Visual attention guided features selection with foveated images

Rafael Beserra Gomes a,⁎, Bruno Motta de Carvalho b, Luiz Marcos Garcia Gonçalves a

a Computing Engineering and Automation Department, Federal University of Rio Grande do Norte, Brazil
b Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte, Brazil

Abstract

Visual attention is a very important task in autonomous robotics, but, because of its complexity, the processing time required is significant. We propose an architecture for feature selection using foveated images that is guided by visual attention tasks and that reduces the processing time required to perform these tasks. Our system can be applied in bottom-up or top-down visual attention. The foveated model determines which scales are to be used on the feature extraction algorithm. The system is able to discard features that are not extremely necessary for the tasks, thus, reducing the processing time. If the fovea is correctly placed, then it is possible to reduce the processing time without compromising the quality of the tasks’ outputs. The distance of the fovea from the object is also analyzed. If the visual system loses the tracking in top–down attention, basic strategies of fovea placement can be applied. Experiments have shown that it is possible to reduce up to 60% the processing time with this approach. To validate the method, we tested it with the feature algorithm known as speeded up robust features (SURF), one of the most efficient approaches for feature extraction. With the proposed architecture, we can accomplish real time requirements of robotics vision, mainly to be applied in autonomous robotics.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Providing an artificial visual system to a robot is an important step to implementing autonomous robotics applications. Moving towards such applications, the idea is to build robots that can perform tasks without human intervention, reacting quickly, if necessary, in response to stimuli observed from the environment. One possible approach to implementing such artificial visual systems is to perform a complete analysis and image indexing, so that image processing tasks that give useful information to the decision making process can be performed with the full data. On one hand, little effort has been done for building system that integrates bottom–up and top–down stimuli, such as also overt and covert attention and how the early vision can be developed so that the whole system performs successfully.

The problem becomes even more critical when the goal is to extract visual information (or features) that are useful for the several general-purpose activities of a robot. Recent works have showed that feature detection and description usage in many applications mainly involve the attention and recognition tasks. Among widely known feature descriptors are: SIFT [1,2], SURF [3], Daisy [4] and GIST [5].

Speeded up robust features (SURF) is one of the most reliable feature extraction algorithms, which is inspired on the previous scale invariant feature transform (SIFT), but with some modifications. The feature descriptors are extracted so that they uniquely identify the region where they are. In this way, it is possible to assign a descriptor to a unique point at the scene. The success of such tasks depends on how well these descriptors can represent their respective features. Descriptors with higher entropy values represent regions very well, thus, allowing higher tasks accuracies. Once the features are detected, several tasks can be performed, such as tracking, 3D reconstruction, simultaneous localization and mapping (SLAM), and object recognition, between other robot tasks. However, the execution times associated with such descriptors are usually much higher than descriptors with lower entropy values, that, on the other hand, do not represent their regions very well.

The processing time associated with the image processing tasks that have to be performed in real time is one of the major drawbacks to the usage of reliable and robust machine vision algorithms in autonomous robotics applications. While visual data acquired by video capturing devices can be quickly grabbed in real time, their processing is a bottleneck for the completion of such systems, unless one uses dedicated image processing architectures, which are usually very expensive. To experimentally exemplify this challenge, we performed some computer vision tasks involving different processing with a digital image of size 320×240 using a Laptop AMD Turion Dual Core 2 GHz. The execution times obtained can be seen in Table 1. It is easy to see

* Corresponding author. Tel.: +55 8432153771.
E-mail addresses: rafaelufrn@gmail.com (R. Beserra Gomes), motta@dimap.ufrn.br (B. Motta de Carvalho), lmarcos@computer.org (L. Marcos Garcia Gonçalves).

0925-2312/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.10.033
that if achieving a rate of 30 frames per second (FPS) is a requirement, the processing time of the tasks listed below is far from being labeled real-time.

The high computational processing cost associated with such tasks suggests that other less computationally expensive approaches should be used, such as biologically inspired systems using reduced image representations. Towards this goal, the system model proposed in this paper is somewhat biologically inspired in the sense that it uses a foveated image, which can be moved according to a visual attention module. The fovea, guided by visual attention, determines the density of features extracted as a function of the distance between a region and the center of the fovea. In other words, the fovea region contains more features than the peripheral zones. We will show that, by representing the input images in this way, it is possible to reduce the processing time, without compromising the quality of the tasks outputs.

