J Real-Time Image Proc (2016) 12:197–213
DOI 10.1007/s11554-014-0396-z

ORIGINAL RESEARCH PAPER

Enhanced low-complexity pruning for corner detection

Nirmala Ramakrishnan · Meiqing Wu · Siew-Kei Lam · Thambipillai Srikanthan

Received: 26 May 2013 / Accepted: 3 January 2014 / Published online: 28 January 2014

Abstract Low-complexity corner detection is essential for many real-time computer vision applications that need to be executed on low-cost/low-power embedded platforms such as robots. The widely used Shi–Tomasi and Harris corner detectors become prohibitive in such platforms due to their high computational complexity, which is attributed to the need to apply a complex corner measure on the entire image. In this paper, we introduce a novel and computationally efficient technique to accelerate the Shi–Tomasi and Harris corner detectors. The proposed technique consists of two steps. In the first step, the complex corner measure is replaced with simple approximations to quickly prune away non-corners. In the second step, the complex corner measure is applied to a small corner candidate set obtained after pruning. Evaluations using standard image benchmarks show that the proposed pruning technique achieves up to 75 % speedup on the Nios-II platform, while yielding corners with comparable or better accuracy than the conventional Shi–Tomasi and Harris detectors.

Keywords Corner detection · Low complexity · Shi–Tomasi · Harris · Pruning

N. Ramakrishnan (✉) · M. Wu · S.-K. Lam · T. Srikanthan
School of Computer Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore
e-mail: rnirms@gmail.com; rama0056@e.ntu.edu.sg

M. Wu
e-mail: wume0007@e.ntu.edu.sg

S.-K. Lam
e-mail: assklam@ntu.edu.sg

T. Srikanthan
e-mail: astsrikan@ntu.edu.sg

1 Introduction

Computer vision algorithms are being extensively used for a wide range of applications such as vision-based navigation for image-guided medical interventions [1], navigation of unmanned vehicles [2] and robots [3], video encoding on unmanned aerial vehicles [4], video tracking [5] and visual SLAM [6]. One fundamental step in these applications is the detection of corners which represent identifiable anchor points in the image. Corners are used for matching (e.g., image registration), tracking (e.g., video tracking) and as robust image representation when combined with feature descriptors for object recognition. Low-level corner detection [7] techniques are evaluated based on two criteria: (1) accuracy which is often measured using repeatability rate [8] of the corners across various image transformations, and (2) efficiency which is usually defined by the computational complexity of the algorithm. For embedded vision applications, it becomes critical to have low-complexity corner detection as they have stringent real-time and/or low-power constraints.

Several corner detectors have been proposed in the literature [7, 8] and comparative evaluations have shown that the Shi–Tomasi [9] and Harris [10] corner detectors achieve some of the best results [8, 11]. Also, wide range of recent real-time applications [1–3, 5, 6] have used Shi–Tomasi or Harris corner detectors. However, both the algorithms require a complex corner measure computation for every pixel in the image. This step is highly compute intensive, requiring floating-point arithmetic and becomes a bottleneck for real-time vision tasks. Reducing the computational complexity of these corner detection algorithms is essential in low-cost and low-power embedded systems, especially those that do not support an on-board floating-point unit (FPU).

In our earlier paper [12], we introduced a low-complexity pruning technique for Shi–Tomasi and Harris corner detectors that created a small pool of corner candidates. The complex corner measure was then applied only on this small pool of candidates. We denote the pruning method in [12] as PP which stands for 'partial pruning'. When PP is combined with the Shi–Tomasi and Harris corner detectors, it is denoted as PP_{ST} and PP_{H}, respectively. The pool of corner candidates generated by PP still contains edge pixels that are obvious non-corners and hence, $PP_{ST/H}$ does not achieve substantial savings in images that contain many strong edges.

In this paper, we extend our previous approach by introducing a low-complexity edge pixels removal step to further reduce the number of corner candidates. We denote the edge pixels removal step as ER, for 'edge removal'. Our enhanced pruning algorithm, which combines PP and ER, is denoted as PP-ER. PP-ER_{ST} and PP-ER_{H} refer to the proposed enhanced pruning method for Shi–Tomasi and Harris, respectively. The proposed PP-ER approach is able to quickly remove the non-corners using simple approximations of the complex Shi–Tomasi and Harris corner measure and create a small but near-complete corner candidate set. The conventional Shi–Tomasi/Harris corner measure is then applied only to this set of corner candidates to extract the final corners, thereby slashing the overall computation cost significantly. The proposed pruning technique is devised based on the premise that in most images, corner regions constitute a very small portion of the image. While the complex corner measure of Shi–Tomasi and Harris can be used to rank and extract the best corners, they incur excessive computations for eliminating non-corners in the entire image. Experimental results reveal that the proposed technique not only leads to significant speedup over the Shi–Tomasi and Harris corner detectors, but also results in final corners with comparable or better accuracy.

This paper differs from our previously reported work on $PP_{ST/H}$ [12] in the following ways:

1. We performed extensive evaluation of the $PP_{ST/H}$ method on a more comprehensive image dataset.
2. The edge removal step (ER) is introduced to remove edge pixels from the corner candidate set obtained after PP. We propose the enhanced pruning approach (PP-ER), which combines the PP and ER pruning strategies to quickly identify a small but complete corner candidate set. The conventional Shi–Tomasi/Harris corner measure is then applied only to this set of corner candidates to extract the final corners.
3. We have investigated the benefits of the proposed pruning methods on two variants of the Shi–Tomasi and Harris corner detectors based on the filter used for computing the auto-correlation matrix. In [12], we showed the results for the more complex Gaussian filter for the auto-correlation matrix. In this paper, we show that pruning accelerates the corner detection even with the simpler box filter for the auto-correlation matrix.
4. We compared the proposed pruning technique with existing techniques for low-complexity corner detection to show the superiority of our method in terms of computation cost and repeatability.
5. Experiments were undertaken on the Nios-II processor with different configurable options (i.e., with and without floating-point unit (FPU) and with and without cache).
6. We show the viability of the proposed technique on feature-based global motion estimation (GME) in video sequences.

The rest of the paper is organized as follows. In the next section, we discuss the previously reported work on corner detectors and techniques to accelerate them. In Sect. 3, we introduce the proposed pruning method (PP-ER). Section 4 provides computational cost analysis of the proposed method in comparison with other related work. Section 5 presents actual implementation results on the Nios-II processor to evaluate the accuracy and performance of the proposed technique with a set of image benchmarks. Section 6 demonstrates the usefulness of the PP-ER method for global motion estimation (GME) in videos. The paper concludes in Sect. 7.

2 Related work

Tuytelaars and Mikolajczyk [7] and Schmid et al. [8] provide excellent surveys on corner detectors proposed in the last 30 years. In this section, we focus on related work in corner detection from the standpoint of efficiency. Earlier work on corner detection involved looking for high curvature points along the contours in the image which typically correspond to true corners in 3D. In [13], such a method is proposed that looks for maxima of curvature where the gradient is large. Recent work in corner detection selects points that are robust, stable and distinctive, and need not always correspond to true corners [7]. Detectors that use the Hessian matrix formed with the second-derivatives of intensity values have been proposed [7]. However, they have been shown to be relatively less reliable. Shi–Tomasi [9] and Harris [10] corner detectors compute an auto-correlation matrix using the first-order derivatives of the intensity values and this matrix represents the degree of intensity variations in various directions around a pixel. SUSAN detector [14] operates directly on the image intensity by computing the fraction of pixels within a neighborhood that have similar intensity as the center pixel. FAST [15] extends this idea to consider only pixels on a

circle around the center pixel and uses an efficient decision tree to classify the center pixel as a corner.

It is well recognized that corner detection is a compute-intensive step. There are typically two approaches that have been reported in the literature for increasing the computation efficiency of corner detection. The first approach employs hardware accelerators while the second approach focuses on algorithmic techniques to reduce the computational complexity.

Hardware acceleration techniques have been proposed to exploit the inherent parallelism in the corner detectors. Efficient FPGA implementation for SUSAN has been presented in [16]. Although FAST is computationally simpler to Harris and Shi–Tomasi detectors, recent evaluations [11] have shown that FAST can be unreliable in many scenes and the Harris detector is preferred. The evaluations in [8] also show that Harris and Shi–Tomasi achieve among the best results in low-level corner detection. Numerous approaches to accelerate these detectors have, therefore, been reported. Various fast implementations of Harris have been proposed in the literature. The target accelerator platforms include Application Specific Integrated Circuit (ASIC) [17], Field Programmable Gate Array (FPGA) [18], Cell Processor [19] and SIMD architecture [20]. In [21], a simpler floating-point format is used by customizing instructions on the Nios-II processor. In [22], a hardware implementation that performs Harris corner detection on a rank transform image instead of the original image is presented. FPGA implementation for Shi–Tomasi in [23] employs an alternative corner measure that uses only integer arithmetic consisting of additions and multiplications and avoids the transcendental operations. In [24, 25], the corner detection step for Shi–Tomasi is implemented on GPU and the final step of non-maximal suppression is parallelized.

Low-complexity techniques for corner detection are achieved through algorithm innovations which are independent of the underlying hardware architecture. In [26], integral image is used and the computation time for corner response computation is kept constant irrespective of the window size used. In [27], a pruning technique based on gradient magnitude is proposed that selects pixels with high gradient magnitude as corner candidates for Shi–Tomasi and Harris algorithms. Our work belongs to this class of techniques and can potentially result in higher computation savings when used with the architecture specific solutions discussed above.

In this paper, we have proposed the PP-ER method to efficiently discard non-corners, which significantly reduces the selection and evaluation effort for the presence of corners to only corner-like regions. The approach proposed in [27] has a similar motivation as our work. However, it uses the gradient magnitude of a pixel alone as an indicator of the corner and does not consider the intensity pattern of the pixel neighborhood. This can potentially miss many good corners. Also, the use of integral image in [26] shows substantial savings only for large window sizes. Small window sizes, i.e., 3×3, have been shown to be sufficient for Shi–Tomasi/Harris detectors [5, 28]. In addition, while the work in [24, 25] achieves speedup by exploiting parallelism on GPU, they are not well suited for low-cost/-power embedded systems. Moreover, compared to [23–26], our technique computes the corner response only on a small set of corner candidates instead of the entire image. As shown in the experimental results, the proposed method leads to significant computation savings without compromising on the accuracy of corner detection.

