Efficient Construction of SIFT Multi-Scale Image Pyramids for Embedded Robot Vision

Peter Andreas Entschev and Hugo Vieira Neto

Graduate School of Electrical Engineering and Applied Computer Science
Federal University of Technology – Paraná, Curitiba, Brazil
peter@entschev.com, hvieir@utfpr.edu.br,
WWW home page: http://www.cpgei.ct.utfpr.edu.br

Abstract. Multi-scale interest point detectors such as the one used in the SIFT object recognition framework have been of interest for robot vision applications for a long time. However, the computationally intensive algorithms used for the construction of multi-scale image pyramids make real-time operation very difficult to be achieved, especially when a low-power embedded system is considered as platform for implementation. In this work an efficient method for SIFT image pyramid construction is presented, aiming at near real-time operation in embedded systems. For that purpose, separable binomial kernels for image pyramid construction, rather than conventional Gaussian kernels, are used. Also, conveniently fixed input image sizes of $2^N + 1$ pixels in each dimension are used, in order to obtain fast and accurate resampling of image pyramid levels. Experiments comparing the construction time of both the conventional SIFT pyramid building scheme and the method suggested here show that the latter is almost four times faster than the former when running in the ARM Cortex-A8 core of a BeagleBoard-xM system.

Keywords: multi-scale image pyramid, binomial filtering kernel, embedded robot vision

1 Introduction

The design of autonomous mobile robots will benefit immensely from the use of physically small, low-power embedded systems that have recently become available, such as the BeagleBoard-xM [1] and the Raspberry Pi [2] boards. These platforms are based on ARM processors that are able to run the Linux operating system and the OpenCV library [3], which makes them attractive for the implementation of embedded robot vision applications.

As robot vision applications are usually computationally intensive and demand relatively large amounts of memory, there are challenges for real-time operation with the limited processing resources of an embedded system. The BeagleBoard-xM is particularly interesting in this sense because it is based on a mid-range single-core ARM Cortex-A8 processor running at 1GHz with 512MB of DDR RAM, and a fixed-point Texas Instruments C64x+ family DSP running at 800MHz. Regarding energetic autonomy, the BeagleBoard-xM consumes as little as 5W at full load, against several dozens of watts consumed by a conventional personal computer.

Moreover, the BeagleBoard-xM is equipped with a dedicated camera bus, in which a CMOS camera can be connected directly to the main processor, virtually eliminating image acquisition overheads that are normally present in traditional types of camera interface, which use USB or FireWire connections. Fully programmable CMOS cameras of up to 5MP are available, whose image size may be conveniently configured to reduce acquisition bandwidth and the resources needed for resampling processes that are often present in multi-scale robot vision algorithms.

There are many powerful object recognition methods available in the literature that support multi-scale feature extraction – for example, SIFT [4, 5] and GLOH [6] – which are particularly interesting for robot vision applications. These methods are relatively expensive to compute as their core algorithms involve the construction of multi-scale image pyramids. In this work our intent is to investigate efficient implementations of the image pyramid construction scheme used in multi-scale object recognition algorithms, in order to allow near real-time execution in an embedded platform such as the BeagleBoard-xM.

2 Related Work

Objects can be detected in images by matching some of their unique visual features, usually edges and corners. In [4], Lowe presented his seminal work on the Scale Invariant Feature Transform (SIFT), demonstrating that it is possible to extract distinctive local features from an object in multiple scales, and match these features successfully afterwards, independently of scale, rotation, affine transformations or occlusions. This technique was later improved in [5].

Stable distinctive features that describe an object can be detected by computing a multi-scale Laplacian pyramid, as originally proposed in [7] and later made more efficient in [8]. In practice, the Laplacian pyramid is obtained by the differences between adjacent levels of a Gaussian pyramid built from successive low-pass filtering and down-sampling of the original input image.

