Wireless smart camera network for real-time human 3D pose reconstruction

Zoran Zivkovic

NXP Semiconductors Research
Eindhoven, the Netherlands

Abstract

A multiple-camera system for 3D pose reconstruction is presented. First, body parts of the user are detected. Each camera has a single-instruction multiple-data (SIMD) processor used to perform this heavy-load image processing task. The detected hand and head candidate positions are then transmitted wirelessly from each camera to a central processor using a low-power ZigBee network. Finally, the 3D pose reconstruction is performed at the central processor by combining the data in a probabilistic manner.

Key words: 3D object tracking, multi-camera systems, wireless camera networks, embedded processing, distributed processing, approximate Bayesian filtering

1 Introduction

Smart devices that respond to user in an appropriate way are expected to become a part of our everyday life [26,20]. Analyzing the behavior of the human user will play the central role and cameras are likely to be an important sensor for the task. Furthermore, it is widely believed that the computing in the smart environments will move from desktop computers to a multiplicity of the embedded computers present in the smart devices around us. Therefore, smart cameras that extract important information about the user from the images using on-board processing might become an important building block for future intelligent environments.

There are two general approaches in vision processing: top-down and bottom-up [21]. Typical for the top-down approach is to build a model that is fitted to the images, for example fitting 3D human model to images [11]. The bottom-up approach has become popular with the idea of extracting various features from images which are combined within an image and then across images from

different cameras in a multi-camera setting. Human pose reconstruction [7,27] is usually a starting point of human behavior analysis applications [1] . Multi-camera systems are used to increase robustness [25,11,14]. Beside the need for more robust algorithms [24], two major practical problems are: the difficulty of sending multiple video streams to a central processor and huge amount of processing that is required. For the bottom-up 3D pose reconstruction usually body parts are detected and then grouped together [14,21]. Distributing the body part detection across smart cameras with embedded processors can address both practical problems [28], see Figure 1. Furthermore, the latency of the processing can be greatly decreased, see Figure 2.

For flexible application smart cameras should communicate wirelessly, have a lot of processing power and remain low power to preferably operate for a long period on batteries. A practical wireless low power camera system for 3D human pose reconstruction is presented here. Real time, low latency, robust 3D upper body pose reconstruction is demonstrated on the system using battery powered wireless smart cameras. The architecture challenges for developing the networked camera application are analyzed and the available solutions compared. The SIMD solution [1] is chosen to build the system giving reasonable processing power, but keeping the power consumption low. A bottom-up approach is implemented for the 3D reconstruction. Head and hands of the user are detected using the embedded SIMD processor. An edge based algorithm is developed [8,12,4,29] taking special care of the parallel processing capabilities and the limited resources of the processor. The detected hand and head candidate positions are transmitted wirelessly from each camera to a central processor (a PC) where the user upper body 3D pose is reconstructed. A probabilistic framework is used for fusing the detected body part positions. The framework addresses the presence of missing and wrong detections, common in the wireless camera setting. Finally, various aspect of the whole system are analyzed and evaluated including the typical issues with the wireless communication. The conclusions and the noted issues are typical for a wireless camera setting, and in this paper they are illustrated and analyzed on a real system.

The paper is organized as follows. First, an overview of smart camera platforms is given and choices discussed. In Section 3 detection of hands and head on the chosen camera SIMD embedded processor is described. Section 4 describes probabilistic combination of the detections from different cameras. Section 5 analyzes the system performance and practical issues. Conclusions and recommendations are in the last Section.

[1] At the moment the gaming industry (e.g., Sony and Microsoft) is aiming to introduce new ways of user interaction based on 3D pose reconstruction.

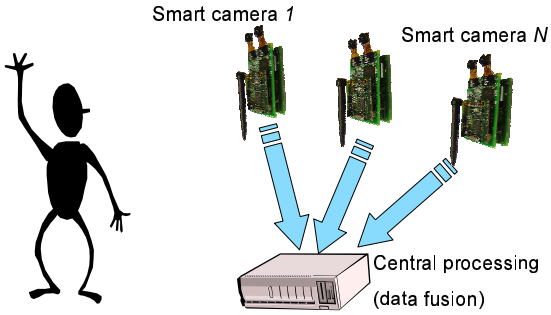

Fig. 1. Wireless smart camera network for real-time human 3D pose reconstruction. The smart cameras perform the computation-intensive detection of the person's head and hands. The extracted coordinates are transmitted using low-power ZigBee network to a central processing point where the data is combined and final reconstruction of the human 3D pose is made.

2 Smart camera architecture

Cameras with embedded processors are used in industrial inspection applications. Texas Instruments IP network camera [2] is an example of a wired smart camera for surveillance applications. It has a megapixel sensor and a TI TMS320DM355 SoC (ARM9 based) running up to 270 MHz and using 3W. It can be extended with a digital media processor TMS320DM6435 for more processing power. The smart camera platforms aimed at autonomous wireless smart camera networks are still in the research phase. The networked smart camera should have powerful onboard processing capabilities. Besides, there are two other major requirements:

- *Low power consumption.* Autonomous operation is a highly desirable feature. Ideally the cameras should work for long periods on batteries so that a large camera network can be easily deployed.
- *Advanced (wireless) communication capabilities.* The cameras should be able to send data to each other and/or to a central processing module. Wireless communication technology is in a mature phase and wireless communication in sensor networks is preferred for many practical reasons.

