Bernhard Rinner *

Klagenfurt University, Institute of Networked and Embedded Computing
9020 Klagenfurt, Austria

Wayne Wolf

Georgia Institute of Technology, School of Electrical and Computer Engineering
Atlanta, GA USA

Towards Pervasive Smart Camera Networks

Abstract

Smart camera networks are real-time distributed embedded systems that perform computer vision using multiple cameras. This new approach has emerged thanks to a confluence of simultaneous advances in four key disciplines: computer vision, image sensors, embedded computing, and sensor networks.

In this chapter, we briefly review and classify smart camera platforms and networks into single smart cameras, distributed smart camera systems and wireless smart camera networks. We elaborate the vision of pervasive smart camera networks and identify major research challenges. As the technology for smart camera networks advances, we expect to see many new applications open up—transforming traditional multi-camera systems into pervasive smart camera networks.

Key words: distributed smart cameras, multi-camera networks, sensor networks, pervasive computing

1 Introduction and Motivation

Smart cameras have been the subject of study in research and industry for quite some time. While some camera prototypes which integrated sensing with some low-level processing were developed in the 1980s, first commercial "intelligent" cameras appeared in the 1990s. However, the sensing and processing

* Corresponding author.
Email addresses: `Bernhard.Rinner@uni-klu.ac.at` (Bernhard Rinner), `wolf@ece.gatech.at` (Wayne Wolf).

capabilities were very limited on these cameras. In the meantime we have seen a dramatic progress in smart camera research and development (e.g., [1,2,3]).

A number of technical factors are converging to cause us to totally rethink the nature of the camera. Distributed smart cameras embody some (but not all) of these trends, specifically: cameras are no longer boxes and cameras no longer take pictures. A smart camera's fundamental purpose is to analyze a scene and report items and activities of interest to the user. Although the camera may also capture an image to help the user interpret the data, the fundamental output of smart cameras is not an image. When we combine several smart cameras together to cover larger spaces and solve occlusion problems, we create a distributed camera. When we furthermore use distributed algorithms to perform smart camera operations, we create a distributed smart camera.

Law enforcement and security are the most obvious applications of distributed smart cameras. Large areas can be covered only by large numbers of cameras; analysis generally requires fusing information from several cameras. However, distributed smart cameras have many other uses as well, including machine vision, medicine and entertainment. All these applications require imagery from multiple cameras to be fused in order to interpret the scene. Because of the complex geometric relationships between subjects of interest, different sets of cameras may need to cooperate to analyze different subjects. Because of subject motion, the sets of cameras that must cooperate may change rapidly. Pulling all of the video from a large number of cameras to a central server is expensive and inherently unscalable. The combination of large numbers of nodes, fast response times, and constantly changing relationships between the cameras pushes us away from server-based architectures. Distributed computing algorithms provide a realistic approach to the creation of large distributed camera systems.

Distributed computing introduces several complications. However, we believe that the problems they solve are much more important than the challenges of designing and building a distributed video system. As in many other applications, distributed systems scale much more effectively than do centralized architectures.

- Processing all the data centrally poses several problems. Video cameras generate large quantities of data requiring high-performance networks for transmitting the video data in steady state.
- Moving video over the network also consumes large amounts of energy. In many systems, communication is 100 to 1000 times more expensive in energy than computation. We do not expect camera systems to be run from batteries for long intervals, but power consumption is a prime determinant of heat dissipation. Distributing larger amounts of power also requires more substantial power distribution networks, which increases the installation

cost of the system.

- Although data must be compared across several cameras to analyze video, not all pairs of cameras must communicate with each other. If we can manage the data transfer between processing nodes, we can make sure that data only goes to the necessary nodes. A partitioned network can protect physically distributed cameras so that the available bandwidth is used efficiently.
- Real-time and availability considerations also argue in favor of distributed computing. The round-trip delay to a server and back adds to the latency of making a decision, such as whether a given activity is of interest. Having available multiple points of computation enables reconfigurations in case of failure which increases the availability of the multi-camera system.

In the progress of smart cameras we can identify three major evolution paths. First, *single smart cameras* focus on the integration of sensing with embedded on-camera processing. The main goal here is to be able to perform various vision tasks on-board and deliver abstracted data from the observed scene. Second, *distributed smart cameras (DSC)* introduce distribution and collaboration to smart cameras resulting in a network of cameras with distributed sensing and processing. Thus, distributed smart cameras collaboratively solve tasks such as multi-camera surveillance and tracking by exchanging abstracted features. Finally, *pervasive smart cameras (PSC)* integrate adaptivity and autonomy to DSC. The ultimate vision of PSC is to provide a service-oriented network which is easy to deploy and operate, adapts to changes in the environment and provides various customized services to users.