A point that we want to emphasize is that we do not want a system designed to perform specific tasks. We want a behaviorally cooperative and active system that can perform several, different tasks in different environments or situations, automatically responding, in real-time, to environment changes. In this way, we believe that data reduction and feature abstraction are the main keys of the system, allowing recognition or an on-line weight tuning (attention) integrating the features extracted from sensory information, according to the task being executed. Thus, the biologically inspired model for feature extraction proposed in this paper, that has allowed us to develop a system with these requirements, is the main issue treated in this work.

If the biologically inspired model proposed in this paper can reduce visual data, and, as a consequence, the execution times of image processing tasks, then, the main remaining question is: how can one adapt a feature detector and descriptor to a foveated model? We describe in details how SURF are extracted in a very fast fashion from the foveated images. As a result of this adaptation, in order to analyze the performance of the foveated system, we also address the issue of how the visual attention influences the feature extraction parameters and performance.

This paper is organized as follows. Section 2 describes related works on visual attention and features extraction. Section 3 describes the foveated model that is used, and some additional formalization for this work and a framework for visual attention tasks, specially for tracking. Section 4 shows several experiments using this model and finally Section 5 presents the conclusions and some possible future work.

### 2. Related works

The usage of full resolution images in vision systems fairly complicates the feature extraction processing. Several models have been proposed in the literature for image data reduction and feature abstraction. Some of these models treat visual data as a classical pyramidal structure. This multi-resolution structure is used in visual search, as proposed by Leonard Uhr [6], while the Laplacian pyramid is introduced by Burt and Adelson [7]. The scale space theory used here is formalized by Witkin [8] and enhanced by Lindeberg [9]. The usage of multi-resolution for visual attention is fostered by Tsotsos [10,11] and Burt [12], and has been used in several attention models, such as the works of Sandon [13,14], Tsotsos [15] and Itti et al. [16], among others.

Several studies suggesting the use of multi-features for selective attention and recognition in biological mechanisms have been proposed, such as the work of Treisman [17]. In latter works, Treisman [18,19] provides a better description of his model for low-level perception, with the existence of two distinct phases in visual information processing: a parallel (simultaneous) low-level feature extraction followed by a sequential processing of the selected regions. Tsotsos [15] also depicts an interesting approach to visual attention based on selective tuning.

A problem when calculating the classical pyramid is that the time spent in this operation does not allow its real-time implementation for robotics purposes, at least when using conventional architectures. In fact, most of the above works consider using only stationary, monocular image frames [20–23] or post-processed sequences of images [24–26], not including temporal aspects like motion or functional and behavioral (real-time) aspects. Moreover, these approaches do not provide real-time feedback to environmental stimuli. That is, do not explicitly deal on-line with the real-time constraints experimented in robotics applications.

There are other approaches that represent the image with non-uniform density. Most of them are inspired on foveated human vision. The foveated model can be computed, for example, using wavelets [27] or space variant sensors [28]. The space variant methodologies [29] use a system where the virtual sensors have non-uniform distribution, usually distributed in a way such that there are more sensors at the fovea region. A log-polar transformation reproduces the image in a new space that tries to reproduce the retina mapping to the visual cortex, resulting in a more compact image [30,31]. Other applications use a wide angle camera for peripheral vision and another one for foveated vision [32].

Unlike what happens at the human eyes, some of these data reduction models allow a free movement of the fovea and do not have only applications in robotic vision, but also in real-time video coding/decoding and transmission [27,33–38]. In the scope of this transmission application, the clients order images that usually have high resolution, on a server. The communication between client and server can be limited, for example, by the connection bandwidth [27]. The fovea position on the images can also be synchronized with the human eye [34]. In this case, a system with a camera keeps information about the current position of the client eyes and changes the image fovea position to the point that the user is looking at. In this way, the user can perceive the image as if there had not been peripheral resolution loss.

When it comes to video transmission, it is feasible to wait a few seconds for the first frame, as long as the target transmission rate is achieved, causing a small lag. However, this can cause serious problems in other applications. For example, in robotics applications where a real-time system with delay restrictions is used, if the system has to wait a non-negligible time before taking a decision, it can critically compromise the whole system. We use our foveated model [39] that was shown to be very fast and easy to implement.

