J Real-Time Image Proc (2019) 16:1057–1076
DOI 10.1007/s11554-016-0591-1

ORIGINAL RESEARCH PAPER

Context-based image acquisition from memory in digital systems

Jianxiong Liu[1] · **Christos Bouganis**[1] · **Peter Y. K. Cheung**[1]

Received: 11 June 2015 / Accepted: 1 April 2016 / Published online: 5 May 2016

Abstract A key consideration in the design of image and video processing systems is the ever increasing spatial resolution of the captured images, which has a major impact on the performance requirements of the memory subsystem. This is further amplified by the facts that the memory bandwidth requirements and energy consumption of accessing the captured images have started to become the bottlenecks in the design of high-performance image processing systems. Inspired by the successful application of progressive image sampling techniques in various image processing tasks, this work proposes the concept of Context-based Image Acquisition for hardware systems that efficiently trades image quality for reduced cost of the image acquisition process. Based on the proposed framework, a hardware architecture is developed which alters the conventional memory access pattern, to progressively and adaptively access pixels from a memory subsystem. The sampled pixels are used to reconstruct an approximation to the ground truth, which is stored in a high-performance image buffer for further processing. An instance of the architecture is prototyped on an FPGA and its performance evaluation shows that a saving of up to 85 % of memory accessing time and 33 %/45 % of image acquisition time/energy are achieved on a set of benchmarks while maintaining a high PSNR.

✉ Jianxiong Liu
mydendrobium@hotmail.com

✉ Christos Bouganis
christos-savvas.bouganis@imperial.ac.uk

Peter Y. K. Cheung
p.cheung@imperial.ac.uk

[1] Imperial College London, London, UK

Keywords Image acquisition · FPGA · Memory · Sampling · Power · Reconstruction

1 Introduction

In modern image processing systems, the process of image acquisition has become a major concern during their design stage. It is often seen in state-of-art systems that the bandwidth requirement, time, and energy costs of this acquisition process present a significant impact to the overall cost of the complete image processing system [25].

On one hand this is due to the increasing resolution of image capturing devices, which leads to large size of image data. Access and transmission of the image data with ever growing size poses a constant challenge to the design of memory systems [10, 25].

On the other hand, the different rate of performance increase between the computing engines and memory systems leads to the fact that a modern system is usually bounded by its communicational power [20], leading to the development of communicational aware algorithms. This is reflected from two aspects.

Firstly, the improvement of processing speed of computing engines exceeds that of the memory access. According to [19], the computation performance (floating point operations per second) is increasing by 59 % a year whereas communication performance is improving at a much lower rate (DRAM latency improves by 5.5 % and bandwidth improves by 23 % every year). In particular, the development of processors and memory systems saw an increasing performance gap between the two. The performance in this case measures the speed of the two devices, which is defined to be the average memory requests per second for the processor, and average data accesses per second for memory. In recent years

the design of processors moved towards multi-cores which further increases this performance gap [8]. With faster processors, inadequate memory bandwidth and high access latency limit the processors from performing further image processing actions and therefore increases the overall execution time of image processing operations.

Secondly, on top of the execution time, energy consumption has become a major concern in state-of-art hardware designs, and especially in embedded systems. Memory system design considers the balance between size, system complexity, manufacturing cost to name a few. These restrictions as well as the lowered energy cost of logic computations has made the energy cost of data accessing more significant in the overall energy cost of the system than ever before. As is reported in the work of Zhou et al. [25], in their H.264 video decoding system the decoding process (excluding DRAM access) only spends about 0.36 nJ per pixel decoded, whereas the DRAM access process during the decoding costs 1.11 nJ per pixel accessed. In such image processing systems, the memory access process has such a significant presence that it dominates the overall energy consumption of the system.

In general, the cost of image acquisition process both in execution time and energy consumption is becoming increasingly significant. This leads to various researches and developments in dealing with the cost of the image acquisition process. These efforts include general solutions such as the development of memory hierarchy, and image specific solutions such as access pattern optimization [16], and application code rewriting [4], etc.

This work aims to approach this problem from a novel direction. Inspired from the development of image processing techniques, this project proposes the concept of "Context-based Image Acquisition" (CbIA) which serves as a framework of designing an intelligent hardware architecture of image acquisition from a source, which is often in the form of external memory that holds the target image to access. Such architecture utilizes image processing techniques to aid the process of image acquisition. It combines sampling of the target image and reconstruction using the sampled fractions, to explore the design space of image quality and bandwidth/time/energy cost of this acquisition process, which ultimately leads to the reduction of the overall cost of image processing systems. This trade-off between image quality and various costs of image acquisition is the key idea of the proposed CbIA framework that provides an alternative approach of designing hardware architectures for image processing systems.

2 Background

The core of image accessing within hardware system is an image processing application requesting for the image data from a memory external to the housing computing engine of the application. The *effort* spent on this access process includes but is not limited to:

1. *Acquisition time* the overall time required between acquisition order is issued and the requested image data are acquired from the source memory. This is directly related to how much time it is going to take for the complete image processing task to finish.
2. *Energy consumption* the overall energy required to perform the acquisition of target image data, including the energy consumption of both the source memory and the computing engine.
3. *Bandwidth* the total amount of data (measured in bits) accessed from the source memory and transmitted to the computing engine in a fixed period of time. The bandwidth metric here indicates how frequent the source memory is occupied during the image acquisition process.

These efforts are becoming more dominant to the performance of an image processing system due to the increasing resolution and size of image data, as well as the advancing in hardware technology. Reducing these effort has attracted plenty of research interest, and is indeed the main objective of this work. In the rest of the paper, these metrics are collectively denoted as "the cost of image acquisition". Various researches have approached this problem of reducing cost from different perspectives, aiming to reduce the effort of the image data acquisition process. Existing methods approach this problem from both the hardware and software perspectives, which are shown in Fig. 1.

Natural images are usually stored as two-dimensional data matrices that usually have significant spatial correlation. Such image data are often accessed by image processing algorithms in a "block-type" pattern [10], such as in DCT, feature extraction, and video decoding systems. This pattern refers to the processing of image data in nested loops. The outer loops can be seen as a moving analysis window that captures a local region (macro block) of the image, while the inner loops are operations centred on each

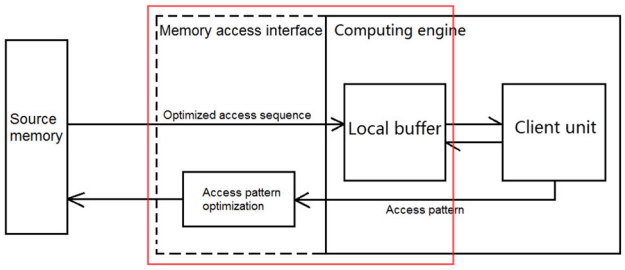

Fig. 1 Image accessing with optimization. Evolving from the basic scenario of image acquisition, existing methods work on the development of memory hierarchy and access pattern optimization (highlighted in *red*)

pixel. These operations often see each pixel used multiple times across consecutive loops and therefore if all pixels within a local area (i.e. "macro block") are readily buffered into the internal fast memory [1], the total cost of the algorithm is reduced. The conventional access pattern sees all pixels of an image macro block read from the memory row by row and stored in the local buffer to be access by the client unit. According to the structure of memory systems, optimizations have been made to accelerate the process and/or lower its energy consumption by reorganising the access patterns [10, 16]. There are also works that construct memory hierarchy specifically for image processing applications [7] and memory structures dedicated to image processing applications [11, 12].

Apart from hardware modifications, a large family of optimization methods is introduced to the application side. Without modifying the source memory from which the data are read, source-to-source code rewriting (such as Data Transfer and Storage Exploration proposed by Catthoor et al. [4]) is done by compilers to transform and optimize the application code in such a way that less temporal storage (e.g. cache) and fewer memory accesses are needed to execute the code. Such methods of automatic code rewriting can be designed to be platform independent [3, 5] which is an advantage for practical use, although the resulting optimized code can produce further improvement if passed through a platform-dependent stage. Last but not least, communication-aware is introduced for linear algebra operations in hardware systems to trade computational effort for communicational effort [9, 20]. While not directly related to image processing hardware, the concept of such trade-off is also explored in this work.

Existing methods all have their strength and deal with specific scenarios. However, most of these methods treat image data as ordinary sequence of numbers without exploiting its contextual features. Moreover, most of the existing methods require modifications to either the source memory or the application itself. This work aims to approach the problem from a different perspective by making use of the contextual features of image data, to design a universally compatible image acquisition architecture that dynamically adapts to the current hardware environment, reduces image acquisition costs, and can be easily plugged into existing hardware systems.

It is worth noting that, in parallel with the works reviewed above, the development of image/video compression algorithms share a similar motivation. Compression of data reduces its size and therefore saves the bandwidth of transmission. However, these algorithms target a more application level of the problem rather than the hardware level, which is the focus of this work. Compression algorithms often require preprocessing of the data as well as a decompression stage, which are not always possible or affordable in intermediate hardware subsystems. An alternative way of illustrating the difference is that, this work is orthogonal to the compression algorithms. These algorithms themselves require the accessing of image data on the hardware level, and therefore can be part of the application scenario this work and the above-mentioned works target. This work is more inline with a subset of the compression algorithms, namely the image recompression [14, 15]. These are essentially light weight compression techniques that offer some of the qualities required by the hardware systems. It still requires preprocessing of the data and is not as generic as this work aims to be, where there is no access to the data before the acquisition happens.

3 The framework of CbIA

3.1 The idea of context-based image acquisition

In this paper, the concept of "Context-based Image Acquisition" (CbIA) is introduced to reduce the cost of image access process within hardware systems. The core idea of CbIA framework is to enable the computing engine to dynamically and adaptively acquire a target region (macro block) of image data from a source memory. Instead of accessing every bit of stored data from the source, CbIA architecture is designed to be able to dynamically select and sample particular fractions of data and fill in the missing parts by combining the sampled data with a certain level of prior knowledge learned from natural images. The end result of this alternative access process is an approximation to the original image, which is stored in the local scratchpad memory and used by a client application. Thus, by reducing the number of times the system accesses the source memory, the proposed CbIA framework aims to reduce the overall cost of image acquisition process, but increasing the computational cost.

Without loss of generality, Fig. 2 describes the difference between the conventional access process and the proposed process. As shown in Fig. 2a, the conventional image access process often sees the data—in this case, pixels—of the target ground truth image being accessed in a sequential order. The memory access interface of the computing engine requests for data by providing addresses to the memory. Each pixel is accessed in this way and transmitted from the source memory back to the local image buffering memory. The image data are not ready and cannot be used before time point t_0. At the end of the process, the target image is completely copied to the local image buffering memory.

The altered access process maintains a modelling of the target image and from the model, it requests for pixels (or

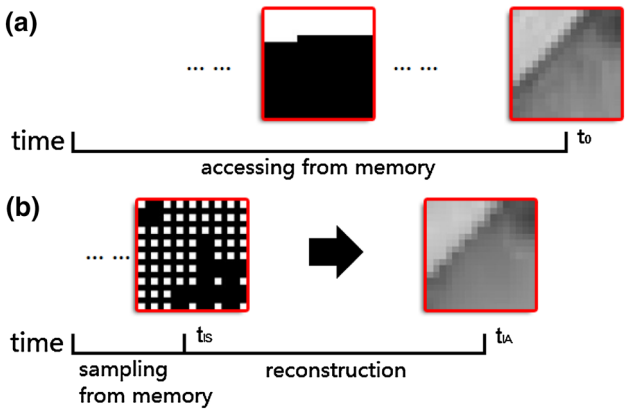

Fig. 2 Comparison between conventional image accessing method and the proposed method. **a** The conventional accessing method; **b** the proposed method

data sample in general) that are considered of most "significance" (Fig. 2b). The "significance" is defined to be the potential of a sample to contribute to the quality of the final approximation. Starting from some initial coarse sampling pattern, the proposed CbIA architecture identifies candidate samples that are the most significant. After acquiring these samples, the modelling of image statistics is updated accordingly, providing another set of candidate sample locations of high estimated significance. The architecture iteratively performs these steps to progressively acquire more data from the source memory and refine the internal prediction of the image. At the point (t_{IS} in Fig. 2b) when the architecture determines that enough samples have been acquired, the sampling process stops. The rest of the missing part are filled in by reconstruction methods using the existing samples. At the end of the reconstruction process (t_{IA} in Fig. 2b), an approximation of the target region is completed within the local image buffering memory.

