THE EVOLUTION FROM SINGLE TO PERVASIVE SMART CAMERAS

Bernhard Rinner[1], *Thomas Winkler*[1], *Wolfgang Schriebl*[1], *Markus Quaritsch*[1] *and Wayne Wolf*[2]

[1] Institute of Networked and Embedded Systems, Klagenfurt University, AUSTRIA
[2] School of Electrical and Computer Engineering, Georgia Institute of Technology, USA

{firstname.lastname}@uni-klu.ac.at[1], wolf@ece.gatech.edu[2]

ABSTRACT

Having seen increased interest from the research community, smart camera systems have gone through a number of evolutionary steps like from single cameras to distributed smart camera systems with collaboration features. This work aims at defining a taxonomy to classify these systems based on their platform capabilities, the degree of distributed processing as well as system autonomy aspects covering self-configuration and mobility. Existing camera systems are classified according to the proposed taxonomy. Besides capturing the design space for smart cameras, the main contribution of this paper is an overview of the research challenges for the envisioned class of pervasive smart camera systems. As defined in this work, future pervasive smart camera systems will be visual sensor networks targeted at end-user applications where special emphasis is put on unobtrusiveness of the cameras as well as simple deployment supported by self configuration capabilities.

***Index Terms*—** Pervasive Smart Cameras, Taxonomy, Classification, Design Space, Research Challenges

1. INTRODUCTION AND MOTIVATION

Smart cameras have been the subject of study in research and industry for quite some time. While in the 1980s some camera prototypes were developed which integrated sensing with some low-level processing, first commercial ”intelligent” cameras appeared in the 1990s. However, the sensing and processing capabilities were very limited on these cameras. In the meantime we have seen dramatic progress in smart camera research and development [1, 2].

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 [3] 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 paper is twofold. First, we want to review and classify smart camera research. This classification is based on a taxonomy addressing the platform capabilities, the degree of distributed processing and the degree of autonomy. Second, we want to elaborate the vision of pervasive smart cameras and identify major research challenges towards this vision. The discussion of the research challenges is based on an exploration of the design space of current smart camera systems.

The remainder of this paper is organized as follows: Section 2 introduces our taxonomy for smart camera classification. Section 3 reviews smart camera systems. Section 4 discusses pervasive smart cameras and addresses research challenges. Finally, section 5 concludes this paper.

2. A TAXONOMY OF SMART CAMERA SYSTEMS

Smart camera systems can be described using different characteristics. In this paper we use the platform capabilities, the degree of distribution and the system autonomy as classification parameters.

2.1. Platform Capabilities

The platform capabilities of smart camera systems are described by the following features:

Image Sensor The image sensor is a crucial component in the overall image processing pipeline. It strongly affects the quality of the captured images. Relevant features include the image resolution, color depth and supported frame rate.

Processing Unit Smart cameras are implemented using a variety of different processing units ranging from general purpose CPUs and digital signal processors (DSPs) over field programmable gate arrays (FPGAs) to system on chip (SOC) solutions.

Some implementations exploit multi-core architectures which are often equipped with dedicated cores for image processing and system management. Memory bandwidth and capacity are further crucial resources for smart cameras since image processing tasks typically deal with huge amounts of data.

Communication Facilities Communication can be either established over wired or wireless interfaces. The available bandwidth can be classified into (1) low bandwidth architectures providing a few hundred kbit/s (e.g., 802.15.4 based wireless networks like ZigBee), (2) intermediate systems with bandwidths ranging from a few to several Mbit/s (e.g., Bluetooth, UMTS) to (3) high performance networks featuring several dozens to hundreds of Mbit/s (e.g., WLAN, Gigabit LAN).

Power Requirements The overall power consumption is an important platform parameter. We distinguish here between cameras capable of running on (1) mains power supply or (2) on battery.

2.2. Degree of Distributed Processing

In multi-camera systems the distribution of processing can be classified as follows: (1) Multiple video streams are analyzed independently without any collaboration. Thus, image processing occurs either on the camera or at a central server where the cameras stream their data to. (2) Multiple video streams are analyzed collaboratively at a central server. Data or abstracted information delivered from multiple cameras is jointly processed but within a single thread of control. (3) Video analysis is performed collaboratively by multiple cameras in a distributed manner. Control is distributed among the camera network, and cameras cooperate to solve given tasks and exchange information potentially at different granularity levels, i.e., pixels within regions of interests, features or decisions.

A clear separation between classes (2) and (3) is not always possible since distributed processing is sometimes realized by exploiting a centralized infrastructure for coordination and management.

2.3. System Autonomy

In the context of smart camera systems we refer to system autonomy as the support for deployment, configuration and mobility.