The goal of this chapter is twofold. First, we briefly review and classify smart camera platforms and networks. Second, we elaborate the vision of pervasive smart camera networks and identify major research challenges towards this vision. The discussion of the research challenges is based on an exploration of trends in current smart camera systems.

The remainder of this chapter is organized as follows: Section 2 starts with a brief overview of architectural issues of smart cameras and focuses then on reviewing the evolution of smart camera systems. In Section 3 we identify current trends and speculate about future developments and applications. Section 4 concludes this chapter with a brief discussion.

2 The Evolution of Smart Camera Systems

Smart cameras are enabled by advances in VLSI technology and embedded system architecture. Modern embedded processors provide huge amounts of performance. However, smart cameras are not simply cost-reduced versions of arbitrarily-selected computer vision systems. Embedded computer vision requires distinct techniques from non-real-time computer vision because of the particular stresses that vision algorithms put on computer systems. Memory is a principal bottleneck of computer system performance because memory speed does not increase with Moores Law [4]. However, computer vision algorithms, much like video compression algorithms, use huge amounts of data and often with less frequent reuse. As a result, caches typically found in general-purpose computing systems may be less effective for vision applications. At the minimum, software must be carefully optimized to make best use of the cache; at worst, the memory system must be completely redesigned to provide adequate memory bandwidth [5].

Beside memory capacity and memory bandwidth computing power is a crucial resource for embedded computer vision. The individual stages of the typical image processing pipeline raise different requirements on the processing elements. Low-level image processing such as color transformations and filtering operates on individual pixels in regular patterns. These low-level operations process the complete image data at the sensor's frame rate, but typically offer a high data parallelism. Thus, low-level image processing is often realized on dedicated hardware such as ASICs, FPGAs or specialized processors [6]. High-level image processing on the other hand operates on (few) features or objects which reduces the required data bandwidth but increases the complexity of the operations significantly. These complex processing tasks exhibit typically a data-dependent and irregular control flow. Thus, programmable processors are the prime choice for these tasks. Depending on the complexity of the image processing algorithms even multi-core or multi-processor platforms may be deployed [7,8].

2.1 Single Smart Cameras

The integration of image sensing and processing on single embedded platforms has been conducted for quite some time. However, research on single smart cameras has intensified over the last decade. Table 1 presents an overview of selected single smart camera platforms.

Moorhead and Binnie [9] presented one of the first fabricated CMOS implementations. Their SoC smart camera integrated edge detection into the image sensor. VISoc [10] represents another smart camera-on-a-chip implementation featuring a 320 x 256 pixel CMOS sensor, a 32-bit RISC processor and a vision/neural coprocessor. Kleihorst et al. [16] engaged in the development of a specialized processor for image processing with high performance and low power consumption. This processor features 320 processing elements allowing to process a single line of an image in CIF resolution in one cycle, or an image in VGA resolution in two cycles respectively.

	Platform Capabilities				Application
System	Sensor	CPU	Comm.	Power	
Moorhead and Binnie [9]	CMOS	custom logic for on-chip edge detection	n/a	mains	low-level edge detection
VISoc (Albani)[10]	CMOS, 320x256	32-bit RICS and vision/neural processor	n/a	battery	low-level edge detection
Wolf [11]	Hi8 Camcorder, NTSC	PC with TriMedia TM-1300 boards	n/a	mains	gesture recognition
Single SmartCam (Bramberger, Rinner) [12]	color, VGA	DSP	n/a	mains	adaptive background substraction
TRICAM [13]	video in (no sensor)	DSP and FPGA, 128MB RAM	Ethernet	mains	Viola-Jones object detection
Bauer [14]	neuromorphic sensor (64×64)	Blackfin DSP	n/a	mains	vehicle detection and speed estimation
Dias and Berry [15]	2048×2048, gyroscope and accelerometer	Altera Stratix FPGA	Firewire (1394)	mains	template-based object tracking

Table 1
Classification of single smart camera systems.

Wolf et al. [11] developed a first generation smart camera prototype for real-time gesture recognition. For the implementation they equipped a standard PC with additional PCI-boards featuring a TriMedia TM-1300 VLIW processor. A Hi8 video camera is connected to each PCI-board for image acquisition.

A completely embedded version of a smart camera was introduced by Bramberger et al. [12]. Their first prototype was based on a single DSP COTS-system (TMS320C64xx processor from Texas Instruments) equipped with 1 MB on-chip memory and 256 MB external memory. A CMOS image sensor is directly connected to the DSP via the memory interface. Communication and configuration are realized over a wired Ethernet connection.

Arth et al. [13] presented the TRICam—a smart camera prototype based on a single DSP from Texas Instruments. Analog video input (either PAL or NTSC) is captured by dedicated hardware, and a FPGA is used for buffering the scanlines between video input and DSP. The TRICam is equipped 1 MB on-chip and 16 MB external memory.

