

# Comparative Analysis of Interest Point Detectors Algorithms on Robotic Operative System

Francisco Javier Valdepeña Rivera<sup>1</sup>, Dante Mújica Vargas<sup>2</sup>, Miguel Ángel Ruíz<sup>1</sup>

<sup>1</sup> Universidad Politécnica del Estado de Morelos  
Jiutepec, Morelos, Mexico

<sup>2</sup> CENIDET-TecNM, Depto. Ciencias computacionales  
Cuernavaca, Morelos, Mexico

{vrfo150252, mruiz}@upemor.edu.mx, dantemv@cenidet.edu.mx

**Abstract.** Applications of robotic vision have had great advances within the artificial intelligence through the processing of images, as well as the automated systems (robots). A comparative analysis of some interest point detector algorithms will be performed, the next analysis will be about a robotic operating system called ROS, by means of a 2D object detector system. For this purpose, a physical architecture will be carried out to carry out the experimentation within a controlled work environment, in order to demonstrate which algorithm will work best in the future for the development of object recognition systems, implementing this system on Robots.

**Keywords:** detector, interest points, ROS, artificial vision.

## 1 Introduction

The artificial vision in these days has been realized through systems that detect characteristics of the objects and classify them by means of neural networks which decide what type of object is, by means of a comparison of weights or learning units which are updated every that an iteration within algorithm.

Deep learning is based on a set of algorithms for machine learning that attempts to model high-level abstractions in data using architectures composed of multiple non-linear transformations [1].

But there is also the way to segment artificial vision in various stages which are essential to divide the number of processes into more concise processes, such as those shown in Figure 1. This is to perform optimal recognition in robotic systems, contrary to only have neural networks for recognition what it is to load a neural network with points or characteristics that are a stage of artificial vision can achieve, that the recognition is more precise and with a minimum of variety to the conditions in which find the object.

In this way it is necessary to take into account which are those algorithms that can extract the most important characteristics to be able to make a good detection, so after having said information the recognition is achieved instantaneously with a neural network or applying the concept of deep learning.

It is proposed to analyze and compare object detection algorithms under rotation, translation and light conditions to conclude which of them is the best for the task and

thus be able to have the analysis data and say which one to use in the future for object recognition.

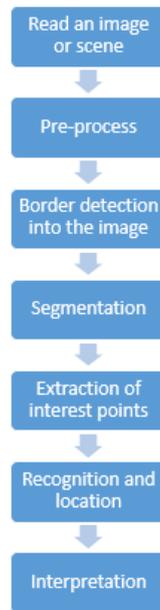


Fig. 1. Stages of artificial vision.

## 2 Theoretical Framework

### 2.1 Interest Point Detector and Feature Descriptor

Many algorithms that focus on robotic vision are based on the location of the points of interest or key points in each image, as well as calculating a description of the characteristic of the pixel region surrounding the point of interest. This contrasts with methods such as correlation, where a larger rectangular pattern is passed over an image within the pixel intervals and the correlation is measured at each location.

The point of interest is the anchor point, as it often provides the attributes of scale, rotation and illumination invariance for the descriptor.

A point of interest can be composed of several types of corners, edges and maxima, as shown in Figure 2.

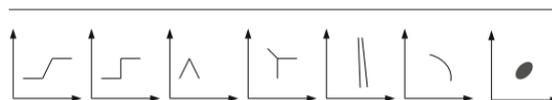


Fig. 2. Interests point [2].

## 2.2 Interest Points Detectors

### FAST

It is a corner detector that is characterized by producing very stable relevant points. This method belongs to the category AST (Accelerated Segment Test), It is the most efficient computationally corners detector that exists so far. This algorithm works through circular windows centered on each of the points of the image. For a point to be detected as a corner there must be a minimum number of points within its window that are a threshold amount more clear than the center or another threshold amount darker than the center [4].

The segment test criterion operates considering a circle of 16 pixels' candidate for the return p. The original detector classifies p as a corner for automatic detection of high-speed corners 5 if there is a set of n contiguous pixels in the circle that are all brighter than the intensity of the candidate pixel  $I_p$  plus a threshold t, or everything else dark than  $I_p - t$ , as illustrated in Figure 3.

n It was chosen to be twelve because it supports a high-speed test that can be used to exclude a large number of non-corners: the test examines only the four pixels in 1, 5, 9 and 13 (the four directions of the compass). This detector itself exhibits high performance, but there are several weaknesses:

- Do not reject so many candidates for  $n < 12$ .
- The choice of pixels is not optimal because their efficiency depends on the order of the questions and the distribution of the appearances of the corners.
- The results of the high-speed tests are discarded.
- Multiple features adjacent to each other are detected.