3 Proposed pruning technique for corner detection

Corner detection in both Shi–Tomasi and Harris detectors is based on the local auto-correlation function that is approximated by matrix M over a small window W for each pixel $p(x, y)$:

$$M = \begin{bmatrix} \sum_W w(x) I_x^2 & \sum_W w(x) I_x I_y \\ \sum_W w(x) I_x I_y & \sum_W w(x) I_y^2 \end{bmatrix} = \begin{bmatrix} a & b \\ b & c \end{bmatrix} \quad (1)$$

I_x and I_y are horizontal and vertical gradients, respectively, and $w(x)$ is an averaging filter that can be a box or a Gaussian filter. The eigenvalues λ_1 and λ_2 of M (where $\lambda_1 \geq \lambda_2$) indicate the type of intensity change in the window W around $p(x, y)$:

- If both λ_1 and λ_2 are small, $p(x, y)$ is a point in a flat region.
- If λ_1 is large and λ_2 is small, $p(x, y)$ is an edge point.
- If both λ_1 and λ_2 are large, $p(x, y)$ represents a corner point.

Shi–Tomasi directly compute the smaller eigenvalue λ_2 as the corner measure C as shown in (2):

$$C = \lambda_2 = \frac{1}{2}\left((a+c) - \sqrt{(a-c)^2 + 4b^2}\right) \quad (2)$$

Harris combines the eigenvalues into a single corner measure R as shown in (3), which avoids the explicit computation of eigenvalues. In (3), k is an empirical constant ($k = 0.04$–0.06).

$$\begin{aligned} C = R &= \lambda_1\lambda_2 - k \cdot (\lambda_1 + \lambda_2)^2 = \det(M) - k \cdot \mathrm{trace}(M)^2 \\ &= (ac - b^2) - k \cdot (a+c)^2 \end{aligned} \quad (3)$$

Once the corner measures for every pixel are computed, a threshold is applied on the corner measures to discard the obvious non-corners. The rest of the pixels are then ranked in the descending order of the corner measure and the pixels with the highest corner measure are then selected as corners after applying the non-maximal suppression.

We make the following observations on the Shi–Tomasi and Harris detectors:

- In most images, the obvious non-corners (i.e., the flat and edge regions) constitute a large majority of the image. Hence, the Shi–Tomasi and Harris detectors incur a lot of redundant computations as they evaluate the entire image for a high corner response.
- Expanding (2), we get

$$\lambda_2 = \frac{1}{2}\left((a+c) - \sqrt{(a-c)^2+4b^2}\right) = \frac{1}{2}\left((a+c) - \sqrt{(a+c)^2-4(ac-b^2)}\right) \quad (4)$$

  λ_2 is most influenced by the term $(ac - b^2)$ as the two $(a + c)$ terms cancel out. For a good corner, λ_2 needs to be a large value. Hence, maximizing $(ac - b^2)$ which is also det(M) can select good Shi–Tomasi corners without explicit eigenvalue computation.
- In Harris, the trace(M) term is introduced so that edges can also be detected. Ignoring the trace(M) term, the det(M) term alone is sufficient to select corner regions.

Based on our observations, det(M) is the key term in the corner measures for both Shi–Tomasi and Harris, and pixels that maximize det(M) represent good corner candidates. We therefore propose a low-complexity pruning technique that employs simple approximations of the det(M) to quickly remove the non-corners. Note that det $(M) = ac - b^2$ consists of two terms and our proposed pruning technique is divided into two steps that handle each term successively: PP that selects pixels with a large value for ac, followed by ER that discards pixels with a large value for b. In effect, PP and ER applied together select pixels that maximize det(M).

3.1 Partial pruning (PP)

Partial pruning [12] is the first step of the proposed enhanced pruning technique and it selects an initial set of corner candidate pixels which have a large value for ac. Applying an appropriate threshold can discard pixels with low ac values. Figure 1b shows the initial corner candidates selected by applying threshold = 0.05 × Max(ac), and it is clear that this covers the final Shi–Tomasi corner regions in Fig. 1d well.

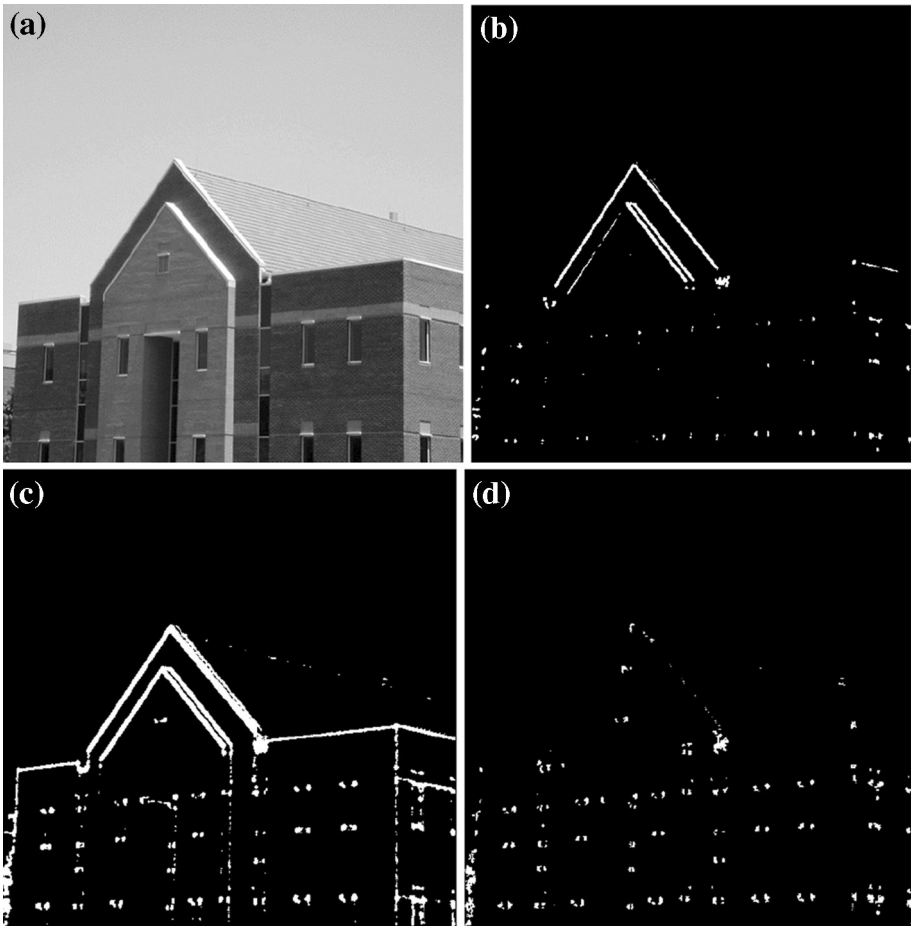


Fig. 1 **a** Original image. **b** Corner candidates selected using ac at threshold = 0.05 × Max(ac). **c** Corner candidates selected using $a'c'$ at threshold = 0.05 × Max($a'c'$) **d** Shi–Tomasi corner regions with λ_2 at threshold = 0.05 × Max(λ_2)

Instead of computing a and c for every pixel explicitly, we propose to approximate the I_x^2 and I_y^2 terms in the expression for ac with the absolute values for I_x and I_y, respectively, as follows:

$$a' = \sum |I_x|, c' = \sum |I_y| \tag{5}$$

This eliminates the multiplication operations involved in the squared gradients. Figure 1c shows that the $a'c'$ map covers the ac map in Fig. 1b and the final corner regions in Fig. 1d well.

Figure 2 shows that as the threshold for $a'c'$ map is reduced under uniform illumination, corners and slanted edges are released first. This is followed by vertical and horizontal lines, and finally faint textures and flat regions. This shows that when the threshold is applied to $a'c'$ map of the image, non-corner regions are likely to be removed, while retaining a significant amount of corners. Hence, $a'c'$ can be used as an effective corner indicator measure as it elevates the corner regions above the non-corner regions. At the end of PP, an initial corner candidate set denoted as C_1 is generated and further processing takes place on C_1. As seen in Fig. 2, C_1 generated after PP contains many edge pixels, which are obvious non-corner pixels. This is because PP only looks for pixels with high values for ac and does not consider b^2, which is the second term in det(M).