Bauer et al. [14] presented a DSP-based smart camera realizing a neuromorphic vision sensor. This smart sensor delivers only information about intensity changes with precise timing information which is then processed to identify moving objects and estimate their speed.

Dias et al. [15] described a generic FPGA-based smart camera. The FPGA is used to implement several standard modules (e.g., interface to the image sensor, memory interface, Firewire interface) along with a programmable control module and a flexible number of processing elements. The processing elements can be interconnected arbitrarily according to the algorithm's data-flow.

2.2 Distributed Smart Cameras

Distributed smart cameras not only distribute sensing but also processing. However, the degree of distribution may vary substantially. On the one hand, smart cameras can serve as processing nodes that perform some fixed preprocessing but still delivering data to a central server. On the other hand, processing may be organized in a completely decentralized fashion where the smart cameras organize themselves and collaborate in a dynamic manner.

Implementing and deploying distributed smart cameras with decentralized coordination pose several new research challenges. Multiple threads of processing may take place on different processing nodes in parallel. This requires a distribution of data and control in the smart camera network. The required control mechanisms are implemented by means of dedicated protocols.

In [17,18] we have already discussed that a substantial system-level software or middleware would greatly enhance application development. Such a middleware has to integrate the camera's image processing capabilities and provide a transparent inter-camera networking mechanism. In [19] we propose to use agents as top-level abstraction for the distribution of control and data. A distributed application comprises several mobile agents, whereas agents represent image processing tasks within the system. Combining agents with a mobility property allows to move the image processing tasks between cameras as needed. To demonstrate the feasibility of this agent-oriented approach, we have implemented an autonomous and fully decentralized multi-camera tracking method [20].

Patricio et al. [21] also use the agent-oriented paradigm. But in their approach, the agent manages a single camera, and an internal state representing beliefs, desires and intentions. Collaboration of cameras hence corresponds to collaboration of agents, i.e., an agent can inform its neighbor about an object expected to appear or ask other agents whether they currently track the same object.

Fleck et al. [22] demonstrate a multi-camera tracking implementation where camera coordination and object hand-off between cameras is organized centrally. Each camera uses a particle-filter based tracking algorithm to track the individual objects within a single camera's field of view. The camera nodes report the tracking results along with the object description to the central server node.

Norouznezhad et al. [23] present an FPGA-based smart camera platform which is developed for large multi-camera surveillance applications. The most distinctive features compared to the other platforms is the large CMOS image sensor (2592 $\times$ 1944) and the GigE vision interface. The processing unit is partitioned into pixel-based processing and ROI processing which can be executed in parallel.

2.3 Smart Cameras in Sensor Networks

Wireless sensor networks are receiving a lot of attention in the scientific community [25]. While many networks are focused on processing scalar sensor values such as temperature or light measurements, there are some networks focusing on visual sensors. Since a core feature of sensor networks is that they are designed to run on battery power, one of the main challenges is to find a reasonable tradeoff between computing power, memory resources, communication capabilities, system size and power consumption.

The Meerkats sensor nodes [31] use an Intel Stargate mote equipped with a 400 MHz StrongARM processor, 64 MB SDRAM and 32 MB Flash. Wireless communication is realized with a 802.11b standard PCMCIA card. A consumer USB webcam serves as imager (640$\times$480 pixels). The sensor nodes are operated by an embedded Linux system. A focus of this work is to evaluate the power consumption of different tasks such as flash memory access, image acquisition, wireless communication and data processing. In [30], further details on deploying Meerkats in a multi-node setup are given. For detection of moving objects, image data is analyzed locally on the cameras. Nodes collaborate for handover using a master-slave mechanism. Compressed image data is

	Platform Capabilities				Application
System	Sensor	CPU	Comm.	Power	
Distributed SmartCam (Bramberger, Quaritsch, Rinner) [8]	VGA	ARM and multiple DSPs	100Mbps Ethernet, GPRS	mains	local image analysis; cooperative tracking
BlueLYNX (Fleck) [22]	VGA	PowerPC, 64MB RAM	Fast Ethernet	mains	local image preprocessing; central reasoning
GestureCam (Shi) [24]	CMOS, 320x240 (max. 1280x1024)	Xilinx Virtex-II FPGA; custom logic plus PowerPC core	Fast Ethernet	mains	local image analysis; no collaboration
NICTA smart camera (Norouznezhad) [23]	CMOS 2592x1944	Xilinx XC3S5000 FPGA; microBlaze core	GigE vision interface	mains	local image analysis; no collaboration

Table 2
Classification of distributed smart camera systems.

transmitted to a central sink. Feng et al. [35] presented the Panoptes—a very similar system which is also based on Stargate motes and USB webcams.

