An Unsupervised Machine Learning Algorithm for Visual Target Identification in the Context of a Robotics Competition

Camila Barbosa, Orivaldo Santana and Bruno Silva

Abstract— **Computer Vision and Machine Learning are the key to develop autonomous robots. While engaged with a IEEE Open Challenge, in which the robots need to recognize a miniature of a cow, we saw a solution in these areas. The main contribution of this paper is the algorithm implemented to identify and follow a known object, the miniature of a cow. We are constructing an application based on Image Processing that can detect in images this previously known object. The method yields the limits and the mass center of the entity and appropriates known algorithms, as well as Shi-Tomasi Corner Detector, the clustering K-means and local binary patterns.**

I. INTRODUCTION

The problem of determining the location and orientation of a designated object in images arises in many diverse areas of Computer Vision and Image Processing. This happens mainly because a determinate pattern can occur in various natural and man-made objects and many complex objects can often be identified by their distinct combination of linear features. In fact, edge-like structures and contours seem to have an essential role in the human visual system: a few lines in a caricature or illustration are often sufficient to unambiguously describe a scene or an element [5]. As computers are becoming more and more powerful, a growing range of applications can be implemented in real time with all the benefits that follow and still be embedded in a Raspberry Pi, for example. Furthermore, visual trackers that rely on prior knowledge on the structure but not at the motion of the scene and are insensitive to environmental changes, for instance illumination, open possibilities to a wide variety of applications [14].

However, even with low interference of illumination, it is possible to misunderstand the target object with various natural and man-made ones. A contribution of this paper is the application of a modified unsupervised Machine Learning clustering algorithm [9] with Computer Vision methods that together decrease the interference of possibles inaccuracies.

This paper proposes the solution to visually detect a specific target in the context of the IEEE Open Challenge 2016/2017. The document that announces the rules of the challenge [17] states that there has been an increasing demand for quality food in the recent past and now, for instance, consumers demand that the meat they consume comes from cage free and hormone free animals. Therefore, there has been an increasing trend to demand that animals are not mistreated during the production process. The task focuses on the problem of milking the cows on site, to avoid the stress involved in taking them to the milking barn, and thus improving milk quality [17]. To accomplish this assignment, we must create a robotic system that does the necessary steps:

1) grab a recipient to store the milk;
2) identify a cow;
3) go towards it;
4) milk the cow;
5) deposit the milked milk at a 'main' storage.

The cow is represented as a miniature 55 cm tall made of wood [17]. Its texture is a checkered pattern with black and white rectangles (Fig. 1a). These aspects are crucial to the identification of our target object.

In this paper we propose an algorithm that identifies and track the described object based on its visual characteristics. These attributes distinguish the target from its environment and filters possible noise that may be caused by similar structures.

This article is structured as it follows: The section II presents the related works as well as the theoretical background used in these papers. The proposed algorithm is given at section III, the experiments exposed at section IV and in section V, we make our final remarks.

II. RELATED WORKS

As the problem of determining the location and orientation of a designated object in images is regular dilemma in Computer Vision, there are published works that try to solve it in its own way. For instance, Jin et al. [14] propose strategies to track point features undergoing affine deformations and changes in illumination. Besides a similar contribution, our paper also determine how to track a visually defined target. The two approaches are far from being the same.

While Jin et al. [14] focused in matrix transformations, tangent planes and image deformations, our algorithm is focused at the geometrical approach of the known object. This way, when the lines, planes and interceptions are used to track the target, the algorithm reliance on colors, for example, is decreased.

On the other hand, Yadav & Singh [15] produced an image based tracker with steps similar to the ones described in Section III. They used canny edge detector, a machine learning algorithm, detected feature points and the method is also designed with a robotics purpose. The main differences between the papers are the approaches and the refining

The authors are with the School of Science and Technology, Federal University of Rio Grande do Norte, Natal, RN, Brazil. Contact: camilagomss@gmail.com, orivaldo.santana@ect.ufrn.br, bruno.silva@ect.ufrn.br

978-1-5386-0956-9/17/$31.00

with the data processing algorithm. While ours tracks a known object due to the characteristics that define it as it is, Yadav & Singh match an image previously stored in a library. Additionally, whereas the machine learning algorithm presented in Section III is unsupervised and takes advantage of previous known characteristics of the target to have a good outcome, the one described at [15] is a supervised method trained with the image it is supposed to track.