### SIFT

The SIFT algorithm is an artificial vision algorithm originally published by David Lowe in 1999, which is responsible for extracting distinctive features from grayscale images. It is used primarily for reconnaissance tasks. The selected points are invariant to scale, rotation and partially invariant to changes of point of view and illumination. The extraction algorithm is based on the location of the maximum and minimum of a Gaussian difference applied in the space of scales. This feature extraction allows the algorithm to be executed in a hierarchical way so that the characteristics are calculated from a pyramid of images with sampling between each level [2].

#### Detection of extremes in space-scale

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma). \quad (1)$$

Here:

$$L(x, y, \sigma) = I(x, y) * G(x, y, \sigma), \quad (2)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}. \quad (3)$$

### Exact location key point

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x. \quad (4)$$

Here D and its derivatives are always evaluated at the point in question and  $x = (x, y, \sigma)^T$  is the position relative to it.

Deriving the previous approximation and equaling it to zero, we obtain:

$$\underline{x} = -\left(\frac{\partial^2 D^{-1}}{\partial x^2} * \frac{\partial D}{\partial x}\right). \quad (5)$$

The previous formula is replaced with the first Taylor series of grade 2, to obtain the local maximum value:

$$D(\underline{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \underline{x}, \quad (6)$$

$$Traza(H) = \frac{\partial^2 D}{\partial x^2} + \frac{\partial^2 D}{\partial y^2} = \alpha + \beta, \quad (7)$$

$$Det(H) = \frac{\partial^2 D}{\partial x^2} * \frac{\partial^2 D}{\partial y^2} = \alpha * \beta. \quad (8)$$

Let  $\alpha = r.\beta$ , the condition is reduced to:

$$\frac{Traza(H)^2}{Det(H)} < \frac{(r+1)^2}{r}. \quad (9)$$

See that as the relation r between both eigenvalues increases, so does the relation between the square of the trace of the Hessian matrix and its determinant.

### Orientation assignment

By assigning an orientation to each point of the image based on local characteristics of the image, the key points can be described relative to these orientations and in this way achieve invariant characteristics to the rotations.

Where to calculate the gradient m (x, y) is:

$$m(x, y) = \sqrt{(\Delta L_x)^2 + (\Delta L_y)^2}. \quad (10)$$

Here m (x, y) equals to:

$$m(x, y) = \sqrt{[L(x+1, y) - L(x-1, y)]^2 + [L(x, y+1) - L(x, y-1)]^2}. \quad (11)$$

Where to calculate the phase  $\theta$  (x, y), we use:

$$\theta(x, y) = \left( \frac{\Delta L_y}{\Delta L_x} \right). \tag{12}$$

Here  $\theta(x, y)$  equals:

$$\theta(x, y) = \left( \frac{L(x, y + 1) - L(x, y - 1)}{L(x + 1, y) - L(x - 1, y)} \right). \tag{13}$$

### SURF

The SURF algorithm consists of the following phases:

- Detection of interest point.
- Assignment of orientation to interest point.
- Calculation of descriptors associated with interest point [2].

#### Detection of Interest point

$$H(p, \sigma) = [L_{xx}(p, \sigma) \ L_{xy}(p, \sigma) \ L_{xy}(p, \sigma) \ L_{yy}(p, \sigma)]. \tag{14}$$

Where  $L_{xx}(x, \sigma)$  is the second-order convolution of the Gaussian,  $\partial^2/(\partial x^2) g(\sigma)$  with the image  $I$  at point  $x$ , and similarly for  $L_{xy}(p, \sigma)$   $L_{yy}(p, \sigma)$ .

The approximations of the partial derivatives are denoted as  $D_{xx}$ ,  $D_{xy}$  and  $D_{yy}$ , and the determinant is calculated as follows:

$$\det \det (H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2. \tag{15}$$

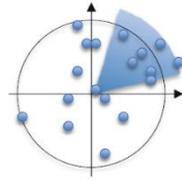
#### Assignment of orientation to interest point

The first step to grant the mentioned orientation consists of the calculation of the answer of Haar in both directions  $x$  and  $y$ .



Fig. 3. Calculation of the answer Haar. Black: -1 White: +1 [2].

After having made all these calculations, integral images are used again to proceed to the filtering through the Haar masks and thus obtain the answers in both directions. Only 6 operations are necessary to obtain the answer in the  $x$  and  $y$  directions. Once the wavy responses have been calculated, they are weighted by a Gaussian value of  $\sigma = 2.5s$  centered on the point of interest.