After the computation of the difference of Gaussians, the location of distinctive local features (keypoints) can be found by detecting extrema (maxima and minima) among adjacent levels of the Laplacian pyramid, a function called Laplacian jet. Keypoints that are stable both in scale and space usually correspond to corners of objects.

Differences of Gaussians are less expensive to compute than computing the Laplacian directly, but even so, building the difference of Gaussians pyramid is one of the most computationally expensive processes that are executed in order to extract object features using the SIFT framework. For this reason, in this work we investigate techniques which are less computationally expensive but maintain the main property of detecting stable keypoints to describe objects.

In [9], it is demonstrated that a binomial difference of Gaussians can be used to approximate a conventional difference of Gaussians in a less computationally expensive way. The main difference is that scales are approximated by convolving the input image with a binomial kernel instead of a Gaussian kernel.

There are several other methods available in the literature that use the same principles to extract object features that can be used for scale invariant recognition [6]. Another well-known method called SURF (Speeded-Up Robust Features) builds the scale-space using a rather different approach, which involves the concept of integral images [10]. However, here we concentrate our efforts in methods that use standard convolution techniques.

3 Image Pyramid Construction

The original SIFT algorithm proposed in [4] uses an image pyramid, in which each scale consists of the previous scale convolved with a Gaussian kernel. Successive convolutions with a Gaussian kernel are applied in order to obtain different scales.

The pyramid construction method we use in this work is based on [9], in which instead of using Gaussian kernels in order to obtain different scales of the input image, a binomial kernel is used. The main advantage of using a binomial kernel instead of a Gaussian kernel is the reduced computational cost for the convolution process. For example, if the construction of a Gaussian pyramid with scales separated by a factor of $\sigma = \sqrt{2}$ is desired, it is necessary to convolve the input image in both horizontal and vertical directions with a separable 1D Gaussian kernel with a minimum length of seven elements; in order to achieve the same scale separation using successive convolutions with a binomial kernel, a length of only three elements is needed.

Building Gaussian pyramids using 2D kernels is also possible, but this is often avoided because it is more computationally expensive and has the exact same result of using separable 1D kernels. The binomial kernels studied here also present the property of separability, which is used throughout this work because our aim is specifically to reduce the processing time needed for image pyramid construction. For the work described here, two separable binomial kernels are especially relevant – one is the three-element kernel given by $\frac{1}{4} \times [1\ 2\ 1]$ and the other is the auto-convolution of the first, which is the five-element kernel given by $\frac{1}{16} \times [1\ 4\ 6\ 4\ 1]$.

3.1 Binomial Filtering

The kernel $\frac{1}{16} \times [1\ 4\ 6\ 4\ 1]$ approximates a Gaussian kernel with $\sigma = 1$, i.e. in order to obtain an approximation of a Gaussian blur of $\sigma = 1$, two consecutive convolutions with the three-element kernel $\frac{1}{4} \times [1\ 2\ 1]$ are needed.

In terms of complexity and if only separable 1D kernels are used, the practical meaning of an image convolution with three elements is that three multiplications and two additions per pixel per dimension (horizontal and vertical) are needed.

For a kernel with seven elements, it is necessary to perform seven multiplications and six additions per pixel per dimension.

The advantage of using separable binomial kernels with fixed-point coefficients is that in either one convolution with the kernel $\frac{1}{16} \times [1\ 4\ 6\ 4\ 1]$ or two consecutive convolutions with the kernel $\frac{1}{4} \times [1\ 2\ 1]$, two of the multiplications involved are multiplications by a factor of 1, making them unnecessary. The number of operations per pixel per direction is then reduced to a total of four multiplications and four additions per pixel per dimension for each scale. The total number of operations per scale of the pyramid is then $N = 8 \times R \times C \times 2$, where R is the number of rows and C is the number of columns in the pyramid level.

Yet another implicit advantage of using separable binomial kernels to perform image convolutions is that they can easily be used in implementations for fixed-point DSP cores. For instance, the built-in Texas Instruments C64x+ family DSP available in the BeagleBoard-xM supports fixed-point arithmetic and could be used in future implementations.