As well as Yadav & Singh [15], Iqbal et al. [18] also uses supervised learning methods. With a supervised learning system, their algorithm required a target saliency map (ground truth) along with the input image for the training. The biggest difference between our work and Iqbal et al. [18] is, indeed, the unsupervised learning algorithm, in which there is no need to give emphasis to the object manually.

III. ALGORITHM

The algorithm consists in a series of image transformations and geometrical associations based on the target visual characteristics. All of this with the purpose to select what we called *crucial points*. These *crucial points* are the points (pixels) that are more likely to belong to the target. This collection of points is, actually, the training set of an unsupervised machine learning clustering method that give as output the center of our target. The algorithm consists basically in:

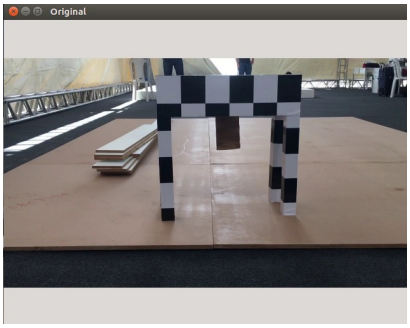
1) image transformations to adequate the frames to the geometrical associations;
2) lines and corners identification;
3) intersection of both outputs from the last step;
4) clustering algorithm.

A. Image transformations and filters

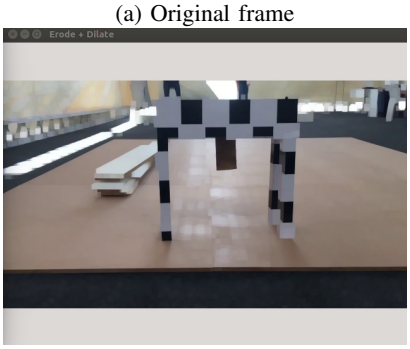
In this section, the series of transformations and filters applied to the frames will be described. Considering that before trying to identify the figure, it was necessary to manipulate the pixels so that the Hough Lines [1], [7] and the Shi-Tomasi [2], [3] algorithms could operate smoothly, we needed to select ways that increased the probabilities of these algorithms finding *crucial points*. These *crucial points* are the ones that have the biggest credibility in the frame, in other words, the ones that are most likely to be in the object of interest. The particular circumstance of our problem is that the object we want to track has only two colors in its body (black and white) and is permeated with different-sized rectangles as it is possible to visualize at Fig. 1a.

First, in order to remove noise, isolate individual elements, and join disparate elements on the image, two basic morphological nonlinear transformations are applied: erode and dilate. The second one causes filled regions within an image to grow, while the other one does the inverse operation. [4]. The original and the transformed frame are displayed on Fig. 1a and Fig. 1b, respectively.

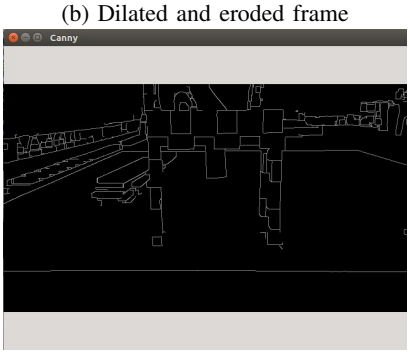
Since the goal is to identify an element which is composed only with black and white pixels, a histogram equalization over a cumulative histogram would decrease some interferences, as well as brightness. This point operation turns the white points clearer and the black points darker, favoring the contrast between the points inside and outside the cow [5].



(a) Original frame



(b) Dilated and eroded frame



(c) Frame with Canny Edge Detector

Fig. 1: Images of the sequence of transformations applied to the frames: a) Example of the original frame used for algorithm testing; b) Original frame with erode and dilate transformations; c) Dilated and eroded frame after canny edge detector filter

With the frame clean enough, the Canny edge detector algorithm can be applied with better conditions. This computational approach simplifies the analysis of images by drastically reducing the amount of data to be processed and still preserves useful structural information about object boundaries [6]. It basically selects only the edge of every element in the image (Fig. 1c). These edges are roughly described as frame positions where the local intensity changes distinctly along a particular direction. The stronger the local intensity change, the higher is the evidence for an edge at that position [5].

