Journal of Real-Time Image Processing (2020) 17:1403–1419
https://doi.org/10.1007/s11554-019-00895-6

ORIGINAL RESEARCH PAPER

# Dynamic multifoveated structure for real-time vision tasks in robotic systems

## A tool for removing redundancy in multifoveated image processing

**Petrucio R. T. Medeiros[1] · Rafael B. Gomes[1] · Esteban W. G. Clua[2] · Luiz Gonçalves[1,2]**

Received: 10 October 2018 / Accepted: 25 June 2019 / Published online: 5 July 2019


**Abstract**
Foveation is a technique that allows real-time image processing by drastically reducing the amount of visual data without loosing essential information around some focused area. When a robot needs to pay attention at two or more regions of the image at the same time, e.g., for tracking two or more objects, multifoveation is necessary. In this case, computing features twice in the intersections between the different foveated structures, which could linearly increase the processing time, must be avoided. To solve this redundancy removal problem, we propose two algorithms. The first one is based on the previous calculation of redundant blocks and the second one is based on a pixel-by-pixel processing at execution time. Experimental results show a gain in processing time for the block-based model in comparison with the pixel-by-pixel and also of both in comparison with other approaches that sequentially calculate various single foveated images. Robotics vision and other tasks related to dynamic visual attention, as recognition, real-time surveillance, video transmission, and image rendering, are examples of applications that can rely on and strongly benefit from such model.

**Keywords** Real time processing · Feature extraction · Multifoveated image

## 1 Introduction

Visual data reduction and extraction of features for real-time applications is generally achieved by applying image preprocessing techniques. Reducing the amount of visual data while keeping essential information can be done using the technique known as foveation [1–13]. This technique provides reduction of 2D [8, 12] or 3D [14–16] data for facilitating further computations necessary for extraction of features, thus allowing the execution of visual tasks in real-time. Also known as multiresolution foveation, it is basically achieved by applying an image transformation from the spatial domain to obtain a dry structure in the multiresolution feature domain [3, 8, 14]. This structure maintains the maximum resolution possible in a small portion of the image, called the fovea (most inner level), and decreases image resolution in the periphery (outer levels), as the distance to the fovea increases, generally described by a logarithmic scale [17].

Foveation has been used in image/video compression for reduction of the amount of data to be codified, graphic

This work is supported by CNPq and CAPES, Brazilian Sponsoring Agencies for Scientific Research and Superior Education Staff Improvement.

✉ Luiz Gonçalves
lmarcos@dca.ufrn.br

Petrucio R. T. Medeiros
petrucior@ufrn.edu.br

Rafael B. Gomes
rafaelbg@dimap.ufrn.br

Esteban W. G. Clua
esteban@ic.uff.br

[1] Federal University of Rio Grande do Norte, Av. Salgado Filho, 3000, Campus Universitário, Lagoa Nova, Natal, RN, Brazil

[2] Fluminense Federal University, Av. Gal. Milton Tavares de Souza, W/N, 24.210-310 Niteroi, RJ, Brazil

rendering [18–21], for streaming data through the Internet or other channels [4, 5, 22] and for stereo disparity estimation [23]. In such application, the foveation method should comply with a pleasant appearance for human visualization purposes, after reconstruction. Other applications, which are the main motivation of our research, are real-time processing for robotics vision [7–11, 24] and real-time objects recognition [13–16, 25]. In such applications, there are limited resources available to process all tasks at a given real-time slot, mainly considering a general purpose robot or 3D camera. It is fine to lose some image features, but there are some that are essential for allowing useful tasks to be done in the available time interval.

One such foveated-based approach was proposed by Gomes et al. [24] and named Multiresolution Moving Fovea (MMF), which is suitable to be used in real-time applications. A single foveated vision system offers attention to a specific region at a time, thus for attending several objects, it is necessary to perform, sequentially, saccadic motions [3]. By providing the mobility of the fovea [8] in software, several regions inside the current view can be rapidly attended without moving physical resources, however, still sequentially. This sequentiality generates an inherent limitation, mainly in visual attention tasks, which is related to the impossibility of applying attention to multiple points of the visual field in a single shot.

Notice that processing and memory are limited resources that should be properly managed by a computational visual system. Applications such as robotic vision must demand all of these tasks to be performed in real time, and therefore, they struggle for computational resources. Applying all of these tasks at once using full resolution images is unfeasible due to resources limitations. Thus, the technique known as multifoveation can be used to address several tasks at once, although each one has fewer data available to work with. This is somehow similar to human behavior: we are able to perform multiple visual tasks at once, although some of them cannot be performed so well as if they were performed in a sequence.

Multifoveation is characterized by the application of several foveae in the image. This can be implemented by replication of a single foveation technique at several points of the image [4, 26] or even exploring parallel array processing implementation in hardware [1, 27]. The work of Camails et al. [1, 27] describes such a hardware implementation using Alters's FLEX 81188 FPGAs to reduce the burden of operations to be performed by the vision system. Basically, parallel projections of lower levels determine the lateral resolution gradients and the hierarchical data structure of the sensor, and this is repeated for several regions of interest (ROI), originating multifoveal structures. The main advantage of the method is the possibility of parallel processing for several ROI. Nevertheless, the described method does not deal with redundancy just replicating the single fovea approach, which are then merged into a memory. With hardware evolution since then, real-time performance can now be achieved with a PC architecture using sequential processing, which is the point of the current work. Besides, a software for parallel implementation of multifoveation using current parallel hardware (nVIDIA GPUs) will be discussed in the conclusions, as future development.

The point here is that the direct replication of various MMF structures, one for each fovea, in the context of multiple foveae, would still be computationally expensive, since there are intersections between regions in the different structures, which are then processed multiple times. As a new step towards solving this problem, we propose a mathematical formulation for multifoveation coming up with two algorithms for calculating multiple resolution structures without redundancy in the computations. Both approaches optimize the processing time in such a way that multifoveation is feasible in real time, thus enabling its application with (several) moving foveae (as it happens in the tracking of moving targets, for example). Our main motivation is to provide greater visual attention capacity to robotic systems.

Since real-time processing is a key, here, we extend the single foveated model as proposed by Gomes [14]. A remaining problem of that work that is solved here is how to compute several foveated structures (in 2D) without redundancy in the processing of regions that belong to the multiple structures and that have been previously computed. We propose two algorithms to get the disjoint regions and eliminate redundancy. The first solution is to perform calculations based on the definition of the redundant blocks, which consists on defining the limits of the regions that will be processed in each structure, for each resolution level, and the subsequent ones, and then, passing these block limits to the functions that effectively calculate features (or other computations). The second algorithm operates based on a pixel-by-pixel testing, looking if a given pixel in a specific resolution has been processed in a previously computed fovea. The mathematics used in the algorithms is introduced in this work in a straightforward way.

We performed experiments to validate the proposed mathematical formulation for both models, verifying their applicability in real-time applications. The results showed a substantial gain of the proposed approaches in comparison with the approach using multiple single foveated structures. The model based on blocks has at least equal or superior performance in comparison with the pixel-by-pixel basis. Specifically, better gains are achieved for foveae that are closer to each other, which shows the best performance of the block-based approach in decreasing processing time. The block-based approach is also better in the general context including multiple foveae randomly distributed on the image.

As an immediate application, this implementation will work as the vision-processing system embarked on our wearable glove device [28], and also in decision-making algorithms for payload applications in our robotic platforms, as drones, and in a robotic sailboat [29]. Applications already being developed for such platforms are such as visual attention and objects/target recognition for harvesting (the glove), navigation, and localization (in the sailboat).

In the remaining of this article, Sect. 2 introduces the theory for understanding the ways of achieving the processing of visual information in machines including techniques for reduction of data based on multiresolution. Particularly, our previous approaches using foveated images are described in detail in Sect. 3, including other approaches for multifoveae. Section 4 details the two proposed algorithms for removing redundancies, which are the core and main contributions of this work. Section 5 verifies the usefulness of our methods including an experimental analysis and comparison with the state of the art. Finally, we discuss the main contributions and propose some directions for research in Sect. 6.

## 2 Reducing and abstracting the complexity of vision

Imitating the human vision system in a computer has been, so far, a very complex and challenging task, requiring a lot of memory and time. Some sort of reduction and abstraction is generally necessary to achieve real-time results. Roughly, in biological vision, the rays of light are filtered through the lens entering through the pupil onto the retina at the back of the eye, where the sensors (cone and rode) are actually located. These two types of photo-receptor cells are not evenly distributed in the retina. Cones are more concentrated in the region with the highest visual acuity, known as fovea. In one of the first attempts towards allowing a computational system to imitate this process, Uhr proposed to approximate the vision processing as a cone of recognition, where lower quality is found closer to the peak levels [30]. It is noteworthy that this form of representation simulates the inverse of the behavior of the biological vision, supposedly from simple to complex.

### 2.1 Approaches for data reduction and abstraction

Following the direction proposed by Uhr [30], several approaches have been devised with the objective of reducing and abstracting data, diminishing the computational complexity required in tasks such as image coding and feature extraction and/or description. The main are the use of the Gaussian [31] and Laplacian [32] for construction of pyramid like structures [33], the scale-space theory [34], the log-polar representation [35], the wavelet transform [36], the multiresolution approach with centralized fovea [3, 7], and the one with moving fovea [8, 14, 24]. The traditional procedure for generation of the Gaussian pyramid [31] is to apply a Gaussian filter on the original source image and perform a downsampling of the smoothed image. Samples with a factor of 2 are generally used, and this process is repeated for the creation of new (upper) levels. The Laplacian pyramid is constructed using the levels of the Gaussian pyramid [32].

The scale-space theory arises from the observation of the representation of real world objects in images, as perceived by humans. According to Lindeberg [34, 37], among the linear transformations, the Gaussian nucleus is the only one capable of generating the space of scales. In fact, this theory has been just extended for incorporation of time scale [38].

Another computational theory inspired from Biology, elaborated in the 70s, is based on researches conducted using animals such as rabbits, cats, and monkeys, to study the mammalian vision system. As a result of these studies, computer scientists have realized that the transmission of information between the retina and the visual cortex follows a log-polar law [35].