**Fig. 4.** The sliding sector window used in SURF to calculate the dominant orientation of HAAR [2].

### Calculation of descriptors associated with points of interest

In summary, each of the sub regions is represented by a vector of components:

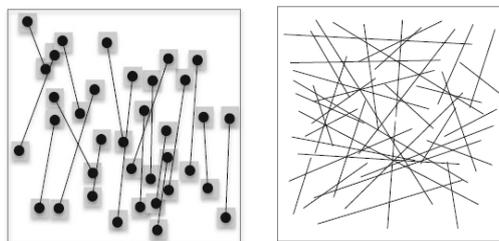
$$v = \left( \sum \partial x, \sum \partial y, \sum |\partial x|, \sum |\partial y| \right). \quad (16)$$

### ORB

ORB is based in part on the BRIEF descriptor, thus the name Oriented Brief, since ORB adds orientation to the BRIEF method and provides other improvements as well.

In BRIEF, as shown in Fig. 5, the sample points are specified in a random distribution pattern based on a Gaussian distribution about the center point within the 31 x 31 patch region; the chosen number of sample points is 256. Selected sample point-pairs are compared to each other to form the binary descriptor vector. The value of each point is calculated via an integral image method to smooth a 5 x 5 region into the point value.

The ORB pattern shown in Fig. 5 is based on choosing point-pairs that have high statistical variance within a bounding 31 x 31 image patch, where the smaller 5 x 5 gray image patch regions are centered at the chosen interest points. Then each 5 x 5 region is smoothed using an integral image method to yield a single value for the point [2].



**Fig. 5.** (Left) An ORB style pattern at greatly reduced point pair count resolution, using < 32 points instead of the full 256 points (Right). A BRIEF style pattern using randomized point-pairs [2].

It is a collection of tools, libraries and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

### The computation graph level

In the Figure 6 shows the computation peer-to-peer of the ROS process.

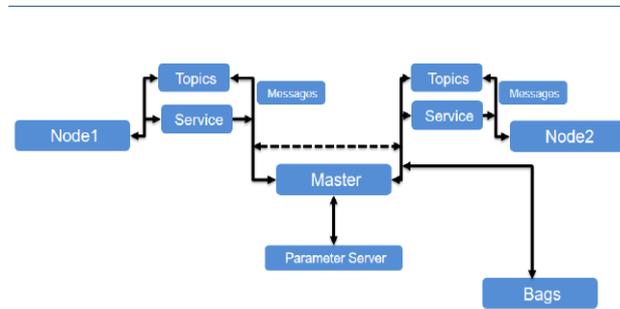


Fig. 6. Computation graph levels [3].

## 3 Methodology

This section explains how the system is working. The interaction begins when the user enters the system the find\_object\_2d, which in turn has the folder configuration that ROS manages, that is, the system itself controls the hardware of both the webcam and the computer that functions as the server of the own system, that is why there is a direct and parallel interaction with the system and the devices, working together it receives the images taken from the test object in order to have the data that finally arrive at the find\_object\_2d system and are processed with the algorithms SIFT, SURF, FAST, ORB which will detect if the camera captures the object or not, the architecture of the system is shown in figure 6.

### 3.1 Functionality of the System the Find\_object\_2d

The find\_object\_2d system works from the communication of the webcam node which is in the variable /usb\_cam/image\_raw, it is necessary to have installed the webcam driver compatible with ROS, if you do not have the driver, you must use the following command [3]:

```
$ sudo apt-get install v4l-utils
```

Once having the webcam driver, the next thing is to start the communication of nodes within ROS, as shown in the figure 7:

```
$ roscore
```

Plug your USB camera into your PC, and launch the ROS usb\_cam driver:

```
$ roslaunch usb_cam usb_cam-test.launch
```

Then the next step is to run the object detector node. The following command will start the object detector node:

```
$ rosrn find_object_2d find_object_2d image: =/usb_cam/image_raw
```

Finally, after inserting the above commands without any problem, the detector system will show the following window as shown in figure 7.

The window is divided into 3 panels.

In the first panel the images of the events captured by the camera are inserted and those images will be compared with the image in panel 2 if they are in both panels the same object the algorithm must detect if there is a match object.

In the third panel are the basic configurations of the detector system. In that panel the different algorithms will be inserted, to perform the comparative analysis of interest point detectors, the option to perform the insertion of the algorithm is Feature 2D.

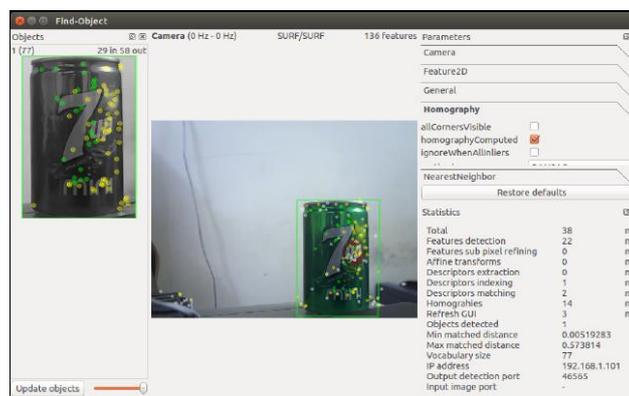


Fig. 7. Main window of the The\_find\_object 2d system [3].