3.2 Removing edge pixels (ER)

To maximize det(M), for all the candidates in C_1, the second term b^2 needs to be substantially smaller than the first term ac. In other words, we need to eliminate pixels that have comparable values for ac and b^2. The det(M) computed over a small window W of a pixel is given by (6):

$$\det(M) = ac - b^2 = \sum_{i=1}^{w} I_{x_i}^2 . \sum_{i=1}^{w} I_{y_i}^2 - \left(\sum_{i=1}^{w} I_{x_i} I_{y_i} \right)^2 \tag{6}$$

The gradient direction of the pixel is represented by $k_i = I_{y_i} / I_{x_i}$ and det(M) can be rewritten in terms of the gradient direction k_i as:

$$\det(M) = ac - b^2 = \sum_{i=1}^{w} I_{x_i}^2 . \sum_{i=1}^{w} (k_i I_{x_i})^2 - \left(\sum_{i=1}^{w} k_i I_{x_i}^2 \right)^2 \tag{7}$$

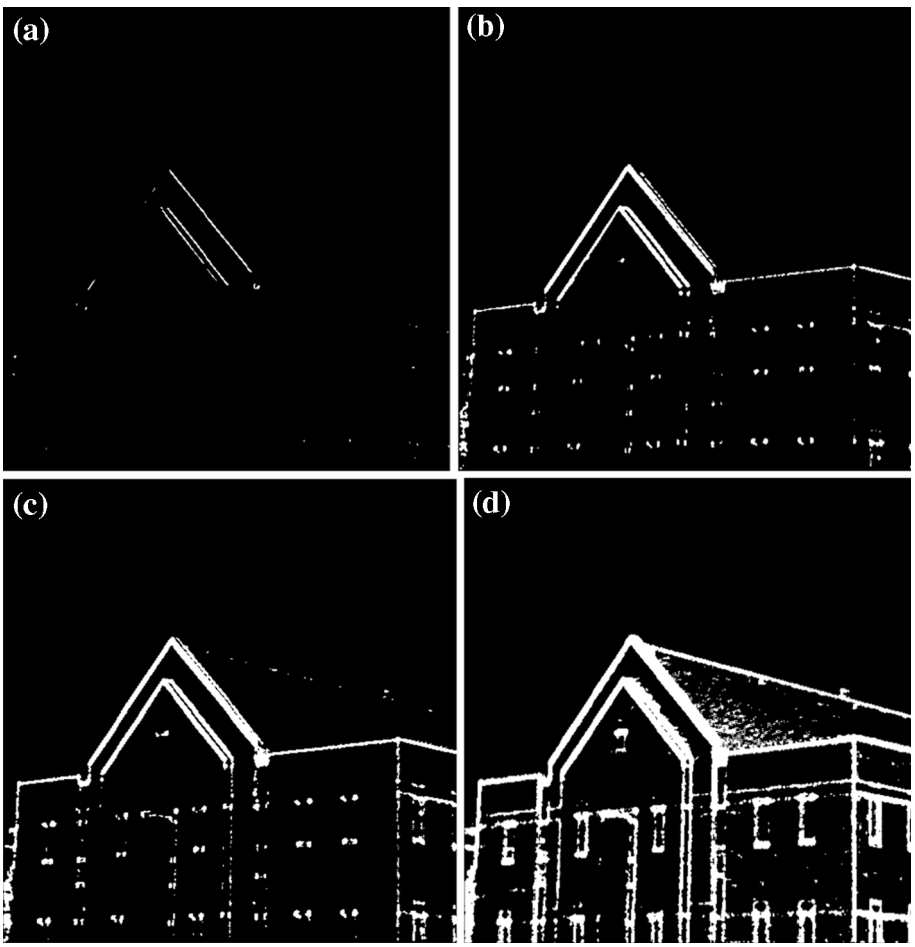


Fig. 2 Initial corner candidate set C_1 ($a'c'$ map) at various thresholds **a** 0.5 **b** 0.1 **c** 0.05 **d** 0.01

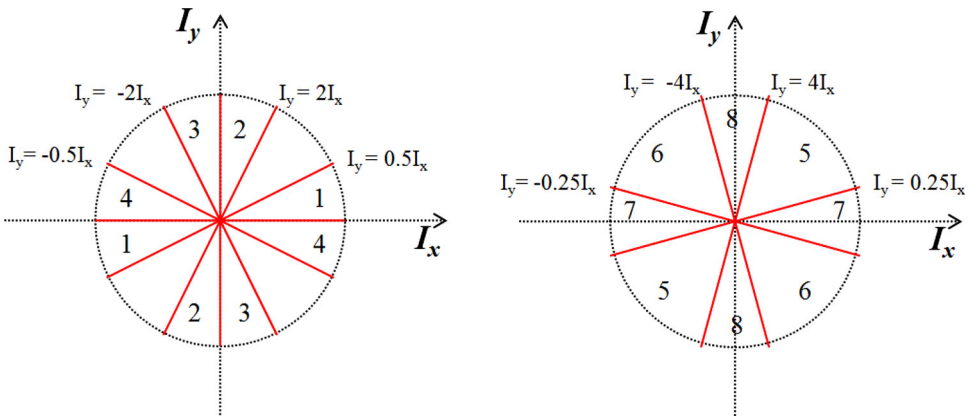
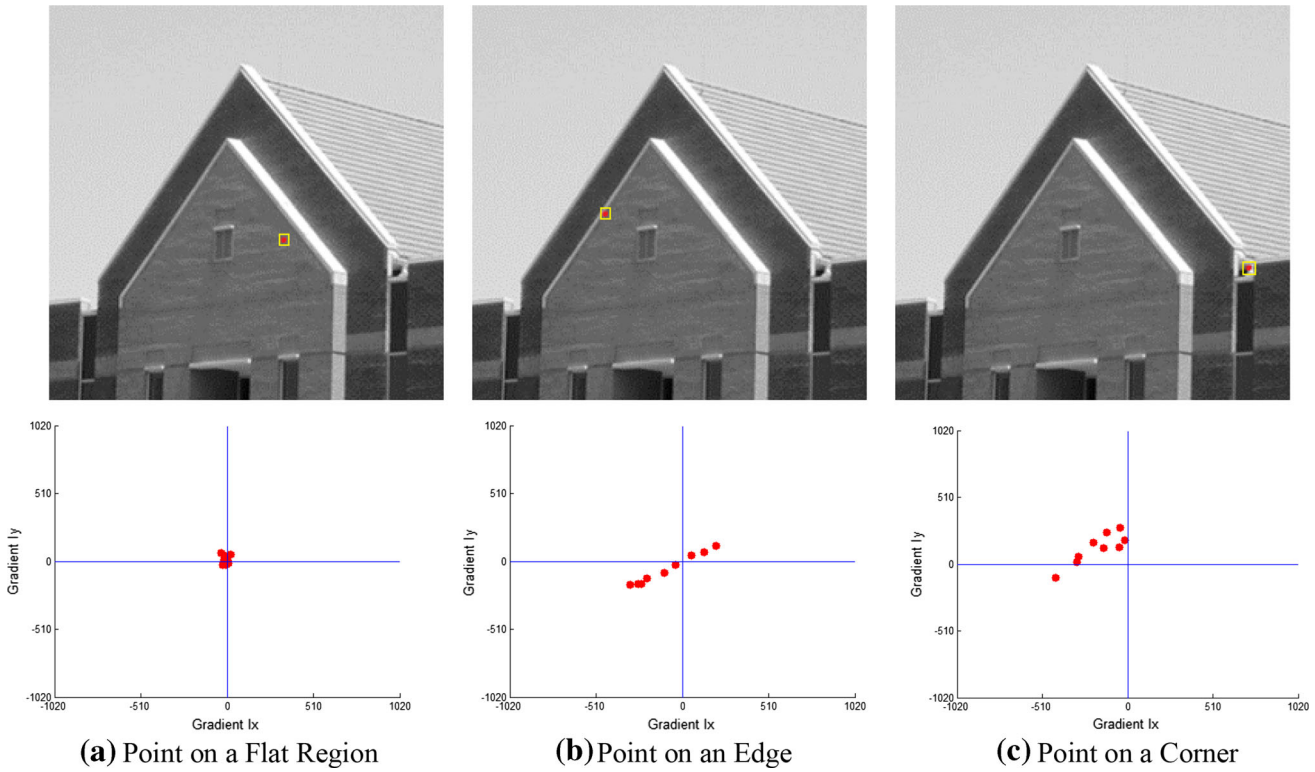


Fig. 3 Bin boundaries for I_x–I_y plots for the edge removal (ER) step in pruning



(a) Point on a Flat Region **(b)** Point on an Edge **(c)** Point on a Corner

Fig. 4 $I_x - I_y$ plots for 3×3 neighborhood of pixel in various types of intensity patterns in the ER step

When $k_i \approx k$ for all the neighbors of a pixel in window W, the two terms, ac and b^2, converge and $\det(M) \approx 0$ as shown in (8):

$$\begin{aligned}\det(M) &= ac - b^2 \approx \sum_{i=1}^{w} I_{x_i}^2 \cdot \sum_{i=1}^{w} (kI_{x_i})^2 - \left(\sum_{i=1}^{w} kI_{x_i}^2\right)^2 \\ &= k^2 \cdot \left(\sum_{i=1}^{w} I_{x_i}^2\right)^2 - k^2 \cdot \left(\sum_{i=1}^{w} I_{x_i}^2\right)^2 = 0\end{aligned} \tag{8}$$

This happens when the pixel and its neighbors fall on an edge with an edge direction given by k. Hence, we need to remove edge pixels in C_1 as they do not maximize $\det(M)$. We propose a novel compute-efficient method to remove the edge pixels by analyzing the gradients I_x and I_y of the 3×3 neighbors of the candidate pixels. The proposed ER method avoids an explicit computation of the b^2 term.

We divide the $I_x - I_y$ space into two sets of overlapping bins as shown in Fig. 3. Edge pixels will have all their neighbors in the same bin whereas corner pixels will show a spread and this allows quick detection of the edge pixels. Overlapping bins are necessary to capture edge pixels located along the bin boundaries in the $I_x - I_y$ space. Also, a 3×3 window is localized around the candidate pixel and hence is the best option for our analysis.

Figure 4 illustrates the $I_x - I_y$ plot of the 3×3 patches of various intensity patterns for an example image:

- Pixels with a relatively flat 3×3 neighborhood have all the points clustered together close to the origin.
- Edge point and its neighbors fall along the edge direction.
- Corner point and its neighbors are spread wider.

To identify an edge point, for each candidate pixel in C_1, we count the number of neighbor pixels in each bin in the $I_x - I_y$ space. Ideally, a candidate pixel and all its 3×3 neighbors should fall in the same bin for it to be considered an edge point. However, this is often not the case in practice as images are noisy and the gradient computation is a numerical approximation. To overcome this problem, a pixel is considered an edge point and is discarded when the number of neighboring pixels in a particular bin exceeds a predefined threshold (possibly lower than $3 \times 3 = 9$).

During the bin allocation for a candidate pixel in the $I_x - I_y$ space, if a neighbor pixel has a low gradient magnitude, i.e., below the thresholds (C_x, C_y) as computed in (9) and (10), it then contributes equally to all the bins:

$$C_x = \text{mean}\ (I_x) + k \cdot \text{SD}\ (I_x) \tag{9}$$

$$C_y = \text{mean}(I_y) + k \cdot \text{SD}\ (I_y) \tag{10}$$

Here, I_x and I_y represent the x- and y-gradient maps of the image and $k = 0.25$ is set empirically. SD refers to the standard deviation.