Deployment Deployment refers to the process of installing the smart camera network. We distinguish between (1) static deployment which often relies on expert users and (2) ad-hoc deployment which requires some self-configuration mechanisms provided by the network.

Configuration Configuration refers to the process of adapting or modifying the network in operation. We consider three different configuration approaches: (1) Static configuration is performed once during setup of the network. (2) Remote configuration supports modifications during operation using a standardized interface. (3) In self-configuration, the cameras perform modifications during operation by themselves—with or without initial learning.

Configuration can be performed at the node level (configuring a single camera) or at the network level (configuring groups of cameras).

Mobility Regarding mobility of nodes, four different types are distinguished: (1) Static cameras which are mounted at fixed locations, (2) semi-static cameras which locations can be changed at some specific operation modes (e.g., suspend), (3) PTZ cameras and (4) mobile cameras performing image analysis while moving.

3. ANALYSIS AND CLASSIFICATION OF SMART CAMERA SYSTEMS

This section describes smart camera systems and research prototypes available today and classifies them according to the taxonomy presented in the previous section (cp. table 1).

3.1. Single Smart Cameras

The processing unit of a smart camera executes the individual stages of a typical image processing pipeline. 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. 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.

Moorhead and Binnie [4] presented one of the first fabricated CMOS implementations. Their SoC smart camera integrated edge detection into the image sensor. VISoc [5] 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.

Wolf et al. [6] developed a first generation smart camera prototype for detecting people and analyzing their movement in real-time. 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. [7]. Their first prototype is based on a single DSP COTS-system (TMS320C64xx processor from Texas Instruments). This camera is 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 is realized over a wired Ethernet connection.

Arth et al. [8] 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. [9] 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.

Dias et al. [10] 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.

Kleihorst et al. [11] 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.

3.2. Distributed Smart Cameras

The subject of distributed smart cameras is to integrate a number of smart cameras in a common network. The main reasons therefore are to (1) resolve occlusion, and (2) extend sensor coverage. As each camera has reasonable computing and communication capabilities, such a network of smart cameras can also be treated as a distributed system for image processing. Distributed smart cameras can be organized either in a centralized manner, where some sort of pre-processing is done on the individual cameras but a central node controls each camera and merges information provided by individual cameras, or in a completely decentralized fashion, where cameras have to organize themselves and collaborate on a certain task.

Implementing and deploying distributed smart cameras with decentralized coordination poses several new research challenges. In [12, 13] we have already discussed that a substantial middleware would greatly enhance application development. Such a middleware has to integrate the camera's image processing capabilities and provide a transparent inter-camera communication mechanism.

The data movement mechanisms must be designed for timely data delivery to support real-time and low power computation. One open question in middleware architectures is the breakdown between generic and application-specific services. Generic services can be shared among more applications, saving development time and cost, but application-specific versions of services can be tuned to the characteristics of the application.

In [14] we propose to use agents as top-level abstraction. A distributed application comprises several mobile agents, whereas agents represent image processing tasks within the system. Attributing agents the 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 [15].

Patricio et al. [16] also use the agent-oriented paradigm. But in their approach, an agent manages a single camera. The agents have an internal state representing beliefs, desires and intentions. Collaboration of cameras hence maps 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. [17] demonstrate a multi-camera tracking implementation, but camera coordination and object hand-off between cameras is done on a central host. 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.

3.3. Smart Cameras in Sensor Networks

Right now, sensor networks are receiving a lot of attention from the scientific community [18]. While many networks are focused on processing scalar sensor values such as temperature or light measurements, there are some networks using 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 and memory resources, communication capabilities, system size and power consumption.

The Meerkats sensor nodes developed by Margi et al. [19] are using an Intel Stargate mote equipped with a 400MHz

StrongARM processor, 64MB SDRAM and 32MB Flash. For wireless communication the nodes are equipped with a 802.11b standard PCMCIA cards. The image sensor is implemented using consumer USB webcams delivering images at 640×480 pixels. The sensor nodes are operated by an embedded Linux system. They evaluate the power consumption of different tasks such as flash memory access, image acquisition, wireless communication and data processing. Additionally, it is shown that the power consumption for state transitions as part of duty-cycling can not be neglected for many components. In [20] additional 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 transmitted to a central sink. A very similar system, also based on Stargate motes and USB webcams is Panoptes presented by Feng et al. [21].