Bottom–up attention refers to the attention caused by stimulus from the environment. A stone that is thrown in the direction of a person, for example, diverts his/her attention to the danger that is coming. A usual way of making the computer aware of these stimuli coming from the environment is to use salience maps. Koch and Ullman [40] propose the use of salience maps, by performing peak selection within those maps, and successive fixations via inhibition of return. Other works have been directed

### Table 1

<table>
<thead>
<tr>
<th>Task</th>
<th>Time per frame (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Four filters</td>
<td>6</td>
</tr>
<tr>
<td>Saliency map</td>
<td>45</td>
</tr>
<tr>
<td>SURF calculation</td>
<td>600</td>
</tr>
<tr>
<td>Stereo disparity</td>
<td>1100</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1751</strong></td>
</tr>
</tbody>
</table>
towards producing reasonable mathematical models for the defini-
tion of low-level features. For example, Van der Laar [20], Itti
[21], Vandapel [24], Luo [23], and Milanesi [25] propose the use of
a transfer function to gather information from the feature maps to
construct a salience map that controls attention. Butko proposes a
fast approximation to a Bayesian model of visual saliency in order
to reduce computational cost 41.

Cheng et al. [42] propose the usage of two saliency maps that are
based on global contrast differences, with one of them also
taking into account spatial coherence as well as global contrast
differences. Both techniques produce full resolution saliency maps
and compared favorably against other existing saliency detection
methods.

Another approach, proposed by Li et al. [43] proposes a
bottom–up paradigm for detecting visual saliency that performs
a scale-space analysis of the amplitude spectrum of natural
images. They apply low-pass Gaussian kernel to the image ampli-
titude spectrum and use the result as an image saliency detector.
The saliency map can then be obtained by reconstructing the
image using the original phase and the amplitude spectrum, but
filtered at the scale selected by the minimal saliency map entropy.
Another approach based on frequency domain analysis is the one
proposed by Guo adn Zhang [44]. They propose a multisolution
spatiotemporal saliency detection model called phase spectrum of
quaternion Fourier transform (PQFT), that computes the spatio-
temporal saliency map of an image by processing its quaternion
representation. One advantage of this model is that the usage of
motion allows the phase spectrum to code spatiotemporal sali-
cy, thus allowing the method to perform attention selection for
images and also for videos [44].

On a different approach, Rybak et al. [22] treat perception and
cognition as behavioral processes. Several behavior work simulta-
neously, each one with a predicate that can enable or disable
them. Westelius [45], using a simulation platform for a robot,
proposes an interesting and relatively complete approach. The
algorithm turns the regions visited invisible to the attention
seeking mechanism in an internal attention map.

In a purely descriptive work, Kosslon [46] suggests that features
extracted from visual images are combined with mental image
completion for recognition purposes, while Lindeberg [47] detects
features using an automatic scale selection algorithm. Lowe
proposed the use of the scale-invariant feature transform (SIFT)
for recognition purposes [1,2], dealing with detection of features
that are invariant to scale, while Young [48] suggests the use of
retinal mechanisms based on Gaussian derivatives for spatial
vision. Basically, the image is transformed into a large set of
feature vectors obtained based on the maximal and minimal of
Gaussian differences in several scales. The resulting features are
known to be invariant (to some degree) to scale, translation,
rotation, and illumination changes.

The idea of using iconic representations (based on Gaussian
filters and multi-scale) and salience is explored by Rao and Ballard
[49] to control attention. Connecting an attention model to a robot
head is first done by Clark and Ferrier [50], and a more extensive
model, that includes change detection, was later proposed by
Tsotos et al [15]. Speeded up robust features (SURF) was proposed
by Bay [3] to be used also in 3D reconstruction besides recognition.
It is inspired in the SIFT, but introduces the concept of integral
images to compute an integer approximation of the determinant
of the Hessian matrix. Haar wavelet responses are used as features
also computed using integral images. Furthermore, recent works
have improved aspects of visual features extraction [51] and their
matching [52].

As it was shown in the previous section, besides being fast
when compared to SIFT, for example, it is still time consuming to
calculate SURF in the whole scale space (pyramidal) image. In our
approach, we do not use this kind of structure but a more compact
one that can be derived from it. We describe in our approach how
SURF can be computed over our much more compact representa-
tion. Surprisingly, while reducing the amount of features their
quality are kept, as described in the experiments.

3. Feature selection with foveated images

As mentioned before, we use our foveated model [39] as the
key structure for reducing data. In this paper, we use the same
notation but add other variables for providing more flexibility in
the proposed feature selection model. The proposed architecture
for features selection (see Fig. 1) in this paper is mainly based on
SURF features, but any feature that can be extracted at multiple
scales can be applied using the same ideas.

