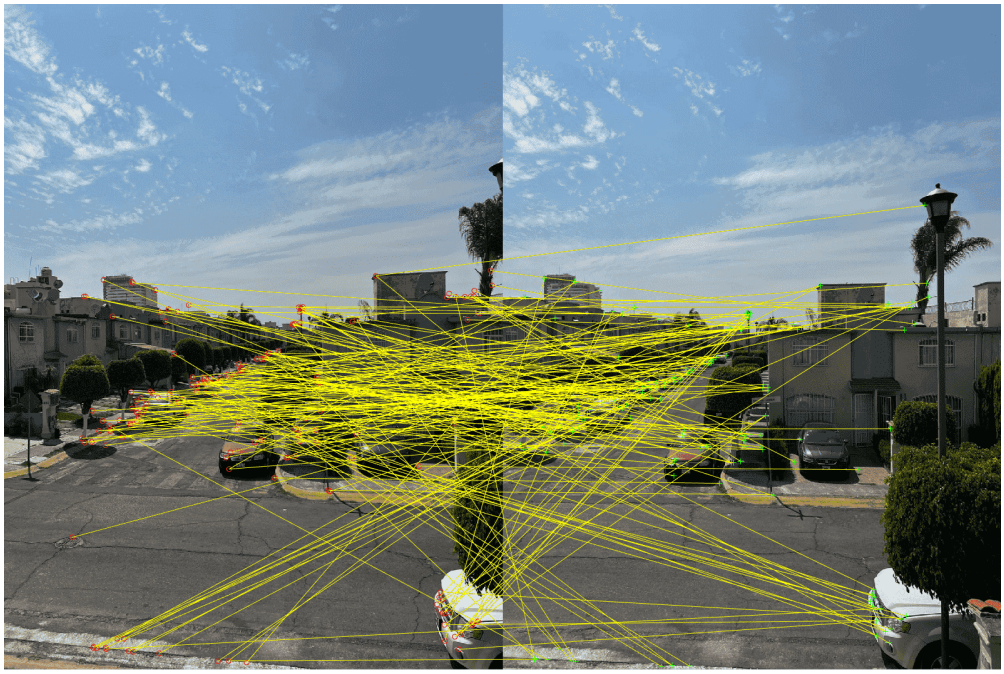


Figure 14. Original SIFT [7]. Results for scene recognition showed a lot of mismatches were generated, and in this case, it could be difficult to determine if the scene was recognized or not. Accuracy = 57%.

Table 2. Processing speed for the proposed algorithm compared with previous approaches in the literature. Input image resolution 2160 × 1080 (Full HD).

Method	Processing Time (fps)
SIFT [7]	1.0044
SURF [36]	3.6597
FAST [37]	3.9117
ORB [9]	0.3820
This work	0.5726

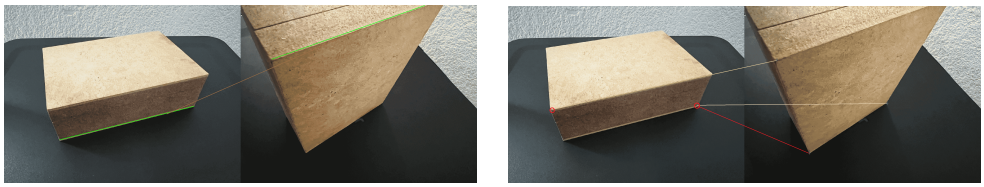
Table 3. Processing speed for the proposed algorithm compared with previous approaches in the literature. Input image resolution 1080 × 540 (HD).

Method	Processing Time (fps)
SIFT [7]	3.3245
SURF [36]	11.4322
FAST [37]	11.4230
ORB [9]	1.0328
This work	2.0012

Limitations: Since the direction in the proposed Directional Pixel Voting Descriptor is defined in the 2D image pixel coordinate, rotations which deforms or occlude primitive shapes in template are not supported, as shown in Figure 15a. However, even classical methods such as SIFT or SURF suffer from this in the case of textureless objects Figure 15b.

Asa final discussion, experimental results demonstrate the effectiveness and robustness of our proposed approach for object and scene recognition tasks. We have demonstrated that our method outperforms previous approaches, particularly in scenarios with variations in rotation, scale, and texture. Despite the increased computational complexity of our method compared to traditional point-based approaches, our algorithms achieve promising performance, laying down the basis for future improvements and applications in real-world applications. Additionally, while our implementation in MATLAB provides a solid implementation, future optimizations and hardware accelerations are expected

to further improve processing speed and efficiency. We believe that our work allows for improvements such as additional constraints for the descriptor previously laid down, non-maximal suppression methods (to avoid multiple matches for some features in the template and query), and so on.



(**a**) Proposed: Accuracy = 11% (**b**) SIFT: Accuracy = 21%

Figure 15. Object detection under shape occlusion. (**a**) Rotations which deform or occlude primitive shapes in template are not supported. (**b**) Even classical methods such as SIFT suffer from this scenario by looking for a textureless object.