There are four recent hardware platforms designed specifically as low-power wireless smart cameras: MeshEye [17], CMUcam3 [22], WiCa [1] and CITRIC [5], see Table 1. MeshEye and CMUcam3 are based on an ARM7 processor and CITRIC platform on an ARM9. WiCa [1] is based on the NXP Xetal IC3D processor with 320 parallel processing elements. Besides the SIMD processor, an 8051 microcontroller is present.

[2] http://embedded-system.net/dm355-ip-network-¿¿camera-ipnc-reference-design.html

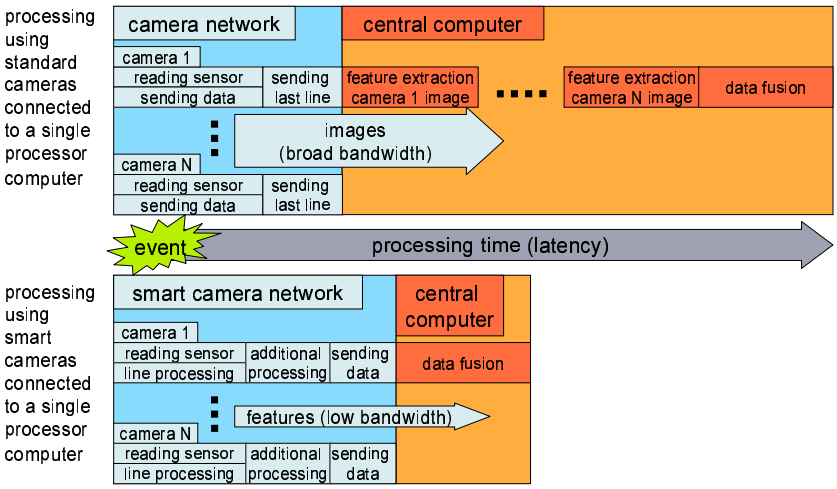

Fig. 2. Typical bottom-up computer vision algorithm using a standard camera network and a smart camera network. In the standard camera system processing usually starts when the whole frame is transferred to the main computer. On a smart camera most of the processing can be performed while reading the camera sensor, which reduces latency. Although smart cameras send much less data, the transmission is typically via a low-bandwidth data path that can extend the data transfer time.

Platform	Sensor	Processor	Transceiver	Power consumption
MeshEye [17]	three CMOS (one 640x480, color + two 30x30, 6 bit gray scale)	Atmel AT91SAM7S (ARM7 based), 55 MHz	ZigBee(802.15.4, 250 Kbps)	≈ 1W (power-efficient use of sensors tested to reduce this)
CMUcam3 [22]	CMOS (351x288, color)	NXP-LPC2106 (ARM7 based), 60 MHz	ZigBee(802.15.4, 250 Kbps)	≈ 650 mW
WiCa [1]	two CMOS (640x480, color)	NXP Xetal (SIMD 320 processors), 80 MHz + 8051, 24 MHz	ZigBee(802.15.4, 250 Kbps)	≈ 1W (3.75 W peak)
CITRIC [5]	CMOS (1280x1024, color)	Intel PXA270 (ARM9 based), up to 520 MHz	ZigBee(802.15.4, 250 Kbps)	≈ 1 W

Table 1
Comparison of selected wireless smart camera platforms.

Comparison of different embedded systems is difficult as it highly depends on the algorithm. The Xetal SIMD processor is highly efficient for operations that can be executed in parallel. For example on the CITRIC system Canny edge detection takes around 350ms, while on Xetal it takes around 2ms for 640×480 images. From the previous work on 3D pose reconstruction it is evident that detecting human body parts is computation-intensive [27,14,21]. The developers of MeshEye, CMUcam3, and CITRIC report the processing power as the main bottleneck, even for the simpler tasks of movement detection. Therefore, the WiCa wireless camera platform seems the only choice at the moment for the presented application.

3 Detection and tracking of head and hands

The first stage of a bottom-up algorithm for human pose estimation is to extract the body part candidates from the images. The presented algorithm is designed for the chosen camera embedded SIMD processor, Figure 3.

3.1 Features

The approach described here is based on edges and Chamfer distance. Chamfer distance has certain robustness to illumination and has been used successfully for example for pedestrian detection [13] and hand tracking [4]. Another reason for an edge based approach is that binary images can be efficiently stored in the memory which is a scarce resource on the NXP Xetal processor which has only 64 lines of memory available. One line-memory of the processor has 320 elements with 10 bits. The 10th bit is reserved as the sign bit, thus 9 lines of a 320-pixel-wide binary image are stored in one line memory of the processor. All operations are on Y component of the available YUV VGA (640×480) sensor. Due to the limited memory the resolution is reduced to 1/2 of its width so that a whole video line fits into 320 wide processor line-memory. Furthermore, every 4th line is processed.