### 3.2 Considerations

To perform the comparative analysis of the different algorithms, the following hardware and software points were taken into account.

#### Hardware

- With a computer with the following minimum characteristics.
  - o Intel processor with a speed of 2 to 3 GHz.
  - o RAM memory with a minimum capacity of 2 GB, recommended 4 GB.
  - o Hard disk with a minimum storage capacity of 500 GB.
  - o Have common input / output peripherals (mouse, monitor, keyboard).
- Count a digital camera with a minimum resolution of 640 x 480 pixels with USB input.
- (Optional) Infrared sensor with VGA camera (Kinect Xbox). Said device will be implemented to calculate distances between the object and the image receiver if the system requires it.

#### Software

- Linux distribution operating system recommended an Ubuntu LTS version 16.04.
- Activate updates of Ubuntu LTS 16.04, to update devices drivers
- Install ROS Kinect kame.

## 4 Obtained Results

**Table 1.** Characteristics of a controlled environment in the experimentation test.

Change	1	2	3	4	5
Illumination	75%	75%	75%	40%	40%
Angle of rotation	0°	0°	45°	0°	0°
Number of objects	1	3	1	1	3

**Table 2.** Analysis of points by pixels in the SIFT and SURF algorithms.

Objects	SIFT			SURF		
	Average	max	min	average	max	min
One object	95.065%	98.571%	81.904%	87.582%	97.619%	81.008%
Three objects	90.906%	98.571%	64.577%	91.482%	97.058%	85.401%
A rotated object	87.197%	97.744%	67.057%	83.66%	95.555%	60.191%

In the table 2 show data of points by pixel that means the interest points that every algorithm, that is to say the quotient of the interest points on all points detected.

In section of the algorithm SURF where the average is 91.482% shows an increment with regard to the algorithm SIFT is for this reason that the algorithm SURF detects better than algorithm SIFT. Every point by pixel is a right matching that means interest point and finally detected objects.

A characteristic of the algorithm SIFT is the quantity of points by pixel to detect objects and it's not the more important to detect a matching or interest point.

The implementation in the algorithm SURF about of points eliminated with a condition of maximum between scales neighbors change the maximum interpolating between scale and the octave position.

**Table 3.** Analysis of points by pixels in the FAST and ORB algorithms.

Objects	FAST			ORB		
	average	max	min	average	max	min
One object	2.352%	3.253%	0.216%	75.370%	85.106%	59.788%
Three objects	1.177%	3.770%	0.0714%	73.635%	82.954%	57.971%
A rotated object	0.604%	1.449%	0.0907%	74.378%	82.954%	61.111%

In the table 3 show data of points by pixel of FAST and ORB algorithm to detect objects in this proof we observe interesting results although the FAST algorithm detects a big quantity of data or points by pixel, the percent to detect interest points is low because is more sensible by the conditions environment we lost the umbral and this situation changes the result of matching.

Another example was ORB algorithm because if you change the rotation or illumination on the object, you have a partial matching that is to say in the next 3 seconds approximately the algorithm loses interest points to detect the object.

However, the answer of ORB algorithm is faster than FAST algorithm. But FAST algorithm has a better detection of points by pixel than ORB and it's less sensible on rotation of objects.

## **5 Conclusions**

In conclusion, every algorithm has characteristics that improve the detection of objects through of interest points but some algorithms weren't showing results waited in the theory.

The SIFT and SURF were constant in their results but the SURF algorithm have a better performance than SIFT, with less points by pixel was able to detect all objects in a short answer of processing and all environments of proof. But the best conclusion about of SURF algorithm was the invariability of rotation, illumination and kind of object is for this reason that SURF has an important position in the artificial vision as detector of interest points for the recognition of objects in uncontrolled environments.

## **References**

1. Alvear, V., Farinango, H., Navarrete, I., Rosero-Montalvo, P., Noguera, J., Cuzme, F., Peluffó, D.P.: *Internet de las Cosas y Visión Artificial, Funcionamiento y Aplicaciones: Revisión de Literatura. INCISCOS 2016*, pp. 113-116 (2016)
2. Krig, S.: *Computer vision metrics*. Switzerland: Springer International Publishing (2016)
3. Joseph, L.: *Projects, ROS Robotics*. Birmingham, UK: Packt Publishing Ltd. (2017)
4. Sucar, E.: *Visión computacional*. México: Instituto Nacional de Astrofísica, Óptica y Electrónica (2017)