The proposed method for removing edge pixels from the initial corner candidate set C_1 relies on the extremely compute-efficient bin allocation strategy that is enabled by the use of bin boundaries, which are factors of two of the gradient magnitudes. This enables the bin boundary computation and bin allocation to be achieved using only comparisons and bit shifts. The corner candidate set generated at the end of ER is referred to as C_2 and the baseline algorithm (Shi–Tomasi or Harris) is now applied only on C_2.

4 Cost analysis

Algorithm 1 describes the proposed low-complexity enhanced pruning technique applied to the baseline corner detector. The proposed PP-ER$_{ST/H}$ algorithm first applies

Algorithm 1 Algorithm for Pruning based Corner Detection: (PP-ER$_{ST/H}$)

Input: Gradients I_x, I_y for Image I, feature quality threshold T_c, pruning threshold T_p
/* Pruning */
PP: For each pixel in I
  Compute $|I_x|$, $|I_y|$
  Compute *a'c'* = $\sum|I_x| * \sum|I_y|$
  Threshold *a'c'* map with threshold = T_p *max(a'c') to get C_1

ER: For each pixel in C_1
  Compute bin boundaries: 0.25* I_x, 0.5* I_x, 2* I_x, 4* I_x
  Allocate bins for each of the 3x3 neighbors in the $I_x - I_y$ plot
  If *#bin_members* for any bin < *bin_threshold*, add corner candidate to C_2

/* Corner Response Function */
B1a: For each pixel in C_2N (C_2 and its neighbors in window W)
  Compute I_x^2, I_y^2, I_xI_y

B1b: For each pixel in C_2
  Compute $a=\sum w(x) * I_x^2$, $c= \sum w(x) * I_y^2$, $b=\sum w(x) * I_xI_y$

B2: For each pixel in C_2
  Compute Corner Response $C = R$ or λ_2

B3:
Threshold corner response map with threshold=T_c *max(C)
Sort in descending order of C
Apply non-maximal suppression

Table 1 Per-pixel operations for the pruning and baseline algorithms

Per-pixel operations			
Steps	Multiplications	Additions	Sq. root
PP	1 (16-bit)	2 W (16-bit) + 2	
ER	1 (32-bit)	39 (16-bit)	
B1a: I_x^2, I_y^2, I_xI_y	3 (16-bit)		
B1b: $w(x)$for M			
Box		3 W (32-bit)	
Gaussian	3 W (FP)	3 W (FP)	
B2: C			
Shi–Tomasi (Box)	2 (32-bit)	4 (32-bit)	1
Shi–Tomasi (Gaussian)	2 (FP)	4 (FP)	1
Harris (Box)	3 (32-bit) + 1 (FP)	2 (32-bit) + 1 (FP)	
Harris (Gaussian)	4 (FP)	3 (FP)	

PP to generate a corner candidate set C_1. It then removes edge pixels from C_1 using ER to generate C_2. The baseline algorithm is then applied only to the corner candidate set C_2. It is noteworthy that PP is applied to the entire image, but subsequent steps of the pruning and baseline algorithm are applied to only smaller candidate sets (i.e., C_1 and C_2).

The Shi–Tomasi and Harris algorithms have been used as the baseline algorithms for this paper. They consist of the following key steps—B1: compute auto-correlation matrix M for each pixel, B2: compute corner measure (Shi–Tomasi or Harris) C for each pixel, and B3: Sort all the pixels based on the corner measure and apply non-maximal suppression to generate the final corners.

Table 1 shows the computations incurred by the pruning and the baseline algorithms per pixel. The pruning step PP only has 16-bit additions and multiplications and ER only has bit shifts and comparisons (which are reported as additions). It is evident that the computations of PP and ER are less complex compared to the steps B1 and B2 of the baseline algorithms, thereby potentially resulting in substantial savings. When the baseline algorithm is used without pruning, for each pixel, 3 (16-bit) multiplications are incurred for B1a (computing the I_x^2, I_y^2, I_xI_y). When pruning is applied, these values need to be computed for the corner candidate as well as its neighbors in the window W. Hence, B1a needs to be applied to all pixels in C_1 and their neighbors (represented by C_1N for PP), and all pixels in C_2 and their neighbors (represented as C_2N for PP-ER).

Table 2 Comparison of computations for each pixel for PP-ER$_{ST}$

Method	Multiplications	Additions	Square root
Shi Tomasi [9]	5	31	1
Perona [21]	5	30	
SLC-KLT [24]	5	19	1
PP$_{ST}$ [11]	$1 + pC_1N \times 3 + pC_1 \times 2$	$20 + pC_1 \times 31$	pC_1
PP-ER$_{ST}$	$1 + pC_1 + pC_2N \times 3 + pC_2 \times 2$	$20 + pC_1 \times 39 + pC_2 \times 31$	pC_2

Table 3 Comparison of computations for each pixel for PP-ER$_{H}$

Method	Multiplications	Additions
Harris [10]	6	30
SLC-Harris [24]	6	18
PP$_{H}$ [11]	$1 + pC_1N \times 3 + pC_1 \times 4$	$20 + pC_1 \times 30$
PP-ER$_{H}$	$1 + pC_1 + pC_2N \times 3 + pC_2 \times 4$	$20 + pC_1 \times 39 + pC_2 \times 30$

Figure 5 shows the number of pixels on which each step of the PP-ER$_{ST/H}$ is applied.

We compare the computations of PP-ER$_{ST/H}$ with the baseline algorithms (i.e., Shi–Tomasi and Harris), PP$_{ST/H}$, and recently reported work that aim to reduce complexity of the baseline algorithms [23, 26] in Tables 2 and 3. In these comparisons, we assumed that a box filter i.e., $w(x) = 1$ and a 3×3 window (i.e., $W = 9$) is used for the auto-correlation matrix M. When a Gaussian window is used for $w(x)$, additional floating-point (FP) multiplications are incurred on the corner measure computation step B1b. When compared to the box filter, the use of a Gaussian window will clearly lead to higher savings for PP-ER$_{ST/H}$ as the overall number of multiplications increases significantly in the baseline algorithms. Note that both [23] and [26] reduce complexity of the baseline algorithm by removing the square root in Shi–Tomasi and reducing the number of additions for the window operations, respectively. However, the computations are still applied to all pixels in the image. In contrast, our proposed pruning approach can potentially result in significant savings as the values for $|C_1|$, $|C_2|$, $|C_1N|$ and $|C_2N|$, which represent the size of the corner candidates at various stages in the algorithm, are typically substantially smaller than the image size. To illustrate the savings in computations per pixel, the corner candidate sizes for C_1, C_2, C_1N and

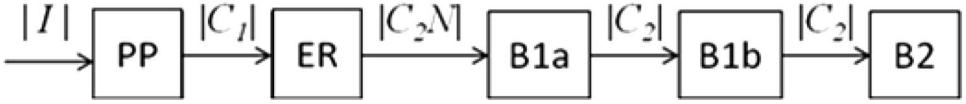


Fig. 5 Relationship of the candidate sizes with the PP-ER algorithm steps

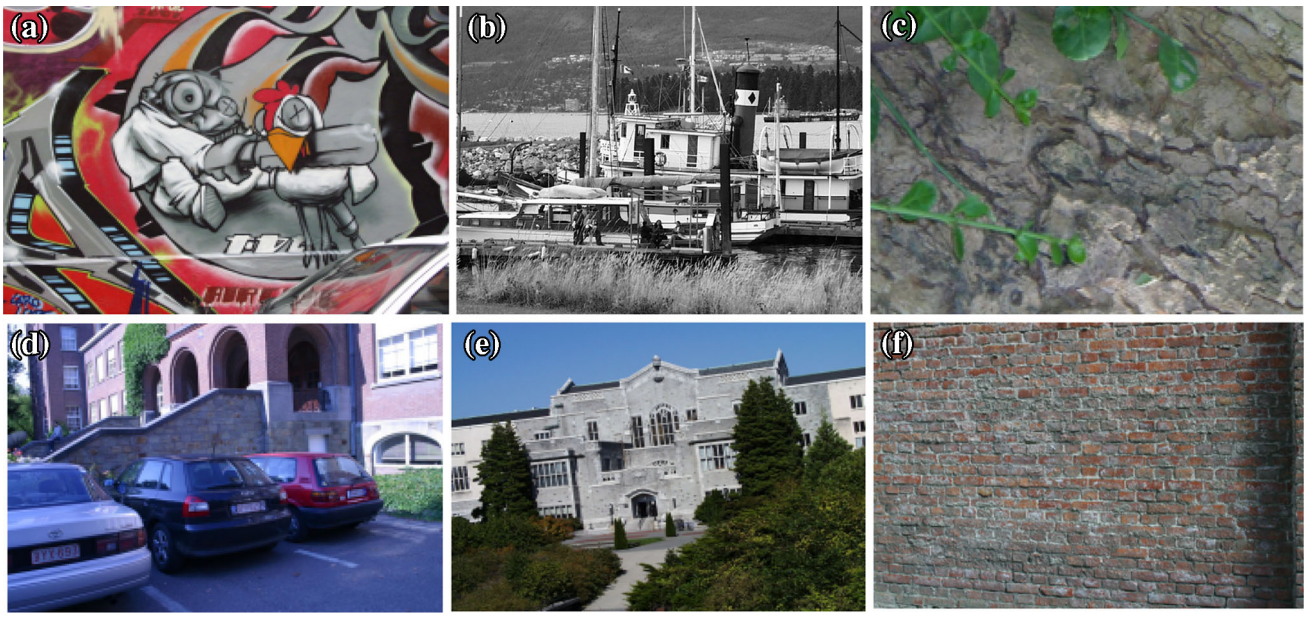


Fig. 6 Image data set used for the evaluation **a** "graf" **b** "boat" **c** "bark" **d** "leuven" **e** "ubc" **f** "wall"

Table 4 Comparison of memory load/store for each pixel for PP-$ER_{ST/H}$

Method	Loads	Stores
Baseline Shi–Tomasi and Harris	$4\ W + 2$	5
PP-$ER_{ST/H}$	$3\ W + pC_1 \times W + pC_2N \times 2 + pC_2 \times 3\ W$	$2 + pC_1 + pC_2N \times 3 + pC_2$

C_2N have been normalized with the image size and are represented as pC_1, pC_2, pC_1N and pC_2N in Tables 2 and 3. Our experiments with the chosen image dataset in Fig. 6 show that the average values for the normalized corner candidate sizes of 300 corners for Shi–Tomasi and Harris are $pC_1 = 0.083$, $pC_2 = 0.052$, $pC_1N = 0.173$ and $pC_2N = 0.134$. The number of operations per pixel can be estimated by substituting these values of pC_1, pC_2, pC_1N and pC_2N in Tables 2 and 3. For example, the total number of multiplications for PP_{ST} in Table 2 is $1 + pC_1N \times 3 + pC_1 \times 2 \approx 1.6$. We can compare this to the corresponding baseline algorithm, Shi–Tomasi—5 multiplications per pixel.