As shown in [9], when the image is convolved multiple times with the separable kernel $\frac{1}{16} \times [1\ 4\ 6\ 4\ 1]$, if the images at every three convolutions are stored, the resulting pyramid is separated by scale steps of $\sigma = \sqrt{2}$.

In order to improve efficiency by reducing the amount of data to be processed, instead of blurring the input image multiple times at the same octave, i.e. maintaining its original dimensions, the input image is down-sampled to half its size in each dimension every time that the scale reaches $\sigma = 2^N$ [9].

As described in [5], in order to be able to detect SIFT keypoints, at least four different scales are necessary for each octave of the Gaussian pyramid, but performing multiple convolutions of the image with a binomial kernel results in only three scales per octave – the original and two blurred ones, with $\sigma = \sqrt{2}$ and $\sigma = 2$ with respect to the original scale, respectively.

As can be seen in Fig. 1, the third image of the current octave is down-sampled in order to result in the initial scale of the next octave. Because this third image already has twice the size of the desired image in each dimension, it is possible to down-sample it using a nearest-neighbour approach with minimal loss of information. In this case, only every other pixel of each column and each row is kept, which is computationally inexpensive.

3.2 Image Acquisition and Resampling

For the resampling process, there is a great advantage provided by the built-in camera port of the BeagleBoard-xM. It is possible to keep the original image borders in all resampled scales if the acquired image has $2^N + 1$ pixels in each dimension – e.g. 129×257 pixels or 513×1025 pixels. With a fully programmable CMOS camera, images can be acquired with conveniently configured dimensions, which is a capability not always available in conventional USB or FireWire cameras for personal computers, for example.

The down-sampling process of images with $2^N + 1$ pixels in each dimension is straightforward and can be done using a nearest-neighbour approach, in which

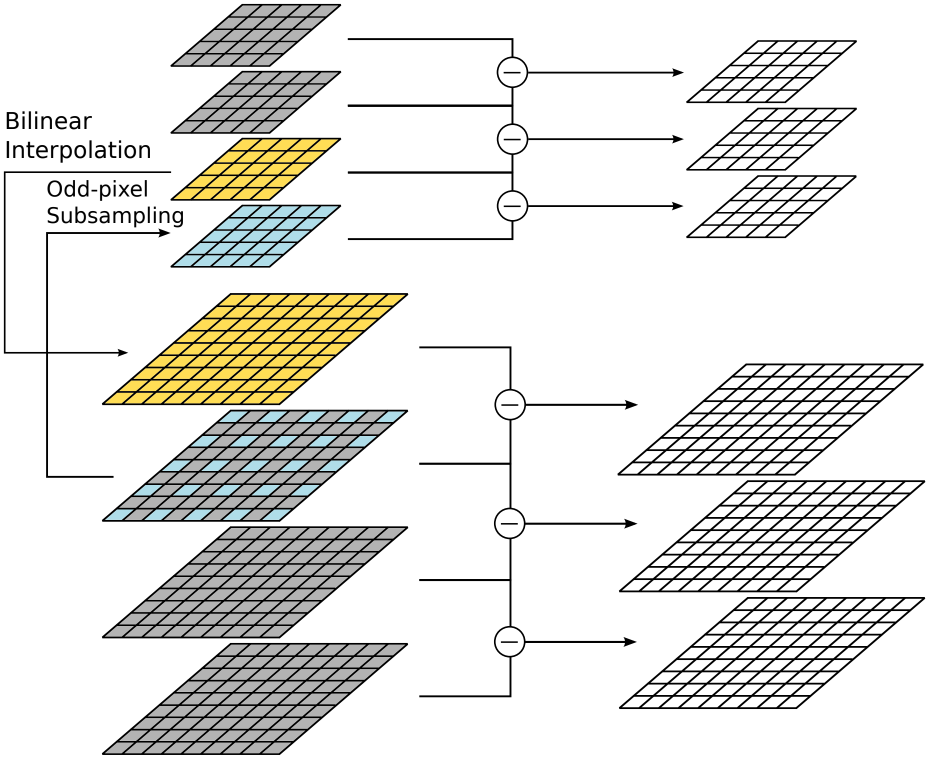

Fig. 1. Binomial difference of Gaussians pyramid construction. In adjacent octaves, the first scale is obtained by nearest-neighbour down-sampling and the fourth scale by bilinear interpolation (up-sampling).

every other pixel in each dimension is kept, including the pixels of the image borders.