Another representative of a smart camera for sensor networks is the Cyclops camera by Rahimi et al. [28]. This node is equipped with a low-performance ATmega128 8-bit RISC microcontroller operating at 7.3 MHz, 4 kB on-chip SRAM and 60 kB of external RAM. The CMOS sensor can deliver 24 bit RGB images at CIF resolution (352×288). The Cyclops platform does not provide on-board networking facilities but it can be attached to a MicaZ mote. Medeiros and Park [29] use a network of Cyclops cameras to implement a protocol supporting dynamic clustering and cluster head election. They demonstrate their system in the context of an object tracking application.

The MeshEye sensor node by Hengstler et al. [32] combines multiple vision sensors on a single node. The platform is equipped with two low resolution image sensors and one VGA color image sensor. One of the low resolution sensors is used to constantly monitor the field of view of the camera. Once an object has been detected, the second low resolution sensor is activated and

	Platform Capabilities				Application
System	Sensor	CPU	Comm.	Power	
CMUcam 3 (Rowe) [26]	color CMOS, 352×288	ARM7 at 60 MHz	none onboard (802.15.4 via FireFly mote)	battery	local image analysis; inter-node collaboration [27]
Cyclops (Rahimi) [28]	color CMOS, 352×288	ATmega128 at 7.3 MHz	none onboard (802.15.4 via MicaZ mote)	battery	collaborative object tracking [29]
Meerkats (Margi) [30]	webcam, 640×480	StrongARM at 400 MHz	802.11b	battery	local image analysis; collab. object tracking; image transmission to central sink [31]
MeshEye (Hengstler) [32]	2× low resolution sensor, 1× VGA color CMOS sensor	ARM7 at 55 MHz	802.15.4	battery	unknown
WiCa (Kleihorst) [7]	2× color CMOS sensor, 640×480	Xetal 3D (SIMD)	802.15.4	battery	local processing; collab. reasoning [33]
CITRIC (Chen) [34]	OV9655 color CMOS sensor, 1280×1024	XScale PXA270	802.15.4	battery (428 - 970mW)	compression, tracking, localization

Table 3
Classification of wireless smart camera systems.

the location of the detected object is estimated using simple stereo vision. The estimated object's region of interest is then captured by the high resolution sensor. The main advantage of this approach is that the power consumption can be kept at a minimum as long as there are no objects in the field of view of the system. The processing is done on an ARM7 microcontroller running at 55 MHz. The MeshEye is equipped with 64 kB of RAM and 256 kB flash memory. An 802.15.4 chip provides wireless networking capabilities.

The WiCa wireless camera by Kleihorst et al. [7] is equipped with the SIMD processor IC3D operating at 80 MHz. This processor features 320 RISC processing units operating concurrently on the image data stored in line memory. In addition to the line memory, the platform also provides access to external DPRAM. For general purpose computations and communication tasks, the WiCa is equipped with an 8051 microcontroller. The WiCa can be extended with an 802.15.4 based networking interface used for inter-node communication. The WiCa platform has been designed with respect to low-power applications and hence could be operated on batteries. Distributed processing between four WiCas has been demonstrated in a gesture recognition system [33].

The CMUcam 3 is the latest version of an embedded computer vision platform developed by Rowe et al. [26]. It consists of a color CMOS sensor capable of delivering 50 frames per second at a resolution of 352×288 pixels. Image data is stored in a FIFO and processed by an ARM7 microcontroller operating at 60 MHz. The CMUcam 3 is equipped with 64 kB RAM and 128 kB flash memory. It comes with a software layer implementing various vision algorithms such as color tracking, frame differencing, convolution or image compression. Networking capabilities can be achieved by attaching an external mote via a serial communication channel, e.g., by combining it with FireFly motes [27]. This FireFly Mosaic relies on tight time synchronization for multi-camera cooperation. The nodes are statically deployed in the context of home activity monitoring.

The CITRIC mote [34] is a wireless camera hardware platform with a SXGA OmniVision CMOS sensor, an XScale processor, 64 MB RAM and 16 MB flash memory. Wireless communication using IEEE 802.15.4 is achieved by connecting the CITRIC board to a Tmote Sky board. The CITRIC platform is similar to the prototype platform used by [36], which consists of an iMote2 connected to a custom-built camera sensor board. The CITRIC platform has been demonstrated in image compression, single traget tracking via background subtraction and camera localization using multi-target tracking [34].

3 Future and Challenges

Distributed smart camera networks can be used in a variety of applications. The different constraints imposed by these applications and their associated platforms require somewhat different algorithms:

- Best results are obtained for tracking when several cameras share overlapping fields of view. In this case, the cameras can compare the tracks they generate to improve the accuracy of the overall track generated by the network.
- When covering large areas, we may not be able to afford to install enough cameras to provide overlapping fields of view. Tracking must then estimate the likelihood that a person seen in one view is the same person seen by a different camera at a later time.
- Not all cameras may be alike. We may, for example, use low-power, low-resolution cameras to monitor a scene and wake up more capable cameras when activity warrants. We may also use cameras in different spectral bands, such as infrared.
- Some cameras may move, which causes challenges for both calibration and background elimination. A moving camera may be part of a cell phone that captures opportunistic images; the camera may also be mounted on a vehicle.