This altered access process is dynamic in that, it can be tuned according to available resources in the current environment, such as memory bandwidth, energy consumption, and execution time, etc. By allowing a reduction in the quality of the acquired image, both t_{IS} and t_{IA} can be tuned on demand unlike the completion time of the conventional approach. which is always fixed.

For the rest of the paper, the following two terms are defined:

1. *Image sampling process* refers to the process of accessing actual data samples from the source memory, i.e. the time period upto t_{IS}. This is directly related to the bandwidth requirement of the image acquisition process.
2. *Image acquisition process* refers to the complete process starting from the client application requiring to access an image data from source memory, till the end of reconstruction when the approximation of the ground truth is ready and returned to the client application (from start to t_{IA}).

From the perspective of the client application the data are not ready until the end of the complete *image acquisition process*. Although that makes the time of the image acquisition process a major performance metric, a short *image sampling process* still carries benefits as it can reduce the bandwidth requirement of the memory as between t_{IS} and t_{IA} the source memory is freed and is available to be used by other entities that potentially exist alongside the computing engine in question. Therefore, the bandwidth requirement is also part of the main metrics that this work addresses for.

3.2 Application scenario

There are various hardware computing engines that are commonly used in modern times. Starting from the custom hardware platforms (ASIC, FPGA, etc.), to the commercially popular general purpose computers, these computing engines all have their own structure and individual features. There are also a large collection of different designs of memory chips that all have distinguishing accessing protocols, making the collection of different hardware system designs even richer.

The idea behind CbIA is to introduce such a framework of hardware architecture which does not conflict with existing hardware protocols while providing the functionality described in the previous section. It endeavours to have universal compatibility with existing hardware systems and therefore is easy to use and of low implementation cost. In this paper, however, the custom hardware is targeted as the research platform and the following scenario is set on which the problem is discussed (Fig. 3).

This scenario sticks with the core mechanism of digital image data accessing within hardware systems, despite the possible varying designs on either end of the transmission. It is based on the very fundamental addressing-accessing

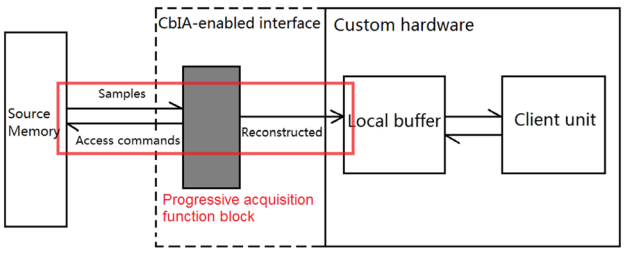

Fig. 3 Scenario setup. The proposed CbIA architecture replaces the conventional image access process (Fig. 1) with a dynamic and progressive sampling procedure (marked in *red*)

mechanism of memory access protocols. It does not conflict with common custom hardware structures (e.g. ASIC, FPGA) and allows for a rich selection of source memories to act as the holder of the target image data. It is worth noting, however, that a local buffering mechanism is required at some point during the acquisition process, for the CbIA to take advantage of the redundancy of image data. Therefore in the extreme case of streaming data in a pixel basis, the proposed idea is not applicable.

Within the computing engine, an implementation of a client image processing application is installed which issues the order to fetch the target image data. The data required are stored in the source memory in the form of pixel matrix. This source memory must have a certain degree of random accessibility: data content in this memory should be able to be accessed by addressing. A local buffer memory is embedded between the client application and the source memory, as is the case for most image processing systems.

The CbIA architecture is responsible of the image acquisition process. According to the request from the client application to access a region of the target image, the architecture computes and generates an optimized sequence of addresses for the samples to be accessed. The architecture works on the local buffer memory, storing sampled data and filling in the missing parts by reconstruction algorithms. At the end of the process, an approximation of the ground truth exists in the local buffer, ready to be used by the client application.

Although in this paper, discussions are focused on accessing image data from source memory chips, this source may as well be of other types such as cameras or other image capturing sensors. The proposed approach can be applied to any sensor that provides some type of addressing mode to access the data. The memory block used in the explanation of the proposed architecture in this paper can be used as an abstraction of such sensor. The proposed architecture is expected to be beneficial in any situation where the accessing of data is relatively costly compared with computational effort, i.e. where a dynamic trade between accessing effort and computational effort is desired.

This scenario serves as a guideline for the design of CbIA architectures so that the design and discussion of the idea are applicable to any hardware environment that agrees with the scenario.

4 Design of a CbIA architecture on reconfigurable hardware

In this section, an example design of the CbIA architecture is given. The design follows the framework of Context-based Image Acquisition introduced in Sect. 3.1 and is based on the scenario set in Sect. 3.2. The core of the proposed architecture is image point sampling which helps the architecture to achieve the progressive data acquisition process and define an instance of the "sample significance" described in Sect. 3.1.

Among various custom hardware platforms, FPGA is chosen as the evaluation platform due to its customizable structure and relatively low design cycle. Moreover, in this paper structured-ASIC chips from Altera are used to estimate the ASIC implementation performance because of its low design cost and easy compatibility with FPGA design.

It is worth noting that, there are many other possibilities of designing an CbIA architecture. Depending on the reconstruction algorithm, the unit of sampling data can vary from pixels to blocks or even data cell resulted from image recompression methods; the definition of "sample significance" is also heavily dependent on the reconstruction methodology [17, 18].

4.1 Scenario setup

To allow for a practical design, the target scenario in this section is more detailed than the generic scenario (Fig. 3). Among various types of memories, popular SDRAMs are used as source memory which contains the target image to acquire. The architecture is implemented and evaluated on FPGA and structured ASIC devices because of their reconfigurability and short design cycle. The designed CbIA architecture makes use of the logic and storage resources on these reconfigurable hardware. In this scenario, the task of the CbIA architecture is to adaptively and dynamically acquire the target image from the external SDRAM, and eventually prepare for an approximation of the ground truth within on-chip memory for the client application to access.

As is briefly reviewed in Sect. 2, image processing algorithms often access image data in nested loops [10]. Because of this featured "block-type access", hardware systems that are designed specifically for image processing applications often use high speed high bandwidth local buffers to temporarily store local regions of the target image. Context-based Image Acquisition does not change the way conventional memory hierarchy works but instead introduces adaptivity and intelligence to the memory accessing protocols. In the scenario defined in this section, the design of the CbIA architecture aims to adaptively and dynamically move macroblocks from the source SDRAM to the local buffer, which is the on-chip BRAMs/registers.

The buffering of the block-type image fractions also gave rise to various data organizing strategies developed to store image data in SDRAMs [14, 24]. In this section, the discussion is focused on the linear and block mapping strategy of storing image data as they are the most popular

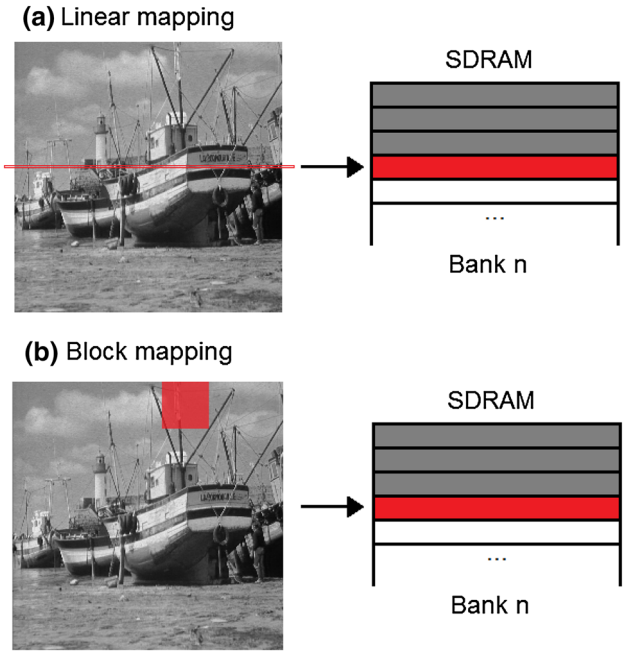

Fig. 4 Linear mapping (**a**) and block mapping (**b**) of image data in SDRAM

methods (Fig. 4) [10]. For linear mapping strategy, pixels of an image are stored row by row within the source memory, each row of the image corresponds to a single page of the SDRAM. The block mapping strategy on the other hand stores each macro block of the image in a single page of SDRAM to minimize the number of row switching activity.

4.2 Design of the sampling procedure

An CbIA procedure requires a image reconstruction algorithm and a sampling algorithm. In this paper, we focus on the sampling of raw image pixels and use interpolation-based method to reconstruct.

The simplest approach along this line is to uniformly refine the image. Starting from a relatively coarse sampling distance,[1] during each iteration the sampling process reduces the sampling distance by a factor of 2 and samples all missing pixels belonging to this sampling distance. The process stops when available bandwidth is depleted and a reconstruction of image can be retrieved from the sampled pixels. Although this scheme requires only the minimum amount of controlling logic, it does not have any data adaptability, therefore the pixels sampled are not always statistically significant. This leads to the inability of the system to make efficient trade between bandwidth and image quality. Moreover, the step size in the number of samples between each sampling distance is fixed and is large (6.25 % to 25 % to 100 %), rendering such strategy of limited use in practice. Such fixed step size limits the flexibility of the sampling procedure to adapt to current hardware environment, i.e. the procedure cannot stop at any time on request.

With the practical hardware scenario in mind, this work takes a different approach to uniform sampling and adopts the estimation of priority scores [6] of candidate pixels. The use of variants of priority scores holds key to many progressive point sampling techniques [6, 22]. Most sampling procedures start with a coarse sampling pattern of the target image, and iteratively identify and sample the pixels of most significance to the improvement of the reconstruction quality. Although defined differently, variants of priority scores share a similar base concept. The priority score introduced in [6] is extended in this work to a more general form that describes such concept:

$$f(\mathbf{x}_i) = d_{\mathbf{x}_i,P} * v_i, \quad \mathrm{P} : \mathrm{sampledpixellocations} \tag{1}$$

where $\mathbf{x}_i$ is the coordinate vector of a candidate unsampled pixel, and the *distance term* $d_{\mathbf{x}_i,P}$ measures the likelihood of determining pixel $\mathbf{x}_i$ with existing samples. This *distance* therefore includes, but is not limited to Euclidean distance of pixel coordinates. The *variance term* v_i is the estimated variance of the distribution of pixel $\mathbf{x}_i$. Given the interpolation based reconstruction methods which assumes general continuity of natural signals, this priority score reflects the least confident locations of the reconstruction process, hence is the measurement of "sample significance".