Table 4 shows the comparison of the total number of estimated memory accesses per pixel. The number of memory accesses per pixel for pruning can be estimated by substituting these values of pC_1, pC_2, pC_1N and pC_2N in Table 4 as follows: total number of loads per pixel $\approx 3.24\ W + 0.26$ and stores per pixel ≈ 2.54. Therefore, compared to the baseline algorithms, there is a reduction in the number of memory accesses per pixel.

Tables 1, 2, 3 show that the per-pixel complexity of the operations in the proposed algorithms is lower than that in the baseline algorithms. For example, the PP and ER computations only require 16-bit operations whereas the baseline corner measure computations require 32-bit and floating-point operations. The actual realization of these operations depends on the target processor. For example, if the target processor does not support a floating-point unit, multiple integer operations will be used to emulate the floating-point operations in the conventional corner measure computation. The floating-point emulations will incur a high computational complexity. Similarly, processors that support only 16-bit memory transfers will result in a higher number of memory load/store operations for the conventional corner measure computation.

5 Evaluation and results

In this section, we evaluate the performance of our proposed PP-$ER_{ST/H}$ methods from the standpoint of accuracy and efficiency, and compare it with $PP_{ST/H}$ and the corresponding baseline algorithms. Two variants of each of the baseline algorithms have been used, which is based on the weights $w(x)$ of the filter for the auto-correlation matrix M: (1) Simple averaging filter (also called Box) resulting in Shi–Tomasi-Box (STB) and Harris-Box (HB), and (2) Gaussian filter resulting in Shi–Tomasi–Gaussian (STG) and Harris–Gaussian (HG). The window size is set to $W = 3 \times 3$ and $\sigma = 0.5$ for the Gaussian filter, as in [5]. We used the normalized 3×3 discrete Gaussian kernel as shown below:

$$w(x) = \begin{bmatrix} 0.0113 & 0.0838 & 0.0113 \\ 0.0838 & 0.6193 & 0.0838 \\ 0.0113 & 0.0838 & 0.0113 \end{bmatrix} \quad (11)$$

We used the following 3 × 3 Sobel filters for computing the horizontal and vertical gradient images I_x and I_y:

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}; G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (12)$$

Figure 6 shows the image dataset used, which contains six sequences ("graf", "boat", "bark", "leuven", "ubc", "wall") of images with various image transformations such as changes in viewpoint, zoom, rotation and illumination [29]. The image resolutions are as follows: "graf"—800 × 640, "boat"—850 × 680, "bark"—765 × 512, "leuven"—900 × 600, "ubc"—800 × 640 and "wall"—880 × 680. We generate a feature set of 300 corners for all the algorithms considered. For Shi–Tomasi, we set the threshold on cornerness $T_c = 0.05$ and threshold on pruning filter $T_p = 0.05$. For Harris, we set $T_c = 0.005$ and $T_p = 0.05$.

5.1 Accuracy evaluation

We used the following criteria to evaluate the accuracy of the proposed methods ($PP_{ST/H}$ and $PP\text{-}ER_{ST/H}$):

1. *Feature matches*: We compared the feature set of the proposed methods with the baseline algorithm and counted the number of corners in the baseline algorithm that was also reported by the proposed methods. We report this value as a percentage of the total number of features (e.g., if 270 corners matched out of the total of 300, then we report this as a 90 % match).
2. *Repeatability rate:* Repeatability rate of a feature set is the percentage of features that are simultaneously present in two images and was introduced in [8]. A higher repeatability rate of corner detection algorithm between two images indicates that more features can be matched between these two images and hence these features can be reliably used for aligning the two images. We computed the repeatability rate for the feature sets generated by the baseline and the proposed methods on the six image sequences in Fig. 6. As we expect the pruning to generate feature sets which are very close to the feature sets generated by the baseline algorithm, we report the difference in the repeatability rate of each of the proposed methods and the corresponding baseline method as follows:

$$\Delta r = \text{average}\left(r_{\text{proposed}} - r_{\text{baseline}}\right) \quad (13)$$

Table 5 Accuracy results for PP and PP-ER in comparison to the baseline algorithms

Proposed Methods	# Feature matches		Difference in repeatability rate Δr		
	Min (%)	Max (%)	Min	Max	Average
PP_{STB}	99.7	100.0	0.00	0.33	0.01
PP_{STG}	95.7	100.0	0.00	0.00	0.00
PP_{HB}	100.0	100.0	−0.33	0.00	−0.01
PP_{HG}	99.0	100.0	0.00	0.00	0.00
$PP\text{-}ER_{STB}$	98.0	100.0	−1.84	0.33	−0.13
$PP\text{-}ER_{STG}$	95.0	100.0	−3.07	0.51	−0.17
$PP\text{-}ER_{HB}$	96.7	100.0	−0.67	0.74	0.02
$PP\text{-}ER_{HG}$	98.0	100.0	−3.57	1.26	−0.10

Table 6 Accuracy results for corner candidate pruning method

	#Feature matches		Difference in repeatability rate Δr		
	Min (%)	Max (%)	Min	Max	Average
CCP_{STB}	3.7	16.0	−28.91	1.04	−12.41
CCP_{STG}	2.7	20.3	−36.84	5.18	−9.66
CCP_{HB}	4.7	20.7	−30.46	0.01	−13.40
CCP_{HG}	10.7	38.0	−48.16	−0.51	−15.55

We report the range and the average of the difference in repeatability Δr over all image sequences for each method. When Δr is negative it implies that the repeatability of the proposed method was lower than the baseline method and vice versa. In Table 5, we report the results for the accuracy of the proposed algorithm with respect to the corresponding baseline algorithm for our chosen evaluation criteria: % of feature matches in the feature sets, and difference in repeatability rate. It is clear that the proposed pruning methods are able to match the feature sets produced by the baseline methods to a very high degree. This is also reflected in the repeatability results which show that on an average there is no difference in the repeatability of the feature sets produced by the pruning methods and the baseline algorithms. This shows that pruning is highly efficient in retaining the best quality corners while removing the non-corner regions.

The proposed PP-ER method is compared with the cascaded candidate pruning (CCP) method [27]. CCP selects corner candidates based on the gradient magnitudes of the pixel and then performs non-maximal suppression based on the gradient magnitude. For our analysis, 300 corners are found with both the Box and Gaussian versions of Shi–Tomasi and Harris, and compared with the corners generated by applying pruning using CCP to these baseline

detectors. Table 6 shows the difference in repeatability for each image sequence between the CCP method and the baseline algorithm. From Tables 5 and 6, it can be seen that the proposed technique PP-ER outperforms CCP. It is clear that for almost all image transformations, CCP has substantially lower repeatability than the baseline algorithm. The main reason for this is that CCP selects pixels that are local maxima in the gradient values without considering the pixel neighborhood. The Shi–Tomasi and Harris on the other hand, rely on the variability of the gradient magnitudes in the pixel neighborhood and not the pixel alone. Hence, potential corner pixels of Shi–Tomasi or Harris may not be local maxima in the gradient values. This is the reason why many such pixels are eliminated by the CCP resulting in a lower quality corner set. This analysis exemplifies the need for considering the pixel neighborhood at the pruning stage so as to retain all the potential good corners. The proposed PP-ER method is able to achieve repeatability rates very close to the baseline algorithms because the pruning steps PP and ER have been derived directly from the baseline detectors and are hence able to approximate the final corner measure with high degree of accuracy.

5.2 Efficiency evaluation

The computational efficiency of the pruning techniques $PP_{ST/H}$ and PP-$ER_{ST/H}$ is demonstrated by running the algorithms on the Nios-II embedded platform [30]. The execution time results are obtained for the first image in each image sequence in Fig. 6. To show the overall savings in computations achieved by introducing pruning, we compute the relative speedup (%) of the proposed pruning techniques with respect to the corresponding baseline algorithms as follows:

$$\text{Speedup} = \frac{\left(t_{\text{baseline}} - t_{\text{proposed}}\right)}{t_{\text{baseline}}} \times 100 \tag{14}$$

where t_{baseline} and t_{proposed} are the execution times (in seconds) of the corresponding baseline algorithm and the pruning-based algorithm, respectively. As the images in Fig. 6 have varying image sizes, we use the speedup to normalize the overall savings in execution time over varying image sizes.

As FPU often leads to high-cost and high-power dissipation in embedded processors, we used two configurations of the Nios-II soft core: with FPU disabled and FPU enabled. Figure 7a shows the execution times (in seconds) and speedup (%) for Shi–Tomasi on Nios-II when FPU is disabled. The images have been arranged from left to right from smallest to the largest in terms of their image size. The execution time of the baseline algorithms i.e., STB and STG depends on both the image size and the nature of the image content. Therefore, images of larger size have higher execution times—for example, "wall" has a higher execution time than "bark".