A traditional approach that has been widely used is the one based on wavelet decomposition. A combination of low-pass and high-pass filters is used to approximate the frequency range of the two-dimensional signal [36]. The basic functions of the wavelet decomposition are required to meet orthonormal constraints to keep the data in a Hilbert space. Several studies have been carried out about wavelet (mainly in the 80s), such as the Daubechies systematical method for constructing the compact support for orthogonal wavelet, and its employment by Mallat and Meyer's in the multiresolution concept, which is somehow related to our proposal. In fact, wavelet complexity is solved with the fast wavelet transform, as proposed by Mallat [36], allowing its application in computer vision.

### 2.2 Multiresolution with multifeatures (MRMF)

Notice that some of the above approaches, such as the ones based on the scale-space theory and using Gaussian pyramid, increase the amount of data and facilitate the extraction of features. On the opposite direction, an interesting spatial approach that substantially reduces the amount of data also allowing multiresolution multifeatures (MRMF) extraction in real time is proposed in 1998 with the thesis of Gonçalves [2, 3]. The main idea of his approach is to start with the directly mapping of every pixel (just picking one by one) from an initially small and centered region in the original image to the highest resolution level (at the top) of a parallelepipedal structure that has the same resolution and size as the region on the original image. Considering that a 32×32 central pixel region is taken, this will be called the fovea. Thereafter, a larger, also image centered region

of the original image is taken for feature extraction, around the first one and containing it (for redundancy reasons), with double the size of the previous region (if a scale 2 is used), but reducing the scope in the resulting image by downsampling (depending on the used feature), so that the output is an image that has the same size as the first parallelepipedal level. This process is repeated by increasing the scope of the region in the original image (always image centered) and even reducing the size of the output thus keeping the same size as the fovea, until the coarser level of the structure has the whole and complete original image, however, reduced to a very coarse resolution [7]. Feature extraction can be any one as desired by the user, such as SURF [14], and it can be extracted from this structure or straight from the original image through this structure.

The result of this approach is a multiresolution structure with much less data than the original image, named multiresolution multifeatures (MRMF). In fact, the structure is not pyramid like [33]; instead, it is a structure in which all of the levels have the same size [3]. For instance, this structure drops down the amount of data from 1M to 6K pixels if using 6 levels of resolution with the size of 32×32 each one, and scales of 2 for an original image of 1K × 1K pixels. If 8 features are extracted for each spatial element using this approach, the total amount of features would be 48K, which is much less than the original 1M pixels to be processed if the whole image is used. Notice that this is a very usual amount of total features to be calculated from an image, mainly in general recognition tasks. The first implementation of this approach in real time was originally made with a dedicated image processor board from Datacube [2]. With hardware evolution, the model could be latter implemented in PC-based architectures [7, 39], also achieving real time.

### 2.3 Multiresolution with moving fovea (MMF)

An enhancement to this approach is the multiresolution model, where the fovea is not always centered, called moving fovea (MMF) as formalized by Gomes [8]. Based on the above MRMF, this improvement aims to mathematically define the dimensions of each level of the foveated structure and to provide mobility to the structure, so that the position of the fovea can be placed anywhere in the image, and not necessarily in the center. This is interesting in applications, where changing of attention without performing physical motions of the cameras is a key feature. This is often necessary for robotic vision systems such as a stereo head for paying attention to a new region of interest. Using this improved structure, recognition tasks could be done in real time [14, 25]. Gomes et al. [14, 24] demonstrate that it is possible to reduce processing time using the MMF model within recognition tasks, without loss of effectiveness, provided that the position of the fovea is properly controlled.

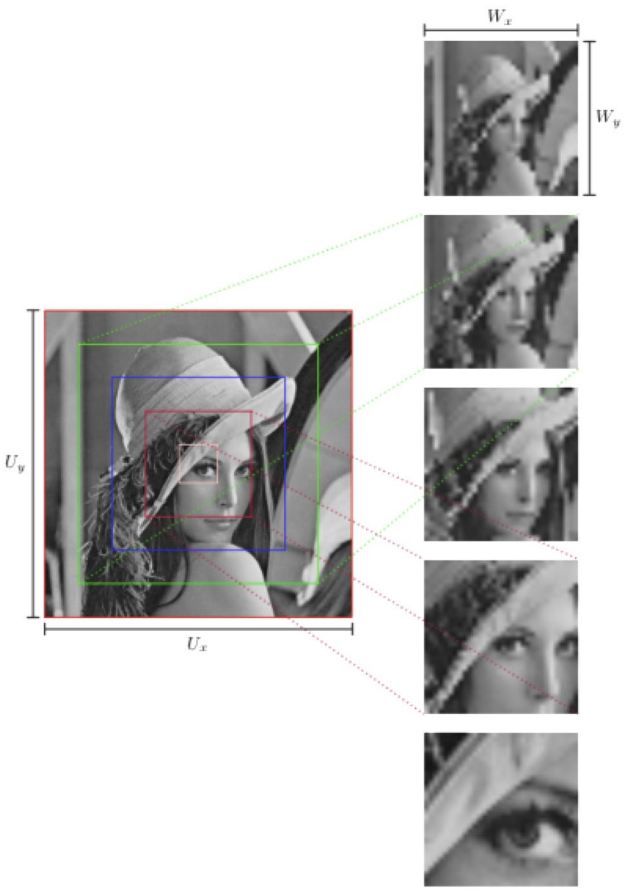



**Fig. 1** Construction of levels with the MMF

The building of this improvement can be summarized as follows [24]. The construction of the parallelepiped layers is the beginning of the preprocessing required for the construction of the MMF structure. In this layered construction, the image is mapped to a set of $k$ levels, with constant size $W$, with indexes from 0 to $m$, where $m$ is the level of the fovea, as shown in Fig. 1. Let $I$ be an image of size $U = (U_x, U_y)$, and for each level $k$, it is delimited a portion of size $S = (S_x, S_y)$ of $I$, which will be mapped to the multiresolution domain. It is defined that $S_0 = U$ and $S_m = W$, whereas the intermediate levels are obtained by interpolation, according to Eq. 1 [24].

The great advantage of this method is that during the extraction or description of features, it does not allocate memory for $k$ images of size $W$, because Eq. 1 already informs, in $I$, the size of the portion that will be processed:

$$S_k = \frac{mU + Wk - kU}{m}. \tag{1}$$

In his work, Gomes states that the mobility of the fovea can be controlled using a fovea vector $F$ inside the image domain [24]. Therefore, it is established that the vector

$F$, originating from the center of the image $I$, is between $(W-U)/2$ and $(U-W)/2$. Consequently, $F=(0,0)$ when the fovea is positioned in the center of the image $I$. Equation 2 indicates the starting position of each region, in the space domain, which must be transformed.

$$\delta_k = \frac{k(U-W+2F)}{2m}. \tag{2}$$

### 2.4 Our current contribution

While the MMF approach substantially reduces the amount of data and at the same time makes easier the abstraction of features, even in real-time recognition tasks, it still has the main drawback of being a single foveated approach. This would be fine for imitating human vision behavior [24], but in robotics vision, it sometimes does matter paying attention to several regions at a single fixation. This is the case in monitoring or vigilance missions, for instance. It is noteworthy that vigilance and monitoring can rely only on coarse scales to detect relevant events. However, if it is necessary to go to multiple targets in the details' scales, then multiple foveae are welcome. Nonetheless, this agree with researches in visual attention who provide evidences that our brain is not able to process all of the information that is contained in the physical world [40]. In fact, some researches in selective attention indicate that substantial information is lost in a top-down context [41]. As a solution to this, we suggest the inclusion of multiple foveae, one for each stimulus.

This limitation is inherent to a computational system with a single fovea, mainly because the system does not have enough time for realizing the several saccades to extract all of the visual information, and consequently, the system loses relevant information from several of the stimuli. Several foveae can be defined in the image, and a straightforward way to allow multifoveation is to take this approach that would have to be reproduced for each single fovea detected in the image (each region of attention). Notice that redundancy appears in this straight approach with the same regions eventually appearing in the different parallelepipeds that represent each foveated structure. The removal of this redundancy is one of the improvements that we develop in the current work. Other works on this same subject are not found in the literature thus indicating an original contribution of our work.

Using a single foveated image at several places eliminates the redundancy problem. However, it may not be suitable for the real-time execution of a task in which attention should be kept at several places. This also comes from the fact that the foveated model is derived from the biological vision. Selective attention using a single fovea and provision of saccadic movements cause loss of visual information [41] in despite of the time spent for a saccade, which could go to about half of a second in a robot [2]. In such applications, saccadic movements should be avoided when possible.

## 3 Related work

Few works are found to implement the multifoveation technique, in different applications and scenarios. The work of Dario et al. [42] explores multifoveation in hardware for the analysis of the arrangement of pyroelectric and piezoelectric sensors in a tactile system. He seeks to analyze the accuracy of the sensors regularly (orthogonal and hexagonal) and non-regularly (foveated and multifoveated) distributed, considering that all sensors have a circular shape. The result of the multifoveated configuration shows that the calculated position and orientation errors of the standards do not increase significantly in relation to the other approaches, but it is as accurate as the other dispositions [42].

Another application that uses multifoveation is the tracking of moving objects as proposed by Lim [43]. The approach uses the log-polar representation to track multiple objects in a video sequence. The log-polar domain is divided into four regions and established that identification would be performed only in two of these regions, called the saved and the active region. After detection, the camera is positioned in a way that the center of the object is in the center of the image. In addition, the object is identified with a tag that distinguishes it in the scene. Another work developed for the tracking task is proposed by Camails et al. [1, 27] based on the Gaussian pyramid. Factors for subdivision are defined on each side of the fovea for the construction of the levels of the structure. According to Camails, multifoveation is performed by the execution of this procedure several times.

Image compression is also greatly exploited using the multifoveation technique. Dhavale and Itti [4, 5] make use of the replication methodology of the gaussian pyramid. By weighting the value of the smallest salience and the smallest distance from the point to the object, a value is generated that defines which output of the pyramid will be used to compose the multifoveated image. Basu et al. [44–46] seek to compress visual information using the Cartesian Variable Resolution (CVR) technique. They propose two methodologies, so that during the insertion of more than one region of interest, it would be possible to define when to reduce the resolution around the fovea as a compensation, or if additional information should be retained by reducing the compression ratio.