In these transformed frames, the algorithms to identify lines and corners are able to perform better. The Hough

Lines [1], [7] operates with the Canny-transformed image and the Shi-Tomasi [2] works with the lookup-tabled matrix originated from the histogram equalization.

B. Lines and corners detection

To identify the cow, a simple strategy is used: select all lines and extract only the parallel ones. Afterwards, a reasonable number of points is sampled (with corner detection [2]) and then, analyzed, to select which of these points belong to the parallel lines. Subsequently, we will have a set of marks that should be in its majority composed with cow-edge definers.

In frames similar to Fig. 1c (With the Canny Operator [6]), the Hough Lines algorithm is applied [7]. The theory behind this method is that any point in a binary image could be part of some set of possible lines. When each line is parameterized by, for example, a slope a and an intercept b, then a point in the original image is transformed to a locus of points in the *(a, b)* plane corresponding to all of the lines passing through that point, of which it could potentially be a part [4]. In other words, the algorithm is going to analyze the edges of the figure and select the ones that match the equation of a line. That is why this method can also be used to recognize other elements as long as they can be described with an equation.

The initial set of lines acquired with the Hough Lines [7] would certainly have plenty of noise. Inasmuch as any environment where the algorithm could be tested, it would be likely to have a great quantity of others lines defining random elements in the frame, so, filtering is still needed inside the collection of lines.

An important assertion to solve the problem is that the target object is composed only with parallel and perpendicular edges [13]. Therefore, in order to decrease the number of *crucial lines*, only the ones with the same slope as another's would still be considered *crucial*.

At this point, some of the noise is increased but it is possible to still have, in the surroundings, parallel lines that do not delimit the edges of our target. As mentioned, a determinate pattern can occur in various natural and man-made objects. Therefore, another approach is neccessary: the Shi-Tomasi Corner Detector [2]. Like the Canny edge detector [6], the code uses the first derivative of the image, but, it goes beyond. Besides looking for points in which its surroundings have a rapid change on the derivative, it searches for rapid changes in direction (corners), a two-dimensional structure [2], [3].

At this point, there are 2 major collections of points: the ones that define the 90-degrees lines and the ones extracted with the Shi-Tomasi algorithm (Fig. 2). The second set still passes through one more filtering: just the points that belong to a line are kept as *crucial*. These two groups merged are the training data of the applied unsupervised machine learning algorithm.

C. Data Processing to eliminate remaining noise

Although after plenty of Computer Vision algorithms, most of the remaining *crucial points* are in the desired object

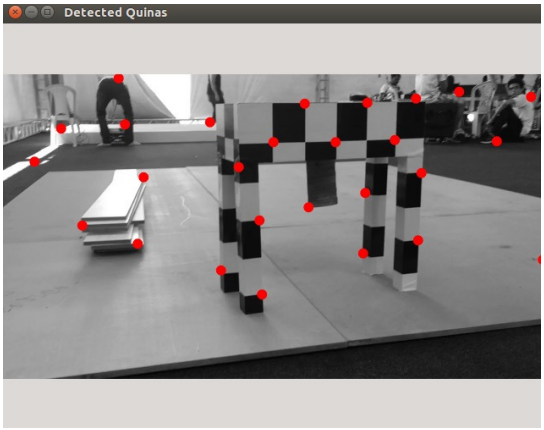


Fig. 2: One of the tested video frames with the product of the Good features to Track algorithm

to track, there are still some points that are mistakenly selected. This happens due to the similar visual characteristics between our target and its surroundings. This noise, depending on a various number of factors, as well as luminosity, contrast, similar figures and intense movement, can disturb the final outcome. Therefore, we apply a clustering algorithm in the points left as *crucial*.

Since we want to eliminate the noise, a clustering algorithm with centroids labeled as "in" and "out" of the target object is a good solution. The method implemented is a simple K-Means. It consists in the attribution to each cluster a set of points (*the crucial points*) based on its euclidean distance between them. In other words, to each crucial point a cluster would be associated, and it would be the one with the smallest euclidean distance [8], [9].