The overall execution time for STB is in the range of (3.3–6.5) s. The execution time for STG is in the range of (11.6–21.5) s due to the more complex corner measure involving the Gaussian filter. The average speedup achieved by PP_{STB} is 38 and 76 % for PP_{STG}. It is to be noted that the execution time of PP_{ST} for both the Box and Gaussian is almost the same. This shows how effective the pruning has been in discarding most of the non-corner regions resulting in very small corner candidate sets. The nature of the image content determines the amount of savings that can be achieved. For images that have a combination of textures and homogenous regions—such as "boat"—the savings are higher with pruning. In comparison, images with a comparable image size but with very rich textures and less homogenous regions—such as "wall"—the benefits from PP_{ST} are lower. Figure 7b shows the additional speedup that is achieved with PP-ER_{ST} when compared to PP_{ST}. It is evident that all the images benefit from the removal of edge pixels resulting in much smaller corner candidate sets that need the cornerness computation. "graf" has very distinct edges that PP_{ST} was unable to prune and, therefore, PP-ER_{ST} shows clear benefits. "wall" does not benefit from PP_{ST}, but shows positive speedup with PP-ER_{ST}.

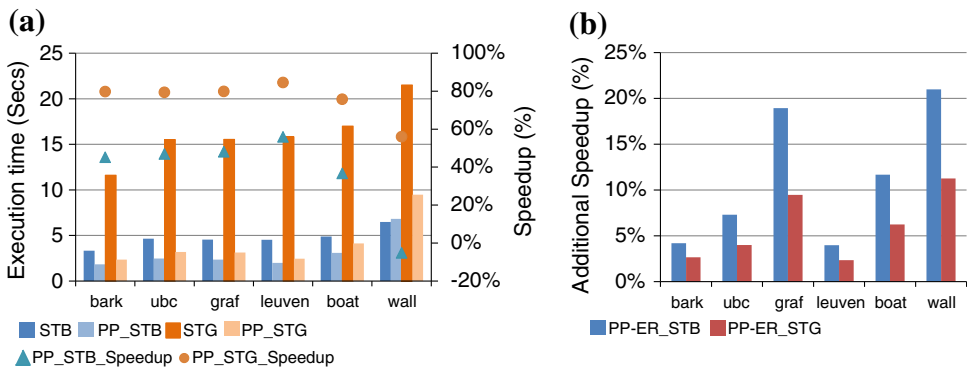


Fig. 7 **a** Speedup in execution time on Nios-II with FPU disabled for Shi–Tomasi with PP_{ST} **b** additional speedup achieved with PP-ER_{ST} when compared with PP_{ST}

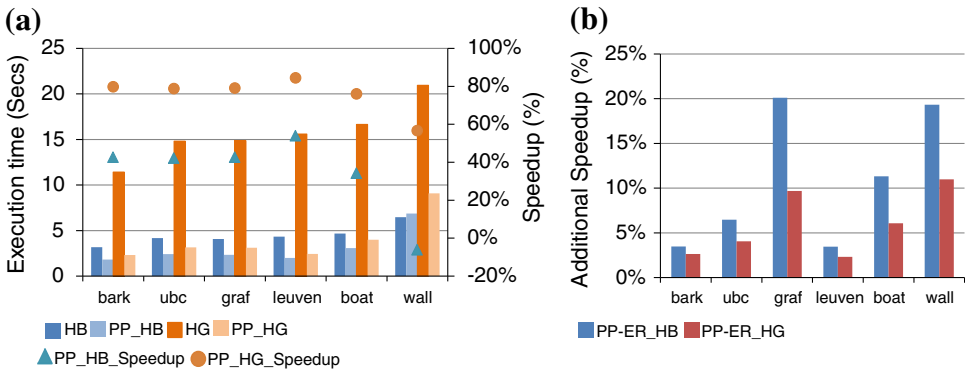


Fig. 8 **a** Speedup in execution time on Nios-II with FPU disabled for Harris with PP_H **b** additional speedup with PP-ER_H when compared with PP_H

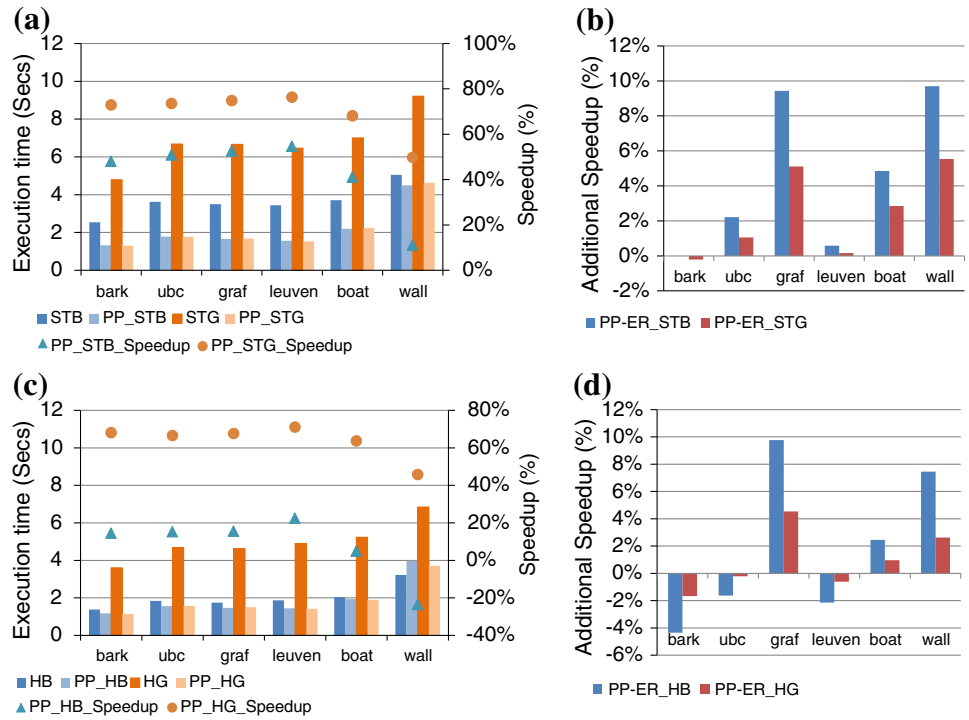


Fig. 9 On Nios-II with FPU enabled **a** speedup in execution time for Shi–Tomasi with PP_{ST}. **b** Additional speedup with PP-ER_{ST} when compared with PP_{ST}. **c** Speedup in execution time for Harris with PP_H. **d** Additional speedup with PP-ER_H when compared with PP_H

Figure 8 shows the corresponding timing results for Harris. The execution times of HB are in the range of (3–6) s. In comparison to STB, the corner measure of HB is less complex and hence, the execution times are lower. The average speedup achieved by PP_{HB} and PP_{HG} is 35 and 76 %, respectively. PP-ER_H achieves higher average speedup of 46 and 82 % for PP-ER_{HB} and PP-ER_{HG}, respectively.

Figure 9 shows the timing results with FPU enabled on Nios-II soft core. When the FPU is available the floating-point operations are handled by the FPU and hence the corner measure computation is executed much faster compared to the software emulation of floating-point arithmetic in the absence of an FPU. However, we still get an average speedup with PP_{STB} of 43 %, PP_{STG} of 69 %, PP_{HB} of 8 % and PP_{HG} of 64 %. PP-$ER_{ST/H}$ still shows significant improvements for "graf" and "wall" when compared to $PP_{ST/H}$.

To analyze the impact of cache on the proposed method, we implemented the baseline and proposed algorithms on two configurations of the Nios-II processor: with and without cache (FPU was disabled in both cases). We report the speedup achieved by the PP-$ER_{ST/H}$ in Fig. 10. When the cache is disabled, then the memory access time becomes the major component of the total execution time, in comparison to the time computation time. However, even in this case, we show substantial speedup of 33 % for PP-ER_{ST} and 37 % for PP-ER_H.

Finally we evaluate the impact of varying the threshold on the corner detection and the speedup achieved with the proposed pruning methods. Figure 11 shows the number of corners released when the threshold T_c is varied for the

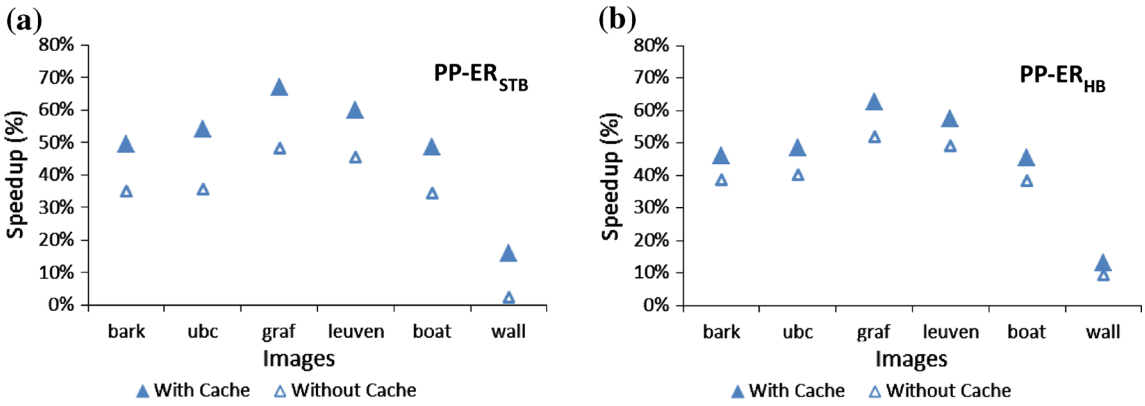


Fig. 10 On Nios-II with FPU disabled. **a** Speedup in execution time for Shi–Tomasi (*Box*) with PP-ER$_{ST}$. **b** Speedup in execution time for Harris (*Box*) with PP-ER$_{H}$