Rodriguez et al. [47] propose to compress the static information from the video and perform the transmission at different rates for different resolutions making use of the exponential Cartesian method (SFMG) and the multifoveation technique. This method is developed to be applied in traffic

**Table 1** Works related to the technique of multifoveation

| Article | Method | Context | *R-T* | Dom. |
|---|---|---|---|---|
| [42] | Variable space | Recognition | No | Spat |
| [43] | Log-polar | Tracking | Yes | Spat |
| [1, 27] | Gaussian pyramid | Tracking | Yes | Spat |
| [44–46] | Cartesian variable | Compression | Yes | Spat |
| [47] | Cartesian Expon. | Compression | No | Spat |
| [4, 5] | Gaussian pyramid | Attention | No | Spat |
| [22] | Multipoint Adapt. | Compression | No | Spat |
| | | | Yes | Freq. |
| [49] | Multipoint Adapt. | Compression | No | Spat |
| | | | Yes | Freq. |
| [23] | Adapted MRF | Stereo disparity | Yes | Spat |
| [15] | MMF | Recognition | Yes | 3D |

tracking, security, and supervision. In such applications, it is often necessary to provide information with quality in a wide field of view, but this requires a greater network resource.

Stereo disparity estimation using single foveated images is done by Gomes et al. [8]. Based on it, a method for stereo disparity estimation in multifoveated images is further developed [23] using an adapted multiscale loopy belief propagation (BP) method on a Markov random field (MRF) [48]. However, multifoveation is used only to restrict the region that will be subject for a more accurate (however, slower) disparity method.

A multifoveated (3D) approach inspired by models of (2D) multifoveation [1, 4, 5, 27, 47] and using the MMF [14] has been recently developed for dealing with point clouds to reduce the processing time in the detection of objects [15, 16]. While this approach avoids the processing of redundant regions, it differs from our proposal, because it removes the intersection between levels of the same fovea structure, not between multiple foveae, as this is necessary in point cloud processing.

Multifoveation for image compression is provided by Sankaran et al. [22] who use frequency filtering. Multifoveation is achieved through the insertion of several cut frequencies. Inspired by this technique, it is further realized that the high-resolution information that remains unchanged in the next frame has no need to decrease its resolution and that unchanged information may come from the previously processed frame [49].

Table 1 resumes the existing techniques that are most related to ours according to the method used, application (context), real time (yes or not), and the domain in which the work is developed (spatial or frequency). It is possible to notice from the table that the multifoveation approach is applied in several contexts through the replication of a single multiresolution structure, as the one detailed above [24]. All the related works found that are developed in software use the extraction of motion features to infer knowledge about the context, except that of Hunsberger et al. [23] that define the regions of interest applying a task-specific cost function in the rough disparity image. Among those works, there are some that perform the technique in real time, but using dedicated hardware (FPGA) to allow parallel (array) processing. Notice that this replication causes an intersection between the fovea levels if it is not previously treated and the only works in the literature dealing with this redundancy of information are the ones of Camails et al. [1, 27], Rodríguez et al. [47], and Oliveira et al. [15, 16]. Oliveira's work deals with multifoveation in point clouds using a completely different approach for that, based on boxes. In the first two works [1, 15, 16, 27, 47], however, the foveae are processed separately, from rings, which are levels of the fovea structure around the fovea, to eliminate redundant processing. The solution proposed by Camails to separate the fovea from the structure ends up being palliative, because there is still redundancy of information between the rings. Furthermore, the differential of the method proposed in this paper is the identification and elimination of all existing redundancy for the replication of the (multi)foveated structures.

## 4 Removing redundant information

As previously mentioned, MMF defines the level $k$ of the structure using $\delta_k$ and $S_k$, as defined in Eqs. 1 and 2, respectively [8, 14, 24]. The *MMF* method transforms an image into the spatial domain into $m$ images with the same dimension as the fovea that gives the width and length of the parallelepiped (the height is the number of levels).

We define $P$ as a set of ordered pairs (Eq. 3). In this way, $P$ is formed by all values of $\delta_{k,j}$ and $S_{k,j}$, where $k$ is the level and $j$ is the foveated structure index. The replication of at least one region in the multiresolution domain is guaranteed, which occurs at the first level of the two foveae, since $S_{0,i} = U$ and $\delta_{0,i} = (0, 0)$, whatever value $i$ has

$$P = \{(\delta_{k,j}, S_{k,j}) | k \in [0, 1, \ldots, m] \wedge j \in [1, 2, \ldots, n]\}. \quad (3)$$

Based on the definitions in Eq. 3, two approaches are introduced next for the extraction of features without redundancies in computations. In the first approach, a pixel-by-pixel verification is performed. In the latter approach, disjoint blocks of pixels that guarantee no redundancy are directly submitted to the feature extraction algorithm.

### 4.1 Pixel-by-pixel-processing approach

In this approach, given a pixel at one fovea structure, and using a point inside the rectangle test, it is tested whether that pixel has already been processed in other fovea structure at the same level. If it has, then the pixel can be skipped.

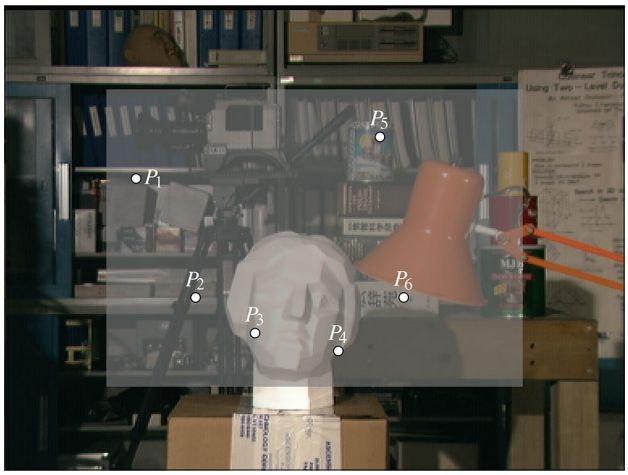



**Fig. 2** Example of distribution of intersection vertex within an image

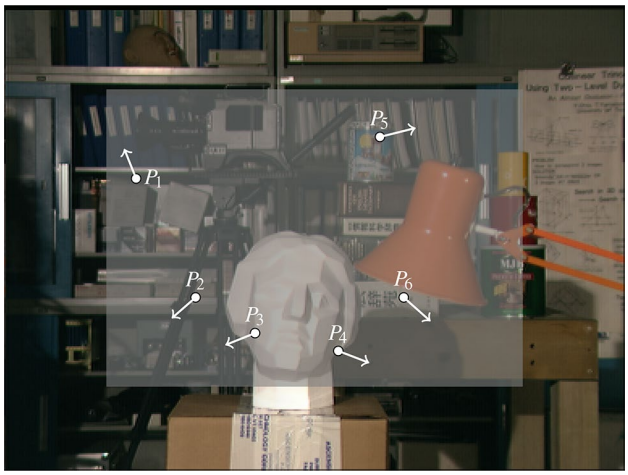



**Fig. 3** Example of direction of regions to be eliminated

The exact number of tests is context dependent, since not all pixels inside a foveated level are necessarily evaluated, like in the feature extraction algorithm proposed by Gomes et al. [14] for which an octave is associated with each level.

## 4.2 Block-based processing approach

In the second approach, the algorithm determines disjoint regions based on the foveae positions, which are free to be processed without any further redundancy checks. There is an extra computation to get these regions; however, it does not affect overall performance.

The problem addressed here is how to reassemble the $P$ set defined in Eq. 3 to a new set $\tilde{P}$, such that regions defined by any two elements of the set with the same level indexes are disjoint and they cover exactly the same regions. The idea for obtaining regions without intersections consists in discovering, for each level of each foveated structure, a set of rectangular regions that represents the region of this level, not necessarily convex, due to the elimination of the intersections with the regions of the same level of the previous foveae. For this purpose, for each level, the reference vertexes of the previous foveae that are within the analyzed region (see Fig. 2) and the directions of their intersection regions are calculated (see Fig. 3).

Given two foveated structures, the intersection computation between them can be optimized considering that they have equal numbers of levels, and considering that $U$ and $W$ (vectors with sizes of the original image and of foveated structure levels as defined above) are the same for both foveated structures. Figure 4 shows an arrangement of projections of the foveated structure $A$ and $B$ in the plane $xz$, where $a$ and $b$ represent the projections of the fovea centers, respectively, and $a \leq b$. In this case, the projection on $x$-axis of the intersection region at level $k$ is the interval $[\delta_{k,B,x}, \delta_{k,A,x} + S_{k,A,x}]$. This is the case that is considered from now on, because this assumption allows the development of a fast multifoveated algorithm, described in the next section.

The intersection region between two foveae referred by $r$ and $s$ is defined as the Cartesian product: $I_x \times I_y$, where $I_x$ and $I_y$ are defined by Eqs. 4 and 5. These equations represent the interval in each component that delimits the intersection region of the level $k$ between two foveae $r$ and $s$:

$$I_x = [\delta_{k,B,x}, \delta_{k,A,x} + S_{k,A,x}] \tag{4}$$

$$I_y = [\delta_{k,B,y}, \delta_{k,A,y} + S_{k,A,y}], \tag{5}$$

where independently for each component:

$$A = \underset{i \in \{r,s\}}{\operatorname{argmin}} f_i \tag{6}$$

$$B = \underset{i \in \{r,s\}}{\operatorname{argmax}} f_i. \tag{7}$$

Furthermore, for the purposes of the proposed algorithms, we define the reference vertex and orientation between two levels.

### 4.2.1 The reference vertex and orientation

There are four possibilities for two axis-aligned rectangular regions' overlapping: (1) an empty set (2), a line (3), a point (4), a rectangular region. Our proposed algorithm works based on a single point of this intersection, which is called here as the reference vertex.