An instance of progressive sampling using priority scores (denoted as Full Adaptive Sampling in this article) is shown in Fig. 5. This adaptive sampling procedure works on regular grid and the priority of pixels is determined by the priority score of its containing image block:

$$f(b_i) = \mathrm{area}(b_i) * [\max_j(p(\mathbf{v}_j)) - \min_j(p(\mathbf{v}_j))] \tag{2}$$

where $\mathrm{area}(b_i)$ is the area of the block b_i and $p(\mathbf{v}_j)$ is the pixel value of one of the four vertices of b_i at location $\{\mathbf{v}_j = (x_j, y_j) | j = 1, 2, 3, 4; \mathbf{v}_j \in b_i\}$. The area of the block b_i is measured by the total number of pixels contained in this block. In every iteration, the block of the highest score is refined to a finer resolution and the process keeps running until a user defined quality requirement is met or there is no more pixel to sample from. At each step of sampling five more pixels, this sampling procedure samples the pixels that are considered by the procedure to be the most significant. Therefore, compared with uniform sampling, this sampling procedure has a much finer step size of sampling, being more responsive to the surrounding hardware environment.

[1] In this work, the "sampling distance" of image uniform sampling refers to how dense the sampling is. A sampling distance of 4 means the image is sampled every four pixels in both horizontal and vertical direction.

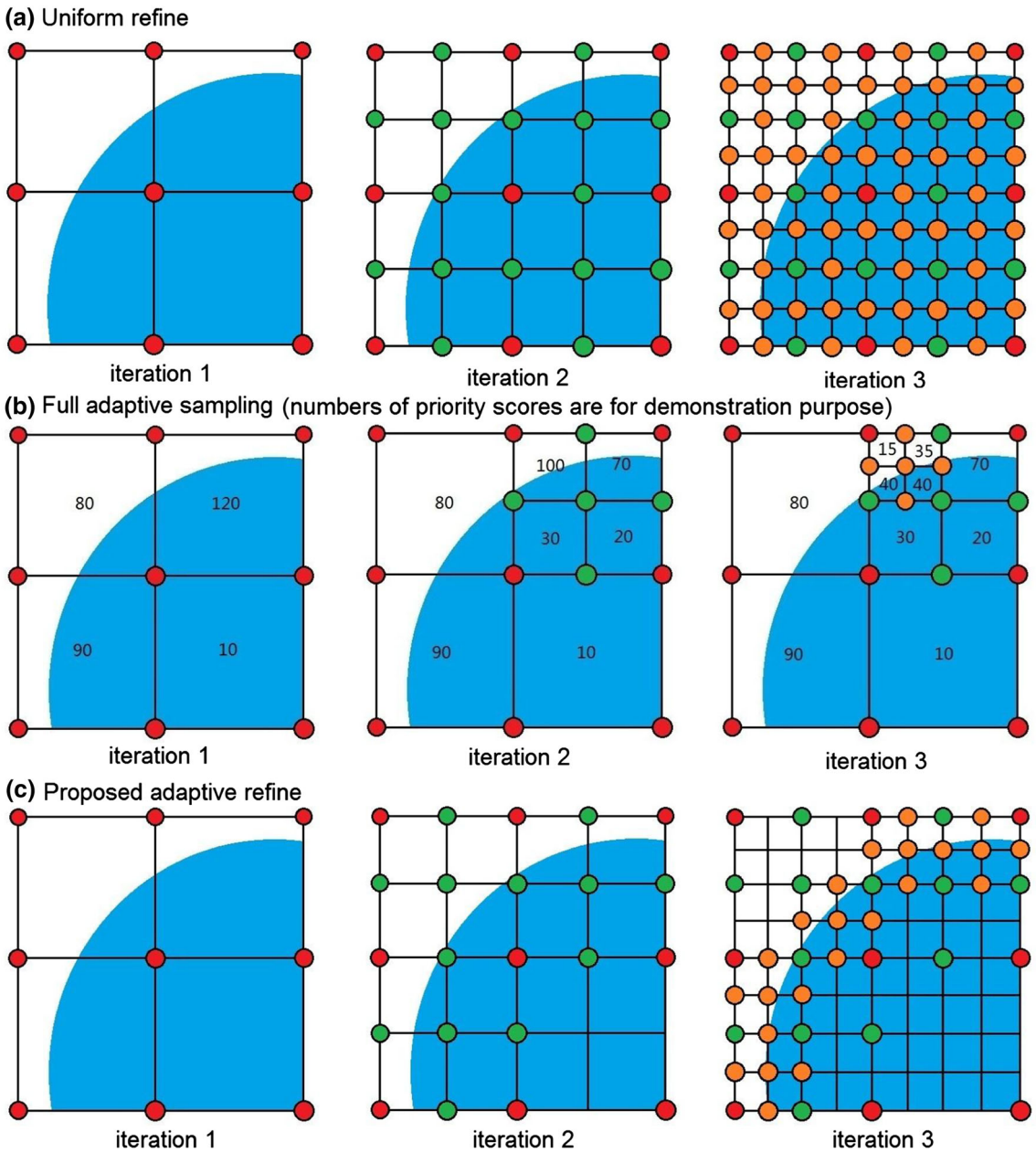

Fig. 5 Progressive sampling methods: uniform sampling (*top*); full adaptive sampling (*middle*); HW-driven adaptive refine (*bottom*)

However, it can be seen that such adaptive sampling is to some extent against the structure of existing hardware systems and DRAMs in concept. While the design of modern hardware systems emphasises the use of data locality to reduce the cost of accessing, full adaptive sampling utilizes the data locality in a different way. It decouples data transmitted in a stream in an attempt to achieve maximum entropy gain with a limited bandwidth. The design of such sampling procedure is based on the assumption that switching sampling location has no cost, which is not true for DRAM access patterns. Therefore this work proposes the HW-driven Adaptive Refine procedure which adopts a modified process based on Full Adaptive Sampling, but is more suitable for DRAM (Fig. 5c).

The HW-driven Adaptive Refine procedure follows the same steps as the Full Adaptive Sampling but adaptively refines every block belonging to the current sampling distance, that has a priority score higher than a given threshold, instead of refining only the block that has the highest priority score at the moment. In practice, the threshold can increase gradually as well to adapt to currently available bandwidth. Although this Adaptive Refine

process cannot guarantee the best sampling pattern in between different threshold levels, it still produces the same sampling pattern as full adaptive sampling does when each threshold is met. The HW-driven Adaptive Refine procedure uses the threshold parameter to control the length of buffering candidate sampling addresses, which allows the architecture to better arrange the actual order of accessing these candidate samples from the DRAM. It also allows for deeper pipelining of the memory access process and the address generating process, while maintaining the data adaptivity of Full Adaptive Sampling.

4.3 Structure of the proposed architecture

The proposed CbIA architecture of image acquisition is implemented on a reconfigurable hardware device. The implemented CbIA architecture is responsible of progressively generating addresses of pixel to sample from a source memory, reconstructing an approximation of the original image data using sampled pixels, and finally store the reconstruction in a local buffer for the potential client application to use.

Figure 6 shows a block diagram of the system. In general the proposed design generates pixel addresses for the DRAM interface (Fig. 6a) to access the requested macro block from the original image. The CbIA procedure operates on a local buffer (Fig. 6b) that is prepared for buffering the macro block of the target image. Sampled pixels are filled into this buffer and based on these samples, the system decides where to sample next. Starting from a coarse uniform sampling pattern, the system checks priority scores of existing blocks (Fig. 6c) and refines their resolution accordingly (Fig. 6d) by requesting to sample more pixels or, if the priority score meets the threshold, passing the block to be interpolated (Fig. 6e). After all blocks of the current sampling distance are processed, the system moves onto next resolution level. When the sampling process achieves a given quality threshold it stops and the remaining missing pixels in the buffer are filled by Bilinear interpolation (Fig. 6e). During the whole process, the target macro block is divided into a number of smaller blocks depending on the sampling statistics. The proposed design characterises each block by its resolution level and anchor, which is the coordinates of its upper left pixel. In detail, the system consists of three major units: *refine_unit*, *addr_translator*, and *interp_unit*. The connection and block diagram of these units are shown in Fig. 6. The process is described in Algorithm 1.

Algorithm 1 The working mechanism of the proposed CbIA architecture

Require: An initial uniform sampling pattern; a given priority threshold thr; identities of initial blocks stored in $FIFO\_A$.
Ensure: the acquired image macro block
while sampling distance > 1 **do**
  while $FIFO\_A$ is not empty **do**
    $refine\_unit$ fetches a block stored in $FIFO\_A$ and check its priority score (Eq 2)
    if priority score $> thr$ **then**
      Stores the block in $FIFO\_C$
      $addr\_translator$ fetches blocks from $FIFO\_C$ and generates the addresses of pixels to sample from these blocks
      Each block refined by $addr\_translator$ is broken down to four sub-blocks
      Newly generated sub-blocks are stored in $FIFO\_B$
    else
      Stores the block in $FIFO\_D$
      $interp_unit$s fetch blocks from $FIFO\_D$ and interpolate for the missing pixels
      The interpolation results are stored into the local canvas buffer
    end if
  end while
  $FIFO\_A$ and $FIFO\_B$ swap place
  Sampling distance divided by 2
end while
return the acquired image macro block

5 Evaluation of the design

The designed CbIA architecture was synthesised and placed and routed on Stratix IV FPGA and Hardcopy IV structured ASIC from Altera. The SDRAM (source memory) response and the rest of the SDRAM access interface are both simulated with Modelsim testbench. The generated SDRAM accessing addresses are passed to SDRAM power models designed by Rambus and HP, which in turn report the corresponding SDRAM energy consumption of the input access pattern. Various 1Gb DDR3s are simulated by the power model from Rambus [23] as the target SDRAM, and the test is also carried out on two smaller sized SDRAM memories modelled by the CACTI tool designed by HP [21]. The SDRAMs simulated by Rambus model and CACTI model are of 55 and 45 nm technology, respectively, while both are of eight bit I/O and have a burst length of 8.

The synthesised architecture performs sampling and reconstruction of three benchmark images which are: lena, barbara, and boat. All of them are of size 527×527 and transformed to grayscale image with each pixel represented by a 8-bit value. The system works on macroblocks (non-overlapping) of size 17×17 on the target image. For each