Another representative of a smart camera for sensor networks is the Cyclops camera by Rahimi et al. [22]. In terms of computing power the system is equipped with an ATmega128 8-bit RISC microcontroller operating at 7.3MHz. The system provides 4KB of on-chip SRAM and an additional 60KB of external RAM. The CMOS sensor can deliver 24bit RGB images at CIF resolution (352×288). For image capturing a CPLD is acting as an intermediate device between the sensor and the microcontroller. The Cyclops platform does not provide on-board networking facilities but it can be attached to a MicaZ mote. In [23] Medeiros and Park 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 presented by Hengstler et al. [24] combines multiple vision sensors on one node. The platform is equipped with two low resolution image sensors as used in an optical mouse, and one VGA color image sensor placed in the middle of the mouse sensors. 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, based on stereo vision, the location of the detected object is computed. This 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 computations are done on an ARM7 microcontroller running at 55MHz. The system is equipped with 64KB of RAM and 256KB flash memory. Additionally, an SD/MMC card slot is provided. For wireless networking, an 802.15.4 chip is included.

The WiCa wireless camera by Kleihorst et al. [25] is using a SIMD processor called IC3D operating at 80MHz and is specifically designed with image processing applications in mind. The processor internally features 320 RISC processing units operating on the image data stored in parallel memory. In addition to the line memories, the platform provides access to external DPRAM allowing for e.g., non-line based image manipulation. For general purpose computations and communication tasks, the WiCa is equipped with an 8051 microcontroller. Optionally, 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 [26].

The CMUcam 3 is the latest version of an embedded computer vision platform developed by Rowe et al. [27]. It consists of a color CMOS sensor capable of delivering 50 frames per second at a resolution of 352×288 pixels. Image data is written into a FIFO from where images are fetched by the processing unit, an ARM7 microcontroller operating at 60MHz. The system is equipped with 64KB of RAM and 128KB of flash memory. The CMUcam comes with a software layer implementing various vision algorithms such as color tracking, frame differencing, convolution or image compression. The CMUcam 3 is not equipped with on-board networking capabilities but an external mote can be attached via a serial communication channel. This approach is used to combined it with FireFly motes [28]. The resulting system, called FireFly Mosaic, relies on tight time synchronization for multi-camera cooperation. The nodes are statically deployed in the context of home activity monitoring. The configuration of the system is done as part of an initial learning phase.

3.4. Design Space for Smart Cameras

Our taxonomy based on platform capabilities, degree of distributed processing and autonomy can also be used to express the design space of smart cameras. Figure 1 summarizes the design space of single, distributed and sensor network smart cameras.

Recent single smart cameras provide fairly sophisticated computing power and memory capacity. They rely on fixed infrastructure for power supply and networking. In terms of autonomy, today's systems typically offer remote configuration capabilities or very limited self configuration functionality.

Distributed smart cameras have very similar platform capabilities as single smart cameras. However, they are deployed in groups enabling distributed data acquisition and data processing. Many of today's systems provide local data pre-processing and feature extraction. Fully distributed processing, where nodes collaboratively work on a given task with little or no involvement of some central facility, is much less common. Regarding system autonomy, most systems are designed to be deployed statically and offer remote configuration capabilities. Many implementations provide basic functionality for inter-node cooperation (e.g., formation of camera clusters). Complete self-configuration has not yet been demonstrated. While limited mobility is supported by some

	Platform Capabilities				Distributed Processing	Autonomy		
System	Sensor	CPU	Comm.	Power		Deployment	Configuration	Mobility
Moorhead and Binnie [4]	CMOS	custom logic for on-chip edge detection	n/a	mains	local image analysis; no collaboration	static	unknown	static
VISoc (Albani)[5]	CMOS, 320x256	32-bit RICS and vision/neural processor	n/a	battery	local image analysis; no collaboration	static	unknown	static
Wolf [6]	Hi8 Camcorder, NTSC	PC with TriMedia TM-1300 boards	n/a	mains	local image analysis; no collaboration	static	unknown	static
Single SmartCam (Bramberger, Rinner) [7]	color, VGA	DSP	n/a	mains	local image analysis; no collaboration	static	remote configuration	static
TRICAM [8]	video in (no sensor)	DSP and FPGA, 128MB RAM	Ethernet	mains	single node; multiple video inputs	static	static configuration	static
Bauer [9]	neuromorphic sensor (64×64 pixels)	Blackfin DSP	n/a	mains	local image analysis; no collaboration	static	unknown	static
Dias and Berry [10]	2048×2048, gyroscope and accelerometer	Altera Stratix FPGA	Firewire (1394)	mains	local image analysis; no collaboration	static	active vision	static
Distributed SmartCam (Bramberger, Quaritsch, Rinner) [29]	VGA	ARM and multiple DSPs	100Mbps Ethernet, GPRS	mains	local image analysis; cooperative tracking	static	remote and dynamic configuration	static
BlueLYNX (Fleck) [17]	VGA	PowerPC, 64MB RAM	Fast Ethernet	mains	local image preprocessing; central reasoning	static	unknown	static
GestureCam (Shi) [30]	CMOS, 320x240 (max. 1280x1024)	Xilinx Virtex-II FPGA; custom logic plus PowerPC core	Fast Ethernet	mains	local image analysis; no collaboration	static	unknown	static
CMUcam 3 (Rowe) [27]	color CMOS, 352×288	ARM7 at 60MHz	none onboard (802.15.4 via FireFly mote)	battery	local image analysis; inter-node collaboration [28]	static [28]	self configuration with initial learning phase [28]	static
Cyclops (Rahimi) [22]	color CMOS, 352×288	ATmega128 at 7.3MHz	none onboard (802.15.4 via MicaZ mote)	battery	collaborative object tracking [23]	static	dynamic clustering and cluster head election [23]	static
Meerkats (Margi) [19]	webcam, 640×480	StrongARM at 400MHz	802.11b	battery	local image analysis; collab. object tracking; image transmission to central sink [20]	static	static	static
MeshEye (Hengstler) [24]	2× low resolution sensor, 1× VGA color CMOS sensor	ARM7 at 55MHz	802.15.4	battery	unknown	unknown	unknown	unknown
WiCa (Kleihorst) [25]	2× color CMOS sensor, 640×480	Xetal 3D (SIMD)	802.15.4	battery	local processing; collab. reasoning [26]	static	static configuration	static