K-Means [10] proceeds by selecting k initial cluster centers and then iteratively refining them as follows:

1) Each point is assigned to its closest cluster center;
2) Each cluster center is updated to be the mean of its constituent points;
3) And repeat.

The algorithm converges when there is no further change at the assignment of points to clusters, when the clusters' position does not change anymore. Only two centroids are necessary to identify the target, $k = 2$: each one identifies one of the object's boundaries at the $x - axis$.

The training data of this unsupervised machine learning algorithm is basically a set of points and have 5 features:

- the position x of the point;
- the position y of the point;
- a histogram-based representation of the white points around the (x,y) provided;
- a histogram-based representation of the black points around the (x,y) provided;
- a histogram-based representation of the points with a *middle gray scale value* around the (x,y) provided;

The first two represent the points' coordinate at the 2D space. To acquire the other three ones a histogram was assessed and a method called Local Binary Patterns (LBP)

is adopted [12]. Originally, the LBP operator was designed for texture description. To every pixel, it assigns a label by thresholding the neighborhood of each pixel with the center pixel value and considering the result as a binary number. Then, the histogram of the labels can be used as a texture descriptor. Defining the local neighborhood as a set of sampling points evenly spaced on a circle centered at the pixel to be labeled allows any radius and number of sampling points [16], [11].

For this case, a $n \times n$ cell defined the LBP, where $n/2-1$ is the radius from the cell with center at the *crucial point*. The application of the operation occurs as in these steps:

1) For each pixel considered *crucial point*, compare the pixel to each of its neighbors (a cell built with a square with every n^2-1 points around the center);
2) Compute the histogram, over the cell, resulting in a 256-dimensional feature vector;
3) Transform this 256-dimensional feature vector into a 3-dimensional feature vector by basically dividing it into 3 proportional pieces;
4) These 3 are the histogram-based representations that form the training data.

As the training data enters the looped k-means algorithm has 5 features, the outcome is two centroids, two pairs (x, y). These points define the horizontal limits of the target object and its median represent the center of mass of the cow's miniature.

IV. EXPERIMENTS AND RESULTS

This section explains the experimentation carried out to test our proposal. The algorithm to identify and track the explained element was tested in each and every step of the development. As each segment was implemented, the theory was proved with visual confirmation. The algorithm was implemented in C++ with assistance of OpenCV library [4].

The parameter adjustments for each situation where the code was tested were calibrated in a series of trial and error tests. The Hough Lines [1], the Canny edge detector [6], the LBP [16], [11] and also the thresholds for some angle intervals assumed different values until fit to one with a high rate of accuracy.

The experiments to estimate the algorithm's accuracy consists in taking account of the frames where the algorithm was tested: how many were legitimately at the center of mass and the amount that was wrongly identified. Short videos with distinguished environments were adopted as samples, named *Video 1* and *Video 2* .

The scenario at *Video 1* can be considered the hardest. The recording is not stable and has a lot of elements that could produce noise, or even be mistakenly labeled as our target. Some of these elements are trellis, lumbers, tables, chairs and people moving right behind our target. That is the main reason of the algorithm's low rate accuracy at defining the center of mass from the target object in *Video 1*, 85.50% (Fig. 3).

The *Video 2* could be categorized as "easy", meaning that the level of difficulty for the algorithm was not high. There is almost no intense movement and the environment surrounding the cow is clean (without objects defined by lines and without persons at a close distance, for example). That explains why it has the highest rate of accuracy within all videos, 93.75% (Fig. 3).

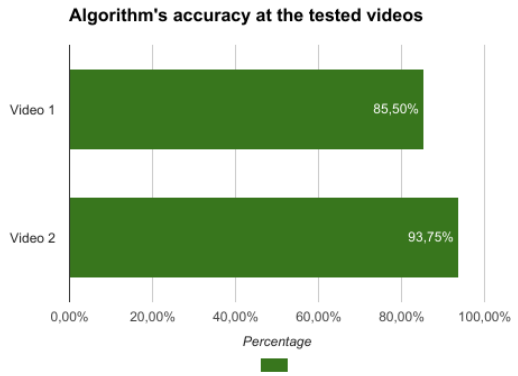


Fig. 3: Graphic that displays the accuracy of the algorithm at the distinct tested videos

The visual confirmation is provided by drawings in distinguished video files that captured the target in different conditions and angles. This way, it is possible to validate the algorithm as well as the code. The points intersect by the parallel lines were considered *crucial points* and would be part of the machine learning algorithm's training data.

