Networked Computer Vision: the importance of a holistic simulator

Juan C. SanMiguel and Andrea Cavallaro

Smart-camera networks enable a wide range of services for vehicular ad-hoc networks, smart cities, home automation, wide-area surveillance, and search and rescue operations. These networks of cameras with inbuilt processing and communication capabilities generate large volumes of data, share high-data-rate messages and generally operate with limited resources. To design and test new applications for smart-camera networks a suitable simulator is needed to support the development and accurately predict the performance of vision algorithms before deployment.

***Index Terms*—Visual sensor networks, smart cameras, simulator, distributed, resource consumption.**

I. Introduction

The success of smart-camera networks (SCNs) depends upon the availability of simulators that facilitate fast algorithmic prototyping and validate performance objectives before deployment. Simulation tools may help predict performance and provide feedback on the models to be employed for real-world systems. Such tools need to account for the myriad of operational conditions and heterogeneity of devices that compose a SCN. While early works on camera networks assumed infinite bandwidth or cost-free data exchange [1], real-word SCNs must consider the constraints imposed by resource-limited platforms. Consider for example battery-powered cameras on board self-driving vehicles that communicate wirelessly to main-powered static cameras for tracking pedestrians, without exhausting their energy and the available bandwidth.

Because cameras capture, process and transmit much larger volumes of data than traditional sensor networks (SN), specific design and operational challenges arise to efficiently use the available resources and existing simulators lack the necessary functionalities (see Box in the next page). Designing SCN simulators requires interdisciplinary expertise covering algorithms, hardware and networking in order to model the camera hardware, to identify appropriate resources and to emulate communication protocols and channels [2]. In order to simulate a range of application scenarios, including dynamic decision-making with moving cameras, collaborative sensing and fusion with high-data-rate exchange and standalone operation, we have developed WiSE-Mnet++, a holistic SCN simulator that:

- models sensing, processing, communication and decision-making, which are the key operations in smart cameras;
- offers power-consumption models for smart-camera hardware; and
- simulates realistic multi-camera networks with both real-world and synthetic datasets.

WiSE-Mnet++ extends the WiSE-Mnet simulator [3] and is based on the OMNeT++ and Castalia SN simulators (see Figure 1). WiSE-Mnet++ is available as open source to the research community at http://www.eecs.qmul.ac.uk/~andrea/wise-mnet.html, along with supplementary material describing how to incorporate new simulation features and SCN algorithms. The WiSE-Mnet++ simulator facilitates smart-camera research by enabling one to easily compare solutions for specific research problems, such as the impact of real communication channels or limited computing capabilities on performance, and to activate or deactivate each simulated feature.

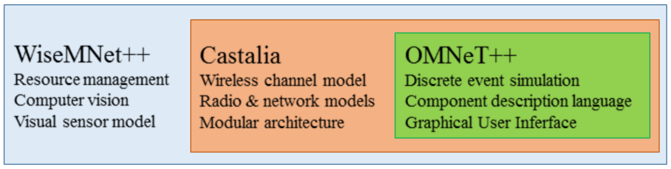


Figure 1: The relation between the WiSE-Mnet++ camera-network simulator, Castalia (http://castalia.forge.nicta.com.au/) and OMNeT++ (http://omnetpp.org/).

In this article we discuss the main features of the WiSE-Mnet++ simulator and two examples that show the effectiveness in profiling performance and energy consumption for networked computer-vision applications.

II. Camera node

WiSE-Mnet++ provides a generic yet descriptive modeling of the camera operations for sensing, processing and communication. A smart-camera is defined by layers that cover specific functionalities (see Fig. 2a). The functionality of a layer can be easily extended following an object-oriented scheme. The hardware associated to each layer is also simulated to determine the camera operational capabilities (e.g. processing frequency) and resources (e.g. battery) [4]. A message-passing structure enables inter-layer communication.

A. Sensing

The `WiseBaseSensor` layer provides input data by measuring the physical phenomena observed by the camera net-

Juan C. SanMiguel is with Universidad Autónoma de Madrid (Spain), email: juancarlos.sanmiguel@uam.es. Andrea Cavallaro is with Queen Mary University of London (UK), e-mail: a.cavallaro@qmul.ac.uk

CAMERA-NETWORK SIMULATORS

Early simulators of camera networks focused primarily on the use of video datasets for multi-camera surveillance and sport games (http://datasets.visionbib.com/). More comprehensive simulators were later proposed to account for communication and coordination with smart cameras. Table I summarizes these simulators that can be classified as local or global.