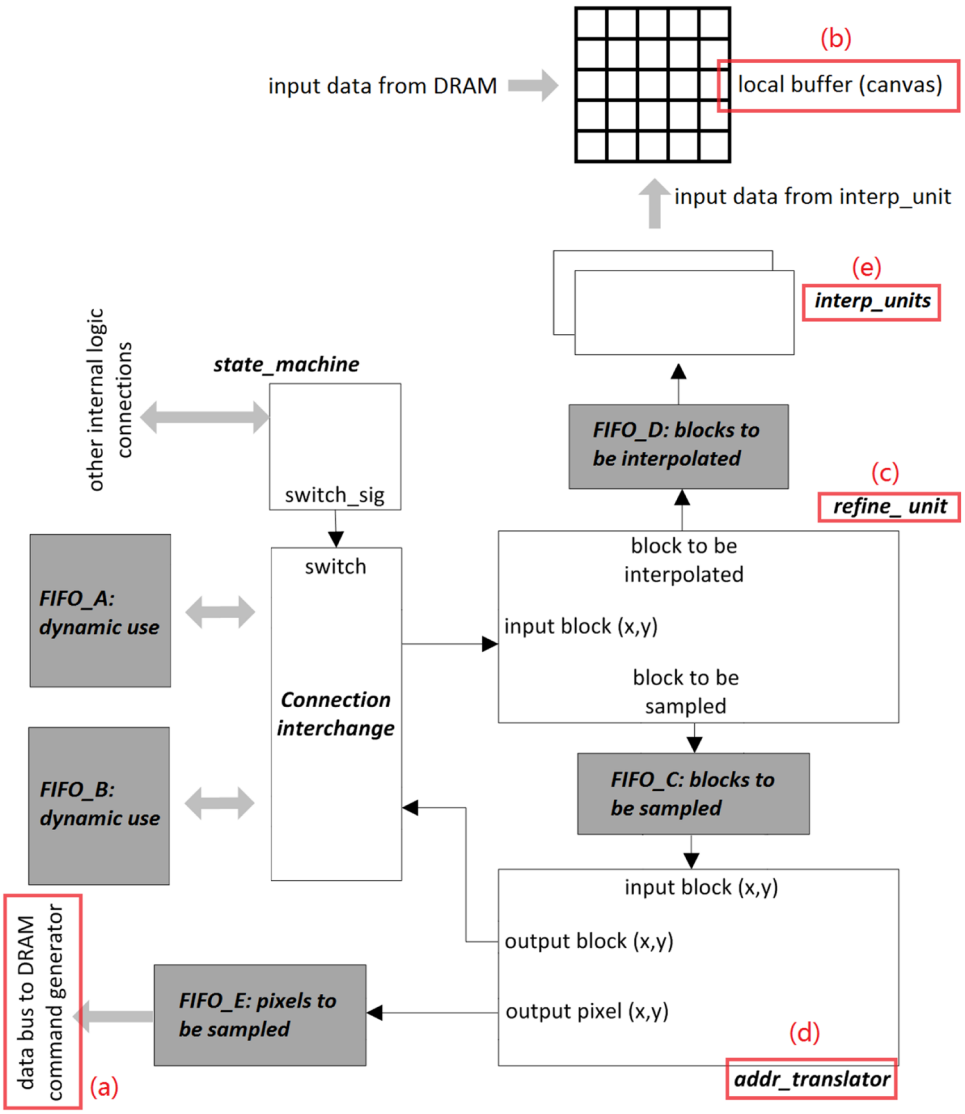

Fig. 6 The structure of the proposed CbIA architecture. **a** The proposed design generates pixel addresses for the DRAM interface; **b** a local canvas buffer stores sampled pixels as well as interpolated pixels; **c** the *refine_unit* checks priority scores of each block; **d** the *addr_translator* generates sampling addresses if a block is to be refined; **e** an array of *interp_units* interpolates the missing pixels for blocks that do not need further refining/sampling

test, the architecture progresses the threshold score (thr) from 1800 (worst quality) to 150 (best quality), gradually increasing the number of samples. Notice that the exact value of threshold score depends on the actual application and image type. In this work the evaluation sweeps through a range of threshold scores that roughly lead to a total of 5–20 % pixels sampled from the benchmark images. In practice the threshold score can be constantly changing to allow more samples in.

Although benchmark images are shown as evaluation results, in fact for the proposed CbIA architecture it acquires these images in macroblocks. The acquired result of these benchmark images are displayed only as a reference of visual quality for the PSNR numbers. For image processing systems, macroblocks of size of $2^n \times 2^n$ are often used due to the nature of digital systems. For the proposed method, however, for the sampling procedure to uniformly refine image blocks, the size of the blocks has to be $3 \times 3, 5 \times 5, 9 \times 9, 17 \times 17$, etc. (Fig. 7a). There are various approaches to mitigate the effect of the irregular block size. For example, when consecutive non-overlapping 4×4 macroblocks (Fig. 7b) are requested by the client application, the proposed acquisition procedure can fetch 5×5 blocks that overlap by one row/column to minimize the overhead of computations on the last row/column. In worst case scenario if a single macro block of size 2^n at random location is requested, the proposed acquisition procedure fetches a $(2^n + 1) \times (2^n + 1)$ macro block without interpolating pixels on the last row/column to minimize the overhead cost. In the evaluation of the proposed CbIA architecture, the experiments treats each macro block as an independent task and their size is set to be 17×17, providing performance measurements of the architecture in ideal situation.

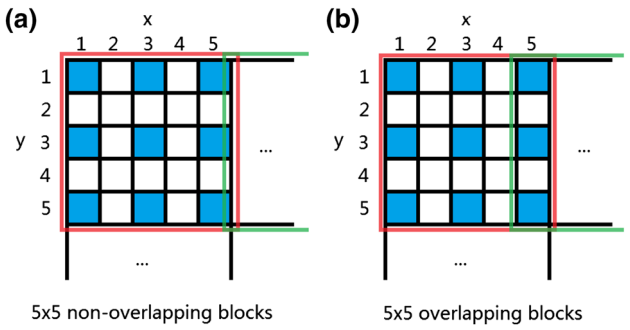

Fig. 7 The size of macroblocks. *Red and green rectangles* mark blocks processed by the proposed system. Pixels marked in *blue* are samples taken during the refining process. The uniform refining of each block requires the size of blocks to be $(2^n + 1) \times (2^n + 1)$

In the following sections, the HW-driven Adaptive Refine method is firstly evaluated against various reference sampling methods for a performance measurement of image quality vs. sampled pixels. Then a detailed evaluation of the proposed CbIA architecture is given together with an evaluation of the impact of the proposed CbIA procedure to an image compression application. Finally the mapping from FPGA implementation to ASIC implementation is discussed.

5.1 Evaluation of the proposed HW-driven adaptive refine procedure

The HW-driven Adaptive Refine is first evaluated against three reference methods for its ability to trade PSNR for reduced SDRAM access time and access energy. The references are: conventional accessing pattern that reads every pixel from SDRAM, uniform refine, and Full Adaptive Sampling on a regular grid. This test first gives an evaluation of the image quality (PSNR) vs. percentage of pixels sampled. Then the test analyses the upper limit of the performance of the proposed system in reducing bandwidth, time and energy consumption of the image acquisition process, temporarily ignoring the cost overhead introduced by the proposed architecture itself. For this test both linear and block mapping strategies are used. For linear mapping each row of the image is stored in a single page within the DRAM, whereas for block mapping each macro block is stored in a single page such that the row switching activities are reduced while reading a macro block.

Figure 8a shows the target image that needs to be acquired, where Fig. 8b shows an instance of the reconstructed image from the HW-driven adaptive refine sampling method where the threshold was set to 600. One block after another, the image is sampled and reconstructed. Figure 8c shows the achieved PSNR of the reconstructed image[2] as a function of the percentage of the pixels sampled. In this graph, performance lines of Full Adaptive Sampling and HW-driven Adaptive Refine coincide because they provide the same sampling pattern at each threshold level and their difference lies in the flexibility of the procedure to optimize the actual order of address sequence sent to the memory. The graph shows that the HW-driven Adaptive Refine, as original Full Adaptive Sampling does, improves the ratio of image PSNR vs. number of pixels sampled and it fills the large gaps between data points from uniform refine method. Moreover, in this evaluation the way memory contents are mapped (linear or block) makes no difference to this performance metric.

The graphs in Fig. 8d, e are the corresponding SDRAM access energy and access time at each achieved PSNR level. Under the assumption of no cost overhead of operating the proposed CbIA architecture, these two metrics represent the overall cost in energy and time of the image acquisition process. Therefore these two graphs show an upper limit of the performance of the proposed CbIA architecture as well as the margin of trade-offs. Additionally, SDRAM access time in Fig. 8e also represents the reduction in memory bandwidth requirement. Both graphs show that the introduced cost overhead (SDRAM access energy and time) from progressive sampling methods is more obvious on linear mapped memory content, but the HW-driven Adaptive Refine allows the system to more flexibly organise the accessing pattern, resulting in a much lowered cost overhead than Full Adaptive Sampling (blue lines vs. green lines). In the case of block mapped memory content such overhead is minimized, and therefore a greater reduction of access energy and access time can be seen. Nevertheless, for both mapping strategies an overall reduction in SDRAM occupation time and access energy can be seen with PSNR up to 35 dB.

Apart from the cost overhead of SDRAM access, the implementation and execution of the proposed system also inevitably introduces overhead. Any access time or energy saved from the SDRAM side has to be compared with the cost of implementing the method. Results in Fig. 8 shows an overview of the trade-off margin offered by the proposed sampling procedure. It shows the ability of the CbIA procedure to trade image quality for reduced number of access times. To reduce the overall time and energy cost of image acquisition process, following conditions have to be met:

1. For the CbIA-based memory interface to reduce the total time of image acquisition compared with the conventional acquisition process, the average time cost

[2] Note that the conventional method (accessing all pixels) has infinite PSNR and therefore is not plotted in this graph.

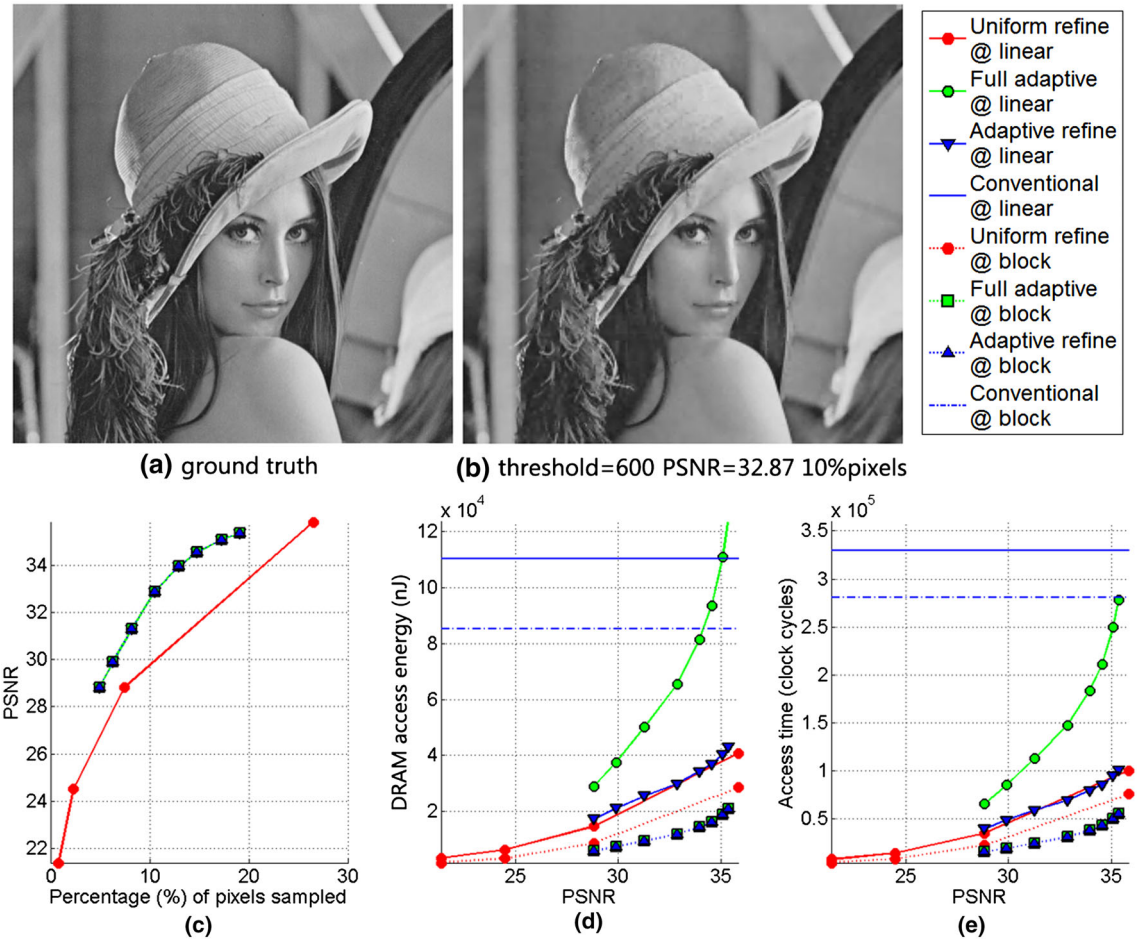

Fig. 8 **a**, **b** Comparison between ground truth image and the reconstruction using pixels sampled at a threshold of 600. **c** Reconstruction PSNR versus percentage of pixels sampled; **d** SDRAM access energy versus reconstruction PSNR; **e** SDRAM access time versus reconstruction PSNR; from *left to right*, the data points of HW-driven adaptive refine and full adaptive sampling algorithms in these graphs are results from threshold of 1800, 1300, 900, 600, 400, 300, 200, and 150, respectively; the data points of uniform refine algorithm are results from sampling rate of 16, 8, 4, and 2, respectively

of reconstructing for one pixel has to be smaller than the average time cost of retrieving that pixel directly from the memory;

2. To reduce the overall energy consumption of the image acquisition process, the average energy spent to reconstruct for an unsampled pixel has to be smaller than that of accessing one pixel from the memory.