3.1. Foveated model

The foveated model proposed by Gomes [39] transforms an
image of size $U$ into a set of $m+1$ small images of size $W$ each
one. This set composes a foveated image. The model defines image
patches $A_k$ of size $S_k$ from the original image, with $k=0,1,...,m$.
These patches are arranged in a sequence of $m+1$ levels. The first
level (level 0) is a mapping of the whole original image. The last one
(level $m$) is a mapping of a patch placed at the original image that
has the same size of each image level, i.e., size $W$ (see Figs. 2–4).

The patch of the last level is guided by a fovea $F$ at $A_m$ center.
For formalization convenience, the fovea coordinate system is
$(0,0)$ at the center of the image. Let $\delta_k = (\delta_k x, \delta_k y)$ be the displace-
ment of patch $A_k$ then $(\delta_k x, \delta_k y) = (0,0)$ and $S_m+S_m/2=F$, where
$F=F–U/2$ and $F$ is the fovea $F$ at original image coordinate
system.

Using linear interpolation with these restriction, the displace-
ment of each patch is given by

$$\delta_k = \frac{k(U–W+2F)}{2m}$$

(1)

Note that $\delta_k$ is defined only for $m>0$; in other words, the foveated
model should have at least two levels.

The size of first patch is $S_0 = U$ and the size of the last patch is
$S_m = W$ by definition. Using linear interpolation with these restric-
tion, the size of each patch $A_k$ is given by

$$S_k = \frac{kw(kU+mU)}{m}$$

(2)
3.2. Feature selection

The foveated model changes the density of features through the levels. Depending on the task at hand, some levels can even be discarded. In order to do that, we define a vector $\eta$ to obtain different feature densities and a vector $B$ to discard levels.

The vector $\eta$ contains the octave of each level (the scale interval is obtained by using the power of 2 of that octave). Similarly to what is done in SURF [3], the filter size is changed, instead of re-sampling the original image to a specific scale. Each scale has a different frequency. In the original SURF paper [3], all filters in the same octave have the same spacial frequency, even though they belong to different scales. The vector $B$ is a binary vector, where a value 1 at the $i$-th position indicates that the $i$-th level will be used and 0 indicates the $i$-th level will be discarded.

In this paper, each foveated level contains scales from the same octave. For simplification purposes, the vector $\eta$ has the correspondent octave for each foveated level. For example, if the model has four levels, then the vector $\eta = (4,3,2,1)$ indicates that the features from the foveated level 0 are at fourth octave, from the level 1 are at the third octave, from the level 2 are at the second octave and, finally, from the level 3 are at the first octave. In the SURF paper [3], the samples for the $p$-th octave have a step of $2^p$ pixels. For example, at the second octave ($p=1$), the sampling step is of 2 pixels. An example of that model can be seen in Fig. 5.

We affirm that it should be possible to discard levels. So, for example, in the following configuration: a foveated model with eight levels, $B = \{1,0,0,1,0,1,1\}$ and $\eta = \{4,0,0,3,0,2,2,1\}$, i.e., there are four levels scanning from the fourth to the first octave, and in the vector positions for the levels that are not available ($B_i = 0$), the octave order number is not set ($\eta_i = 0$).

Since each Hessian filter has a different size, the ideal is that their centers coincide, so that the non-max suppression is applied to the same pixel. In order to make the centers coincide, it is...
enough to start at the position where all filters do not come out of the image. Then, all filters can be equally sampled (because they are at the same octave) with coincident filters. The Hessian filter should be limited by the foveated model regions. This means that all filters should start from the central pixel of the biggest filter, delimited by the current foveated level. Fig. 6 illustrates this idea.

Here, we introduce a growth fovea factor \((G_x, G_y)\) (see Fig. 7). As will be more clear in Section 4, this factor increases the levels’ feature density by enlarging their areas. Observe that this model behaves like there is no foveation when \(G\) goes to \(\infty\) (Fig. 8).

By adapting the equations from the foveated model \([39]\), the starting pixel is the maximum between the results of Eqs. (3) and (4), where \(H(k) \times H(k)\) is the size of the biggest Hessian filter at level \(k\) (e.g., the fourth Hessian if there are four filters per octave). In the same way, the Hessian filter scanning should be limited to the foveated level. Then, the central pixel should not exceed the minimum between the results of Eqs. (5) and (6)

\[
\left(\delta x_k + \frac{H(k)}{2} - G_x, \delta y_k + \frac{H(k)}{2} - G_y\right) \\
\left(\frac{H(k)}{2} - \frac{H(k)}{2}\right) \\
\left(\delta x_k + S_k - \frac{H(k)}{2} + G_x, \delta y_k + S_k - \frac{H(k)}{2} + G_y\right) \\
\left(U_x - \frac{H(k)}{2}, U_y - \frac{H(k)}{2}\right)
\]

3.3. Top–down attention

In most of top–down attention tasks it is important to keep track of the object. In this way, we propose to select only part of the features that seem to be important in keeping up with the tracking. Features can exist at different scales, and if the object is almost parallel to the camera plane, then the object features that will be matched can be estimated. For example, if the object is near the camera, the lower scale features from the object are matched with the high scales features from the model. On the other hand, if the object is far from the camera, the high scale features from the object are matched with the low scales features from the model.