In the next octave of the pyramid, we can continue blurring the image with the binomial kernel with a size of five elements and get new scale levels, also separated by steps of $\sigma = \sqrt{2}$. However, the previous pyramid octave still has only three different scales, and at least a fourth one is needed to find SIFT keypoints. This problem can be solved by doubling each dimension of the second level of the next level using bilinear interpolation (see Fig. 1).

4 Experiments and Results

In this section, we present experiments and results obtained while executing the algorithms in the ARM Cortex-A8 core of a BeagleBoard-xM system running the Linux operating system. Both image pyramid construcion schemes, based on binomial and Gaussian kernels, were implemented using the OpenCV library.

For these experiments, we executed the construction of binomial and Gaussian pyramids for several different image sizes ranging from 129×65 to 2049×1025 pixels, with total sizes doubling at each step. Each instance was executed 200 times – the average execution times obtained, along with their standard deviations, are shown in Table 1.

Table 1. Average execution times for the construction of binomial and Gaussian pyramids on the BeagleBoard-xM (ARM Cortex-A8 core). The construction of Gaussian pyramids takes at least 3.72 times the necessary amount of time taken for the binomial pyramids construction.

Input image size (pixels)	Number of octaves	Binomial pyramid execution time (seconds)	Gaussian pyramid execution time (seconds)
129 × 65	4	0.0207 ± 0.0001	0.0770 ± 0.0022
129 × 129	5	0.0398 ± 0.0002	0.1512 ± 0.0027
257 × 129	5	0.0768 ± 0.0007	0.2968 ± 0.0044
257 × 257	6	0.1508 ± 0.0011	0.5725 ± 0.0019
513 × 257	6	0.2893 ± 0.0007	1.1712 ± 0.0049
513 × 513	7	0.5914 ± 0.0012	2.2671 ± 0.0025
1025 × 513	7	1.1764 ± 0.0027	4.7103 ± 0.0108
1025 × 1025	8	2.3447 ± 0.0149	9.3758 ± 0.0103
2049 × 1025	8	4.6954 ± 0.0287	18.8793 ± 0.0183

Inspecting the average execution times in Table 1, it is clearly visible that the construction of the binomial pyramid is faster than the Gaussian pyramid. Calculating the ratio between the average time to build the Gaussian pyramid and the binomial pyramid, a minimum of 3.72 times is observed for images of 129 × 65 pixels in size, raising up to 4.02 for images of 2049 × 1025 pixels in size.

In order to obtain the results in Table 1, equivalent binomial and Gaussian pyramids were computed. We kept down-sampling the image to half its size while both dimensions were still greater than eight pixels. In other words, for the image of 129 × 65 pixels, four octaves were computed, for the image of 129 × 129, five octaves, and so on.

In Fig. 2, it can be seen graphically how the execution time of both pyramid construction schemes is affected by the size of the input image.

5 Conclusions

Autonomous mobile robots that aim to use vision as perceptual input will benefit greatly from designs using physically small, low-power embedded systems which can run the Linux operating system and the OpenCV library. However, efficient implementation of robot vision algorithms are necessary in order to allow near real-time operation.