This is discussed in the following subsection based on the evaluation data of the implemented system on actual hardware platforms.

5.2 Evaluation of the proposed architecture on reconfigurable platforms

A detailed evaluation of the system, implemented on reconfigurable hardware, is discussed in this section taking into account the overheads imposed by the proposed system. The implementation is on Stratix IV FPGA and Hardcopy IV structured-ASIC. The proposed system is compared against the conventional address generator in SDRAM access interface, for the difference in image acquisition time and overall energy consumption of the acquisition process. Block mapped image content is used in the following test as it is the most popular storing strategy used in image processing hardware systems.

In this set of evaluations, the proposed CbIA architecture is assumed to work at the same frequency as the memory data bus, i.e. it has the ability to random access a single 8-bit pixel from the memory without using the memory supported enhanced modes of pre-fetching/burst-reading consecutive pixels. However, for the baseline performance (the conventional image acquisition method) the target image patch is read into the local buffer in burst mode with burst length of 8. Therefore the evaluation provided here is a lower bound performance of the proposed CbIA architecture.

Table 1 Hardware resource usage of the proposed system, on Stratix IV

	ALUTs	Logic Registers	Block RAM usage		DSP block 18-bit Elements
			Bits	M9K	
Refine_unit	147	101	0	0	0
Interp_unit (×3)	919	474	0	0	36
Addr_translator	94	72	0	0	0
FIFOs	971	365	1751	9	0
Control	83	15	0	0	0
Total	2214	1027	1751	9	36
Total (%)	0.61 %		0.008 %	0.7 %	3.52 %

The percentage resource usage in the last line shows the percentage of total resource of the corresponding type used on the device

Table 2 Hardware resource usage of the proposed system, on Hardcopy IV

	Hcells	Block RAM Bits	DSP block 18-bit elements
Total	49,693 (0.55 %)	1751	36

A total of 0.55 % of the total HCell resource on device is used

Table 3 Reported maximum frequencies of the design

	Model ID	Fmax at 900 mV 85C	Fmax at 900 mV 0C
Stratix IV	EP4SGX530KH40C2	200 MHz	357 MHz
Hardcopy IV	HC4GX35FF1517	327 MHz	593 MHz

5.2.1 Hardware resource usage

The conventional address generator in SDRAM interface often acts as a simple counter with minimum implementation cost. Therefore the hardware resource required to implement it is omitted in this evaluation. Table 1 reports the added cost of hardware resources for implementing the prototype system on Stratix IV FPGA. The table shows that a significant proportion of the hardware resources, including all the block RAM bits and the majority of the ALUTS/registers, are used to maintain the intermediate data structure. The array of *interp_unit*s also uses a major proportion of ALUTs and registers as they are the most computational intensive parts of the system.

Table 2 reports the added cost of hardware resources for implementing the prototype system on Hardcopy IV structured-ASIC. Finally, the estimated maximum frequency of the prototype system is given in Table 3.

5.2.2 Acquisition time

The time spent on the sampling procedure (in clock cycles), as well as the total image acquisition time spent including the final interpolation stage are reported in Fig. 9. Again, this evaluation assumes that the prototype system works under a full random accessing situation, which is the worst case scenario for operating on the SDRAM. The reference lines in the plots show the image acquisition time required by conventional image access process, transformed to equivalent clock cycles. For reference, the raw number of memory accesses (i.e. number of accessed pixels) are also reported in Table 4, as well as the SDRAM access time in clock cycles corresponding to the reported number of accessed pixels without taking into account any extra clock cycles that are introduced by the control logic (this is captured by the "sampling process only" line in Fig. 9).

It can be seen that the achievable PSNR differs with test subject. The image "barbara" has more complex local structures than the other two images and therefore the PSNR of its reconstruction is relatively lower, even when a same priority score threshold is met. In general, due to the reduced number of sampled pixels, the proposed system has a much lower SDRAM occupation time (black lines) than that of the conventional access method (blue reference lines). This results in a much reduced bandwidth requirement of SDRAM and when needed, it frees the SDRAM early on to be accessed by other potential processing units in a large system. On the other hand, a significant amount of time is spent on interpolating the image. Nevertheless, the total *image acquisition time* is reduced in most cases in this test. In this particular test three *interp_unit*s are used, but more of this module can be added to accelerate the process at the expense of more hardware resources. This is because blocks that need interpolation are recorded in *FIFO_D* (Fig. 6) and the interpolation task can be completed by multiple *interp_unit*s in parallel.

The designed CbIA architecture demonstrates that the proposed framework is capable of reducing the total *image acquisition time* by employing on-chip computational resources to compensate for the selective and lossy

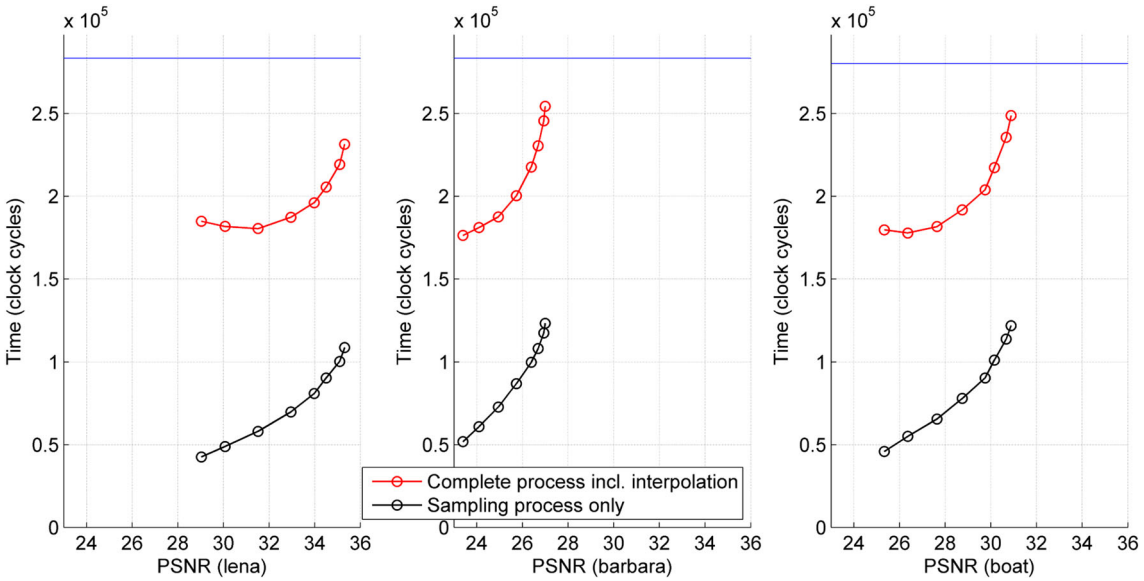

Fig. 9 Time requirement for sampling process, and complete acquisition process including interpolation. The *X* axis shows the achieved PSNR given different levels of thr. Data points from *left to right* represents thr of 1800, 1300, 900, 600, 400, 300, 200, and 150, respectively. *Reference lines* show the time requirement of conventional image accessing method in equivalent clock cycles

Table 4 The number of memory accesses and the SDRAM access time in clock cycles

Image	Threshold								
	0 (full image)	150	200	300	400	600	900	1300	1800
Lena									
# Access	277,729	56,240	50,797	43,826	38,799	31,911	25,246	19,719	15,942
Time (clock cycles)	283,495	62,008	56,560	49,593	44,559	37,674	31,005	25,478	21,715
Barbara									
# Access	277,729	65,350	61,461	55,351	51,019	42,965	34,300	26,162	19,719
Time (clock cycles)	283,495	71,124	67,217	61,113	56,793	48,744	40,078	31,916	25,480
Boat									
# Access	277,729	64,905	59,823	51,630	45,603	37,132	29,467	23,024	17,664
Time (clock cycles)	283,495	70,678	65,586	57,398	51,359	42,889	35,222	28,789	23,421

sampling process. The time cost overhead of computation, including that for controlling the sampling process and for interpolation, is mitigated by the advanced computation capability of the computing engine as well as by design methods such as parallelism in the interpolation process.

5.2.3 Energy consumption

By reducing the number of pixels accessed from the source memory, the proposed CbIA system has the potential to reduce the overall energy consumption of image acquisition process. However, as is expected in Sect. 5.1, the proposed CbIA procedure complicates the conventional address generating process and it employs extra computational effort for the reconstruction of the target image. By evaluating the energy consumption of the implemented CbIA architecture on reconfigurable hardware, this section aims to demonstrate the actual impact of the CbIA procedure on energy consumption.

To evaluate the energy consumption of the complete acquisition process, the core dynamic energy consumed by the proposed system is analysed by Quartus PowerPlay analyser, as the cost of executing the CbIA procedure on top of the conventional address generator in SDRAM access interface. The implemented architecture is run on both Stratix IV and Hardcopy IV platforms to demonstrate the energy consumption of the proposed image acquisition procedure, as well as the difference in energy consumption brought by different choices of housing platform. Again, in this test the conventional address generator in SDRAM