Local simulators test a particular aspect of cameras. For example, the Object Video Virtual Tool (OVVT) [a] and the Software Laboratory for Camera Network Research (SLCNR) [b] use virtual worlds to emulate the sensing of real-life scenarios. The Visual Sensor Network simulator (VSNSim) [c] also supports coordination and control, but lacks models for camera resources and communication channels thus making it difficult to implement realistic coordination approaches. Moreover, extending the functionalities of these simulators is not straightforward as they are provided as bundled packages. Finally, the CamSim simulator [d] defines protocols for communication between cameras, but without realistic communication models and without real-world video data as input.

Global simulators focus on realistic camera networking by extending OMNeT++, a popular discrete-event simulator for Wireless Sensor Networks. The Wireless Video Sensor Network (WVSN) simulator [e] determines the visual coverage of cameras over static 2D images, but without using video streams or visual analytics. The Mobile MultiMedia Wireless Sensor Network (M3WSN) [f] simulator addresses multimedia transmission without enabling collaborative processing. Although these simulators are extensible and can use communication protocols, they are mainly focused on 2D measurements, without support for video data, visual tools or resource-consumption models for smart-camera platforms.

WiSE-Mnet++, our smart-camera network simulator, takes advantage of discrete-event simulation to address the above-mentioned shortcomings.

Ref	**Name**	**Type**	**Calibration**	**Camera**	**Sensing**		**Processing**		**Communication**		**Coordination**		**Resources**		**Extensible**
				Mobility	**Synthetic**	**Real**	**Scalable**	**Visual**	**Ideal**	**Realistic**	**Topology**	**Modes**	**Consumption**	**Allocation**	
[a]	OVVT	CS	✓	✓	V			✓				SY			
[b]	SLCNR	CS	✓		V			✓				SY			✓
[c]	VSNSim	CS			V			✓	✓			SY			
[d]	CamSim	DS			MP				✓		CG, VG	SY			✓
[e]	WVSN	DS		✓	MP	I	✓			✓	CG	SY	C	S	✓
[f]	M3WSN	DS		✓	MP	I				✓	CG	SY	C	S	✓
Ours	**WiSE-Mnet++**	**DS**	✓	✓	**V,MP**	**I,R,L**	✓	✓	✓	✓	CG,VG	AS, SY	P	D	✓

Table I: Simulators for smart-camera networks and their main features. Empty cells represent features not offered by the corresponding simulator. KEY -- CS: Continuous simulation (real time). DS: Discrete Simulation. MP: Moving Points. V: Virtual video. R: Recorded video. L: Live video. CG: Communication Graph. VG: Vision Graph. AS: Asynchronous. SY: Synchronous. C: Constant. P: Parametric. S: Static. D: Dynamic.

REFERENCES

[a] G. Taylor et al., "OVVT: Using Virtual Worlds to Design and Evaluate Surveillance Systems", IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1-8, Jun. 2007. Available: http://development.objectvideo.com/

[b] W. Starzyk and F. Qureshi, "Software Laboratory for Camera Networks Research", IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 3, no. 2, pp. 284-293, Feb. 2013. Available: https://github.com/vclab/virtual-vision-simulator

[c] M. Gruber et al., "Demo: The extended vsnsim for hybrid camera systems", in Int. Conf. on Distributed Smart Cameras, pp. 203-204, Sept. 2015

[d] L. Esterle et al., "CamSim: A Distributed Smart Camera Network Simulator", in IEEE Int. Conf. on Self-Adaptive and Self-Organizing Systems Workshops, pp. 19-20, Sept. 2013. Available: https://github.com/EPiCS/CamSim

[e] C. Pham and A. Makhoul, "Performance study of multiple cover-set strategies for mission-critical video surveillance", IEEE Int. Conf. on Wireless and Mobile Computing, Networking and Comms., pp. 208-216, Oct. 2010. Available: http://cpham.perso.univ-pau.fr/WSN-MODEL/wvsn.html

[f] D. Rosario et al., "An OMNeT ++ Framework to Evaluate Video Transmission in Mobile Wireless Multimedia Sensor Networks", ICST Conf. on Simulation Tools and Techniques, pp. 277-284, Mar. 2013. Available: http://home.inf.unibe.ch/~zhao/M3WSN/

work. To introduce new sensor functionalities, this layer can be extended with sub-layers such as the `WiseCameraManager` to control the sensing and the capturing parameters, such as the focal length.