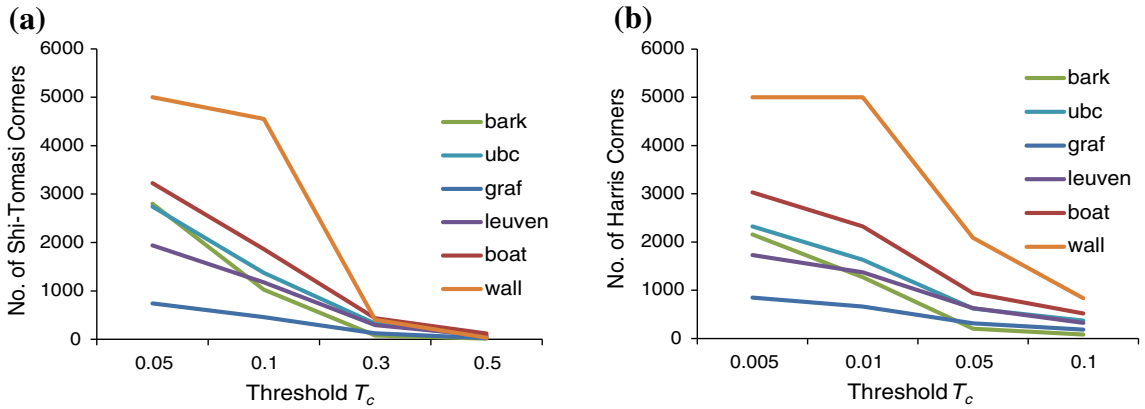


Fig. 11 Impact of varying the threshold T_c on the number of corners in baseline **a** Shi–Tomasi and **b** Harris algorithms

baseline Shi–Tomasi and Harris algorithms and it can be clearly seen that the number of corners released at a given threshold depends on the image content. Also Shi–Tomasi and Harris corner measure values have different ranges. Harris corner measure is directly proportional to the determinant of the auto-correlation matrix, det(M). Shi–Tomasi corner measure is proportional to the root of the determinant of the auto-correlation matrix det(M). Figure 11b shows that Harris requires much lower threshold values for T_c to select comparable number of corners with Shi–Tomasi.

Figure 12 shows the impact of varying the threshold T_p (T_c is set to 0.05 for Shi–Tomasi and 0.005 for Harris to ensure that 300 corners are found). A high value of T_p results in insufficient number of corners whereas a low value of T_p reduces the overall speedup achieved by pruning. Our investigations show that when threshold T_p is set such that 300 corners is found, the repeatability and %feature matches does not vary much with the threshold and is similar to what we have reported in Table 5.

In addition, the benefit of PP-ER when compared to PP can be clearly seen when the thresholds are set lower. At lower thresholds such as when $T_p = 0.01$, PP is unable to prune away many non-corner candidates and incurs an overhead in computations as seen in images "bark", "ubc", "boat" and "wall". However, due to the efficient pruning by the ER step, PP-ER achieves a speedup of an average of 59 % for Shi–Tomasi (Box) and 56 % for Harris (Box) at $T_p = 0.01$. This shows that PP-ER is a highly efficient pruning technique compared to our previously proposed method, PP.

6 Case study with GME on aerial videos

Unmanned aerial vehicles (UAVs) with a camera on-board are increasingly being used for surveillance. Global motion estimation (GME) is an important computer vision task for these UAVs for video encoding [4] and vision-aided takeoff and landing [31]. In this section, we apply the proposed methods for corner detection to video sequences

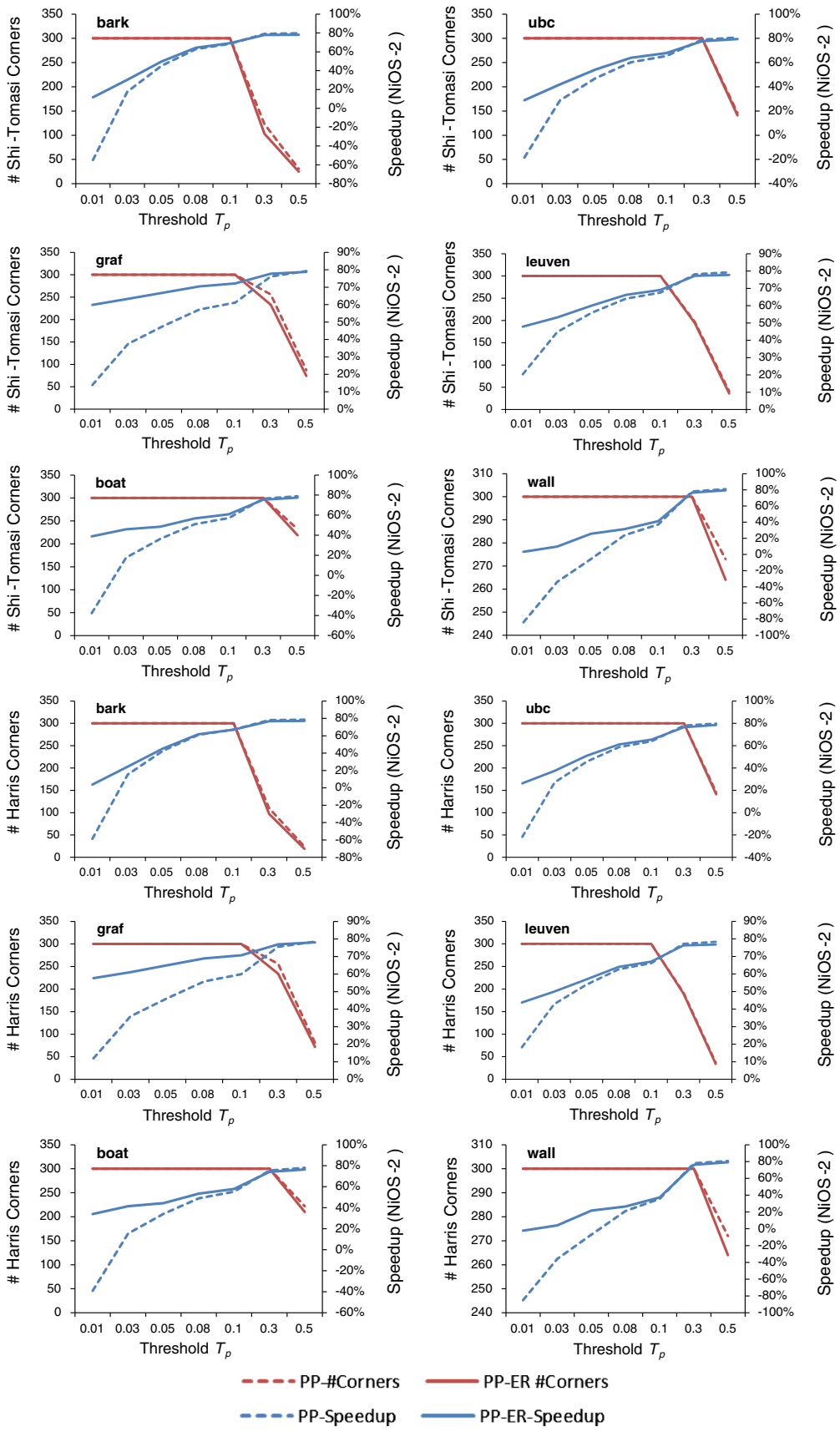


Fig. 12 Impact of varying the pruning threshold T_p on PP_{STB}, $PP\text{-}ER_{STB}$, PP_{HB} and $PP\text{-}ER_{HB}$

Fig. 13 Sample frames from the video sequence used for GME with pruning-based feature detection

Table 7 Quality evaluation for GME using pruning-based corner detection

Methods	PSNR	
	Min (%)	Max (%)
PP_{STB}	−0.3	0.1
PP_{HB}	0.0	0.0
PP-ER_{STB}	−0.4	0.2
PP-ER_{HB}	−0.4	0.2

captured from a camera aboard a moving aerial vehicle, and perform inter-frame registration using a feature-based global motion estimation scheme. We chose a video sequence of 150 frames from the VIRAT surveillance dataset [32]. Figure 13 shows sample frames from this sequence.

We perform feature-based GME on every frame pair in the chosen video sequence. For every frame a maximum of 300 corners is detected using the box variants of the baseline algorithms and their corresponding proposed pruning-based methods. The thresholds are set as follows: Shi–Tomasi-Box ($T_c = 0.01$, $T_p = 0.01$) and Harris-Box ($T_c = 0.001$, $T_p = 0.01$). Then, we applied the Lucas–Kanade feature tracking algorithm [33] to track the corners in the next frame. Then the RanSaC algorithm is applied to the tracked corners and a 2D affine transform is estimated for the GME. This is used to reconstruct the next frame from the current frame. The PSNR between the reconstructed frame and the original frame is computed as in (15):

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{255^2}{\text{MSE}}\right) \tag{15}$$

MSE is the mean of the squared error between the two frames. The PSNR is used to evaluate the registration accuracy of the GME. Table 7 shows the error margin in PSNR for the video frames when the $PP_{ST/H}$ and PP-$ER_{ST/H}$ are used in comparison to the corresponding baseline algorithms. The error margin is within ±0.5 % and shows that the pruning-based corner detection achieves comparable accuracy in GME with the baseline corner detection methods.

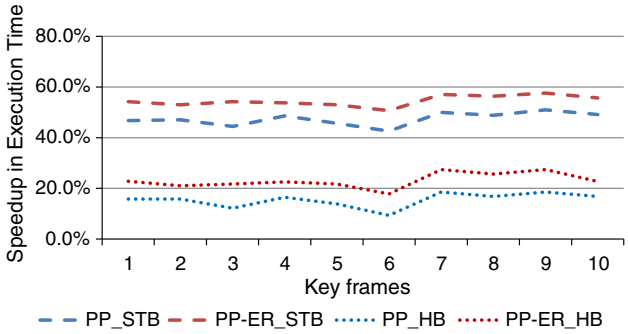


Fig. 14 Speedup in execution time on Nios-II for pruning-based corner detection in GME

As the image content does not change substantially in every frame, in order to demonstrate the computation savings with pruning, we chose a key frame to represent a batch of 15 consecutive frames. Figure 14 shows the speedup achieved in the pruning-based corner detection in these key frames, when executed on Nios-II platform with the on-board FPU enabled.