Table 1. Classification of smart camera systems.

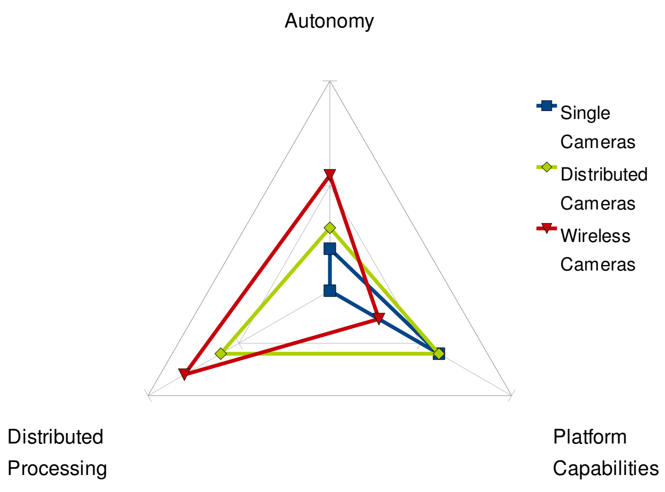

Fig. 1. The design space for smart cameras.

systems in the form of PTZ cameras, static installations are generally used. Figure 1 illustrates the similarities of single and distributed smart cameras with respect to platform capabilities and autonomy as well as their difference regarding distributed processing.

Wireless smart cameras, in contrast, can be clearly distinguished from distributed smart cameras regarding their platform capabilities. Core criteria are a wireless communication channel and battery powered operation. An immediate consequence of these requirements is that systems offer considerably less computing power and on-board memory than single or distributed smart cameras to keep their power consumption as low as possible. Directly related to reduced computing power is the fact that many wireless cameras use lower resolution image sensors than single or distributed cameras. Being small and self-contained, wireless cameras offer an even larger freedom regarding distribution of the system than conventional distributed smart cameras. Another reason for the increased level of distributed processing shown in figure 1 stems from the necessity for increased collaboration to compensate the fairly limited computing capabilities of the individual nodes. The enhanced autonomy results from the fact that wireless camera systems should be designed with ad-hoc deployment and self configuration capabilities in mind. Practical realizations of these aspects still are at early stages.

3.5. Trends

The expected trend for single smart camera systems is that computing power will see a steady increase. System autonomy is expected to slightly increase, shifting more towards self-configuration to simplify system deployment.

The aim for distributing smart cameras is mainly enhanced reliability, coverage of larger areas and the ability to process data on-site. Therefore, future research of distributed smart cameras is not expected to focus on increasing the degree of distribution substantially, but will aim at improving performance and platform capabilities. Furthermore, system autonomy is expected to increase to support easier deployment by introducing nodes with self-calibration capabilities.

Research in the area of wireless smart cameras is expected to aim at enhancing ubiquitous usage of wireless cameras. The trend towards networks of *pervasive smart cameras* mainly implies improvements in system autonomy to enable installation and operation of camera networks by non-expert users. Therefore, a focus is expected to be on improving inter-node collaboration for supporting system autonomy in a self-organizing and self-configuring manner, and to optimize the shared usage of processing and memory resources by enforcing collaborative image processing. In the area of embedded systems, advances in chip manufacturing technologies are expected to be mainly used for lowering the power consumption and therefore increase the overall operating time of the system while only moderately improving system performance.