The tracking flow chart is shown in Fig. 10, and the tracking module uses the foveated model. If the object is still detected, then the fovea is moved to a new position. In this experiment, the detected object center is set as the new fovea position. Otherwise, we need to recover the fovea position. Several strategies can be applied in order to recover the tracking. Four strategies are shown on the flow chart. The so-called based on the last \(N\) frames uses the \(N\) last frames to predict a new position. We can use, for example, a weighted average of the last \(N\) frames or apply a Kalmann filter. The easier implementation option is to just disable foveation. In this way, a sudden peak of processing time appears, due to the amount of features, until the object is re-detected. Another option is to increase the fovea growth factor, so the recovery process happens in a gradual way, with the amount of features increasing with the processing time. We can impose a maximum fovea size, so that we do not increase the processing time too much.

One question that emerges in this kind of system is how to know whether the object was detected or not without prior knowledge. In the experiments an object was considered as detected when the difference between the two diagonal of the plane was less than 10% of the biggest diagonal and the average distance between descriptors that was matched is less than 0.20. The descriptors were normalized so that the maximum possible distance is 2. Examples of unsuccessful detection in the proposed system are shown in Fig. 9.
3.4. Bottom–up attention

If there is more than one object, there are two possibilities. Even though it is hard for humans to foveate more than a region at once, it is possible to foveate as many as possible regions via software. This will, of course, increase the processing time if the same parameters are used. One option is to reduce attention or visual information for each object, so that the overall processing time remains in the same range. Another option would be to foveate each object and process them in sequence. The proposed model in this paper uses only one foveation at time. If two objects, for example, ask for top–down attention, then the visual process pay attention to one in a frame and to the another one object in the next frame.

3.5. Fovea position

The fovea position and foveated parameters depend on the task to be implemented. For example, if this task is tracking, then it is most suitable to keep the fovea around the most relevant features of the object. If the features are equally distributed along the object, in a general way, it is better to keep the fovea at the center of the detected object. One problem happens when the visual system loses the fovea. In this case, if the fovea is placed far from the object, the system can become unstable without finding the object. Several strategies can be applied using the foveated model. Another example is an examination of the environment by the visual system. In this case, one can move the fovea around the most salient regions.

A possibility is to use bottom–up attention and move the fovea to the most salient region. Another option is to eliminate the fovea model and compute the features in the whole image for a while. In this case, the extra processing time will cause a decrease in the frame rate, but the fovea can be recovered when the object is found.

4. Experiments

For the experiments described on next, a Laptop Intel Core i5 2.3 GHz with 4 GB memory was used. Some of the experiments performed in this work are done using the ground truth proposed by Zimmermann [53]. For each frame one of the three scenes (a towel, a phone and a mouse pad), there are four corners that indicate where the almost planar object is. Four of the frames for the mouse pad ground truth can be seen in Fig. 11.

In order to do a better analysis, some measurements were collected. The mouse pad video has 6946 frames with 640 × 480 pixels and shows a mouse pad on a cardboard being moved by a person. The object moves from near to far relative to the camera (see Fig. 12) at different speeds (see Fig. 13).

4.1. Foveation distance

This experiment demonstrates how the distance of the fovea from the object center influences its detection. If the fovea is far from the object, it is expected that the object should be detected with less precision. On the other hand, if the fovea is at the object center, it is expected that the model performs as well the non-fovea model.

The image used was the mouse pad of the ground truth proposed by Zimmermann [53]. A Hessian threshold of 200 and a descriptor size of 128 are used. Several experiments were done varying the distance between the center of the object from the ground truth and the fovea. The radius is in the interval from 1 to 300 with a step of 30 pixels. The fovea is placed at one of the assumed eight possible degress (multiples of π/4). If the position is outside the image, the next degree is tried until one of them is possible. As the radius is less than the minimum dimension size of the image, then as least 1° is feasible. The same procedure is used for 18 different models whose differences can be seen in Table 2.