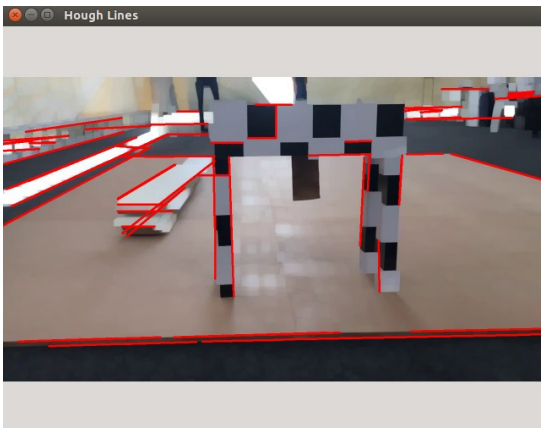


Fig. 4: One of the tested video frames with displayed lines identified with the Hough Lines algorithm

The expected outcome and the real outcome from the unsupervised machine learning algorithm are two contrasting situations. The clustering method is applied to divide the training data into two blocks of points: *in the cow* and *noise*. But it turns out to divide the set into *left leg* and *right leg*, setting the clusters to define the target's horizontal limits.

Link to Video 1: https://www.youtube.com/watch?v=l_IJTcQ5k64

Link to Video 2: https://www.youtube.com/watch?v=lNC2jCKTR5M

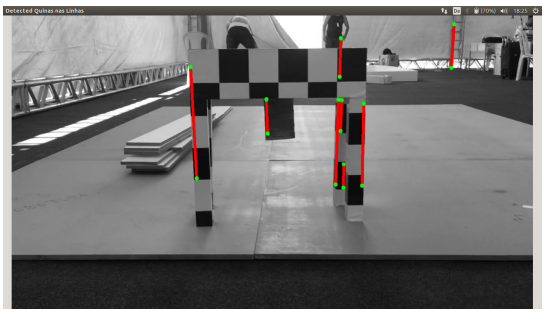


Fig. 5: One of the tested video frames with only 90 degrees parallel lines

As the unsupervised machine learning algorithm used is a clustering algorithm, it basically categorized all points in "K" clusters based on the euclidean distance [8], [9]. In other words, it attributed each point to the closest cluster, and as the noise is considerably small in a majority of frames, the two distinct group of points that the algorithm identified are the two limits of the target object, as it is visible at Fig. 6.

The experiments formulated to test our proposal presented us some interesting conclusions:

- the stability of the camera (if there is or there isn't much movement) influences the accuracy of the algorithm;
- the environment (the surroundings of the target) influences the accuracy of the algorithm;
- the processing time needs to be faster;

The first two are circumstances that can be treated, but not extinguished. While the algorithm runs, the robot is going to be in movement, and there is going to be instability within the frames. But, the stability of the camera can be increased by slowing down the movements that the robot makes. As well as we cannot erase the line-based elements from the surroundings of the target, there is going to be persons, tables, chairs and another variety of objects at the robot's vision field. And as for the third, the optimization key of this method lies at the data processing in the unsupervised machine learning algorithm.

As mentioned before, the algorithm is going to be implemented in a robot. This means, by all cases, that the computational power of the machine is limited and that the process should occur in real time. To have an idea of how the algorithm would be processed, a series of simulations at the robot simulator V-REP were executed. In these tests, a simple PID code was implemented, so that the robot could move according to the localization of the target's mass center. The outcomes can be seen in this *playlist* .

V. CONCLUSION

In this paper, we have presented the tracking of a previously known object based on its visual characteristics. It is noticeable that although this paper explores the solution for a IEEE Competition [17] and within a very specific target, the concepts developed here can be applied to the recognition of other objects with similar characteristics.

As a future work, we plan to optimize the algorithm by decreasing the processing time. As it will be implemented in a robot, the resources for the image processing and the learning algorithm are scarce, and the shorter the processing time per frame, the better.