4. THE CHALLENGES OF PERVASIVE SMART CAMERAS

This section describes the vision of pervasive smart cameras, system requirements and challenges.

4.1. The Vision of Pervasive Smart Cameras

Typical scenarios for deploying traditional smart cameras are applications in well-defined environments with static sensor setups, as traffic surveillance scenarios [7]. The focus in designing these systems is usually put on the integration of sufficient processing power and system functionality to execute well-known image processing algorithms in real-time. In contrast to that, efforts for deployment, operation and maintenance play a minor role in design. Therefore, systems usually cannot be deployed by non-expert users, and cannot be deployed without adapting the existing infrastructure. This lack in system autonomy and adaptivity limits the range of feasible application scenarios dramatically. In this section, we want to present our vision of pervasive smart cameras (PSC) as one direction for more user-centric developments.

The vision of pervasive smart cameras is to have a smart camera infrastructure that allows for smooth embedment into activities of our daily life. Small and adaptive ubiquitous smart camera nodes form a dynamic network, which can be used for a wide range of applications. Nodes can be easily removed and added by non-expert users, supported by self-organizing functionality of the network. The nodes work autonomously in the sense of energy and sensor calibration and adapt themselves to environmental changes. The service-oriented architecture of the network makes installation and integration of new applications easily possible.

Typical scenarios for using pervasive smart cameras touch very different fields of application. Beyond emerging new applications such as elderly care [31] or home entertainment [26], the generic platform can be used to supplant other infrastructure by augmenting or replacing established methods with vision based approaches, e.g. gesture recognition for human computer interaction.

4.2. The Challenges

This section discusses emerging research challenges towards pervasive smart cameras. While most of the discussed challenges are of technical nature, this section also briefly covers application scenarios and questions related to user acceptance.

Hardware Challenges. The envisioned PSC system consists of a large number of camera sensors, which are integrated inconspicuously into the environment. Therefore, the physical dimensions of the sensor nodes are very limited, and wired connections for communication and power supply are not available. Purchasing costs and installation efforts must be kept at a minimum to get the system accepted by the user. Both, the limited size and the low price lead to lower processing power, smaller memory size and lower image sensor quality than available in comparable non-pervasive systems. Furthermore, the integration of additional hardware devices providing particular functionality, e.g., environmental sensors or localization facilities supporting autonomous operation must be considered. The main challenge in hardware design for PSC is to provide acceptable processing power and communication capabilities, while offering sufficient battery power for maintenance-free operation [11, 25].

System-Software Challenges. System level software covers low level operating system components including hardware drivers as well as higher level functionality and services, often subsumed under the term *middleware*. Considering the limited capabilities of the platform, the system software has to be kept as lightweight as possible. Functionality to be covered typically includes services for naming and lookup whereas the latter likely is not limited to devices but could also include services and resources offered by the individual nodes. Taking into account the limited capabilities of the nodes, sharing of resources as in grid computing applications might be a suitable option to accomplish more complex tasks. Related to that is distributed storage and dissemination of data throughout such a network with respect to unreliable communication channels and non-fixed system topology. A further aspect that is of importance in computer vision applications are real-time requirements including synchronization as well as timely delivery and availability of data. To meet these specific requirements, optimizations likely can not be limited to individual layers of the middleware or the operating system but a cross-layer approach might the required.

Privacy and Security Challenges. When deploying camera systems in end-user environments like homes or public places, the general user acceptance of such systems is of crucial importance. The increasing amount of video surveillance people are facing in their daily life is raising the awareness of privacy, confidentiality and general security issues in video surveillance applications [32]. Aside from questions like how to ensure the confidentiality of private data and enforcing access restrictions, one of the key challenges is to find proper mechanisms to allow users to check if a given system really is behaving as advertised by the implementer. If this trust in the behavior of the system can not be established, users likely will not accept the deployment of camera systems in sensitive places no matter how sophisticated the underlying security concept is. One approach to improving privacy is to not let raw imagery leave the camera, sending only partial or full analysis results across the network. However, this makes it much more difficult to ensure the trustworthiness of the system.

Adaptation / Autonomy Challenges. The envisioned PSC network follows a user-centric approach by minimizing efforts for configuration and maintenance purposes. Autonomous operation and automatic system adaptation play a major role and influence the system design on all levels of abstraction.

When altering the network structure by adding or removing nodes, the system must be adapted on network layer, middleware layer and application layer in a *self-organizing* manner. Logical clusters and role assignments must be renewed, processes must be redistributed and alternative data sources must be consulted and integrated.