If there is an ordered pair of such regions $(a, b)$, then the reference vertex is defined as a single vertex from $b$ that lies on $a$. If the intersection is an empty set, there is no such reference vertex. If there are two or more vertexes from $b$, any of them

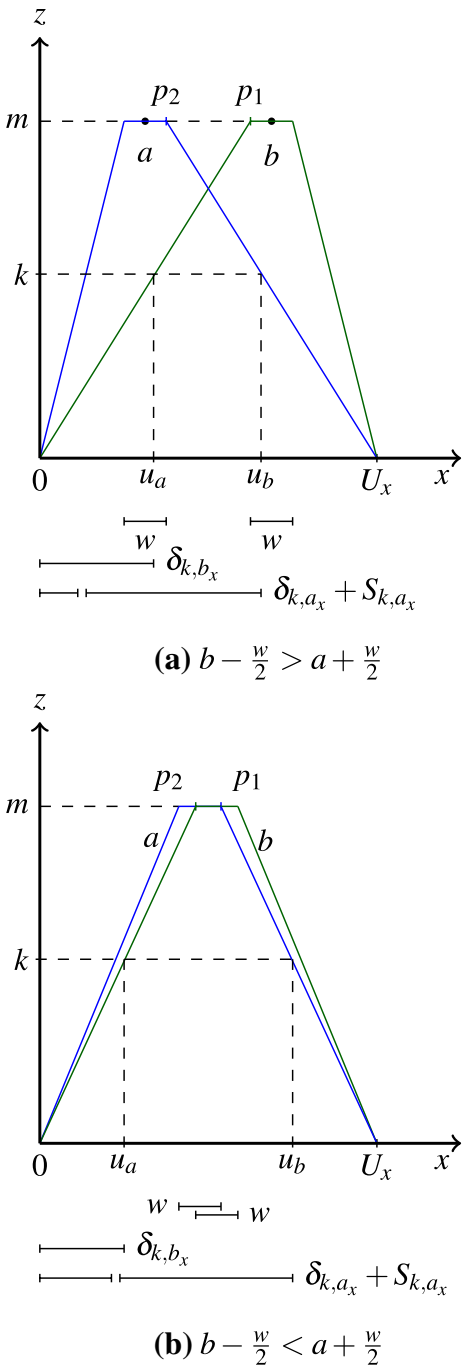



**Fig. 4** Two possibilities of two fovea placement along a single axis. The $z$-axis represents the level despite its discrete domain for visualization purposes

can be chosen, but this decision must be considered later in the proposed algorithm.

**Table 2** Orientation vector definition example in a function of displacement ($\Delta x$, $\Delta y$) between foveae

| $\Delta x$ | $\Delta y$ | Orientation |
|---|---|---|
| Positive | Positive | Northeast |
| Positive | Negative | Southeast |
| Negative | Positive | Northwest |
| Negative | Negative | Southwest |
| Zero | Positive | Northeast |
| Zero | Negative | Southeast |
| Positive | Zero | Southeast |
| Negative | Zero | Southwest |
| Zero | Zero | – |

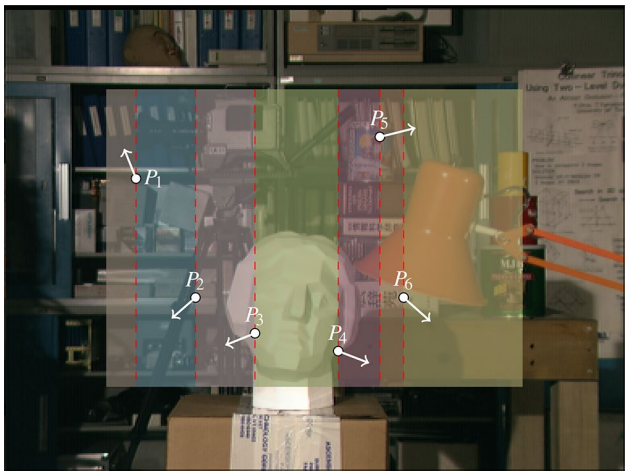



**Fig. 5** Example of level division in vertical regions

To define the orientation of the second region in relation to the first, we consider the difference between the central positions of each region and the variations in $x$ and $y$ to define the orientation vector, as shown in Table 2.

#### 4.2.2 Proposed disjoint sets algorithm

Let $t$ be the number of reference vertexes. The region of the level in question is split into up to $t + 1$ rectangular regions, having as references the abscissa of each vertex, as illustrated in Fig. 5. We consider two arrays with size $t + 1$, where the first is the upper bound and the second is the lower bound of ordinates of the ordinates of each rectangular region. The first one is initialized with the value of $\delta_k$, whereas the second one is initialized with the values of $\delta_k + S_k$ of the level $k$ under analysis. A line sweep algorithm, as described in Algorithm 1, is applied, such that at each vertex, the algorithm changes the upper or lower bounds of the regions toward supremum and infimum, respectively. At the end of the algorithm, these

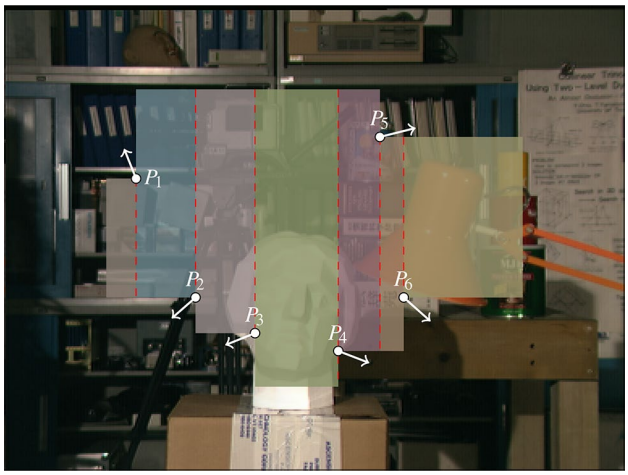



**Fig. 6** Example of level with intersected regions deleted

regions represent the set of regions without redundancies and they are added to the new set $\tilde{P}$ (see Fig. 6).

**Algorithm 1:** Upper and lower bound adjustment.

**Input:** Reference vertexes $(x_i, y_i)$, such that $i \in [0, \ldots, t-1]$; orientation vectors $d_i$, such that $i \in [0, \ldots, t-1]$; upper bound vector (v) with size $t$; lower bound vector (u) with size $t$

**Output:** Minimum and maximum ordinates of each region

```
for each reference vertex i do
    if d_i == northwest then
        v[j] = min(v[j], y_i), for each j ∈ [0, 1, ..., i];
    end
    else
        if d_i == northeast then
            v[j] = min(v[j], y_i), for each j ∈ [i+1, ..., t];
        end
        else
            if d_i == southeast then
                u[j] = max(u[j], y_i), for each j ∈ [i+1, ..., t]
            end
            else
                /* d_i == southwest */
                u[j] = max(u[j], y_i), for each j ∈ [0, 1, ..., t]
            end
        end
    end
end
Q ← v, u;
return Q;
```

## 5 Experiments and results

This section presents the achieved results using our two proposed approaches. The tests aim to analyze the computational performance of these two methods compared to the traditional multifoveation approach, which is achieved by sequentially applying a single MMF, for each structure, several times, without considering redundancy.

We analyze the efficiency of the proposed methods in the elimination of redundancies using the SURF feature extraction technique with a standard (chess) image and also with the Lena's image (Fig. 1) to evaluate the algorithms efficiency in a more real-life image. A similar procedure is used for both images, with foveae in the corners, near the image center, and in the image center.

We chose SURF as the main feature for most of the experiments, because its implementation needs complex calculations, enough to test if our algorithms can do it in real time, as SURF is among the most time-consuming of processing features. In addition, it is one of the most effective features used in robot vision systems. Besides, other features can be used without modifications on our algorithms and keeping performance. In fact, we also show time results for the Canny edge detector, after these main experiments, to better evaluate our methods.

### 5.1 Processing versus distance between foveae

Considering the displacement between two regions of size $(S, S)$ as $(\Delta x, \Delta y)$, the number of pixels in the intersection is given by Eq. 8, as plotted in Fig. 7:

$$t(\Delta x, \Delta y, S) = \begin{cases} 0 & \text{if } \Delta x > S \vee \Delta y > S \\ S^2 - S(\Delta x + \Delta y) + \Delta x \Delta y & \text{otherwise} \end{cases} \quad (8)$$

Hence, the number of pixels in all intersections of two foveated structures is given by Eq. 9. Figure 8 shows an example of this for m = 3, $W \in (30, 30), (60, 60), (90, 90)$ and $U = (500, 500)$:

$$\sum_{k=0}^{m} t(k\Delta x/m, k\Delta y/m, S_{k,x}). \quad (9)$$

### 5.2 Comparison between methods

For all the experiments in this section, a 2.3GHz Intel Core i5 Laptop with 4GB of RAM and Linux Ubuntu 14.04 operating system is used. We present results of the execution of the traditional approach to multifoveation using a simple

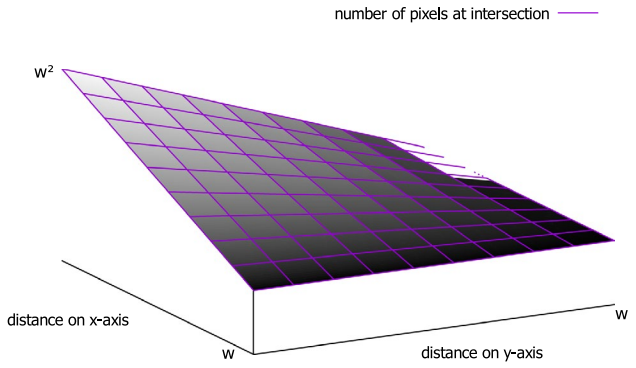



**Fig. 7** Number of pixels in the intersection between two regions of size $(S, S)$

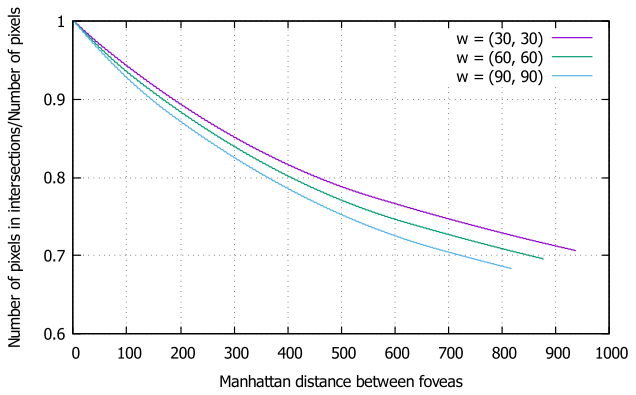



**Fig. 8** Number of pixels in the intersection between two foveated structures in comparison with the total number of pixels

replication of the original MMF method compared to our two approaches. The algorithms are executed for 1, 2, 3, and 4 foveae, with size $W = (60, 60)$, submitted to 5000 iterations. Each fovea contains three levels and it runs the extraction of features for all of them. All of the tests performed in this experiment are done with two images. The first is a chessboard image with dimensions of $250 \times 250$ pixels, with each cell having dimensions of $5 \times 5$ pixels. The second is the Lena's image (Fig. 1) that has dimensions of $500 \times 500$ pixels presented in Sect. 4 that explains the MMF technique.