Note that the scale of the object in this ground truth changes along the frames (see Fig. 12). Sometimes the object appears so small that could not be detected with default parameters of SURF,
even without the foveation. If the object appears in large scales, then more robust features or others object detection algorithms should be applied.

The full video with 6946 frames with a step of 30 pixels and also three $B$ vectors are used. On non-foveated images, two $B$ vectors are used and the detection rate is 87% for both. This means that the third and the fourth octave are not essential for object detection, what does not necessarily means that they were not used. The models, the average time and the detection rate for some radius are shown in Table 2. It is possible, in this example, to reduce up to four times the processing time in relation to the non-foveated model with the same $B$ vector, but as the fovea moves far from the object center, the object detection decreases. The velocity of this decreasing depends of the model used as shown in Fig. 14. We believe that at higher scales, the processing time saving can be greater due to relation between features matched and features discarded, but the detection algorithm must be robust enough to detect at these scales.

The ideal is that the object is near the fovea, but if the object is far from the fovea, the model is still able to detect it. The distance tolerance depends also on the appearance size of the object on the image. For example, if the object is far from the camera and appears like 50 pixels wide, then the fovea can be at most say 25 pixels far from the object center for obtaining high detection
probability. On the other hand, if the object is very close to the camera then the fovea distance can be larger without decreasing the detection rate. Another variable that influences this question is the original image size. The greater the difference between the appearance size of the object and the size of the original image, more the foveated model is useful, because more features that would not be matched are discarded.

4.2. Tracking

In this task, an object is shown and the visual system must track it. If the same feature extraction parameters are used in the whole image, then the tracking should be lost if the object is outside the field of view. However, if the foveated model is used, the object can be lost more easily if the fovea is not properly placed, as shown in the first experiment. Then, it is important for the tracking system to be able to recover the fovea position as the object is lost. In this way, different speeds of the video were tested, so that it is hard to track objects at high speed.

The ground truth used is the one used by Zimmermann [53] and the foveated parameters are the ones of the Model 6 shown in Table 2. As explained previously, our tracking system has

<table>
<thead>
<tr>
<th>ID</th>
<th>W</th>
<th>B</th>
<th>Avg. time (ms)</th>
<th>Distance from ground truth (px)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>–</td>
<td>[1,1,1,1]</td>
<td>85</td>
<td>87 87 87 87 87</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>–</td>
<td>[0,0,1,1]</td>
<td>56</td>
<td>87 87 87 87 87</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(256, 256)</td>
<td>[1,1,1,1]</td>
<td>53</td>
<td>89 89 89 82 42</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(192, 192)</td>
<td>[1,1,1,1]</td>
<td>47</td>
<td>90 90 88 67 26</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(160, 160)</td>
<td>[1,1,1,1]</td>
<td>43</td>
<td>90 89 84 58 20</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(128, 128)</td>
<td>[1,1,1,1]</td>
<td>39</td>
<td>90 87 76 48 17</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(128, 256)</td>
<td>[0,1,1,1]</td>
<td>34</td>
<td>90 90 89 67 26</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(64, 64)</td>
<td>[1,1,1,1]</td>
<td>34</td>
<td>69 64 57 30 11</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(32, 32)</td>
<td>[1,1,1,1]</td>
<td>33</td>
<td>45 48 44 23 8</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(16, 16)</td>
<td>[1,1,1,1]</td>
<td>31</td>
<td>43 44 38 21 7</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(8, 8)</td>
<td>[1,1,1,1]</td>
<td>31</td>
<td>43 42 35 19 7</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>(256, 256)</td>
<td>[0,0,1,1]</td>
<td>26</td>
<td>89 89 90 82 40</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(8, 8)</td>
<td>[0,1,1,1]</td>
<td>21</td>
<td>42 42 35 19 7</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(16, 16)</td>
<td>[0,1,1,1]</td>
<td>21</td>
<td>43 44 37 20 7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(128, 128)</td>
<td>[0,0,1,1]</td>
<td>14</td>
<td>90 88 77 47 10</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(16, 16)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>36 38 30 12 2</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(8, 8)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>36 35 26 11 2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(32, 32)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>40 45 39 16 3</td>
<td></td>
</tr>
</tbody>
</table>

ID, Model number identification.

Fig. 11. Frames number 25, 90, 175 and 420, respectively, from mouse pad ground truth.

Fig. 12. Approximation length of the object that indicates the variation of scale of the object on the mouse pad ground truth (an average filter was applied for visualization purposes).

Fig. 13. Speed of the object ground truth center (an average filter was applied for visualization purposes).