Collaborative video processing strongly relies on calibrated image data in spatial and temporal domains. Each camera sensor has to provide perspective information about its viewpoint and about its field of view relative to a common coordinate system, and must provide timing information relative to a common time base. Synchronizing clocks in distributed systems with high accurateness is well researched, and mainly depends on the networking technology used [13]. Spatial calibration of nodes in a camera network is more complex and a very active field of research. Many methods for camera auto-calibration are based on cameras with an overlapping field of views (FOV), and also approaches for self-organization of sensors without overlapping FOV where presented [33, 34, 35, 36, 37, 38]. The envisioned PSC network must enable *self-calibration* by adopting methods for temporal and spatial calibration, with preferably no contraints for orientation and distribution of the sensors.

A further challenge in PSC design is the aspect of maintenance-free operation. The camera nodes in the PSC network are not designed to use mains power supply but integrate energy sources in the enclosure, e.g., batteries or solar panels. The power sources are limited in lifetime and therefore the trade-off between system functionality and node size is crucial for achieving an intended maintenance-free runtime of several days or weeks.

Collaboration Challenges. PSCs heavily rely on collaboration. The main rationales are (1) limited fields of view of the individual sensors, (2) low resolution image data, and (3) limited computational power. While the former argument is inherent for camera installations the other arguments are mainly due to the limited size and energy constraints of PSCs. One approach to cope with these limitations is by collaboration, where several nodes jointly work on a certain task. Collaboration can be either established by a central instance or the nodes autonomously organize themselves and form groups that investigate on a certain task. Since PSCs are intended for ad-hoc deployment, self-organization is required.

Collaboration further requires a substantial system-level software that allows individual nodes to form groups and exchange data within these groups [23]. In wireless sensor networks the agent-oriented approach is used to build autonomous systems [39, 40]. But also in general purpose computing the agent-oriented approach is used to manage distributed systems [41, 42, 43]. One of the challenges will be to adapt existing technologies for PSCs such that special requirements of computer vision applications are satisfied. An example is the localization within such networks: Existing technologies can provide a rough map of the network based the characteristics of the employed radio technology. This rough positioning could be enhanced by exploiting information from the visual sensors.

For collaborative computer vision algorithms it is crucial to know the position and orientation of the involved cameras. In the simplest case, this could be a graph representing the neighborhood relations. But most algorithms require a common world coordinate system which requires to calibrate each camera and establish homographies. Doing this manually requires a lot of time and effort. Hence researchers investigate in automatic camera calibration (e.g. [33, 34, 35, 36]).

Existing computer vision algorithms often are not designed with collaboration of distributed nodes in mind. For PSCs, however, this aspect is highly important. Hence, ways have to be found how algorithms can be adopted for such environments [13].

User Centric Challenges. Besides all technological considerations and challenges, one of the major aspects is easily forgotten: Smart camera systems should be designed for users. That is even more true for the envisioned class of pervasive smart cameras, being targeted at consumers who not necessarily have a technological background. Consequently, one of the main challenges is finding and implementing applications truly useful and desirable for users. Aside from obvious surveillance scenarios, applications that are frequently mentioned are e.g., personal health and elderly care where the environment is monitored for unusual events such as a fall of a person [31]. Another, related scenario are smart homes where pervasive smart camera networks could be employed to simplify the life of the inhabitants. This could include an adaptation of the environment (e.g., lighting conditions, preferred TV or radio station, etc.) based on the present persons. Another useful application in this context, taking into account concepts from human computer interaction research, would be gesture recognition for controlling and interacting with devices. This not necessarily has to be limited to e.g., opening the window blinds, but could very well be extended to leisure activities such as interactive gaming applications. Regardless of the actual application scenario, one of the biggest challenges will be to build systems that can be set up and operated by customers with little or no technical knowledge.

5. CONCLUSION

In this paper we have proposed a taxonomy for smart camera systems considering obvious aspects such as system capabilities and performance as well as less obvious but equally important aspects like the degree of distributed processing and system autonomy. Based on this taxonomy, selected representatives of existing smart camera systems have been described and classified. To provide a better overview of the field of smart camera development, we sketched the design space for single, distributed and wireless camera systems followed by an analysis of the trends for these three classes.

Having a look at existing smart camera systems, it becomes apparent that many systems are still limited in the degree of distributed processing. Even more evident is the lack in the field of system autonomy including support for ad-hoc deployment, self-configuration and mobility. These however are central requirements for smart cameras to become truly omnipresent system that can be handled by average users. To turn our vision of pervasive smart cameras into reality, a number of technological and user centric problems have to be addressed. We described major challenges on the way towards such systems together with potential directions and approaches.