The chessboard image is chosen, because it has a uniform distribution of SURF features, thus ensuring that the positioning of the fovea will not compromise the comparison of results between the methods.

### 5.2.1 Foveae in the corners of a chessboard

The foveae are distributed and positioned at the corners of the image of a chessboard. For this experiment, the foveae were distributed considering the following positions $(-95, -95)$, $(95, 95)$, $(95, -95)$, and $(-95, 95)$, respectively.

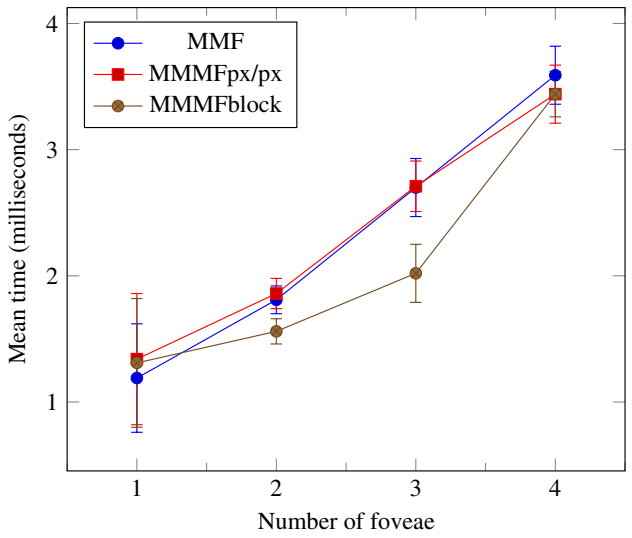



**Fig. 9** Foveae on chessboard corners: mean time (milliseconds) and standard deviation for obtaining keypoints

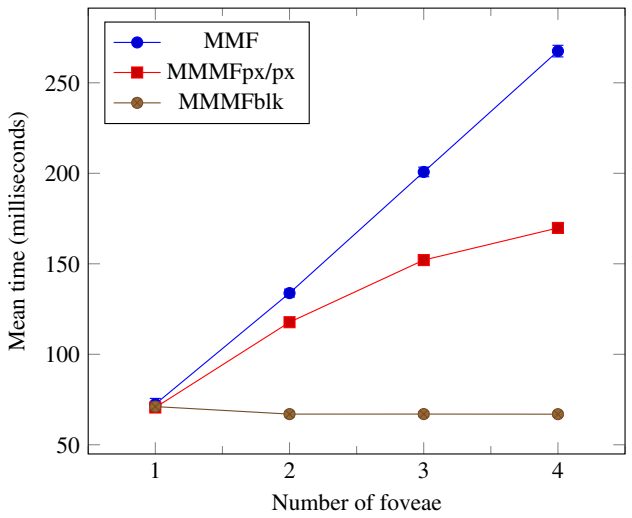



**Fig. 10** Foveae on chessboard corners: mean time (milliseconds) and standard deviation for computation of the descriptors

As shown in Figs. 9 and 10, we notice that, for the detection of keypoints, the sending block approach performs better than the MMF and the pixel-by-pixel verification. We had already hypothesized that the pixel-by-pixel approach would be very costly, because this algorithm spent a lot of time analyzing whether the pixel is or not in a redundant region. It was possible to observe that the MMMF construction with the sending blocks approach is faster in almost all tests than the simple MMF replication for the detection of keypoints, except in the case of using only one fovea, as expected. Notedly, in the computation of the descriptors, both MMMF approaches are superior to the MMF.

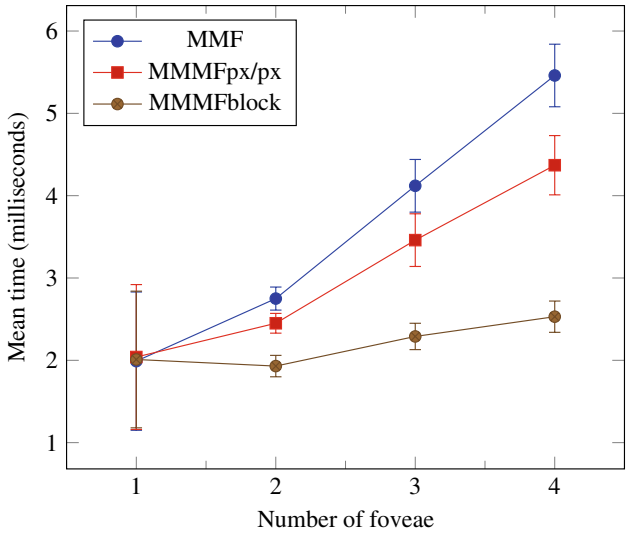



**Fig. 11** Foveae around chessboard's center: mean time (milliseconds) and standard deviation for obtaining keypoints

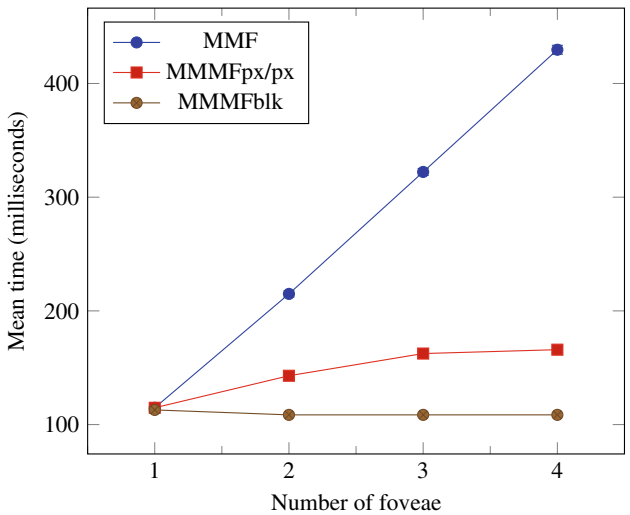



**Fig. 12** Foveae around chessboard's center: mean time (milliseconds) and standard deviation for computation the descriptors

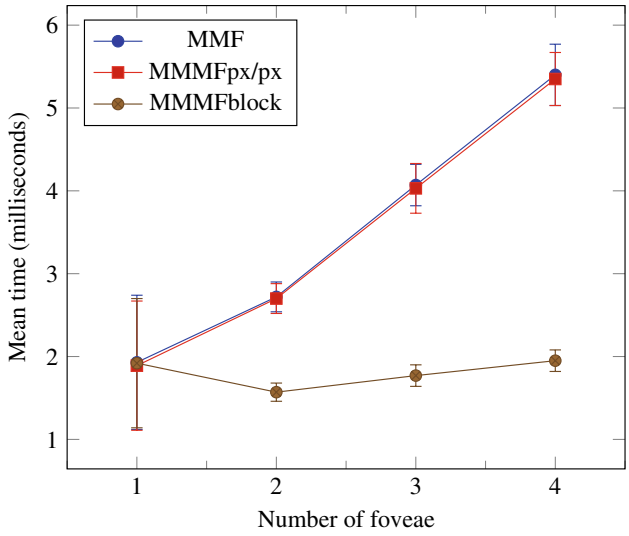



**Fig. 13** Foveae on chessboard's center: mean time (milliseconds) and standard deviation for obtaining keypoints

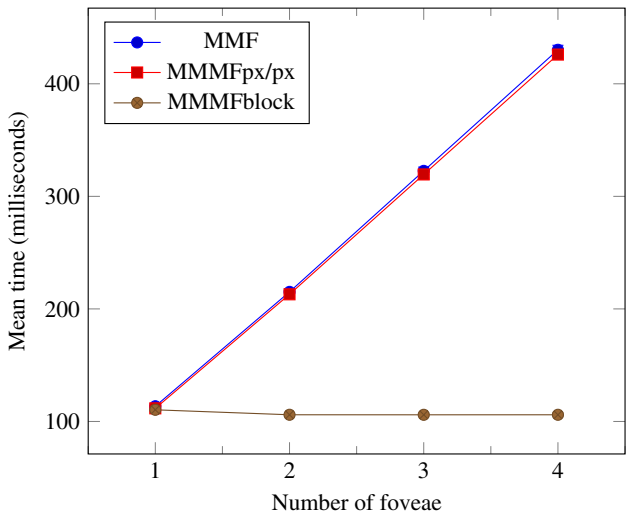



**Fig. 14** Foveae on chessboard's center: mean time (milliseconds) and standard deviation for computation the descriptors

### 5.2.2 Foveae around the center of the chessboard

In this testing case, the foveae are homogeneously distributed around the center of the image, but it is guaranteed that there is no intersection at the last level. That is, the foveae are sufficiently far from each other, at least more than the size of each level in both directions. For this experiment, the foveae were distributed in $(-30, -30)$, $(30, 30)$, $(30, -30)$, and $(-30, 30)$, respectively.

As shown in Figs. 11 and 12, our methods perform better than the MMF in almost all fovea distributions, except when there is only one fovea, as expected. Notice that this happens, because the MMMF approaches perform some additional operations.

### 5.2.3 Foveae in the center of the chessboard

As a last experiment, we place all foveae at the position (0, 0), that is, all of them totally overlapping each other. With this, we expect that all of the levels of the computed foveae present intersections with the first fovea. Consequently, it is supposed that the processing of foveae 2, 3, and 4 will not be very time consuming.