5. Conclusions

The importance of object detection under several applications, including surveillance, autonomous navigation, and augmented reality, cannot be overstated. While AI-based approaches like Convolutional Neural Networks (CNNs) have demonstrated high performance in object detection, there are limitations for unknown objects being detected. Feature-based methods such as SIFT, SURF, and ORB support unknown object detection but have limitations under visual variations. In response, this work presented a novel edge-based object/scene recognition technique. Our proposed directional pixel voting descriptor, based on image segments, represents a significant advancement in this domain. Experimental results demonstrated high performance compared with previous works in the current literature. In future work, improvements such as additional constraints for the descriptor previously laid down, non-maximal suppression methods (to avoid multiple matches for some features in the template and query), optimizations, and hardware accelerations will be implemented in order to improve processing speed and efficiency.

Author Contributions: Conceptualization, Investigation: A.A.-G. Validation and Writing—Original Draft: A.M.S. and J.A.d.J.O.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ji, Y.; Zhang, H.; Zhang, Z.; Liu, M. CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances. *Inf. Sci.* **2021**, *546*, 835–857. [CrossRef]
2. Liu, G.; Hu, Y.; Chen, Z.; Guo, J.; Ni, P. Lightweight object detection algorithm for robots with improved YOLOv5. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106217. [CrossRef]
3. Zhao, X.; Cheah, C.C. BIM-based indoor mobile robot initialization for construction automation using object detection. *Autom. Constr.* **2023**, *146*, 104647. [CrossRef]
4. Napier, T.; Lee, I. Using mobile-based augmented reality and object detection for real-time Abalone growth monitoring. *Comput. Electron. Agric.* **2023**, *207*, 107744. [CrossRef]
5. Nguyen, D.D.; Nguyen, D.T.; Le, M.T.; Nguyen, Q.C. FPGA-SoC implementation of YOLOv4 for flying-object detection. *J. Real-Time Image Process.* **2024**, *21*, 63. [CrossRef]
6. Zhu, H.; Huang, Y.; Xu, Y.; Zhou, J.; Deng, F.; Zhai, Y. Unmanned aerial vehicle (UAV) object detection algorithm based on keypoints representation and rotated distance-IoU loss. *J. Real-Time Image Process.* **2024**, *21*, 58. [CrossRef]