With the hardware building blocks becoming cheaper and smaller, we believe that a trend is clearly going into the direction of pervasive smart cameras. The primal challenge will be to design systems that can easily be deployed by non-expert users to make them suitable for low-cost and consumer oriented applications.

6. REFERENCES

[1] B. Rinner and W. Wolf, “Introduction to Distributed Smart Cameras,” *Proceedings of the IEEE*, vol. 96, no. 10, Oct. 2008, to appear.

[2] H. Aghajan and R. Kleihorst, Eds., *Proceedings of*

the ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 07), Vienna, Austria, Sept. 2007.

[3] S. Khan, O. Javed, Z. Rasheed, and M. Shah, "Human Tracking in Multiple Cameras," in *Proc. of the 8th IEEE Int. Conference on Computer Vision ICCV '01*, vol. 1, 7–14 July 2001, pp. 331–336.

[4] T. W. J. Moorhead and T. D. Binnie, "Smart CMOS Camera for Machine Vision Applications," in *Proc. of the IEE Conference on Image Processing and its Applications*, Manchaster, UK, July 1999, pp. 865–869.

[5] L. Albani, P. Chiesa, D. Covi, G. Pedegani, A. Sartori, and M. Vatteroni, "VISoc: A Smart Camera SoC," in *Proc. of the 28th European Solid-State Circuits Conference*, Sept. 2002, pp. 367–370.

[6] W. Wolf, B. Ozer, and T. Lv, "Smart Cameras as Embedded Systems," *IEEE Computer*, vol. 35, no. 9, pp. 48–53, Sept. 2002.

[7] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach, "Real-time Video Analysis on an Embedded Smart Camera for Traffic Surveillance," in *Proc. of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium RTAS '04*, May 2004, pp. 174–181.

[8] C. Arth, H. Bischof, and C. Leistner, "TRICam - An Embedded Platform for Remote Traffic Surveillance," in *Proc. of the Conference on Computer Vision and Pattern Recognition CVPR '06 Workshop on Embedded Systems*, 2006, pp. 125–125.

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

[10] F. Dias, F. Berry, J. Serot, and F. Marmoiton, "Hardware, Design and Implementation Issues on a Fpga-Based Smart Camera," in *Proc. of the 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, Sept. 2007, pp. 20–26.

[11] R. Kleihorst, A. Abbo, A. van der Avoird, O. de Beek, L. Sevat, P. Wielage, R. van Veen, and H. van Herten, "Xetal: A Low-Power High-Performance Smart Camera Processor," in *Proc. of the IEEE Int. Symposium on Circuits and Systems ISCAS '01*, vol. 5, 2001, pp. 215–218.

[12] B. Rinner, M. Jovanovic, and M. Quaritsch, "Embedded Middleware on Distributed Smart Cameras," in *International Conference on Acoustics, Speech, and Signal Processing*, no. 4, Apr. 2007, pp. 1381–1384.

[13] C. H. Lin, W. Wolf, A. Dixon, X. Koutsoukos, and J. Sztipanovits, "Design and Implementation of Ubiquitous Smart Cameras," in *Proc. of the IEEE Int. Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing SUTC '06*, vol. 1, June 2006, pp. 32–39.

[14] M. Quaritsch, B. Rinner, and B. Strobl, "Improved Agent-Oriented Middleware for Distributed Smart Cameras," in *Proc. 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, 2007, pp. 297–304.

[15] M. Quaritsch, M. Kreuzthaler, B. Rinner, H. Bischof, and B. Strobl, "Autonomous Multi-Camera Tracking on Embedded Smart Cameras," *EURASIP Journal on Embedded Systems*, vol. 2007, p. 10, 2007.

[16] M. A. Patricio, O. Carbó, J.and Pérez, J. García, and J. M. Molina, "Multi-Agent Framework in Visual Sensor Networks," *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 226–226, Jan. 2007.

[17] S. Fleck, F. Busch, and W. Straßer, "Adaptive Probabilistic Tracking Embedded in Smart Cameras for Distributed Surveillance in a 3D Model," *EURASIP Journal on Embedded Systems*, vol. 2007, p. 17, 2007.

[18] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A Survey on Wireless Multimedia Sensor Networks," *Computer Networks*, vol. 51, pp. 921–960, 2007.

[19] C. Margi, V. Petkov, K. Obraczka, and R. Manduchi, "Characterizing energy consumption in a visual sensor network testbed," in *Proc. of the 2nd Int. Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities TRIDENTCOM '06*, Mar. 2006, p. 8.

[20] C. B. Margi, X. Lu, G. Zhang, R. Manduchi, and K. Obraczka, "Meerkats: A Power-Aware, Self-Managing Wireless Camera Network for Wide Area Monitoring," in *Int. Workshop on Distributed Smart Cameras DSC '06*, 2006.