Distance transform: The model and the target shape are described by sets of 2D edge points $\{x_1^m, ..., x_N^m\}$ and $\{x_1^t, ..., x_N^t\}$. The Chamfer distance of the model shape to the target shape is computed as:

$$d_{chamfer} = \frac{1}{N} \sum_{i=1}^{N} min_{x^t} ||x_i^m - x^t|| \tag{1}$$

Distance transform is used to compute the Chamfer distance. Full distance transform can be performed using 2 image passes [13,4]. While it is appropriate for a general purpose processor, sequential pixel processing is highly inefficient on a SIMD processor. Therefore, distance transform is implemented in a straightforward manner by initializing all edge pixel to zero distance and all other to some maximum allowed distance and then recursively applying a 3×3 distance filter. The result of the distance transform is an image where value of each pixel indicates the distance to the nearest edge pixel, see Figure 3. In the recursive implementation, the maximal distance that is computed depends on the number of times the distance filter is applied. Only 3 passes of the distance transform were performed for each image line on a buffer of last 14 image lines. This means that for the new line maximal distance of only 3 is computed. Larger maximal distances are computed for the lines that are longer in the buffer. Standard Canny edge detector was implemented using 5×5 Laplacian and Gaussian gradients.

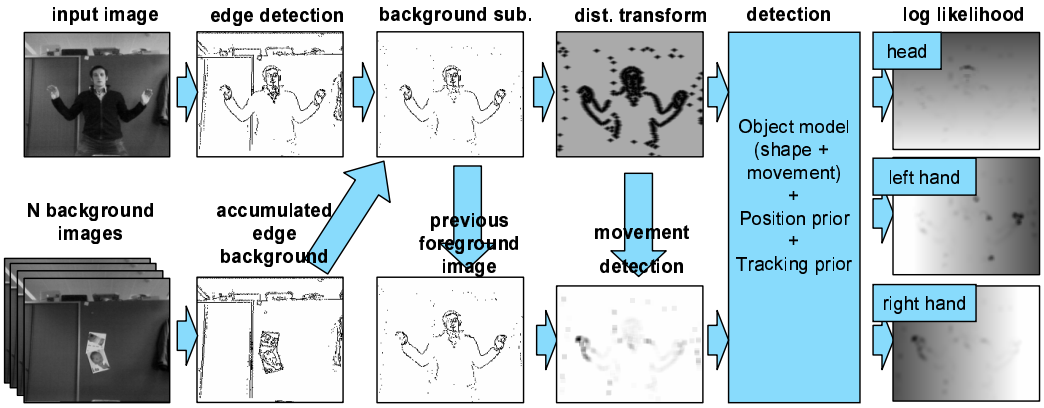

Fig. 3. Overview of the algorithm for user head and hands detection implemented on the smart camera SIMD processor.

Background subtraction: When camera is static, proper background subtraction can focus the processing on the foreground object. Due to limited resources, only simple shape representation is used for the Chamfer distance. It was observed that the number of false detections can be greatly reduced using a background model. Model of the background is constructed from a set of background edge images that are simply added together using OR operator. The result is a binary edge background which can be stored efficiently. We used 13 line memories of the processor to store $9*13 = 117$ lines of a 320-pixel-wide background image. Note that the simple binary background removes also some parts of the foreground object. However, the Chamfer distance is considered to be robust to missing points. This simple model was used due to the limited resources [8]. With more memory dynamic background model estimation could be implemented [33,30,32]. Furthermore, the smart camera platform has 2 image sensors and depth extracted from the two images could be used also to separate the foreground object from the background more robustly [19,10].

Movement detection: For fast movement images get blurred when captured by most standard camera sensors. This makes shape information less reliable especially for detecting person's hands. Therefore motion is used as an additional cue. The previous foreground edge image is saved using 13 line memories of the processor. Instead of the edge images, the already computed distance image is used for the detection. Pixels of the previous edge image get the value equal to the distance to the nearest edge of the current images. The result is smoothed using a 6×6 filter.

3.2 2D person model

The goal is to extract the most likely positions of the user head and hands. For each body part, $part \in \{head, lefthand, righthand\}$ a log-likelihood function over image 2D position x is defined $\log P_{part}(x)$. The likelihood will be

constructed by combining 4 sources of information, described below, that are considered independent:

$$\log P_{part}(x) = \log P_{shape}(x) + \log P_{movement}(x) + \log P_{prediction}(x) + \log P_{prior}(x) \quad (2)$$

The individual likelihoods in the sum are described next.

Shape: is the result of the Chamfer distance comparison. For head and each hand a simple edge template is used to model the average shape of the body part. An edge template is convolved with the distance image. The result is an image with pixel values equal to the Chamfer distance $d_{chamfer}$, see (1), that describes how well does the template fit into the edge map at that position. The edge templates were of 9×14 size. For each body part 10 edge templates are manually selected from a set of recorded images. The template that leads to the best detection results is chosen to be used in the final implementation. The log-likelihood is defined as:

$$\log P_{shape}(x) = -d_{chamfer}(x) \quad (3)$$

Movement: is the additional cue important for detecting fast moving user hands. Let $mov(x)$ be the average edge distance between consecutive frames computed as described in the previous section. For hands we use:

$$\log P_{movement}(x) = \alpha(max_d - mov(x)) \quad (4)$$

where max_d is equal to the maximum average edge distance possible. α is empirically chosen and it used to used to control the influence of the $P_{movement}$ within the whole likelihood.