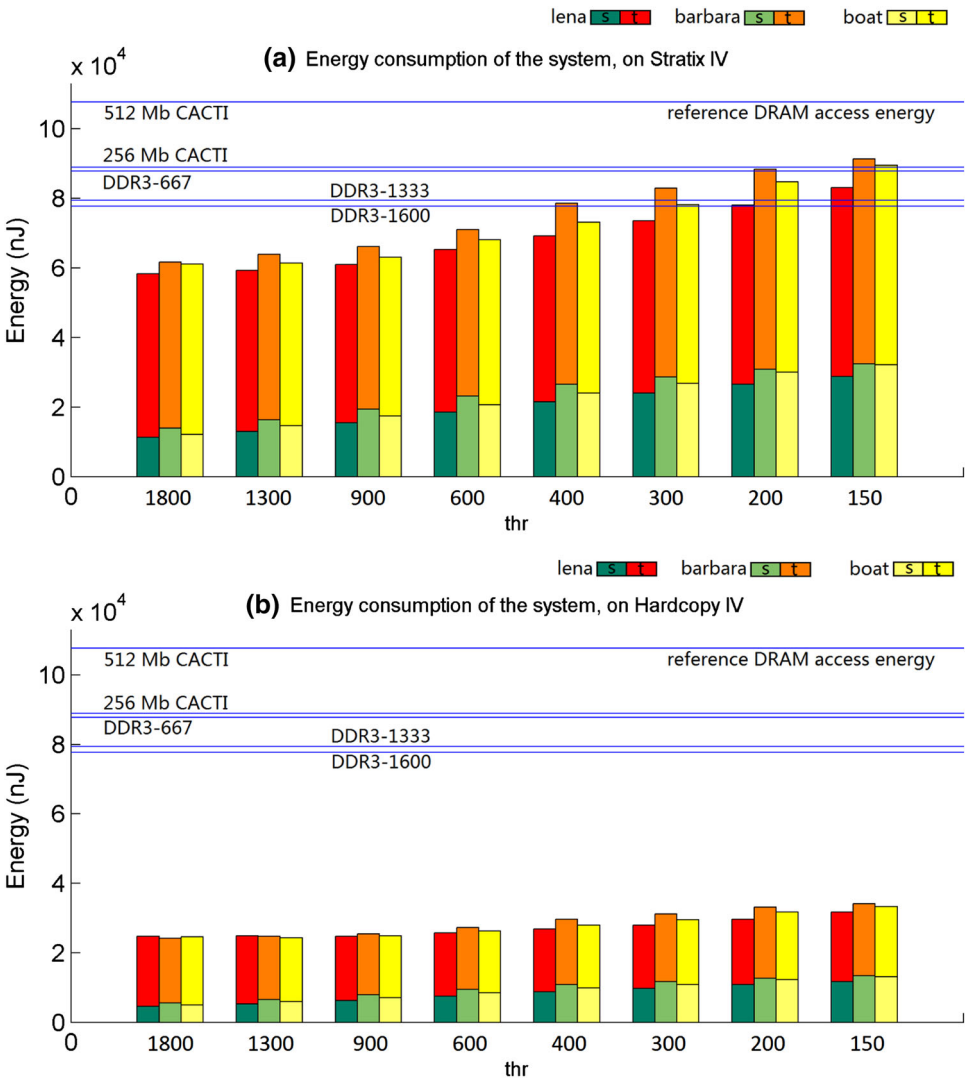

Fig. 10 Breakdown of energy consumption of the proposed system without considering the memory, for sampling process (marked by "s"), and complete process including interpolation (marked by "t"). The *X*-axis "threshold (thr)" shows the predefined threshold parameter for the design. The higher the threshold is, the lower quality the reconstruction will be and fewer samples are acquired from the memory. Reference lines are the energy consumption of accessing the whole target image from SDRAM by the conventional method

access interface is assumed to have a zero cost in the energy consumption department as well. Due to the simplicity of the conventional address generator, this assumption is made to make the evaluation more clear. Any energy consumed by the prototype system is considered to be overhead cost on top of the conventional address generator.

Figure 10 shows the breakdown of energy consumption of the sampling and interpolation process spent purely by the proposed system, i.e. without considering the memory, demonstrating the amount of energy overhead introduced by the proposed CbIA system. It can be seen that among the energy consumed by the proposed system, the reconstruction process (interpolation in this case) is also much more dominating than that of the sampling control mechanism. In the case of Stratix IV implementation (Fig. 10a), the energy consumption overhead of running the proposed CbIA system has a significant presence compared with that of the overall energy consumption of conventional image acquisition process. On the other hand, the energy overhead has a much less impact in the case of Hardcopy IV implementation (Fig. 10b), leading to a much higher potential for the system to reduce the overall energy consumption of the image acquisition process.

The overall energy consumption of the image acquisition process is reported in Fig. 11. For the overall energy consumption in this figure, the energy consumed by the complete system and its devices (including the SDRAM and the CbIA system) during all process (i.e. including sampling and reconstruction) is added together and reported. All measurements are normalized by the required SDRAM energy consumption of acquiring the target image by conventional accessing method. Therefore a ratio lower than 1 indicates that the proposed CbIA system is able to save energy for the image acquisition process.

With the presence of energy overhead introduced by the proposed system implemented on Stratix IV, a reduction of overall energy consumption can still be seen when the

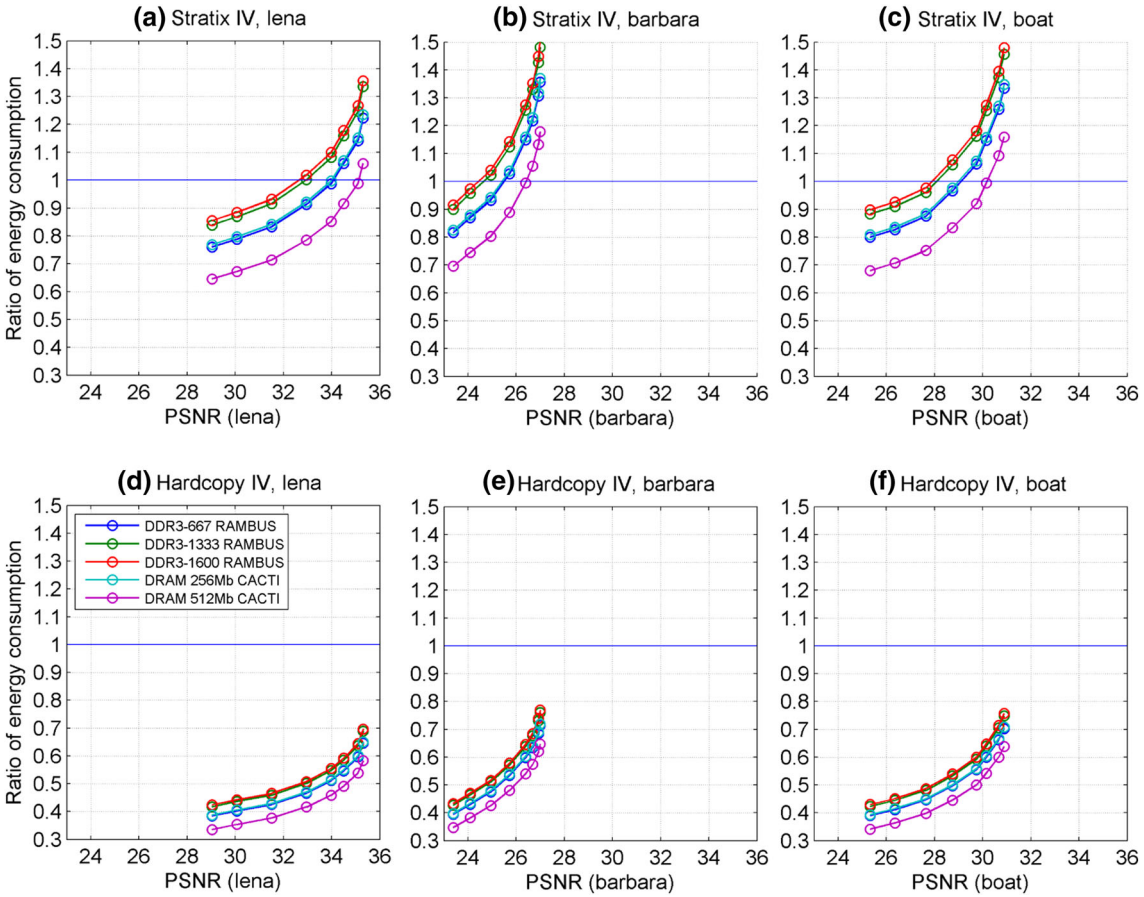

Fig. 11 The ratio of total energy consumption of the complete system (including corresponding energy spent on sampling from DRAM) to that of the memory access by conventional access method. Different DRAM models are used as target memory. Data points from *left to right* represents thr of 1800, 1300, 900, 600, 400, 300, 200, and 150, respectively

threshold is above about 600 if DDR3s are targeted. For general purpose SDRAMs simulated by CACTI, a reduction of energy consumption can be seen across most threshold levels. In the case of "lena", a reduction of up to 30 % can be seen while maintaining a PSNR above 30 dB. On the other hand, the test on Hardcopy IV shows a significant reduction of overall energy consumption across all threshold levels and different memory types. In the case of "lena", a energy saving of up to 75 % can be seen while maintaining a PSNR above 30 dB.

This set of evaluations demonstrates the potential of the proposed CbIA framework in efficiently trading image quality for reduced energy consumption of image acquisition process.

5.2.4 Comparison with existing methods

With the evaluation results given above, it is, however, challenging to directly evaluate the designed architecture against existing solutions. The proposed CbIA framework, as is represented by the example architecture explained in this paper, offers several unique qualities.

Firstly, it is universally applicable to any type of image data and achieve cost reduction regardless. With a controlled loss of image quality, the proposed architecture brings significant reduction of acquisition costs, comparable to some of the application specific optimization methods such as in the work of [2, 10]. Unlike methods that focus on the reduction of either bandwidth (e.g. [14, 15]) or energy consumption (e.g. [16]), the proposed architecture offers both. Secondly, the overhead of implementing the architecture is minimal. Not only is the hardware resource usage small (only a small fraction of the Stratix IV resource), it is also a stand-alone subsystem independent from both the source memory and the processing unit and hence requires minimal changes to be made on existing systems. Thirdly, it offers a truly dynamic and runtime trade-off between image quality and acquisition cost. The acquisition process adapts to the changing hardware environment and is able to stop at any time while still

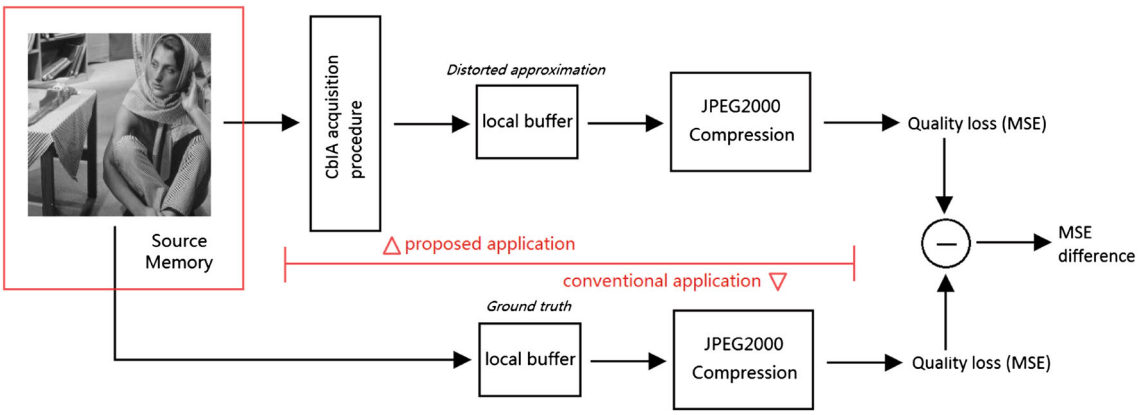

Fig. 12 Case study on JPEG2000: both the ground truth image and the reconstructed image from CbIA system are used for image compression process. This study aims to analyse the impact of reduced image quality brought by CbIA-enabled memory access interface. Source memory is Rambus model of DDR3-667

producing plausible results, which the existing methods (access pattern optimization, storage organization, image recompression) are not capable of.

These features combined, together with the acquisition cost reduction, are the values of the proposed system compared with existing methods. Nevertheless, the number of memory accesses and the SDRAM access time listed Table 4 provide some basic references for cross comparison with other methods.

5.3 Case study on JPEG2000

The proposed CbIA procedure is evaluated under a practical application scenario to assess its impact under a real-life problem. The selected application is the JPEG2000 image compression and it is chosen due to its wide usage. The compression unit accesses image macroblocks read either in the conventional access method, or by the proposed CbIA acquisition procedure. The image quality of the compression output using both image acquisition methods are compared with each other (Fig. 12) to demonstrate the impact of quality reduction resulted from Context-based Image Acquisition process.

Figure 13 shows the corresponding quality of compression outputs, using the two image acquisition methods. In this test, the image quality is shown in Mean Squared Error (MSE) instead of PSNR, to be able to represent the situation where compression ratio is 1, i.e. no compression is performed. In this situation the output image has 0 MSE but its PSNR is $+\infty$ which cannot be plotted on the graph. From this figure, it can be seen that due to the sampling nature of the CbIA procedure, the reconstruction error during the CbIA process is carried over to the client application which in this case is the image compression unit. This additional error increases the quality loss of the compression output, in addition to the compression distortion.