Fig. 14. Object detection rate for some of the 18 models used for the entire mouse pad ground truth sequence (6946 frames). Some of them decay quickly while another ones do not.

Table 2

<table>
<thead>
<tr>
<th>ID</th>
<th>W</th>
<th>B</th>
<th>Avg. time (ms)</th>
<th>Distance from ground truth (px)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>–</td>
<td>[1,1,1,1]</td>
<td>85</td>
<td>87 87 87 87 87</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>–</td>
<td>[0,0,1,1]</td>
<td>56</td>
<td>87 87 87 87 87</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(256, 256)</td>
<td>[1,1,1,1]</td>
<td>53</td>
<td>89 89 89 82 42</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(192, 192)</td>
<td>[1,1,1,1]</td>
<td>47</td>
<td>90 90 88 67 26</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(160, 160)</td>
<td>[1,1,1,1]</td>
<td>43</td>
<td>90 89 84 58 20</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(128, 128)</td>
<td>[1,1,1,1]</td>
<td>39</td>
<td>90 87 76 48 17</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(128, 256)</td>
<td>[0,1,1,1]</td>
<td>34</td>
<td>90 90 89 67 26</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(64, 64)</td>
<td>[1,1,1,1]</td>
<td>34</td>
<td>69 64 57 30 11</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(32, 32)</td>
<td>[1,1,1,1]</td>
<td>33</td>
<td>45 48 44 23 8</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(16, 16)</td>
<td>[1,1,1,1]</td>
<td>31</td>
<td>43 44 38 21 7</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(8, 8)</td>
<td>[1,1,1,1]</td>
<td>31</td>
<td>43 42 35 19 7</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>(256, 256)</td>
<td>[0,0,1,1]</td>
<td>26</td>
<td>89 89 90 82 40</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(8, 8)</td>
<td>[0,1,1,1]</td>
<td>21</td>
<td>42 42 35 19 7</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(16, 16)</td>
<td>[0,1,1,1]</td>
<td>21</td>
<td>43 44 37 20 7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(128, 128)</td>
<td>[0,0,1,1]</td>
<td>14</td>
<td>90 88 77 47 10</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(16, 16)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>36 38 30 12 2</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>(8, 8)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>36 35 26 11 2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(32, 32)</td>
<td>[0,0,1,1]</td>
<td>10</td>
<td>40 45 39 16 3</td>
<td></td>
</tr>
</tbody>
</table>
recovery strategies when it loses the tracking. In this experiment, we apply four different strategies for recovering the tracking: backing to the last fovea position, disabling foveation, bottom–up attention using saliency map and increasing the fovea growth factor.

As explained before, a saliency map can be used in order to move the fovea to salient regions in response to bottom–up stimulus. In this experiment, each time the object is lost, the saliency map is computed, a threshold is applied on this map and the fovea is moved to the centroid of the remaining salient regions. As also stated before, the proposed model uses only one foveation at time, then it is assumed that the salient region is nearby the object center. Several algorithms can be used in such cases, but if the object was lost due to a fast movement of the object, then motion is an important feature on the saliency map computation. With this in mind, we used the saliency map proposed by Butko [41], since it has shown to be fast and also efficient in emphasizing motion information (see Fig. 15). In this experiment, the saliency map was computed over the original image scaled to 20% reducing the processing time from 330 ms per frame to about 6 ms per frame.

For some object movement speeds, it was possible to reduce from 2.7 up to 4.4 times the average time of tracking, without decreasing object detection rate (see Speeds 1x and 5x in Table 3). Experiments have shown that the larger the growth factor is (see Fig. 17), more time is needed and the higher the detection rate is. When the fovea is disabled, there is a peak in processing time (see Fig. 16). The several growth factors can be seen as intermediate steps between the last frame and disabled foveation strategies. The best strategy depends on the context of the scene. If the object movement is slow and the tracking is lost due to a false detection, then going back to the last frame or choosing a slower growing factor speed are good strategies. However, both strategies do not work properly if the object is moving too fast (see the last frame strategy when playing the video 10 times