Prior: it is assumed that the user is in more or less frontal position to the camera, and this prior knowledge can be included by adding some preference to certain positions for certain body parts. For head we define $\log P_{prior}$ as linearly increasing toward the top of the image. This describes that we expect to detect the head at the upper parts of the image. In a similar way for the left and right hand we use functions linearly increasing to the left-side and to the right side of the image.

Prediction: finally to perform some form of temporal smoothing (tracking) the head and hand 2D positions from the previous frame are used to express the preference to detect smooth movements. The likelihood is defined as:

$$\log P_{prediction}(x) = \beta max(|x - x_{prev}| + |y - y_{prev}|, c) \quad (5)$$

where x_{prev}, y_{prev} are the most likely positions of the corresponding body part from the previous frame. Constants β and c are empirically chosen. Parameter

β is used to control the influence of the $P_{prediction}$ within the whole likelihood and c to limit the influence only to a local area of the image ($c = 30$ pixels is used).

Combining the different cues, the final likelihood (2) is computed for each line while reading data from the sensor. Memory of the processor is used to buffer a number of previous video lines used for processing. The size of the buffer depends on type of operation. Typically, the size of the buffer needs to be at least equal to the vertical size of the largest filter that is used. After computing the likelihood, the most likely hand and hand positions are transmitted to a PC. The most likely positions are also computed while reading the sensor in on-line fashion. In this way the low-level processing adds almost no additional latency to the whole system, see Figure 2, and the resulting head and head detections are available almost immediately as reading the sensor data has finished.

4 3D data fusion and tracking

Each camera computes the most likely 2D image positions of the head and hands as described in the previous section. The positions are transmitted to a PC where they are combined.

4.1 Multi-camera point tracking

It is assumed that the cameras are calibrated. Having, the noisy 2D detections $\mathbf{x} = \{x^1, ..., x^{N_{cam}}\}$ from N_{cam} cameras, the 3D position of the corresponding point can be estimated. This is the so called "triangulation problem" [16], Figure 4. Once we calculate the 3D point reconstruction $m = m(\mathbf{x})$. The noise in each image is assumed to be Gaussian with covariance R_{image} and the uncertainty of the reconstruction can be estimated R by local Taylor approximation [3,16].

The true 3D position of the point in the space at time t is denoted by $\mathbf{s}_t = (x_t, y_t, z_t)$. The reconstructed 3D position at time t is m_t and computed as described above. The distribution $p(m_t|s_t) = \mathcal{N}(m_t, R)$ is Gaussian with the mean m_t and covariance R describing the uncertainty of the measured reconstruction. To track a 3D point, we wish to estimate the *the filtering density* $p(s_t|m_{1:t})$ for each time instance $t = 1, 2 \ldots$, given the sequence of measurements $m_{1:t}$. In Bayesian filtering $p(s_t|m_{1:t})$ is calculated recursively. The dynamic model that describes the movement of the 3D point should be a first order Markov process $p(s_t|s_{t-1}) = p(s_t|s_{1:t-1})$ and the measurements are inde-

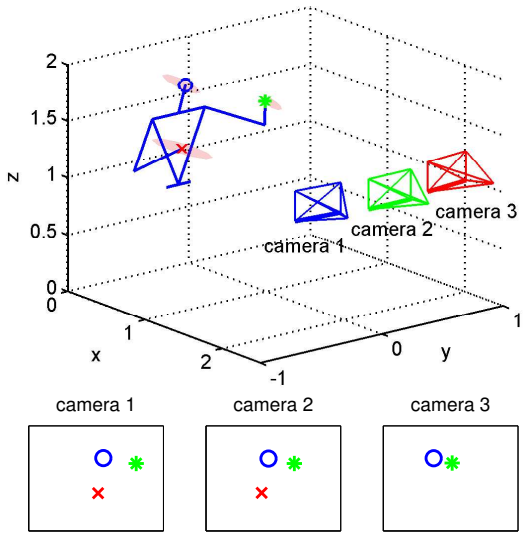

Fig. 4. User upper body model and three cameras. $\circ$ denotes head 3D position, $*$ - left hand, $\times$ - right hand. The corresponding noisy 2D detections in each camera are presented at the bottom row. The right hand is not detected in camera 3. The 2D detections are used find the maximum likelihood reconstruction $m(x)$. The ellipsoidal regions represent the uncertainty. Since all cameras are in front of the user the uncertainty is larger in the x direction, especially for the right hand which is detected by only 2 cameras. Standard pinhole camera model is used.

pendent from each other given the dynamic process, i.e., $p(m_t|s_t) = p(m_t|s_{1:T})$. Recursive updates are performed using:

$$\text{the prediction stage: } p(s_t|m_{1:t-1}) = \int p(s_t|s_{t-1})p(s_{t-1}|m_{1:t-1})ds_{t-1} \quad (6)$$

$$\text{the update stage: } p(s_t|m_{1:t}) = \frac{1}{c}p(m_t|s_t)p(s_t|m_{1:t-1}) \quad (7)$$

where $c = \int p(m_t|s_t)p(s_t|m_{1:t-1})ds_t$ and $p(s_{t-1}|m_{1:t-1})$ is the estimate from the previous time step. We use a simple random walk model $p(s_t|s_{t-1}) = \mathcal{N}(s_t; s_{t-1}, Q)$. We further assume transition noise covariance Q to be diagonal, with values estimated from data. Clearly, more elaborate dynamical models can be envisaged. The tracking is performed using the well known Kalman Filter [2] and all densities in the prediction (6) and update step (7) remain Gaussian.