To further examine the impact on the compression quality, Fig. 14 shows the differences of MSE between compression output using ground truth image and that using the image acquired by the CbIA procedure. The black curve is the MSE difference when no compression is used and it is in fact the same as in Fig. 11 but presented in MSE. When the acquired image is processed by the compression unit, it can be seen that the image quality difference keeps decreasing as the compression rate increases. This shows that some loss of image quality due to progressive sampling is absorbed by the process of compression.

In general, if the client tends to remove image redundancy as the proposed system does then the impact of quality loss due to applying the proposed system is reduced, and therefore the system can achieve an even larger gain in bandwidth and image acquisition time/energy.

5.4 Targeting an ASIC implementation

The evaluation of the CbIA architecture on FPGA and structured-ASIC suggests its potential in saving both bandwidth and time/energy of the image acquisition process. It is desirable to be implemented as ASIC architecture, replacing conventional memory access interface, to provide an alternative image accessing methods when bandwidth or acquisition time/energy is of a major concern. According to [13], the ASIC implementation of a same design has an average of 4.6× decrease in path delay, and an average of 14× decrease in dynamic power consumption running the same test vector. A projection of the proposed FPGA implementation to ASIC by these factors sees the proposed design able to reduce both image

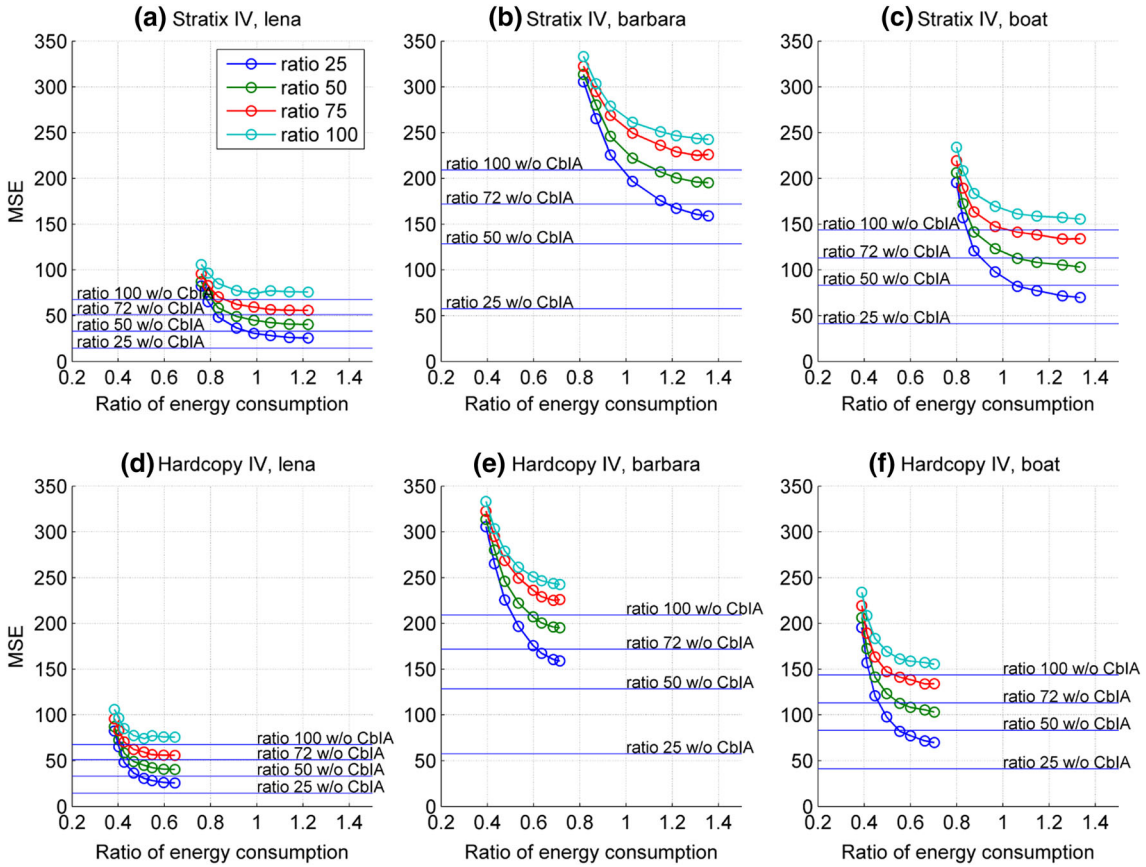

Fig. 13 The quality of compressed image measured in MSE, using both conventional accessing method and the proposed system. DDR3-667 is used as source memory. Data points from *left to right* represents thr of 1800, 1300, 900, 600, 400, 300, 200, and 150, respectively. Because of the additional quality loss introduced by the CbIA procedure, the error of the compression output using CbIA acquired images is higher than that of the conventional image acquisition method (*blue reference lines*)

acquisition time and energy. It will meet the clock frequency of DDR3-800 but for faster models it still requires more *interp_unit* to accelerate the interpolation process, to be capable of reducing total image acquisition time. On the other hand, the energy consumption of the architecture will be reduced greatly, making the proposed architecture promising in saving energy by large margin.

5.5 Performance under burst mode

Modern memory designs allow for pre-fetching (burst mode) capability. By receiving one address, the memory returns a consecutive sequence of data starting at the provided address. When the storage of the content is well optimized for the client application, this functionality can bridge the clock frequency difference between the control bus and data bus of the memory. It also reduces the addressing effort of the memory interface and therefore save energy.

The discussion provided in above sections assumes for a complete random access situation where the proposed CbIA memory interface has full ability to random access individual pixels from the source memory (no DDR feature). Even that the proposed context-based image acquisition does not directly benefit from the concept of pre-fetching, burst reading more pixels than addressed for is design-wise compatible to the proposed acquisition procedure and the CbIA architecture can take advantage of it. As shown in Fig. 15a, in ideal situation five individual samples are requested (marked in green) when the 9×9 block is called to be refined by CbIA procedure. In the situation where such random accessing of individual pixels is not viable and memory pre-fetch is effective, a consecutive sequence of pixels are accessed (Fig. 15b).

A full evaluation of the proposed system working under memory burst mode is done and results are shown in Figs. 16 and 17. In this evaluation, the proposed CbIA system is not able to access individual pixels but the memory returns a burst of pixels every time it receives a sampling address.

It can be seen from Fig. 16 that the longer the burst is, the less informative the sampling patterns are, i.e. a same

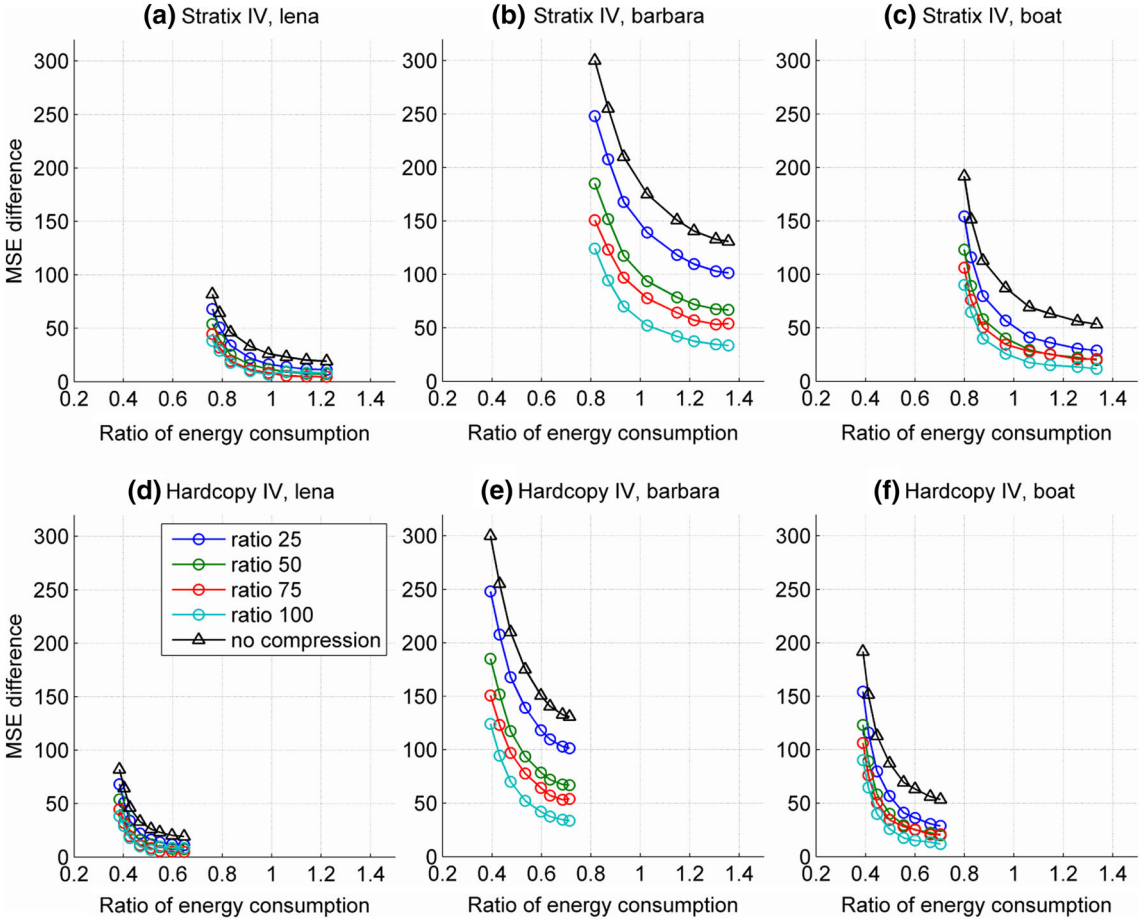

Fig. 14 The quality difference of compressed image, using both conventional accessing method and the proposed system. DDR3-667 is used as source memory. Data points from *left to right* represents thr of 1800, 1300, 900, 600, 400, 300, 200, and 150, respectively

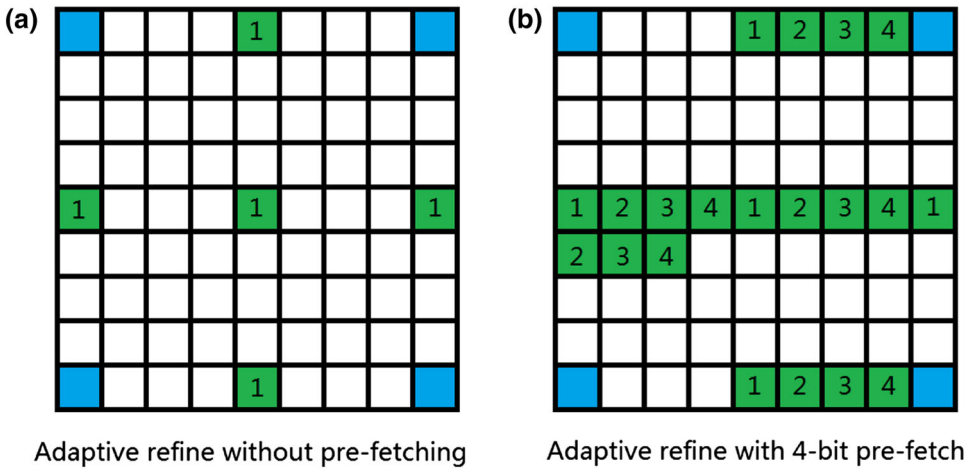

Fig. 15 Pre-fetch in the proposed sampling procedure, assuming a block mapping strategy same as in Fig. 4b

amount of samples leads to a lower reconstruction quality. However, when the overall energy consumption of the image acquisition process is considered it can be seen that the impact of these extra samples is two-fold (Fig. 17). At lower achievable PSNR levels, burst reading more pixels than addressed for is a compromise that increases overall energy consumption; at high achievable PSNR levels where the system refines gradually smaller regions, the energy spent on running the system outweighs the reconstruction quality improvement it brings, so much so that