As these frames contain many edges that need to be removed after $PP_{ST/H}$, PP-$ER_{ST/H}$ shows higher savings in computation time for both STB and HB. For the Gaussian variants of the baseline algorithms, we expect even higher savings in computations than in the box variants. These results show how pruning lowers the per-frame computation complexity of the corner detection enabling GME on resource constrained systems such as robots.

7 Conclusions

In this paper, we have presented a low-complexity pruning technique to accelerate the Shi–Tomasi and Harris corner detectors using an approximate corner indicator derived from the conventional corner measure. Evaluations for repeatability showed that the corner candidates selected by the proposed pruning technique are of the same quality as those found by the baseline detectors. The approximate measure used for pruning allows high thresholds to be applied to remove non-corner regions, while retaining a significant amount of corners. This facilitates the selection

of a small but near-complete set of corner candidates, which results in significant computation savings on corner response evaluation.

Evaluations on a Nios-II processor with FPU show that the proposed PP-ER pruning technique leads to a substantial speedup (in terms of execution time) of 47–71 % in Shi–Tomasi and 10–65 % in Harris for 300 corners. In the absence of the FPU which typifies low-cost/low-power embedded systems, both Shi–Tomasi and Harris benefit from PP-ER with computational savings of 48–82 % and 45–81 %, respectively. When compared to our earlier method PP, the proposed PP-ER pruning technique for 300 corners, shows an average additional speedup of up to 11 % for Harris and 13 % for Shi–Tomasi. However, for lower threshold settings when PP fails to achieve any speedup, PP-ER achieves a much higher average additional speedup of 56 % for Harris and 59 % for Shi–Tomasi. Also, PP-ER enables the use of the more robust but complex Gaussian filter in corner detection especially on systems that do not support FPU. Hence, the proposed low-complexity pruning technique, PP-ER is highly suited for real-time and low-power vision-based embedded systems.

References

1. Mirota, D.J., Ishii, M., Hager, G.D.: Vision-based navigation in image-guided interventions. Annu. Rev. Biomed. Eng. **13**, 297–319 (2011)
2. Ehsan, S., McDonald-Maier, K.D.:On-board vision processing for small UAVs: time to rethink strategy. In: Presented at the Proceedings of the 2009 NASA/ESA Conference on Adaptive Hardware and Systems (2009)
3. Schmidt, A., Kraft, M., Kasiński, A.: An evaluation of image feature detectors and descriptors for robot navigation. Comput. Vis. Graph. **6375**, 251–259 (2010)
4. Bhaskaranand, M., Gibson, J.D.: Low-complexity video encoding for UAV reconnaissance and surveillance. In: Military Communications Conference, 2011—MILCOM 2011, pp. 1633–1638 (2011)
5. Gauglitz, S., Höllerer, T., Turk, M.: Evaluation of interest point detectors and feature descriptors for visual tracking. Int. J. Comput. Vis. **94**, 335–360 (2011)
6. Gil, A., Mozos, O., Ballesta, M., Reinoso, O.: A comparative evaluation of interest point detectors and local descriptors for visual SLAM. Mach. Vis. Appl. **21**, 905–920 (2010)
7. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: a survey. Found. Trends. Comput. Graph. Vis. **3**, 177–280 (2008)
8. Schmid, C., Mohr, R., Bauckhage, C.: Evaluation of interest point detectors. Int. J. Comput. Vis. **37**, 151–172 (2000)
9. Jianbo, S., Tomasi, C.: Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94, 1994 IEEE Computer Society Conference on, 1994, pp. 593–600 (1994)
10. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proceedings of The Fourth Alvey Vision Conference, 1988, pp. 147–151 (1988)
11. Aanæs, H., Dahl, A., Steenstrup Pedersen, K.: Interesting interest points. Int. J. Comput. Vis. **97**, 18–35 (2012)
12. Wu M., Ramakrishnan, N., Lam, S.-K., Srikanthan, T.: Low-complexity pruning for accelerating corner detection. In: Circuits and Systems (ISCAS), 2012 IEEE International Symposium on, 2012, pp. 1684–1687 (2012)
13. Wang, H., Brady, M.: Real-time corner detection algorithm for motion estimation. Image Vis. Comput. **13**, 695–703 (1995)
14. Smith, S.M., Brady, J.M.: SUSAN—a new approach to low level image processing. Int. J. Comput. Vision **23**, 45–78 (1997)
15. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: Presented at the Proceedings of the 9th European conference on Computer Vision, Volume Part I, Graz, Austria (2006)
16. Claus, C., Huitl, R., Rausch, J., Stechele, W.: Optimizing the SUSAN corner detection algorithm for a high speed FPGA implementation. In: Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on, 2009, pp. 138–145 (2009)
17. Chih-Chi, C., Chia-Hua, L., Chung-Te, L., Chang, S.C., Liang-Gee, C.: iVisual: an intelligent visual sensor SoC with 2790fps CMOS image sensor and 205GOPS/W vision processor. In: Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE, 2008, pp. 90–95 (2008)
18. Dietrich, B.: Design and implementation of an FPGA-based stereo vision system for the EyeBot M6. University of Western Australia (2009)
19. Saidani, T., Lacassagne, L., Bouaziz, S., Khan, T.: Parallelization strategies for the points of interests algorithm on the cell processor. In: Stojmenovic, I., Thulasiram, R., Yang, L., Jia, W., Guo, M., Mello, R. (eds.) Parallel and distributed processing and applications, vol. 4742, pp. 104–112. Springer, Berlin (2007)
20. Hosseini, F., Fijany, A., Fontaine, J.-G.: Highly parallel implementation of Harris Corner detector on CSX SIMD architecture. In: Presented at the Proceedings of the 2010 conference on Parallel processing, Ischia, Italy (2011)
21. Piskorski, S., Lacassagne, L., Bouaziz, S., Etiemble, D.: Customizing CPU instructions for embedded vision systems. In: Presented at the Proceedings of the 2007 IEEE International Conference on Application-Specific Systems, Architectures and Processors (2007)
22. Tippetts, B., Lee, D.-J., Archibald, J.: An on-board vision sensor system for small unmanned vehicle applications. Mach. Vis. Appl. **23**(2), 1–13 (2012)
23. Benedetti, A., Perona, P.: Real-time 2-D feature detection on a reconfigurable computer. In: Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on, 1998, pp. 586–593 (1998)
24. Teixeira, L.P., Celes, W., Gattass, M.: Accelerated corner-detector algorithms. In: British Machine Vision Conference, Leeds, United Kingdom (2008)
25. Sinha, S., Frahm, J.-M., Pollefeys, M., Genc, Y.: Feature tracking and matching in video using programmable graphics hardware. Mach. Vis. Appl. **22**, 207–217 (2011)
26. Mainali, P., Qiong, Y., Lafruit, G., Van Gool, L., Lauwereins, R.: Robust low complexity corner detector. IEEE Trans. Circuits Syst. Video Technol. **21**, 435–445 (2011)
27. Alkaabi, S., Deravi, F.: Candidate pruning for fast corner detection. Electron. Lett. **40**, 18–19 (2004)
28. Bouguet, J.-Y.: Pyramidal implementation of the Lucas Kanade feature tracker. Intel corporation, Microprocessor research labs (2000)
29. Affine covariant features. Available: http://www.robots.ox.ac.uk/~vgg/research/affine/
30. Nios II processor: the world's most versatile embedded processor. Available: http://www.altera.com/devices/processor/nios2/ni2-index.html (2012)
31. Claybrough, M., Defay, F.: Stabilization of an unmanned aerial vehicle using real-time embedded motion estimation. In:

Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on, pp. 2189–2194 (2012)
32. Sangmin, O., Hoogs, A., Perera, A., Cuntoor, N., Chia-Chih, C., Jong Taek, L., et al.: A large-scale benchmark dataset for event recognition in surveillance video. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 3153–3160 (2011)
33. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Presented at the Proceedings of the 7th international joint conference on Artificial intelligence, vol. 2, Vancouver, BC, Canada (1981)

Nirmala Ramakrishnan received her B.A.Sc (Honors) in Computer Engineering from Nanyang Technological University (NTU), Singapore in 2002 and Masters in Computing from National University of Singapore in 2005. She had a stint with Hewlett Packard Singapore for 5 years, working on embedded systems development. She is presently pursuing her Ph.D. in School of Computer Engineering, NTU and her research focus is low-complexity computer vision algorithms for embedded systems.

Meiqing Wu received M.S. degree in Computer Engineering from Peking University in 2009. She has worked on a number of challenging projects that involved the development of novel computer vision algorithms for embedded platforms. Meiqing is currently a Ph.D. candidate in the Centre for High Performance Embedded Systems in Nanyang Technological University, Singapore. Her current research interests include computer vision, pattern recognition and embedded systems.

Lam Siew-Kei has been with NTU (Nanyang Technological University, Singapore) since 1994, where he received his B.A.Sc (Honors) and M.Eng in Computer Engineering. He currently holds an appointment as Research Associate in the Centre for High Performance Embedded Systems and has worked on a number of challenging projects that involved the porting of complex algorithms in VLSI. He is also familiar with rapid prototyping and ASIC design flow methodologies. His research interests include embedded system design algorithms and methodologies, algorithms-to-architectural translations, and high-speed arithmetic units. He is a member of IEEE.

Dr Srikanthan joined Nanyang Technological University (NTU), Singapore in 1991. At present, he holds a full professor and joint appointments as Director of a 100 strong Centre for High Performance Embedded Systems (CHiPES) and Director of the Intelligent Devices and Systems (IDeAS) cluster. He founded CHiPES in 1998 and elevated it to a university level research centre in February 2000. He has several years of university experience and his research interests include design methodologies for complex embedded systems, architectural translations of compute-intensive algorithms, computer arithmetic and high-speed techniques for image processing and dynamic routing. He has published more than 250 technical papers including 60 journals in IEEE Transactions, IEE Proceedings and other reputed international journals. His services as a key consultant to embedded systems industry, both locally and internationally are continually being sought for. He was awarded the Public Administration Medal (Bronze) on 2006 National Day in recognition of his contributions to education in Singapore.