4.2 3D pose reconstruction

Each body part, $part \in \{head, lefthand, righthand\}$, can be tracked separately in 3D using the Kalman Filter as described above. Alternatively, the 3 Kalman filter can be combined into a single one performing the same

equations. The part 3D positions are concatenated in a 9 dimensional vector $\mathbf{s}_t = s_{head}, s_{lefthand}, s_{righthand}$ and the tracking can be done using a 9 dimensional Kalman filter where measurements m are also concatenated and the corresponding covariances R-s are diagonal blocks of a new 9×9 dimensional covariance matrix.

Additional "soft constraints" on the positions of the hands with respect to the head can be introduced using Gaussian distribution $p_{pose}(\mathbf{s}_t) = \mathcal{N}(m_{pose}, Q_{pose})$ describing the constraints. The state transition model becomes now:

$$p(\mathbf{s}_t|\mathbf{s}_{t-1}) = \mathcal{N}(\mathbf{s}_t; \mathbf{s}_{t-1}, Q)p_{pose}(\mathbf{s}_t) \tag{8}$$

The static global constraints are imposed by the second part of the model. If the covariance matrix Q_{pose} is diagonal than this model can be seen as describing "string-like" constraints between the body-part positions [9]. The non-diagonal covariance matrix will express additional relations between the positions of the body parts. The transition model remains a Gaussian distribution and the tracking can be performed in the same way as before.

4.3 Missed detections and outliers

A body part might not be detected always by all cameras. Furthermore, in a wireless camera setup, detections from some camera might not arrive to the central node. The measurement distribution $p(m|s) = \mathcal{N}(m, R)$ should be computed accordingly, Figure 4. If a point is observed by only one camera then uncertainty in R becomes infinite along the line connecting the camera center and the detection image position, and the 3D position m can be anywhere on this line. The final reconstruction is still defined because of the temporal filtering (7).

Body part detectors might provide from time to time completely wrong, 'outlier' detections. Explicitly taking outliers into consideration increases robustness [2]. Let h denote hypothesis that certain detections are outliers, e.g., a detection by 2 cameras has 4 options: only first or only second true, both outliers or both true. Given a hypothesis h corresponding 3D reconstruction will be m_h and the observation density approximated by a Gaussian mixture:

$$p(m|s) \approx \sum_{\text{all outlier hypotheses } h} \mathcal{N}(m_h, R_h)\rho_h \tag{9}$$

The probability of the hypothesis component weights ρ_h can be chosen in different ways. Here for each m_h, corresponding 2D image position and detections, and image detection uncertainty R_{image} are used to calculate the probability of the true detections. Uniform distribution over the image plane pBG_h is assumed for the outliers. The resulting filtering distribution (7) becomes a

Processing step	clock cycles/video line	load
Canny edge detection (using 5x5 filters)	163	3%
Distance transform (running a 3x3 filter 3 times)	285	6%
Head and hand shape models convolution (3 times 9x14 filter)	789	14%
Movement detection (6x6 filter)	327	6%
Overhead (background model, combining likelihoods, etc.)	154	3%
Total	1717	31%

Table 2
Processing load of the SIMD processor. Operating at 30 frames/s and with processor clock at 80MHz there are 5555 clock cycles available per video line (80MHz / (30fames/s * 480 video lines)). Due to the memory limitation every 4th video line is processed and therefore the presented numbers need to be divided by 4 for the average processing load.

mixture of Gaussians and the number of modes will grow exponentially. Practical solution is keeping the best K modes and performing inference using a mixture Kalman filter (MKF) [6,2]. In the presented experiment $K = 5$ was sufficient.

5 Experiments

The performance of the system is analyzed in this section.

5.1 Part detection on a smart camera

Table 2 contains the number of clock cycles and total load of the Xetal SIMD processor. Only 30% of the processor power was used for the algorithm from Section 3. Unfortunately due to the memory constraints we processed only every 4th line and only half of the horizontal sensor resolution. Current implementation is using all the memory resources of the processor.

To evaluate the part detection algorithm a dataset is made of 200 images. The user performed different movements remaining more or less frontal to a camera. The current smart camera system can capture a few images and transmit them wirelessly every few seconds. The 2D image positions of head and hands were hand-labeled in all images as the ground truth. The detections by the algorithm were also recorded. The average errors are reported in Table 3. The percentage of errors larger than 25 pixels is also given to indicate the number of 'outlier' detections. The experiments were repeated at two different scenes, one with little detail and the other more complex. Further, the influence of the background subtraction and movement detection on the Chamfer distance detection is studied. From the results we can see that the background model has little influence in case of a simple scene. The accuracy is even decreased since the simple model will remove some parts of the edges. In the complex scene,

the background model helps to focus on the foreground object. Especially the 'outlier' detections are reduced. It can be also observed that the movement detection can help for the hand detection. Chamfer distance with a single shape is unreliable for the complex shape of hands. Furthermore, fast moving hands are blurred in the image and the edges often difficult to detect. The data-set can be used to optimize the parameters of the part detectors from the previous section, but for simplicity we just hand tuned the parameters once.