3.1 Distributed Algorithms

Distributed algorithms have a number of advantages and are a practical necessity in many applications. Centralized algorithms create bottlenecks that limit the scalability of systems. Distributed algorithms can also, when properly designed, provide some degree of fault tolerance.

Two styles of distributed algorithms have been used in distributed smart cameras: consensus algorithms compare information between nodes to improve estimates; coordination algorithms hand off control between nodes. Consensus and coordination algorithms use different styles of programming and have distinct advantages.

Consensus algorithms are typically thought of as message-passing systems. An example of a coordination algorithm for distributed smart cameras is the calibration algorithm of Radke et al. [37]. This algorithm determines the external calibration parameters (camera position) of a set of cameras with overlapping fields of view by finding correspondences between features extracted from the scenes viewed by each of the cameras. It is formulated as a message-passing system in which each message includes a node's estimate of the position.

Consensus algorithms are well-suited to estimation problems, such as the determination of position. Many algorithms showed that matrix algorithms can be formulated as message-passing systems. An important characteristic of consensus algorithms is loose termination criteria. Distributed systems may not provide reliable transmission of messages. As a result, termination should not rely on strict coordination of messages into iterations.

Coordination algorithms can be viewed as token-passing systems. A token represents the locus of control for processing. They are well-suited to problems like tracking, in which the identity of a subject must be maintained over an extended period. These algorithms can be thought of as protocols—each node maintains its own internal state and exchanges signals with other nodes to affect both its own state and the state of other nodes.

Coordination algorithms date back to the early days of distributed smart cameras. The VSAM system [38] handed off tracking from camera to camera. More recently, the gesture recognition system of Lin et al. [39] uses a token to represent the identity of the subject whose gestures are being recognized. Some low-level feature extraction is always performed locally but the final phases of gesture recognition may move from node to node as the subject moves and features from several cameras need to be fused. A protocol manages the transfer of the token between nodes; the protocol must ensure that tokens are neither duplicated nor lost. The tracking system of Velipasalar et al. [40] also uses a protocol to trade information about targets. Each node runs its own tracker for each target in its field of view. A protocol periodically exchanges information between nodes about the position of each target. This system is considered a coordination algorithm rather than a consensus algorithm because only one round of information exchange is performed at each period.

Of course, tracking includes both maintenance of identity (coordination) and position of estimation (consensus). More work needs to be done to combine these two approaches into a unified algorithmic framework.

3.2 Dynamic and Heterogeneous Network Architectures

Dynamic and heterogeneous camera networks provide some advantages over static architectures. They can be better adapted to the requirements of the applications and—even more important—are able to react to changes in the environment during operation.

Such a heterogeneous architecture can be comprised of cameras with different capabilities concerning sensing, processing and communication. We can choose the mode the camera operates and hence determine the configuration of the overall camera network. By that we can set the network into a configuration which best fits the current requirements. There are many optimization criteria possible—energy, response time, communication bandwidth are just a few examples. The optimization goal we want to achieve clearly depends on the application.

Dynamic and heterogeneous architectures are not special to camera networks. These principles are well known for example in sensor networks or communication networks. One such example are multi-radio networks which combine low and high performance radios to adapt bandwidth, energy consumption and connectivity over time. The system of Stathopoulos et al. [41] uses dual radio platforms to implement a protocol to selectively enable high-bandwidth nodes to form end-to-end communication paths. For their work they use a low-bandwidth network that is always on for control and management purposes as well as for transmission of low-bandwidth data. High-bandwidth radios are only enabled when fast response times or large data volumes have to be transferred. Lymberopoulos et al. [42] evaluate the energy efficiency of multi-radio platforms. They compare a 802.15.4 radio providing a data rate of 250 kbps with an 802.11b radio providing a data rate of up to 11 Mbps. They also consider the different startup times of the two radios in their energy evaluation.

A moving camera not only poses challenges for calibration and background elimination. A freely moving camera connected by some wireless links can also change the topology of the overall network. Communication links to some nodes may drop; new links to other nodes may need to be established. This is closely related to mobile ad-hoc networks (MANET) [43] which mainly deals with the self-configuration of the dynamic network.

3.3 Privacy and Security

When deploying camera networks in end-user environments such as private homes or public places, the awareness of privacy, confidentiality and general security issues is rising [44]. By being able to perform onboard image analysis and hence to avoid transferring raw data, smart cameras have great potential for increasing privacy and security. Boult et al. [45] and Fleck et al. [46], among others, have explored smart cameras in privacy-sensitive applications by omitting the transfer of images of some parts of the observed scene.