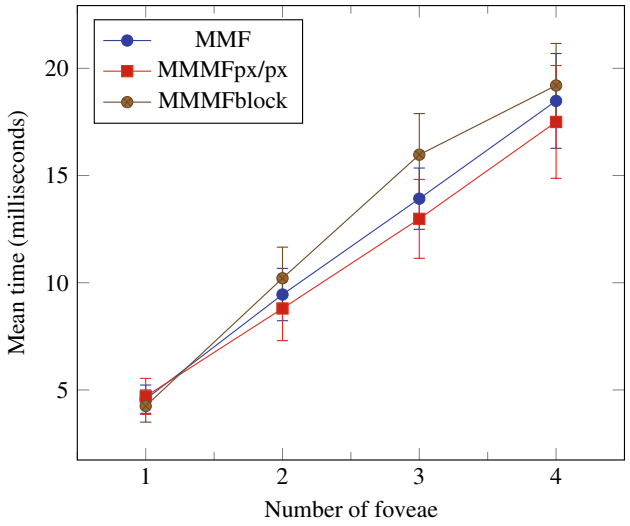



**Fig. 15** Foveae on lena corners: mean time (milliseconds) and standard deviation for obtaining keypoints

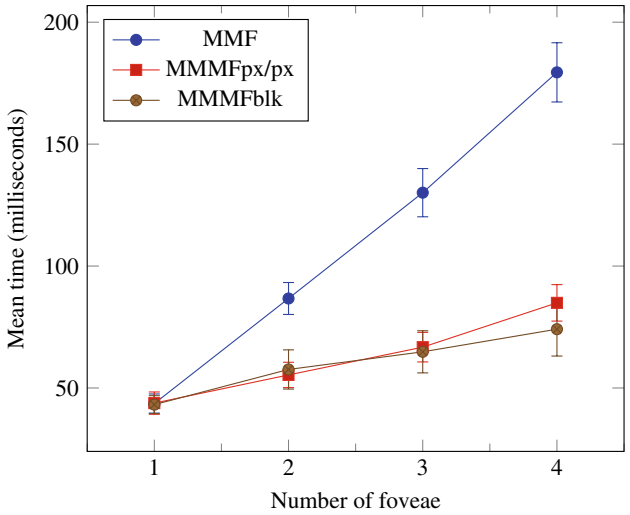



**Fig. 16** Foveae on lena corners: mean time (milliseconds) and standard deviation for computation of the descriptors

Actually, the supposition that the processing of foveae 2, 3, and 4 should not consume much time is verified through the graphics shown in Figs. 13 and 14, for the approach of sending blocks. In the pixel-by-pixel approach, this does not happen, because all of the pixels end up being scanned, what consumes processing. However, this method is more efficient than the traditional MMF with re-execution, both in the extraction of keypoints and in the computation of the descriptors.

#### 5.2.4 Foveae in the corners of Lena's image

Furthermore, we chose Lena's image to evaluate the behavior of the methods on a more real-life image distributing the foveae in the positions: $(-226, -226)$, $(226, 226)$, $(226, -226)$, and $(-226, 226)$, respectively. These values were adjusted according to the image size.

By looking at the corners of Lena's image, we notice that there is not a large amount of SURF features in those places. In this situation, we expected that the sending block approach did not provide the best performance, because it was built without considering the content of the intersection (features). In other words, the identification of the block intersection takes into account only the positioning between the foveae. In fact, for the detection of keypoints, the sending blocks approach does not perform well, and is worse than MMF and the pixel-by-pixel verification, as shown in Fig. 15. Figure 16 shows that the computation time of the descriptors for both the MMMF approaches is fairly inferior to the MMF. Thus, even the block-based loosing in the keypoints detection is overall better than the MMF replicated several times because of the descriptors.

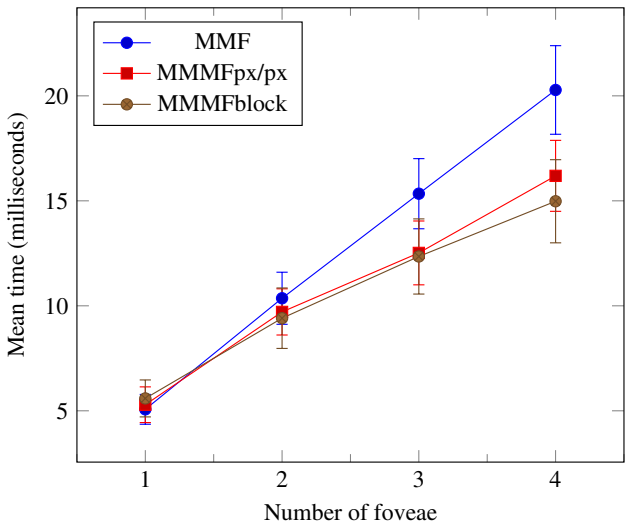



**Fig. 17** Foveae around lena's center: mean time (milliseconds) and standard deviation for obtaining keypoints

#### 5.2.5 Foveae around the center of Lena's image

Similar to the tests of Sect. 5.2.2, the foveae are distributed around the center, ensuring that they are close enough, but in such a way that there is no intersection in the last level. For this experiment, the foveae were distributed at $(-30, -30)$, $(30, 30)$, $(30, -30)$, and $(-30, 30)$, respectively. As shown in Figs. 17 and 18, the results are similar to the one presented for the chessboard image.

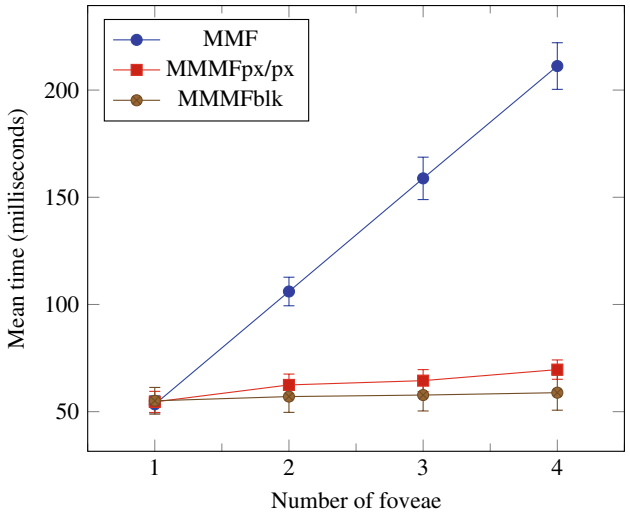



**Fig. 18** Foveae around lena's center: mean time (milliseconds) and standard deviation for computation the descriptors

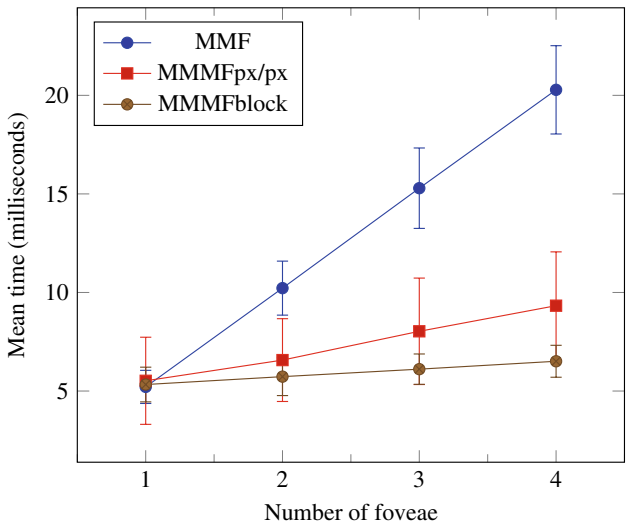



**Fig. 19** Foveae on lena center: mean time (milliseconds) and standard deviation for obtaining keypoints

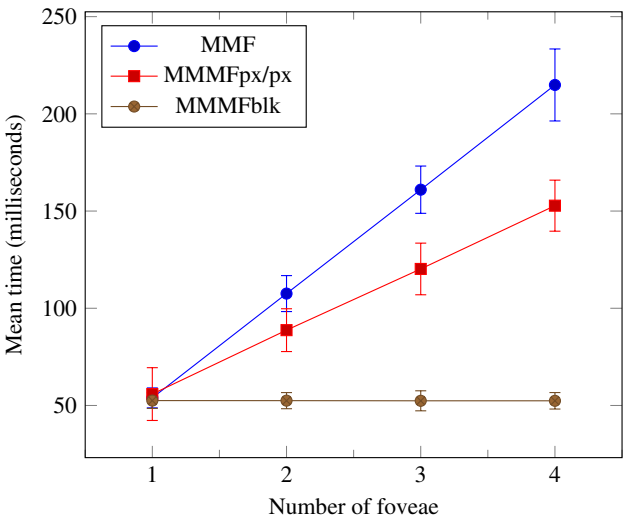



**Fig. 20** Foveae around lena's center: mean time (milliseconds) and standard deviation for computation the descriptors

**Fig. 21** Canny feature extraction without foveation

#### 5.2.6 Foveae in the center of Lena's image

Likewise in Sect. 5.2.3, all foveae are positioned in the center (0, 0) of the image, that is, with total overlapping. By looking at Figs. 19 and 20, it is possible to see that the results were similar to those obtained with the chessboard image.