			Head		Hands	
scene type	backgr. subtr.	movement detection	error std. deviation	error > 25	error std. deviation	error > 25
simple 	off	off	5.5	2%	16.1	17%
	on	off	5.5	2%	16.1	17%
	off	on	5.6	2%	11.1	9%
	on	on	5.5	2%	11.6	9%
complex 	off	off	12.2	9%	18.9	27%
	on	off	**7.2**	**3%**	15.9	18%
	off	on	12.5	8%	17.1	25%
	on	on	7.5	4%	**14.6**	**15%**

Table 3
Detection accuracy of hand and hands (converted to correspond to 320×240 images). Influence of the background model and the movement detection on the Chamfer distance head and head detection.

5.2 Data transmission

# of cameras per channel	**Blocking Mode**		**Non-blocking Mode**		**Robust Mode**	
	Speed (KB/s)	Lost packets	Speed (KB/s)	Lost packets	Speed (KB/s)	Lost packets
1	6.597	0.52%	4.628	0.01%	1.820	0.00%
2	5.657	11.53%	4.013	12.76%	1.810	0.17%
3	5.064	27.8%	3.278	27.77%	1.766	0.37%

Table 4
Communication speed and the percentage of lost packets when multiple cameras are communicating via one channel.

The detected head and hands candidate 2D positions are sent to a PC using a low power ZigBee wireless link. As expected with more cameras the number of lost packets is higher because of collisions and buffer overflow issues during transmitting, see Table 4. Total length of a messages was 96 bytes including the message header. Longer messages are expected to lead to more lost packets [15]. Three sending modes were examined. In "Normal mode" the cameras operate in a fully controllable fashion using whole ZigBee network software library. In "Blocking mode" cameras are just sending data to PC host module. The "Robust mode" is introduced to tackle the problems with sending data from multiple cameras over the same wireless channel. In the "Robust mode" data was sent only once per frame (each 30ms). The results show that using

the "Robust mode" can support 3 cameras using a single channel with very low percentage of the lost data packets.

5.3 Upper body 3D pose reconstruction

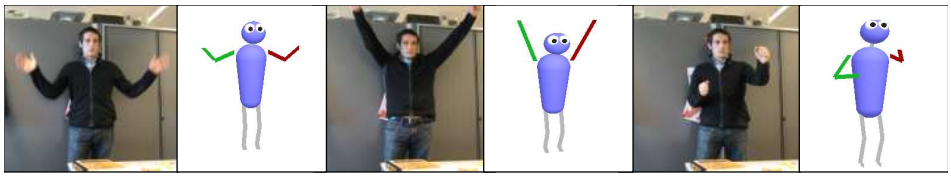

Fig. 5. Example body poses as seen by one of the camera and the avatar showing the resulting 3D reconstruction computed in real-time. For presentation purpose we draw shoulders at fixed distance from the head and elbows in the head-shoulder–hand 3D plane with a proper upper-lower arm length.

The demonstration system was used with two or three smart cameras. The cameras are calibrated using a blinking LED that can be detected robustly by the used smart camera [18]. Images are processed at 30 frames/s. The main time delay occurs due to the wireless communication and the data-fusion on the PC. Currently performing the KF steps to fuse the data on the PC for 2 cameras takes around 5 ms on a 2 GHz PC. For MKF with $K = 5$ modes (Section 4.3), approximately 5 times more processing is needed. The wireless communication delay is below 30ms and therefore the whole system latency around $30 - 50$ms. Furthermore, the current processing on the PC host takes around $5 - 25\%$ of the processor power and other applications can run on the PC simultaneously without any problems. With the current load the wireless cameras were able to work around 4 hours using 4 AA rechargeable batteries of 2300 mAh. An avatar showing a set of example 3D reconstructions is presented in Figure 5.

To evaluate the accuracy of the 3D reconstruction, realistic 3D motion of human hands and head are obtained from the HumanEva data-set [24]. The dataset contains recordings of a number of activities. The 'throw-catch', 'boxing' and 'gesture' activity are used, where the subjects stay more or less frontal to the camera. For three subjects there is ground truth 3D positions data of the body parts corresponding to a few hundred video frames. For each subject and each activity 100 data points are selected for head and hands. Realistic simulation of our system for this 3D data is constructed. The subjects are set at 2.5 meters distance from the middle camera at start. The camera parameters correspond to the real system [18], Figure 4. In the two camera setup the middle camera is removed. The detection accuracy of the part detectors were simulated using the measured accuracy and outlier levels from the previous sections. The outlier data is generated as uniform random 2D detection in the image plane. The detection errors are assumed Gaussian with the stan-

dard deviation from the previous section. The results presented are simulation averages over 1000 random trials.

In Figure 6 the influence of the outlier detections is given. Outliers are added randomly with different probability levels. In case of 2 cameras the difference between the KF and MKF that explicitly considers outliers is small. In case of 3 cameras we see that the accuracy of the KF decreases since now there is a higher chance that there is an outlier among the data. On the other hand the MKF is able to make proper use of more cameras, detect outliers and increase the accuracy with respect to the 2 camera setup. In Figure 7 the influence of the accuracy of the detection and the missing data is presented. Missing data is simulated randomly with different probability levels. Because of the temporal filtering the influence of the missing data seems to be much less severe than the influence of the outliers.