Serpanos et al. [47] identified the most important security issues of smart camera networks and classified the major security requirements at the node level and the network level. Although security issues of distributed smart cameras are analogous to networked embedded systems and sensor networks, emphasis should be given to special requirements of smart camera networks, including privacy and continuous realtime operation. To guarantee data authenticity and protect sensitive and private information, a wide range of mechanisms and protocols should be included in the design of smart camera networks.

3.4 Service Orientation and User Interaction

Besides all technological considerations and challenges, one of the major aspects is easily forgotten: smart camera systems should be designed for users. This is even more important for future camera networks—some of which are targeted at consumer applications. For these applications, service orientation, robustness and and ease of use are important factors for user acceptance.

A main challenge is still identifying and demonstrating multi-camera applications which are really useful and desirable for users. Aside from obvious surveillance scenarios, applications that are frequently mentioned are personal health and elderly care where the environment is monitored for unusual events such as a falling person [48]. Smart homes are another related scenario where pervasive smart camera networks could be employed to simplify the life of the residents. Services for smart homes include an adaptation of the environment (e.g., lighting or air conditioning) based on the detection of the presence of persons. The automatic detection of gestures and activities by the smart camera network supports a more active user interaction. Much progress has been achieved in that field in recent years (e.g., [49]), however a lot of research is still required until human gesture and activity recognition can be applied in real-world settings.

Regardless of the actual application scenario, a big challenges is to develop multi-camera systems that can be deployed, set up and operated by customers with little or no technical knowledge.

4 Conclusion

Smart camera networks have emerged thanks to the simultaneous advances in four key disciplines: computer vision, image sensors, embedded computing, and sensor networks. The convergence of these technical factors stimulates a revolution in the way we use cameras. Image sensors will become ubiquitous and fade into the everyday environment. Their onboard processing and communication facilities foster collaboration among cameras and distributed data analysis.

Considering the recent advances of smart cameras in research and industrial practice, we can identify several trends: First, camera networks currently undergo a transition form static to dynamic and adaptive networks. Second, as the cost of single cameras and the required network infrastructure drops we will see an increase in the size of the camera networks. Finally, we expect to see researchers start to integrate different sensors—audio, seismic, thermal, etc.—into distributed smart sensor networks. By fusing data from multiple sensors, the smart camera exploits the distinct characteristics of the individual sensors resulting in an enhanced overall output. All these advances will stimulate the development of many new applications—transforming traditional multi-camera systems into pervasive smart camera networks.

References

[1] B. Rinner, W. Wolf, Introduction to Distributed Smart Cameras, Proceedings of the IEEE 96 (10).

[2] H. Aghajan, R. Kleihorst (Eds.), Proceedings of the ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 07), Vienna, Austria, 2007.

[3] R. Kleihorst, R. Radke (Eds.), Proceedings of the ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 08), Stanford, USA, 2008.

[4] D. A. Patterson, J. L. Hennessy, Computer Architecture: A Quantitative Approach, 4th Edition, Morgan Kaufman, San Francisco CA, 2006.

[5] W. Wolf, High Performance Embedded Computing, Morgan Kaufman, San Francisco CA, 2006.

[6] R. Kleihorst, B. Schueler, A. Danilin, Architecture and Applications of wireless Smart Cameras (Networks), in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007), Honolulu, Hawaii, USA, 2007.

[7] R. Kleihorst, A. Abbo, B. Schueler, A. Danilin, Camera Mote with a High-Performance Parallel Processor for Real-Time Frame-Based Video Processing, in: Proceedings of the First ACM/IEEE International Conference on Distributed Smart Cameras ICDSC '07, 2007, pp. 109–116.

[8] M. Bramberger, A. Doblander, A. Maier, B. Rinner, H. Schwabach, Distributed Embedded Smart Cameras for Surveillance Applications, Computer 39 (2) (2006) 68–75.

[9] T. W. J. Moorhead, T. D. Binnie, Smart CMOS Camera for Machine Vision Applications, in: Proceedings of the IEE Conference on Image Processing and its Applications, Manchester, UK, 1999, pp. 865–869.

[10] L. Albani, P. Chiesa, D. Covi, G. Pedegani, A. Sartori, M. Vatteroni, VISoc: a Smart Camera SoC, in: Proceedings of the 28th European Solid-State Circuits Conference), Florence, Italy, 2002, pp. 367–370.

[11] W. Wolf, B. Ozer, T. Lv, Smart Cameras as Embedded Systems, Computer 35 (9) (2002) 48–53.