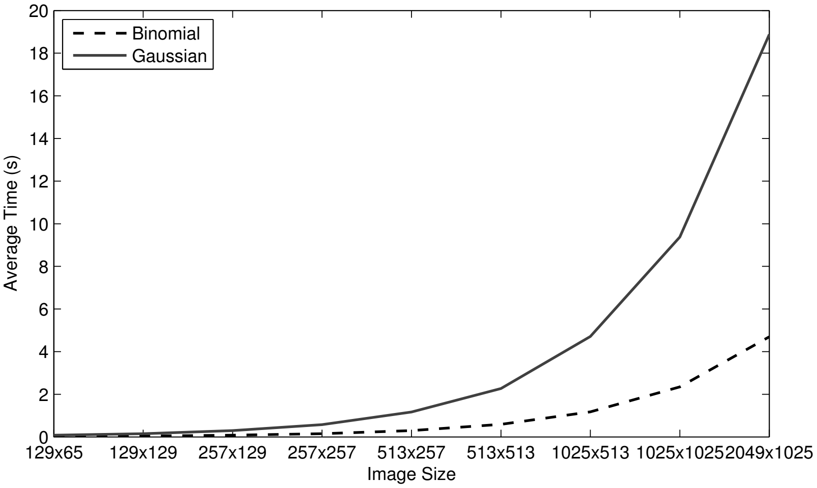

Fig. 2. Average execution times for the construction of binomial and Gaussian pyramids for multiple image sizes on the BeagleBoard-xM (ARM Cortex-A8 core). The black dashed line corresponds to execution times for the construction of binomial pyramids whereas the gray solid line corresponds to execution times for the construction of Gaussian pyramids.

This work has presented a less computationally expensive method for the construction of multi-scale SIFT [4, 5] pyramids. For this, we have focused on the work of Crowley and Riff [9], which describes the advantages of using separable binomial kernels over Gaussian kernels in order to build multi-scale pyramids.

Results of experiments conducted in a real embedded platform based on an ARM Cortex-A8 processor have shown that the binomial pyramid building scheme discussed in this work takes about one fourth of the time needed for building the conventional Gaussian pyramid. The approximation method described here reduces the overall time necessary for extracting SIFT features, making it more suitable for near real-time processing, especially on embedded platforms, in which limited computational resources are available.

Future improvements in the technique detailed here include using the fixed-point DSP core available in the BeagleBoard-xM. As the Texas Instruments C64x+ family DSP shares a limited portion of the available DDR RAM with the ARM Cortex-A8 processor, a hybrid approach using the parallel processing power of both cores to compute the image pyramids is possible. For near real-time continuous image feature extraction, a pipeline technique can be used to share the execution of processes between the DSP and the ARM processor.

Experiments and discussions about the stability of the extracted keypoints using both pyramid construction schemes are the subject of future work – this deserves special attention, given that it is necessary to assess the best parameters for selection of the most stable SIFT features.

References

1. Coley, G. Beagleboard-xM System Reference Manual – Revision A2. BeagleBoard.org (2010)
2. Upton, E., Halfacree, G. Raspberry Pi User Guide. Wiley (2012)
3. Bradski, G., Kaehler, A. Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media (2008)
4. Lowe, D. G. Object Recognition From Local Scale-invariant Features. Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 2, IEEE, 1150–1157 (1999)
5. Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60.2, 90–110 (2004)
6. Mikolajczyk, K., Schmid, C. A Performance Evaluation of Local Descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 27.10, 1615–1630 (2005)
7. Burt, P., Adelson, E.: The Laplacian Pyramid as a Compact Image Code. IEEE Transactions on Communications 31.4, 532–540 (1983)
8. Crowley, J. L., Stern, R. M. Fast Computation of the Difference of Low-pass Transform. IEEE Transactions on Pattern Analysis and Machine Intelligence 2, 212–222 (1984)
9. Crowley, J. L., Riff, O. Fast Computation of Scale Normalised Gaussian Receptive Fields. Scale Space Methods in Computer Vision. Springer, 584–598 (2003)
10. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L. Speeded-Up Robust Features (SURF). Computer Vision and Image Understanding 110.3, 346–359 (2008)