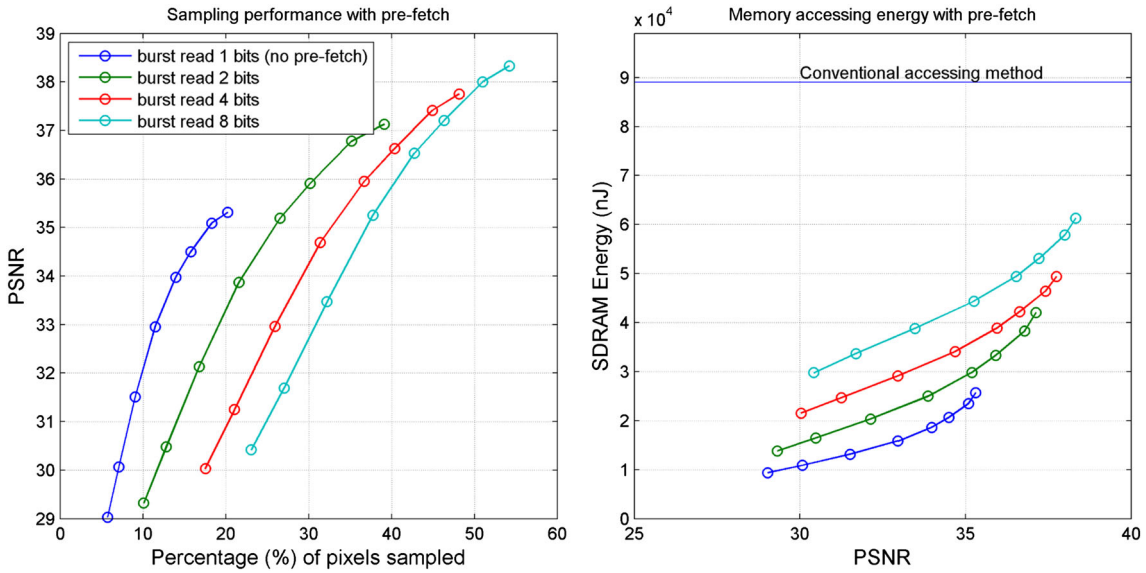

Fig. 16 Sampling process evaluation of the system under memory burst mode. **a** Achievable PSNR vs Number of samples; **b** SDRAM accessing energy versus achievable PSNR. Source memory is Rambus model DDR3-667

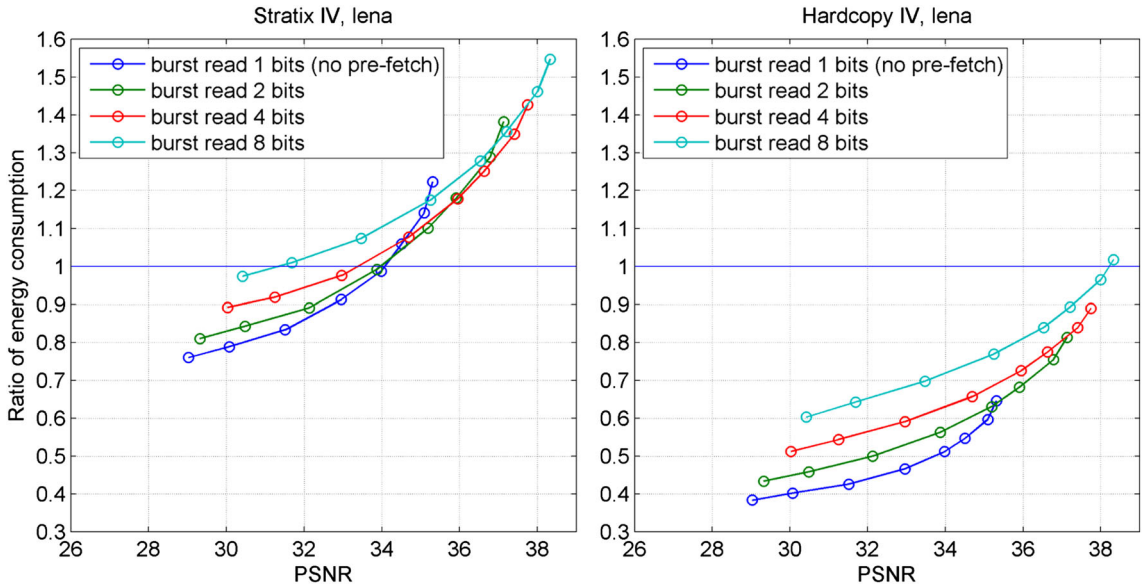

Fig. 17 Energy consumption evaluation of the system under memory burst mode. Source memory is Rambus model DDR3-667

directly burst reading the rest of pixels within the small region in question actually costs less in energy consumption.

In general, the proposed CbIA system is compatible with memory burst mode design-wise. To a larger extent, the proposed framework of context-based image acquisition is compatible with any memory mechanism that allows for a certain level of random access ability. The samples does not have to be individual pixels but instead small regions on the target image. This also means that the proposed framework is compatible with existing techniques such as image re-compression in [14, 15], where compression happens within local sequences of pixels.

6 Conclusion

In this work, the framework of Context-based Image Acquisition is proposed to reduce the cost of image acquisition process in hardware systems. An example design of the procedure and architecture is given and evaluated, demonstrating the ability of the proposed

concept to efficiently trade image quality for a reduced overall cost of the acquisition process. The proposed CbIA framework is based on the essential scenario of image acquisition in hardware systems, and deals with the specific task of moving image macroblocks from source memory to a local buffer. Therefore it is compatible with most existing hardware platforms, as well as many existing methods of reducing memory accessing cost such as code re-writing. There are other ways to design and implement a CbIA architecture besides the proposed architecture in this work. More complex modelling of images [18] can be used and even domain specific learning [17] can be applied to improve the trade between image quality and the cost of memory access. It is the hope of this paper that the work of CbIA will inspire the design and development of future image processing hardware systems.

References

1. Banakar, R., Steinke, S., Lee, B.S., Balakrishnan, M., Marwedel, P.: Scratchpad memory: design alternative for cache on-chip memory in embedded systems. In: Proceedings of the Tenth International Symposium on Hardware/Software Codesign, ACM, pp. 73–78 (2002)
2. Berić, A., Van Meerbergen, J., De Haan, G., Sethuraman, R.: Memory-centric video processing. Circuits Syst. Video Technol. IEEE Trans. on **18**(4), 439–452 (2008)
3. Catthoor, F., Danckaert, K., Kulkarni, C., Omnes, T.: Data Transfer and Storage Architecture Issues and Exploration in Multimedia Processors. Marcel Dekker Inc., New York (2000)
4. Catthoor, F., Danckaert, K., Wuytack, S., Dutt, N.D.: Code transformations for data transfer and storage exploration preprocessing in multimedia processors. IEEE Des. Test Comput. **18**(3), 70–82 (2001)
5. Danckaert, K., Catthoor, F., De Man, H.: A preprocessing step for global loop transformations for data transfer optimization. In: Proceedings of the 2000 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, ACM, pp. 34–40 (2000)
6. Eldar, Y., Lindenbaum, M., Porat, M., Zeevi, Y.: The farthest point strategy for progressive image sampling. Image Process. IEEE Trans. on **6**(9), 1305–1315 (1997)
7. Fischaber, S., Woods, R., McAllister, J.: SOC memory hierarchy derivation from dataflow graphs. J. Signal Process. Syst. **60**(3), 345–361 (2010)
8. Hennessy, J.L., Patterson, D.A.: Computer Architecture: A Quantitative Approach. Elsevier, London (2012)
9. Hoemmen, M.: Communication-Avoiding Krylov Subspace Methods. Doctoral Dissertation, University of California, Berkeley (2010)
10. Kim, H., Park, I.C.: High-performance and low-power memory-interface architecture for video processing applications. Circuits Syst. Video Technol. IEEE Trans. on **11**(11), 1160–1170 (2001)
11. Kim, J.Y., Kim, D., Lee, S., Kim, K., Yoo, H.J.: Visual image processing ram: memory architecture with 2-D data location search and data consistency management for a multicore object recognition processor. Circuits Syst. Video Technol. IEEE Trans. on **20**(4), 485–495 (2010)
12. Kumar Gupta, A., Nooshabadi, S., Taubman, D.: Optimal 2 subbank memory architecture for bit plane coder of jpeg2000. In: Circuits and Systems: ISCAS 2005, pp. 4373–4376. IEEE International Symposium on, IEEE (2005)
13. Kuon, I., Rose, J.: Measuring the gap between fpgas and asics. Comput.-Aided Des. Integr. Circuits Syst. IEEE Trans. on **26**(2), 203–215 (2007)
14. Lee, T.Y.: A new frame-recompression algorithm and its hardware design for mpeg-2 video decoders. Circuits Syst. Video Technol. IEEE Trans. on **13**(6), 529–534 (2003)
15. Lee, Y., Rhee, C.E., Lee, H.J.: A new frame recompression algorithm integrated with h. 264 video compression. In: Circuits and Systems, : ISCAS 2007, pp. 1621–1624. IEEE International Symposium on, IEEE (2007)
16. Li, Y., Zhang, T.: Reducing dram image data access energy consumption in video processing. Multimed. IEEE Trans. on **14**(2), 303–313 (2012)
17. Liu, J., Bouganis, C., Cheung, P.Y.: Domain-specific progressive sampling of face images. In: Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE, IEEE (2013)
18. Liu, J., Bouganis, C., Cheung, P.Y.: Kernel-based image adaptive sampling. In: Proceedings of the International Conference on Computer Vision Theory and Applications (2014)
19. Patterson, C.A., Snir, M., Graham, S.L., et al.: Getting Up to Speed: The Future of Supercomputing. National Academies Press, Washington, DC (2005)
20. Rafique, A., Kapre, N., Constantinides, G.A.: Application composition and communication optimization in iterative solvers using fpgas. In: Field-Programmable Custom Computing Machines (FCCM), 2013 IEEE 21st Annual International Symposium on, IEEE, pp. 153–160 (2013)
21. Thoziyoor, S., Muralimanohar, N., Ahn, J.H., Jouppi, N.P.: Cacti: an integrated cache and memory access time, cycle time, area, leakage, and dynamic power model. Technical Report HPL-2008-20, HP Laboratories (2008)
22. Tzou, K.H.: Progressive image transmission: a review and comparison of techniques. Opt. Eng. **26**(7):267, 581–267, 581 (1986)
23. Vogelsang, T.: Understanding the energy consumption of dynamic random access memories. In: Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture, IEEE Computer Society, pp. 363–374 (2010)
24. Yng, T.L.B., Lee, B.G., Yoo, H.: A low complexity and lossless frame memory compression for display devices. Consum. Electr. IEEE Trans. on **54**(3), 1453–1458 (2008)
25. Zhou, D., Zhou, J., He, X., Zhu, J., Kong, J., Liu, P., Goto, S.: A 530 mpixels/s 4096x2160@ 60fps h. 264/avc high profile video decoder chip. Sol.-State Circuits, IEEE J. **46**(4):777–788 (2011)

Jianxiong Liu received his Ph.D. degree from Imperial College London in 2015, and is working as advanced Design Engineer for Altera in the UK. Jianxiong's research interests are focused on image processing based algorithms and hardware system design, as well as high-performance digital system design. He is also interested in the design and development of machine learning algorithms.