[12] M. Bramberger, J. Brunner, B. Rinner, H. Schwabach, Real-Time Video Analysis on an Embedded Smart Camera for Traffic Surveillance, in: Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2004), Toronto, Canada, 2004, pp. 174–181.

[13] C. Arth, H. Bischof, C. Leistner, TRICam - An Embedded Platform for Remote Traffic Surveillance, in: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW 2006), New York, U.S.A., 2006.

[14] D. Bauer, A. N. Belbachir, N. Donath, G. Gritsch, B. Kohn, M. Litzenberger, C. Posch, P. Schön, S. Schraml, Embedded Vehicle Speed Estimation System Using an Asynchronous Temporal Contrast Vision Sensor, EURASIP Journal on Embedded Systems 2007 (2007) 12.

[15] F. Dias, F. Berry, J. Serot, F. Marmoiton, Hardware, Design and Implementation Issues on a Fpga-Based Smart Camera, in: Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on, 2007, pp. 20–26.

[16] R. P. Kleihorst, A. A. Abbo, A. van der Avoird, Op, L. Sevat, P. Wielage, R. van Veen, H. van Herten, Xetal: a low-power high-performance smart camera processor, in: Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on, Vol. 5, 2001, pp. 215–218 vol. 5.

[17] B. Rinner, M. Jovanovic, M. Quaritsch, Embedded Middleware on Distributed Smart Cameras, in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007), Honolulu, Hawaii, USA (invited paper), 2007, pp. 1381–1384.

[18] C. H. Lin, W. Wolf, A. Dixon, X. Koutsoukos, J. Sztipanovits, Design and Implementation of Ubiquitous Smart Cameras, in: Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous and Thrustworthy Computing, Taichung, Taiwan, 2006, pp. 32–39.

[19] M. Quaritsch, B. Rinner, B. Strobl, Improved Agent-oriented Middleware for Distributed Smart Cameras, in: Proceedings of the ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 2007), Vienna, Austria, 2007, pp. 297–304.

[20] M. Quaritsch, M. Kreuzthaler, B. Rinner, H. Bischof, B. Strobl, Autonomous Multicamera Tracking on Embedded Smart Cameras, EURASIP Journal on Embedded Systems Volume 2007 2007 (2007) 10.

[21] M. A. Patricio, O. Carbó, J.and Pérez, J. García, J. M. Molina, Multi-Agent Framework in Visual Sensor Networks, EURASIP Journal on Applied Signal Processing 2007 (1) (2007) 226–226.

[22] S. Fleck, F. Busch, W. Straßer, Adaptive Probabilistic Tracking Embedded in Smart Cameras for Distributed Surveillance in a 3D Model, EURASIP Journal on Embedded Systems 2007 (2007) 17.

[23] E. Norouznezhad, A. Bigdeli, A. Postula, B. C. Lovell, A high resolution smart camera with gige vision extension for surveillance applications, in: Proceedings of the Second ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC-08), 2008.

[24] Y. Shi, T. Tsui, An FPGA-Based Smart Camera for Gesture Recognition in HCI Applications, in: Computer Vision - ACCV 2007, Springer Berlin / Heidelberg, 2007, pp. 718–727.

[25] I. F. Akyildiz, T. Melodia, K. R. Chowdhury, Wireless Multimedia Sensor Networks: Applications and Testbeds, Proceedings of the IEEE 96 (10).

[26] A. Rowe, A. G. Goode, D. Goel, I. Nourbakhsh, CMUcam3: An Open Programmable Embedded Vision Sensor, Tech. Rep. CMU-RI-TR-07-13, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (May 2007).

[27] A. Rowe, D. Goel, R. Rajkumar, FireFly Mosaic: A Vision-Enabled Wireless Sensor Networking System, in: D. Goel (Ed.), Proceedings of the 28th IEEE International Real-Time Systems Symposium RTSS 2007, 2007, pp. 459–468.

[28] M. Rahimi, R. Baer, O. I. Iroezi, J. C. Garcia, J. Warrior, D. Estrin, M. Srivastava, Cyclops: In Situ Image sensing and interpretation in wireless sensor networks, in: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, ACM, New York, NY, USA, 2005, pp. 192–204.

[29] H. Medeiros, J. Park, A. Kak, A Light-Weight Event-Driven Protocol for Sensor Clustering in Wireless Camera Networks, in: J. Park (Ed.), Proc. First ACM/IEEE International Conference on Distributed Smart Cameras ICDSC '07, 2007, pp. 203–210.

[30] C. B. Margi, X. Lu, G. Zhang, R. Manduchi, K. Obraczka, Meerkats: A Power-Aware, Self-Managing Wireless Camera Network for Wide Area Monitoring, in: International Workshop on Distributed Smart Cameras (DSC-06), 2006.