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Different strategies for tracking playing the ground truth at different play speeds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>Mode</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Last frame</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Grow factor 5px/frame</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Grow factor 15px/frame</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Grow factor 30px/frame</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Grow factor 60px/frame</td>
</tr>
<tr>
<td>1x</td>
<td>Foveated/Disable foveation</td>
</tr>
<tr>
<td>1x</td>
<td>Bottom–up attention</td>
</tr>
<tr>
<td>1x</td>
<td>Non-foveated</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Last frame</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Grow factor 5px/frame</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Grow factor 15px/frame</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Grow factor 30px/frame</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Grow factor 60px/frame</td>
</tr>
<tr>
<td>5x</td>
<td>Foveated/Disable foveation</td>
</tr>
<tr>
<td>5x</td>
<td>Bottom–up attention</td>
</tr>
<tr>
<td>5x</td>
<td>Non-foveated</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Last frame</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Grow factor 5px/frame</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Grow factor 15px/frame</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Grow factor 30px/frame</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Grow factor 60px/frame</td>
</tr>
<tr>
<td>10x</td>
<td>Foveated/Disable foveation</td>
</tr>
<tr>
<td>10x</td>
<td>Bottom–up attention</td>
</tr>
<tr>
<td>10x</td>
<td>Non-foveated</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Last frame</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Grow factor 5px/frame</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Grow factor 15px/frame</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Grow factor 30px/frame</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Grow factor 60px/frame</td>
</tr>
<tr>
<td>20x</td>
<td>Foveated/Disable foveation</td>
</tr>
<tr>
<td>20x</td>
<td>Bottom–up attention</td>
</tr>
<tr>
<td>20x</td>
<td>Non-foveated</td>
</tr>
</tbody>
</table>

Fig. 15. Saliency map is used as strategy to recover fovea position when the object is lost. The first column has the 700th and 740th frames from Zimmermann ground truth (scaled to 14%); the third column has the 780th and 820th frames (scaled to 14%) and the second and fourth columns have the respective saliency map related to their left image (scaled to 70%).

Fig. 16. Difference between time processing using two strategies: back to the last fovea position or disabling foveation. Note that disable foveation causes peaks of processing time. (The processing times for both strategies for the other frames follow a similar pattern.)
5. Conclusion

A mechanism of feature extraction guided by visual attention processes is proposed in this work. This approach can be applied in different tasks like tracking and object detection with considerable reduction of processing time. For this purpose, the first step is to determine where the fovea should be. The fovea vector can be determined, for example, using bottom-up or top-down stimulus. For bottom-up stimulus this can be done using the first level of foveated resolution, where the fovea can be tracked.

Although the proposed model reduces the amount of visual data, and therefore, the time processing, experiments have shown that it is possible to perform successfully tracking and object detection tasks in real time using visual attention concepts. The detection rate depends on the proper fovea placement. If the fovea is far from the object area, then the detection rate decreases. The speed of detection rate decreasing depends on the foveated model used.

The foveated feature selection model is able to track an object reducing 60% the time processing without losing the tracking. This capability of fovea is related to the speed of the object because the fovea can lose the sight of the object thus decreasing the success rate. But, strategies were proposed in this paper to recover the tracking. Disabling the fovea is the simpler and most intuitive solution, but it causes a sudden peak of time processing. Increasing the fovea growing factor can minimize these peaks and is more efficient if the object has not been far from the current fovea position. Going back to the last frame is useful if the tracking lost is due to poor detection, but not due to fast movements.

Experiments have shown that whenever the fovea is at the object center, the visual attention guided feature extraction continues to successfully detect the object. Further improvements rely on features extraction and matching algorithms itself, like the ones proposed by Pang [51,52].

This paper has two major contributions. The first one is a technique for extracting multi-scale features in a foveated image, considering that the density of features depends on the fovea position. The second contribution is a method developed in order for this model to be guided by visual attention. Object tracking and attention tasks were done in order to validate both contributions. As a main result, processing time is substantially reduced in hard phases of the visual system, as feature detection thus increasing performance, allowing our system to run in real time (roughly 30 frames per second are achieved). Based on this, the proposed methodology can be used in robotics vision mainly to allow execution of high level tasks, such as navigation, recognition, localization and object manipulation by robotics devices.

With this work, we verified experimentally that the possibility of changing the focus of attention is the basis not only for common tasks described in the literature, as recognition, human face localization but also for other more complex tasks, such as tracking and object detection involved robot cognition [54]. The proposed model improves a methodology somehow inspired on biological models in the way that more accuracy resolution is at the fovea region. In this way, foveated levels with lower resolution can be used, for example, to detect motion or features to be used in tasks, such as navigation and foveated levels with higher resolution can be used, for example, in recognition, text reading or grasping. As a matter of fact, we intend to use such approach in further developments involving a robot adopted with a stereo head with 5° of freedom available at NataNet Lab (www.natalnet.br). We also plan to investigate the usage of other saliency detection methods that incorporate motion information in their calculation, such as the methods proposed by Li et al. [43] and Guo and Zhang [44].

References