[21] W.-C. Feng, E. Kaiser, W. C. Feng, and M. L. Baillif, "Panoptes: scalable low-power video sensor networking technologies," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 1, no. 2, pp. 151–167, 2005.

[22] M. Rahimi, R. Baer, O. I. Iroezi, J. C. Garcia, J. Warrior, D. Estrin, and M. Srivastava, "Cyclops: In Situ Image sensing and interpretation in wireless sensor networks," in *Proc. of the 3rd Int. Conference on Embedded Networked Sensor Systems SenSys '05*, 2005, pp. 192–204.

[23] H. Medeiros, H. Medeiros, J. Park, and A. Kak, "A Light-Weight Event-Driven Protocol for Sensor Clustering in Wireless Camera Networks," in *Proc. of the 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, J. Park, Ed., Sept. 2007, pp. 203–210.

[24] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan, "MeshEye: A Hybrid-Resolution Smart Camera Mote for Applications in Distributed Intelligent Surveillance," in *Proc. of the 6th Int. Symposium on Information Processing in Sensor Networks IPSN '07*, 25–27 April 2007, pp. 360–369.

[25] R. Kleihorst, A. Abbo, B. Schueler, and A. Danilin, "Camera Mote with a High-Performance Parallel Processor for Real-Time Frame-Based Video Processing," in *Proc. of the 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, Sept. 2007, pp. 109–116.

[26] C. Wu, H. Aghajan, and R. Kleihorst, "Mapping Vision Algorithms on SIMD Architecture Smart Cameras," in *Proc. of the 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, H. Aghajan, Ed., Sept. 2007, pp. 27–34.

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

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

[29] M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *IEEE Computer*, vol. 39, no. 2, pp. 68–75, Feb. 2006.

[30] Y. Shi and 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.

[31] S. Fleck, R. Loy, C. Vollrath, F. Walter, and W. Straßer, "SmartClassySurv - A Smart Camera Network for Distributed Tracking and Activity Recognition and its Application to Assisted Living," in *Proc. of the 1st ACM/IEEE Int. Conference on Distributed Smart Cameras ICDSC '07*, Sept. 2007, pp. 211–218.

[32] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. F. Shu, and M. Lu, "Enabling Video Privacy through Computer Vision," *IEEE Security & Privacy Magazine*, vol. 3, no. 3, pp. 50–57, 2005.

[33] B. Bose and E. Grimson, "Ground Plane Rectification by Tracking Moving Objects," in *Proc. of the Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance VS-PETS '03*, Oct. 2003.

[34] S. Khan and M. Shah, "Consistent labeling of tracked objects in multiple cameras with overlapping fields of view," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1355–1360, 2003.

[35] R. Pflugfelder and H. Bischof, "Online Auto-Calibration in Man-Made Worlds," in *Digital Image Computing: Technqiues and Applications DICTA '05*, Dec. 2005, pp. 519–526.

[36] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar, "Distributed Localization of Networked Cameras," in *Proc. of the 5th Int. Conference on Information Processing in Sensor Networks IPSN '06*. ACM, 2006, pp. 34–42.

[37] D. Devarajan, R. J. Radke, and H. Chung, "Distributed Metric Calibration of Ad-Hoc Camera Networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 3, pp. 380–403, 2006.

[38] R. B. Fisher, "Self-Organization of Randomly Placed Sensors," in *Proc. of the European Conference on Computer Vision ECCV '02*, May 2002, pp. 146–160.

[39] C.-L. Fok, G. C. Roman, and C. Lu, "Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications," in *Proc. of the 25th IEEE Int. Conference on Distributed Computing Systems ICDCS '05*, 2005, pp. 653–662.

[40] D. Georgoulas and K. Blow, "Making Motes Intelligent: An Agent-Based Approach to Wireless Sensor Networks," *WSEAS on Communications Journal*, vol. 5, no. 3, pp. 525–522, Mar. 2006.

[41] M. Yokoo and S. Fujita, "Trends of Internet auctions and agent-mediated Web commerce," *New Gen. Comput.*, vol. 19, no. 4, pp. 369–388, Jan. 2001.

[42] K. Stathis, O. de Bruijn, and S. Macedo, "Living Memory: Agent-Based Information Management for Connected Local Communities," *Interacting with Computers*, vol. 14, no. 6, pp. 663–688, Dec. 2002.

[43] W.-S. E. Chen and C.-L. Hu, "A mobile agent-based active network architecture for intelligent network control," *Information Sciences*, vol. 141, no. 1-2, pp. 3–35, Mar. 2002.