In conclusion, with the approximate accuracy of detection of 15 pixels and outlier level of 15%, the simple part detectors implemented on the SIMD platform can give the accuracy of the 3D reconstruction comparable to the results of other more complex algorithms [24], but only on more or less frontal body poses. The graphs show also expected improvements by the increased accuracy of the part detectors. The Chamfer system can be greatly improved by using more shapes as discussed in detail in [12]. It should also be possible to handle non-frontal views. Current implementation takes 14% of the SIMD processing power for 1 shape per body part, see Table 2. The processing power allows almost 5 more different shapes per body part, but unfortunately memory is the limiting factor.

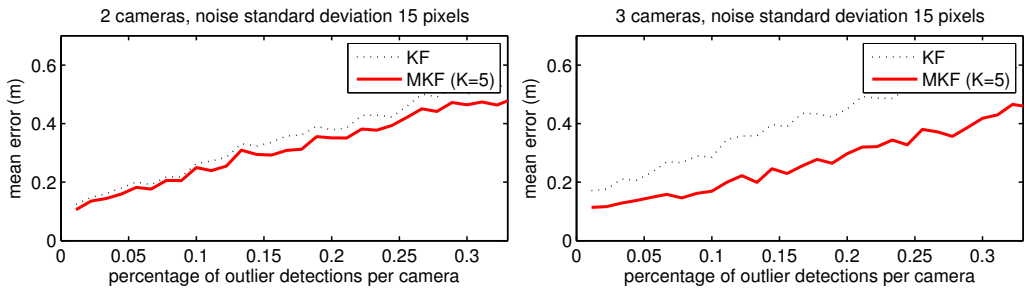

Fig. 6. Influence of outlier detections on the 3D reconstruction error.

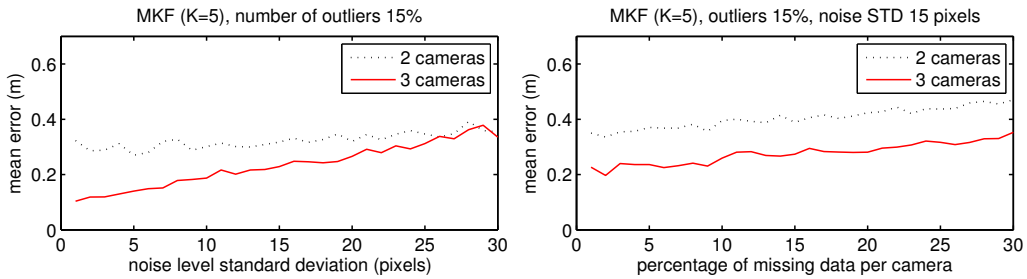

Fig. 7. Influence of the accuracy of the detection and missing data.

6 Conclusions

A wireless smart camera system for user 3D pose reconstruction is presented. The SIMD based camera [1] is selected to provide reasonable processing power, but keep the power consumption low. A bottom-up approach is implemented where head and hands of the user are detected using the SIMD processor. Parallel processing capabilities and the limited resources of the processor were considered. Furthermore, a general fusion framework for combination of the detected positions is described addressing two important issues: wrong (outlier) detections of the detectors and missing detections which might be due to communication problems. The data fusion can be extended in various ways, e.g., more realistic human body model [14]; or auto calibration [16] since currently the cameras are assumed calibrated [18]. Results show that with the current accuracy of detection of 15 pixels and outlier level of 15%, the implemented simple part detectors can give the accuracy of the 3D reconstruction comparable to the results of other more complex algorithms [24], but only on more or less frontal body poses. Non-frontal views and increased accuracy might be possible if more shape models are used for the part detection [12]. The processing power allows almost 5 more different shapes per body part, but the on-chip memory is the limiting factor of the evaluated processor. More cameras could be also used for the non-frontal views and increased accuracy but this can introduce more communication problems. However, it is noted that missing the detection has less influence than the wrong detections, see Figure 7. Furthermore, the percentage of the lost data in the proposed "Robust mode" of wireless communication is low. For very large number of cameras more complex network management could be considered [15].

The presented system operates 4 hours on 4AA batteries. This is still far from desired autonomous wireless solution for future intelligent environments. Power is used for processing and wireless communication. Further developments in processor technology and architecture, power management and energy harvesting techniques are expected to reduce processing power consumption [34]. The selected SIMD processor is an example of using parallelism to reduce the power consumption [1]. Algorithm specific integrated solutions might be a used instead of a general processor, further reducing power consumption. Recently some algorithm specific implementations were tested on FPGA board [31]. Wireless communication consumes much energy [15,34] and only limited amount of information should be communicated. Proper management of the whole wireless camera system might greatly reduce the communication power consumption. For example it was noted that missing detection has less effect on the accuracy than the outliers. This means that if the detection is unreliable it might be better not to transmit it and in this way save communication costs. Furthermore, a camera expected to give unreliable detections, e.g., because of the current user pose, might be put into some stand-by mode.

Acknowledgements

I would like to thank the members of the NXP Semiconductors Research, Richard Kleihorst, Ben Shueler, Alexander Danilin, Joost Hart, Peter Mijer , Chung-Ching Chang, Vitaly Kliger and Giuseppe Arturi.

References

[1] A.A. Abbo and R.P. Kleihorst. A programmable smart-camera architecture. In *Proc. Advanced Concepts for Intelligent Vision Systems*, 2002.