[31] C. Margi, V. Petkov, K. Obraczka, R. Manduchi, Characterizing energy consumption in a visual sensor network testbed, in: Proceedings of the 2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities TRIDENTCOM 2006, 2006, p. 8pp.

[32] S. Hengstler, D. Prashanth, S. Fong, H. Aghajan, MeshEye: A Hybrid-Resolution Smart Camera Mote for Applications in Distributed Intelligent Surveillance, in: Proceedings of the 6th International Symposium on Information Processing in Sensor Networks IPSN 2007, 2007, pp. 360–369.

[33] C. Wu, H. Aghajan, R. Kleihorst, Mapping Vision Algorithms on SIMD Architecture Smart Cameras, in: H. Aghajan (Ed.), Proc. First ACM/IEEE International Conference on Distributed Smart Cameras ICDSC '07, 2007, pp. 27–34.

[34] P. Chen, P. Ahammad, C. Boyer, S.-I. Huang, L. Lin, E. Lobaton, M. Meingast, S. Oh, S. Wang, P. Yan, A. Y. Yang, C. Yeo, L.-C. Chang, J. Tygar, S. S. Sastry, Citric: A low-bandwidth wireless camera network platform, in: Proceedings of the Second ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC-08), Stanford, CA, USA, 2008.

[35] W.-C. Feng, E. Kaiser, W. C. Feng, M. L. Baillif, Panoptes: scalable low-power video sensor networking technologies, ACM Transactions on Multimedia Computing, Communications, and Applications 1 (2) (2005) 151–167.

[36] T. Teixeira, D. Lymberopoulos, E. Culurciello, Y. Aloimonos, A. Savvides, A Lightweight Camera Sensor Network Operating on Symbolic Information, in: Proceedings of the Workshop on Distributed Smart Cameras (DSC 06), Boulder, USA, 2006.

[37] R. J. Radke, D. Devarajan, Z. Cheng, Calibrating Distributed Camera Networks, Proceedings of the IEEE 96 (10).

[38] R. T. Collins, A. J. Lipton, H. Fujiyoshi, T. Kanade, Algorithms for Cooperative Multisensor Surveillance, Proceedings of the IEEE 89 (10) (2001) 1456–1477.

[39] C. H. Lin, T. Lv, W. Wolf, I. B. Ozer, A Peer-to-Peer Architecture for Distributed Real-Time Gesture Recognition, in: Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME 2004), Taipe, Taiwan, 2004, pp. 57–60.

[40] S. Velipasalar, J. Schlessman, C.-Y. Chen, W. Wolf, J. P. Singh, SCCS: A Scalable Clustered Camera System for Multiple Object Tracking Communicating via Message Passing Interface, in: Proceedings of IEEE International Conference on Multimedia and Expo 2006, 2006.

[41] T. Stathopoulos, M. Lukac, D. McIntire, J. Heidemann, D. Estrin, W. J. Kaiser, End-to-end Routing for Dual-Radio Sensor Networks, in: Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM), Anchorage, USA, 2007, pp. 2252–2260.

[42] D. Lymberopoulos, N. B. Priyantha, M. Goraczko, F. Zhao, Towards Energy Efficient Design of Multi-radio Platforms for Wireless Sensor Networks, in: Proceedings of the 7th International Conference on Information Processing in Sensor Networks, Washington, DC, USA, 2008, pp. 257–268.

[43] N. P. Mahalik (Ed.), Sensor Networks and Configuration: Fundamentals, Standards, Platforms, and Applications, Springer, 2007.

[44] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. F. Shu, M. Lu, Enabling video privacy through computer vision, IEEE Security & Privacy Magazine 3 (3) (2005) 50–57.

[45] A. Chattopadhyaya, T. Boult, PrivacyCam: a Privacy Preserving Camera using uCLinux on the BlackFin DSP, in: Proceedings of the Workshop on Embedded Computer Vision (ECVW 2007), Minneapolis, U.S.A., 2007.

[46] S. Fleck, W. Strasser, Smart Camera Based Monitoring System and Its Application to Assisted Living, Proceedings of the IEEE 96 (10) (2008) –.

[47] D. N. Serpanos, A. Papalambrou, Security and Privacy in Distributed Smart Cameras, Proceedings of the IEEE 96 (10) (2008) –.

[48] S. Fleck, R. Loy, C. Vollrath, F. Walter, W. Straßer, SmartClassySurv - A Smart Camera Network for Distributed Tracking and Activity Recognition and its Application to Assisted Living, in: Proceedings of the First ACM/IEEE International Conference on Distributed Smart Cameras ICDSC '07, 2007, pp. 211–218.

[49] D. A. Forsyth, O. Arikan, L. Ikemoto, J. OBrien, D. Ramanan, Computational Studies of Human Motion: Part 1, Tracking and Motion Synthesis, Foundations and Trends in Computer Graphics and Vision 1 (2-3).