### 5.3 Applying the approaches with the Canny feature detector

In this section, we show results for the Canny feature extraction technique [50] to understand how our techniques behave with a different feature extraction method that involves much less computations. The experiments reported here are done using a 2.3GHz Intel Core i5 computer with 4GB of RAM and Linux Ubuntu 14.04 operating system, which is the same machine used in the SURF experiments. Just to illustrate, the resulting image without foveation is shown in Fig. 21, with foveation using the original MMF method in Fig. 22), with MMF using simple re-execution in Fig. 23) and using the sending blocks approach in Fig. 24. For all tests, we consider

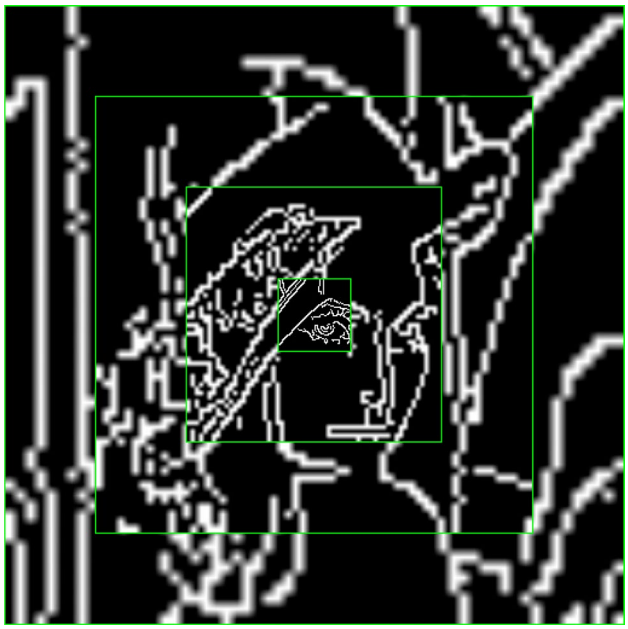


**Fig. 22** Canny feature extraction at multiresolution levels of the MMF method

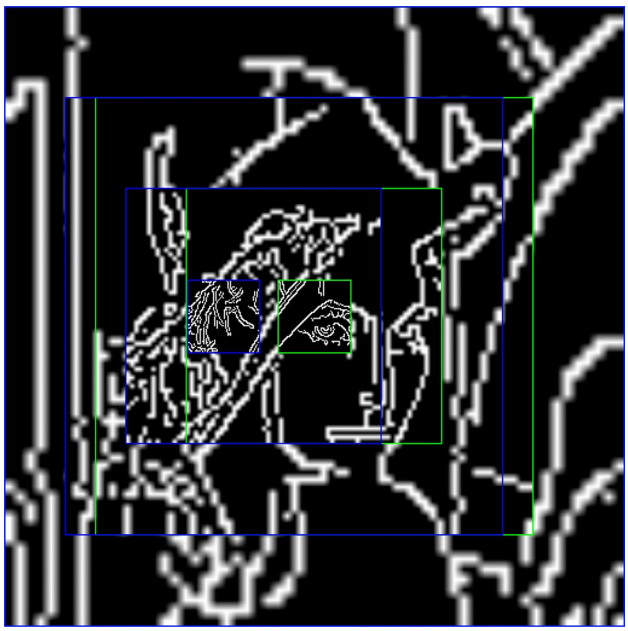


**Fig. 23** Canny feature extraction at multiresolution levels of the MMF approach with re-execution

a Sobel kernel of size 3 and the low and high thresholds of the Canny detector equal to 100 and 200, respectively.

Figures 22, 23, and 24 were obtained considering the size $W = (60, 60)$ with each fovea containing four levels, and it runs the extraction of Canny feature at all levels. The fovea represented by the green color in these pictures was

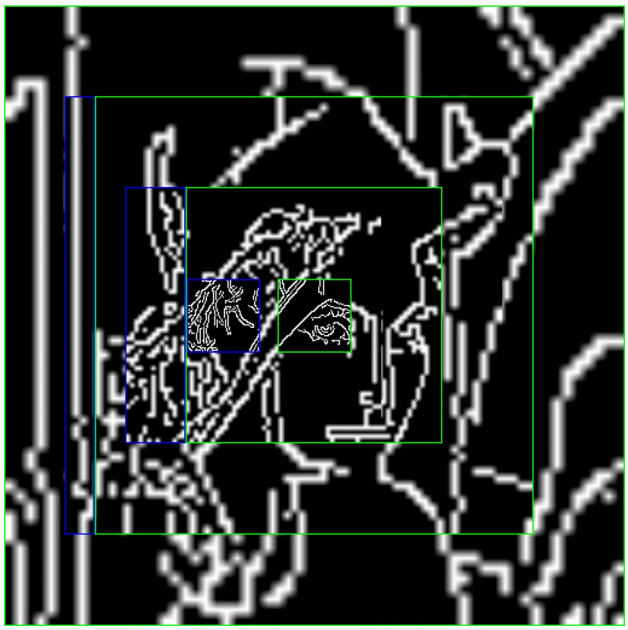


**Fig. 24** Canny feature extraction at multiresolution levels of the MMMF approach with sending blocks

**Table 3** Mean time (milliseconds) to extraction of the Canny feature using Lena's image without foveation, foveated with MMF, multifoveated by MMF with re-execution and MMMF with sending blocks

| Method | Mean time (milliseconds) |
|---|---|
| Original Image | 2.099 |
| Original MMF | 1.682 |
| MMF with re-execution | 2.532 |
| MMMF with sending blocks | 1.882 |

positioned in center of the image (0, 0). The second fovea, represented by the blue color, was positioned to left $(-75, 0)$.

Notice the difference in the color representation of the levels in Figs. 23 and 24, because during the processing of MMF with re-execution, the second fovea ends up processing the content already processed by the first fovea. In turn, the sending blocks' approach represented by Fig. 24 only processes the regions that are not processed by the previous fovea.

The mean time, in milliseconds, for extraction of the Canny feature in each case represented by Figs. 21, 22, 23, and 24 can be seen in Table 3. It is possible to observe that the mean time is much smaller when compared to the SURF feature extraction and there is a behavior similar to the results obtained in the previous tests using the SURF feature. In other words, the mean time obtained by Canny feature extraction in Lena image without foveation is much greater than the mean time obtained by Canny feature extraction at

levels of multiresolution with MMF as already suggested in the literature [2, 3, 8, 14, 24]. It is also noteworthy that the mean time obtained by Canny feature extraction using MMF with re-execution is greater than Canny feature extraction using MMMF with the sending blocks approach, proving that our approach maintains its behavior even after changing features. As suspected, these experiments show that the approaches proposed in this article have results that are very close or similar to those obtained through the use of SURF features, which is known to have more complex calculations. In addition, notice that in robotics vision and other real-time applications that relies on feature extraction, in general, much more than a single feature is extracted, some times several tens of them [3].

## 6 Conclusion

In this paper, we have proposed a novel formulation and implementation of two multifoveation approaches to support the execution of applications that require the use of multiple foveae in real time. This is specially useful in the tracking of multiple targets that include (dynamical) motion of multiple foveae positions from one frame to another. As observed in the literature, all existing methods use the re-execution of the single foveation technique to allow performing multifoveation in 2D images. Nonetheless, we noticed that single foveation, when applied in a sequence to the MMF method, causes redundancies in the extraction and computation of features. To deal with this problem, basically, we extend the MMF approach [24] removing the redundancy observed in the computations. To remove redundancy, we use two approaches, one using a pixel-by-pixel verification and the other based on sending information about redundant region blocks, which depends on the number of levels and distances between the several foveae.

The two approaches proposed in this work are limited to structures that contain the same amounts of levels and the same dimensions of the fovea structures. This limitation can be overcome by rewriting the algebraic equations introduced in this article. Yet, we noticed that this does not affect their usage or effectiveness, as developed. In addition to the limitations of this multifoveated model, we also observed the precedence between the foveae, regarding the extraction of features in the lower layers, since the model only considers the positions of the foveae.

Multifoveation can be exploited in several applications, providing more than one point with visual acuity in the image at the same time. Thus, tasks such as search for specific objects among other objects (distractors) in an image, tracking of several objects at the same time, and simultaneous bottom-up and top-down attentions are some of the practical tasks that can benefit from our technique. With decreasing of processing, other real-time and low-cost solutions can be devised relying on the usage of visual processing and not only on single types of sensors, thus allowing devices and robots to work more effectively with real-time responses. A possible enhancement on this work is to provide a way for removing or adding foveae in the structure. Functions for insertion, changing position, or removal of foveae would be necessary to be introduced in the current methodology. Such functions can be used in a higher hierarchical level, allowing to perform context-based attention tasks, for instance.

Therefore, as future goal, we intend to work on the implementation using OpenMP to use multiple cores and the CUDA library for the Xavier nVidia board. One can notice that parallelizing the block-based approach is a straightforward task as the blocks sizes can be computed previously and passed to each thread. Nevertheless, to better do that, we are analyzing the parallelism of the whole foveae structures, in a function of parameters as the number of levels and image pixels. In addition, the inclusion of attention in the earlier steps of deep learning is planned to be implemented in a dedicated architecture with the GPU (Xavier from nVidia) to speed up processing in these kinds of devices. This future implementation is planned to work as the vision-processing system embarked in our wearable glove device [28], and also in payload applications in our robotic platforms, as drones and a robotic sailboat [29].

**Acknowledgements** We would like to thank the Brazilian Sponsoring Agency for Higher Education (CAPES) for sponsoring this project (88887.091733/2014-01) and the doctorate fellowship of Petrucio R. T. Medeiros (88882.145745/2017-01), and also the Brazilian Agency for Research Sponsoring (CNPq) for the research grant of Luiz M. G. Gonçalves (8546600060032961).



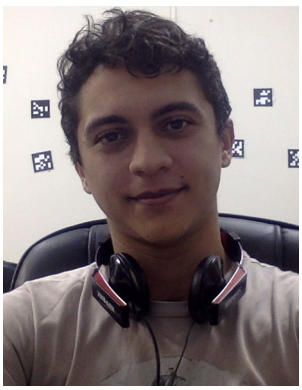
## References

1. Camacho, P., Arrebola, F., Sandoval, F.: Adaptive multifovea sensors for mobile tracking. In: 1998 IEEE International Conference on Electronics, Circuits and Systems. Surfing the Waves of Science and Technology (Cat. No.98EX196), Vol. 2, 1998, pp. 449–452
2. Goncalves, L.M.G.: A robotic control system for integration of multimodal sensory information (in portuguese), Ph.D. thesis, Federal University of Rio de Janeiro (1999)
3. Garcia, L.M., Oliveira, A.A.F., Grupen, R.A., Wheeler, D.S., Fagg, A.H.: Tracing patterns and attention: Humanoid robot cognition. IEEE Intelligent Systems **15**(4), 70–77 (2000)