Link to the Playlist: `https://www.youtube.com/playlist?list=PLc9i48HquG7aTHa-AhTjdXOPsZ6vJj6li`

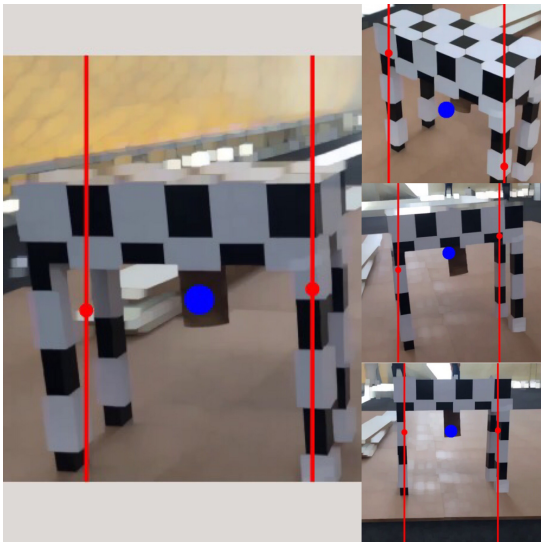


Fig. 6: Four frames with the final result of the algorithm

REFERENCES

[1] Duda, R. O., and P. E. Hart. "Use of the Hough transformation to detect lines and curves in pictures", Communications of the Association for Computing Machinery 15 (1972): 11–15;

[2] Shi, J. & Tomasi, C. "Good features to track", 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1994, 593-600;

[3] Tommasini, T.; Fusiello, A.; Trucco, E. & Roberto, V. "Making good features track better Proceedings". 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231), 1998, 178-183;

[4] Kaehler, A. & Bradski, G. "Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library", O'Reilly Media, Inc., 2016;

[5] Burger, W. & Burge, M. J. "Principles of Digital Image Processing", Fundamental Techniques Springer, 2009;

[6] Canny, J. A. "Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986, PAMI-8, 679-698;

[7] Duda, R. O. & Hart, P. E. "Use of the Hough Transformation to Detect Lines and Curves in Pictures", Commun. ACM, ACM, 1972, 15, 11-15;

[8] Likas, A.; Vlassis, N. & Verbeek, J. J. "The global k-means clustering algorithm". Pattern Recognition, 2003, 36, 451 - 461;

[9] Kanungo, T.; Mount, D. M.; Netanyahu, N. S.; Piatko, C. D.; Silverman, R. & Wu, A. Y. "An efficient k-means clustering algorithm: analysis and implementation", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24, 881-892;

[10] Wagstaff, K.; Cardie, C.; Rogers, S. & Schrödl, S. "Constrained K-means Clustering with Background Knowledge", Proceedings of the Eighteenth; International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2001, 577-584;

[11] Ahonen, T.; Hadid, A. & Pietikainen, M. "Face Description with Local Binary Patterns: Application to Face Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28, 2037-2041;

[12] Lukashevich, M. & Sadykhov, R. "Texture analysis: Algorithm for texture features computation", 2012 IV International Conference "Problems of Cybernetics and Informatics" (PCI), 2012, 1-3;

[13] Garrido-Jurado, S.; noz Salinas, R. M.; Madrid-Cuevas, F. & Marín-Jiménez, M. “Automatic generation and detection of highly reliable fiducial markers under occlusion Pattern Recognition”, 2014, 47, 2280-2292;

[14] Jin, H.; Favaro, P. & Soatto, S. “Real-time feature tracking and outlier rejection with changes in illumination”, Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, 2001, 1, 684-689 vol.1;

[15] Yadav, S. & Singh, A. “An image matching and object recognition system using webcam robot”. 2016, Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC), 2016, 282-286;

[16] T. Ojala, M. Pietikäinen, and D. Harwood (1996), “A Comparative Study of Texture Measures with Classification Based on Feature Distributions”, Pattern Recognition, vol. 29, pp. 51-59;

[17] Rules of the IEEE Open category 2016-2017 [Online] Available: http://ewh.ieee.org/reg/9/robotica/Reglas/IEEE%20Open%202016-2017%20-%20English%20-%20Version%201.1.2.pdf

[18] Iqbal, M.; Naqvi, S. S.; Browne, W. N.; Hollitt, C. & Zhang; “M. Learning feature fusion strategies for various image types to detect salient objects Pattern Recognition”, 2016, 60, 106 - 120