7. Lowe, G. Sift-the scale invariant feature transform. *Int. J.* **2004**, *2*, 2.
8. Du, G.; Su, F.; Cai, A. Face recognition using SURF features. In Proceedings of the MIPPR 2009: Pattern Recognition and Computer Vision, Yichang, China, 30 October–1 November 2009 ; SPIE: Bellingham, WA, USA, 2009; Volume 7496, pp. 593–599.
9. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; IEEE: Washington, DC, USA, 2011; pp. 2564–2571.
10. Doménech-Asensi, G.; Zapata-Pérez, J.; Ruiz-Merino, R.; López-Alcantud, J.A.; Díaz-Madrid, J.Á.; Brea, V.M.; López, P. All-hardware SIFT implementation for real-time VGA images feature extraction. *J. Real-Time Image Process.* **2020**, *17*, 371–382. [CrossRef]
11. Wei, S.; Li, Z. An RGB-D SLAM algorithm based on adaptive semantic segmentation in dynamic environment. *J. Real-Time Image Process.* **2023**, *20*, 85. [CrossRef]
12. Xie, X.; Wu, D.; Xie, M.; Li, Z. GhostFormer: Efficiently amalgamated CNN-transformer architecture for object detection. *Pattern Recognit.* **2024**, *148*, 110172. [CrossRef]
13. Tan, Z.; Fan, W.; Kong, W.; Tao, X.; Xu, L.; Xu, X. An improved ORB-GMS image feature extraction and matching algorithm. In Proceedings of the 2023 IEEE International Conference on Robotics and Biomimetics (ROBIO), Koh Samui, Thailand, 4–9 December 2023; IEEE: Washington, DC, USA, 2023; pp. 1–6.
14. Rathour, S.S.; Ito, T.; Machii, K.; Bando, M.; Shimizu, T. ORB Keypoint Based Flying Object Region Proposal for Safe & Reliable Urban Air Traffic Management. In Proceedings of the 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), Bilbao, Spain, 24–28 September 2023; IEEE: Washington, DC, USA, 2023; pp. 1263–1268.
15. Mogaka, O.M.; Zewail, R.; Inoue, K.; Sayed, M.S. TinyEmergencyNet: A hardware-friendly ultra-lightweight deep learning model for aerial scene image classification. *J. Real-Time Image Process.* **2024**, *21*, 51. [CrossRef]
16. Xie, L.; Lee, F.; Liu, L.; Kotani, K.; Chen, Q. Scene recognition: A comprehensive survey. *Pattern Recognit.* **2020**, *102*, 107205. [CrossRef]
17. Daou, A.; Pothin, J.B.; Honeine, P.; Bensrhair, A. Indoor Scene Recognition Mechanism Based on Direction-Driven Convolutional Neural Networks. *Sensors* **2023**, *23*, 5672. [CrossRef] [PubMed]
18. López-Cifuentes, A.; Escudero-Vinolo, M.; Bescós, J.; García-Martín, Á. Semantic-aware scene recognition. *Pattern Recognit.* **2020**, *102*, 107256. [CrossRef]
19. Rafique, A.A.; Gochoo, M.; Jalal, A.; Kim, K. Maximum entropy scaled super pixels segmentation for multi-object detection and scene recognition via deep belief network. *Multimed. Tools Appl.* **2023**, *82*, 13401–13430. [CrossRef]
20. Bose, D.; Hebbar, R.; Somandepalli, K.; Zhang, H.; Cui, Y.; Cole-McLaughlin, K.; Wang, H.; Narayanan, S. Movieclip: Visual scene recognition in movies. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–7 January 2023; pp. 2083–2092.
21. Lv, G.; Dong, L.; Zhang, W.; Xu, W. Region-based adaptive association learning for robust image scene recognition. *Vis. Comput.* **2023**, *39*, 1629–1649. [CrossRef]
22. Liu, Z.; Liu, L.; An, L. Intelligent Home Scene Recognition Based on Image Processing and Internet of Things. *Trait. Signal* **2023**, *40*, 1171. [CrossRef]
23. Pathak, S.; Doegar, A. On Real-Time Object Recognition by Single Image Dehazing Method Using Deep Learning Approach. In Proceedings of the 2023 3rd International Conference on Advancement in Electronics & Communication Engineering (AECE), Ghaziabad, India, 23–24 November 2023; IEEE: Washington, DC, USA, 2023; pp. 476–479.
24. Liang, M.; Hu, X. Recurrent convolutional neural network for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3367–3375.
25. Akçay, S.; Kundegorski, M.E.; Devereux, M.; Breckon, T.P. Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; IEEE: Washington, DC, USA, 2016; pp. 1057–1061.
26. Shaha, M.; Pawar, M. Transfer learning for image classification. In Proceedings of the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; IEEE: Washington, DC, USA, 2018; pp. 656–660.
27. Luo, S.; Mou, W.; Althoefer, K.; Liu, H. Novel tactile-sift descriptor for object shape recognition. *IEEE Sens. J.* **2015**, *15*, 5001–5009. [CrossRef]
28. Bansal, M.; Kumar, M.; Kumar, M.; Kumar, K. An efficient technique for object recognition using Shi-Tomasi corner detection algorithm. *Soft Comput.* **2021**, *25*, 4423–4432. [CrossRef]
29. Rashid, M.; Khan, M.A.; Sharif, M.; Raza, M.; Sarfraz, M.M.; Afza, F. Object detection and classification: A joint selection and fusion strategy of deep convolutional neural network and SIFT point features. *Multimed. Tools Appl.* **2019**, *78*, 15751–15777. [CrossRef]
30. Kwitt, R.; Vasconcelos, N.; Rasiwasia, N. Scene recognition on the semantic manifold. In Proceedings of the Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; Proceedings, Part IV 12; Springer: Berlin/Heidelberg, Germany, 2012; pp. 359–372.
31. Zhang, H.; Zheng, Q.; Dong, B.; Feng, B. A financial ticket image intelligent recognition system based on deep learning. *Knowl.-Based Syst.* **2021**, *222*, 106955. [CrossRef]

32. Li, Y.; Zhang, Z.; Cheng, Y.; Wang, L.; Tan, T. MAPNet: Multi-modal attentive pooling network for RGB-D indoor scene classification. *Pattern Recognit.* **2019**, *90*, 436–449. [CrossRef]
33. Prewitt, J.M. Object enhancement and extraction. *Pict. Process. Psychopictorics* **1970**, *10*, 15–19.
34. Vale, A.; Ucchesu, M.; Di Ruberto, C.; Loddo, A.; Soares, J.; Bacchetta, G. A new automatic approach to seed image analysis: From acquisition to segmentation. *arXiv* **2020**, arXiv:2012.06414.
35. Mičušík, B.; Hanbury, A. Automatic image segmentation by positioning a seed. In Proceedings of the European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 468–480.
36. Bay, H.; Tuytelaars, T.; Van Gool, L. Surf: Speeded up robust features. In Proceedings of the Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; Proceedings, Part I 9; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
37. Viswanathan, D.G. Features from accelerated segment test (fast). In Proceedings of the 10th Workshop on Image Analysis for Multimedia Interactive Services, London, UK, 6–8 May 2009; pp. 6–8.
38. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
39. Lv, Q.; Lin, H.; Wang, G.; Wei, H.; Wang, Y. ORB-SLAM-based tracing and 3D reconstruction for robot using Kinect 2.0. In Proceedings of the 2017 29th Chinese Control and Decision Conference (CCDC), Chongqing, China, 28–30 May 2017; IEEE: Washington, DC, USA, 2017; pp. 3319–3324.