4. Dhavale, N., Itti, L.: Saliency-based multi-foveated mpeg compression. In: Proceedings of Seventh International Conference on Signal Processing and its Applications, IEEE, 2003, pp. 229–232
5. Itti, L.: Automatic foveation for video compression using a neurobiological model of visual attention. IEEE Transactions on Image Processing **13**(10), 1304–1318 (2004)
6. Yeasin, M., Sharma, R.: Foveated Vision Sensor and Image Processing - A Review. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
7. Segundo, J.S.S., Bezerra, J.P.A., Silveira, R.W., Goncalves, L.M.G.: Development of a multiresolution stereo vision system in real time (in portuguese). In: IEEE Latin American Robotics Symposium, SBA, São Luís, MA, Brazil, 2005, pp. 1–8
8. Gomes, R.B., Goncalves, L.M.G., Carvalho, B.M.: Real time vision for robotics using a moving fovea approach with multi resolution. In: IEEE International Conference on Robotics and Automation, IEEE, Pasadena, CA, USA, (2008), pp. 19–23
9. Butko, N.J., Movellan, J.R.: Infomax control of eye movements. IEEE Transactions on Autonomous Mental Development **2**(2), 91–107 (2016)
10. Butko, N.J., Movellan, J.R.: Detecting contingencies: An infomax approach. IEEE Transactions on Autonomous Mental Development **23**(8–9), 973–984 (2010)
11. Butko, N.J.: Active perception, Ph.D. thesis, University of California, San Diego (2010)
12. Gomes, R.B.: Feature selection guided by visual attention in images with fovea (in portuguese), Ph.D. thesis, Universidade Federal do Rio Grande do Norte (2013)
13. Piao, J.-C., Jung, H.-S., Hong, C.-P., Kim, S.-D.: Improving performance on object recognition for real-time on mobile devices. Multimedia Tools and Applications **75**(16), 9623–9640 (2016)
14. Gomes, R.B., de Carvalho, B.M., Gonçalves, L.M.G.: Visual attention guided features selection with foveated images. Neurocomputing **120**, 34–44 (2013). special issue on Image Feature Detection and Description
15. Oliveira, F.F., Souza, A.A.S., Fernandes, M.A.C., Gomes, R.B., Goncalves, L.M.G.: Efficient 3d objects recognition using multifoveated point clouds. Sensors **18**(7), 1–27 (2018)
16. Oliveira, F.F., Goncalves, L.M.G., Fernandes, M.A.C., Gomes, R.B.: Efficient recognition of multiple 3d objects in point clouds with a multifoveation approach. In: Conference on Graphics, Patterns and Images, 31 (SIBGRAPI), SIBGRAPi, Foz do Iguaçu, PR, Brazil, (2018), pp. 1–6
17. Traver, V.J., Bernardino, A.: A review of log-polar imaging for visual perception in robotics. Robotics and Autonomous Systems **58**(4), 378–398 (2010)
18. Patney, A., Salvi, M., Kim, J., Kaplanyan, A., Wyman, C., Benty, N., Luebke, D., Lefohn, A.: Towards foveated rendering for gaze-tracked virtual reality. ACM Trans. Graph. **35**(6), 179:1–179:12 (2016)
19. Lee, S., Cho, J., Lee, B., Jo, Y., Jang, C., Kim, D., Lee, B.: Foveated retinal optimization for see-through near-eye multi-layer displays. IEEE Access **6**, 2170–2180 (2017)
20. Hsu, C.-F., Chen, A., Hsu, C.-H., Huang, C.-Y., Lei, C.-L., Chen, K.-T.: Is foveated rendering perceivable in virtual reality?: Exploring the efficiency and consistency of quality assessment methods. In: Proceedings of the 25th ACM International Conference on Multimedia, MM '17, ACM, New York, NY, USA, 2017, pp. 55–63
21. Meng, X., Du, R., Zwicker, M., Varshney, A.: Kernel foveated rendering. Proc. ACM Comput. Graph. Interact. Tech. **1**(1), 5:1–5:20 (2018)
22. Sankara, S., Ansari, R., Khokhar, A.: Adaptive multifoveation for low-complexity video compression with a stationary camera perspective. In: Proceedings of SPIE - The International Society for Optical Engineering, Vol. 5685, SPIE, San Jose, CA, United States, (2005), pp. 1007–1018
23. Hunsberger, E., Osorio, V.R., Orchard, J., Tripp, B.P.: Feature-based resource allocation for real-time stereo disparity estimation. IEEE Access **5**, 11645–11657 (2017)
24. Gomes, R.B., Gardiman, R.Q., Leite, L.E.C., Carvalho, B.M., Goncalves, L.M.G.: Towards real time data reduction and feature abstraction for robotics vision. Robot Vision 345–362 (2010)
25. Benjamim, X.C., Gomes, R.B., Burlamaqui, A.F., Goncalves, L.M.G.: Visual identification of medicine boxes using features matching. In: 2012 IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS) Proceedings, (2012), pp. 43–47
26. Cavallaro, A., Hands, D., Popkin, T.: Multi-foveation filtering, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP), Vol. 1, IEEE, (2009), pp. 669–672
27. Camacho, P., Arrebola, F., Sandoval, F.: Multiresolution sensors with adaptive structure. In: IECON '98. Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.98CH36200), Vol. 2, (1998), pp. 1230–1235
28. Aroca, R.V., Gomes, R.B., Dantas, R.R., Calbo, A.G., Goncalves, L.M.G.: A wearable mobile sensor platform to assist fruit grading. Sensors **13**(5), 6109–6140 (2013)
29. Santos, D., Junior, A.G.S., Negreiros, A., Boas, J.V., Alvarez, J., Araujo, A., Aroca, R.V., Goncalves, L.M.G.: Design and implementation of a control system for a sailboat robot. Robotics **5**(1), 1–24 (2016)
30. Uhr, L.: Layered "recognition cone" networks that preprocess, classify, and describe. IEEE Transactions on Computers **C–21**(7), 758–768 (1972)
31. Adelson, E.H., Simoncelli, E.: Orthogonal pyramid transforms for image coding. Visual Communications and Image Processing **II**(845), 50–58 (1987)
32. Burt, P., Adelson, E.: The laplacian pyramid as a compact image code. IEEE Transactions on Communications **31**(4), 532–540 (1983)
33. Adelson, E.H., Anderson, C.H., Bergen, J.R., Burt, P.J., Ogden, J.M.: Pyramid methods in image processing. RCA Engineer **29**(6), 33–41 (1984)
34. Lindeberg, T.: Feature detection with automatic scale selection. International Journal of Computer Vision **30**(2), 90–95 (1989)
35. Gomes, H.M., Fisher, R.B.: Primal sketch feature extraction from a log-polar image. Pattern Recognition Letters **24**(7), 983–992 (2003)
36. Mallat, S.G.: A theory for multiresolution signal decomposition: The wavelet representation. IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(7), 674–693 (1989)
37. Lindeberg, T.: Discrete scale-space theory and the scale-space primal sketch, Ph.D. thesis, Royal Institute of Technology (1991)
38. Lindeberg, T.: Time-causal and time-recursive spatio-temporal receptive fields. Journal of Mathematical Imaging and Vision **55**(1), 50–88 (2016)
39. Bezerra, J.P.A.: A vision system for robust navigation of a semi-autonomous robotic platform (in portuguese), Ph.D. thesis, Federal University of Rio Grande do Norte (2006)
40. Katsuki, F., Constantinidis, C.: Bottom-up and top-down attention: Different processes and overlapping neural systems. The Neuroscientist **20**(5), 509–521 (2014), pMID: 24362813. https://doi.org/10.1177/1073858413514136
41. Desimone, R., Duncan, J.: Neural mechanisms of selective visual attention. Annual Review of Neuroscience **18**, 193–222 (1995)
42. Dario, P., Bergamasco, M., Fiorillo, A., Leonardo, R.: Geometrical optimization criteria for the design of tactile sensing patterns. In: IEEE International Conference on Robotics And Automation, IEEE, (1986), pp. 1268–1273

43. Lim, F.L., West, A.W., Venkatesh, S.: Tracking in a space variant active vision system. In: Proceedings of the 13th International Conference on Pattern Recognition, Vol. 1, IEEE, Vienna, (1996), pp. 745–749
44. Basu, A., Sullivan, A., Wiebe, K.J.: Variable resolution teleconferencing. In: Proceedings of IEEE Systems Man and Cybernetics Conference—SMC, IEEE, Le Touquet, France, (1993), pp. 170–175
45. Basu, A., Wiebe, K.J.: Videoconferencing using spatially varying sensing with multiple and moving foveae. In: Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 2—Conference B: Computer Vision and Image Processing, IEEE, Jerusalem, Israel, (1994), pp. 30–34
46. Basu, A., Wiebe, K.J.: Enhancing videoconferencing using spatially varying sensing. IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans **28**, 137–148 (1998)
47. Rodríguez, J.A., Urdiales, C., Bandera, A., Sandoval, F.: Nonuniform video coding by means of multifoveal geometries. International Journal of Imaging Systems and Technology **12**, 27–34 (2002)
48. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. IEEE Access **5**, 11645–11657 (2006)
49. Pioppo, G., Ansari, R., Khokhar, A.A., Masera, G.: Low-complexity video compression combining adaptive multifoveation and reuse of high-resolution information. In: Proceedings of the International Conference on Image Processing, (ICIP), (2006), pp. 3153–3156
50. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) **8**(6), 679–698 (1986)



**Petrucio R. T. Medeiros** holds a BSc degree in Computer Engineering and a MSc in Electrical and Computer Engineering, both from the Federal University of Rio Grande do Norte (UFRN). Currently, he is PhD student at UFRN. His main interests are computer vision and image processing.

**Rafael B. Gomes** holds a BSc degree in Computer Science, and a MSc and PhD degrees in Electrical and Computer Engineering, all from the Federal University of Rio Grande do Norte (UFRN), Brazil. Currently, he is a Professor at the Department of Informatics and Applied Math at UFRN. His main interests are computer vision, computer graphics and image processing.

**Esteban Clua** holds a BSc in Computer Science and a MSc and PhD in Informatics from Pontifical Catholic University of Rio de Janeiro. Currently, he is a professor at the Computer Science Institute of the Fluminense Federal University. His main interests are in Computer Graphics and in Games & Entertainment. He is an nVidia Fellow since 2015 and has worked for many years with GPUs.

**Luiz Gonçalves** holds a Doctorate in Systems and Computer Engineering (1999) from COPPE-UFRJ, Brazil, including a two years PhD stage at the Laboratory for Perceptual Robotics of UMASS, Amherst, MA, USA. He is Full Professor at the Computer Engineering Department of UFRN, Brazil. He has done researches in the several aspects of Graphics Processing including fields as Robotics Vision (main interest), Computer Graphics, GIS, Geometric Modelling, Animation, Image Processing, Computer Vision, and also has several works on Robotics in Education.