[2] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[3] A. S. Bedekar and R. M. Haralick. Finding corresponding points based on Bayesian triangulation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, page 61, 1996.

[4] B.Stenger, A. Thayananthan, P.H.S.Torr, and R. Cipolla. Filtering using tree-based estimator. *In Proc. Intl. Conf. on Computer Vision*, 2003.

[5] 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, D. Tygar, and S.S. Sastry. CITRIC: A low-bandwidth wireless camera network platform. In *Proc. International Conf. on Distributed Smart Cameras (ICDSC)*, 2008.

[6] R. Chen and J. Liu. Mixture kalman filters. *Journal of Royal Stat. Soc. B*, 62:493–508, 2000.

[7] D.Hogg. Model-based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

[8] D. Huttenlocher, J.J Noh and W.J. Rucklidge. Tracking Nonrigid Objects in Complex Scenes. In *Proc. International Conference on Computer Vision*, 1993.

[9] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1):55–79, 2005.

[10] X. Gao, R. Kleihorst, and B. Schueler. Stereo vision in a smart camera system. In *Proc. IEEE CVPR 2008 Workshop on Embedded Computer Vision*, 2008.

[11] D. Gavrila and L. Davis. Tracking of humans in action: A 3D model-based approach. *In Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 1996.

[12] D. M. Gavrila. A Bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Tr. Pattern Analysis Machine Int.*, 29(8), 2007.

[13] J. Giebel, D. M. Gavrila, and C. Schnrr. A Bayesian framework for multi-cue 3D object tracking. *In Proc. European Conference on Computer Vision*, 2004.

[14] A. Gupta, A. Mittal, and L.S.Davis. Constraint integration for efficient multiview pose estimation with self-occlusions. *IEEE Trans. Pattern Anal. Machine Intell.*, 30(3):493–506, 2008.

[15] C. De Santis H. Labiod, H. Afifi. *Wi-Fi, Bluetooth, Zigbee and WiMAX.* Springer, 2007.

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2003.

[17] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan. MeshEye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In *Proc. Information Processing in Sensor Networks*, 2007.

[18] M. Koch, Z. Zivkovic, R. Kleihorst, and H. Corporaal. Wireless camera network calibration using a blinking LED. *In Proc. Advanced Concepts for Intelligent Vision Systems*, 2008.

[19] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Probabilistic fusion of stereo with color and contrast for bi-layer segmentation. *International Journal of Computer Vision*, 76(2):107, 2008.

[20] M. Pantic, A. Pentland, A. Nijholt, and T.S. Huang. Human computing and machine understanding of human behavior: A survey. *Lecture Notes in A.I.*, 4451:47–71, 2007.

[21] D. Ramanan, David A. Forsyth, and Andrew Zisserman. Tracking people by learning their appearance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(1):65–81, 2007.

[22] A. Rowe, A. Goode, D. Goel, and I. Nourbakhsh. CMUcam3: An open programmable embedded vision sensor. In *Carnegie Mellon Robotics Institute Technical Report, RI-TR-07-13*, 2007.

[23] L. Sigal, S. Bhatia, S. Roth, M.J. Black, and M. Isard. Tracking loose-limbed people. *In Proc. IEEE Computer Vision and Pattern Recognition*, 1:421–428, 2004.

[24] L. Sigal and M. J. Black. HumanEva: Synchronized video and motion capture dataset for evaluation of articulated human motion. In *Technical Report CS-06-08, Brown University*, 2007.

[25] S. Velipasalar and W. Wolf. Multiple object tracking and occlusion handling by information exchange between uncalibrated cameras. In *Proc Int. Conf. Image Proc.*, Genova, Italy, Sep. 11–14, 2005.

[26] M. Weiser. The computer for the twenty-first century. *Scientific American*, 265(3):94–104, 1991.

[27] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(7):780–785, 1997.

[28] C. Wu and H. Aghajan. Collaborative gesture analysis in multi-camera networks. In *Proc. ACM SenSys Workshop on Distributed Smart Cameras*, 2006.

[29] J. Garcia, N. da Vitoria Lobo, M, Shah, and J. Feinstein. Automatic detection of heads in colored images. In *Proc. Canadian Conference on Computer and Robot Vision*, 2005.

[30] V. Reddy, C. Sanderson, B.C. Lovell and A. Bigdeli. An efficient background estimation algorithm for embedded smart cameras In *Proc. ACM/IEEE International Conference on Distributed Smart Cameras*, 2009.

[31] P. Chalimbaud and F. Berry. Embedded active vision system based on an FPGA architecture. *EURASIP Journal on Embedded Systems*, 2007.

[32] K. Kiratiratanapruk, P. Dubey and S. Siddhichai. A gradient-based foreground detection technique for object tracking in a traffic monitoring system. In *Proc. Advanced Video and Signal Based Surveillance*, 2005.

[33] Z.Zivkovic. Improved adaptive Gaussian mixture model for background subtraction. In *Proc. International Conference Pattern Recognition*, 2004.

[34] Z.Zivkovic and R.Kleihorst. *Smart Cameras for Wireless Camera Networks: Architecture Overview.* In A.Cavallaro, H.Aghajan, Eds., Multi-Camera Networks: Concepts and Applications, Elsevier, 2009.