The `WiseBasePhysicalProcess` layer defines the observable phenomena (see Fig. 3). The `WiseVideoFile` and `WiseVirtualCam` extensions allow to use video data from real-world datasets and from virtual 3D worlds such as Unity (https://unity3d.com). Moreover, synthetic objects can be modeled as simple moving points on a common coordinate system (e.g. ground plane or zenital view) via the `WiseMovingTarget` extensions. In this case, the directional sensing of the field of view (FoV) is modeled on the ground plane as a 2D polygon defined by the orientation, the angle and depth of the camera view.

Unlike Pan-Tilt-Zoom smart cameras that consider only dynamic FoVs, the `WiseBaseMobility` enables to spatially move cameras by simulating the physical motion of their location that is typical of vision-based robotic applications [5].

B. Processing

The processing of video streams is pivotal for decision making and WiSE-Mnet++ defines a hierarchy of modules to coordinate the execution of the camera operations. The `WiseBaseApplication` layer is the interface with the network and provides basic capabilities to exchange data via the `WiseBaseComm` layer. The `WiseCameraAlgorithm` layer extends `WiseBaseApplication` with functions running at initialization and others called periodically for receiving new data. These functions also define a finite-state-machine that sequentially performs the three main camera operations for each sensed sample (e.g. a video frame). OMNeT++ timers are used to specify response times of the processing capabilities and to control the frequency when collecting data from `WiseBaseSensor`. Moreover, the sub-layer `WiseCameraPeriodicTracker` provides a ready-to-use functionality for target tracking. Finally, user applications are implemented by extending `WiseCameraAlgorithm` or `WiseCameraPeriodicTracker` with custom video analysis tools or third party libraries such as OpenCV.

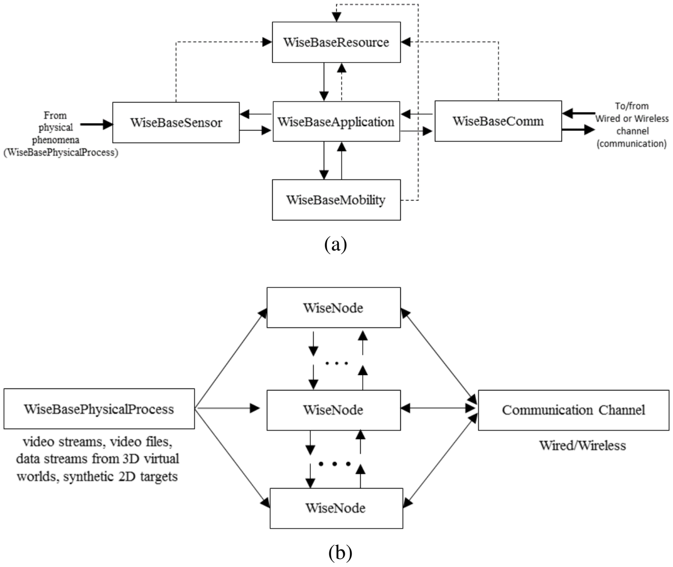

WiseBaseResource
From physical phenomena (WiseBasePhysicalProcess)
WiseBaseSensor
WiseBaseApplication
WiseBaseComm
To/from Wired or Wireless channel (communication)
WiseBaseMobility
(a)
WiseNode
WiseNode
WiseNode
WiseBasePhysicalProcess
video streams, video files, data streams from 3D virtual worlds, synthetic 2D targets
Communication Channel
Wired/Wireless
(b)

Figure 2: Layered WiSE-Mnet++ simulation for smart-camera networks. (a) A smart-camera node (`WiseNode`). Sensing, processing and communication capabilities are handled by the `WiseBaseSensor`, `WiseBaseApplication` and `WiseBaseComm` layers, respectively. The `WiseBaseMobility` changes the camera location and the `WiseBaseResource` monitors the employed resources. (b) A smart-camera network. `WiseNode` cameras are inter-connected by wired/wireless channels or by direct (instantaneous) message passing.

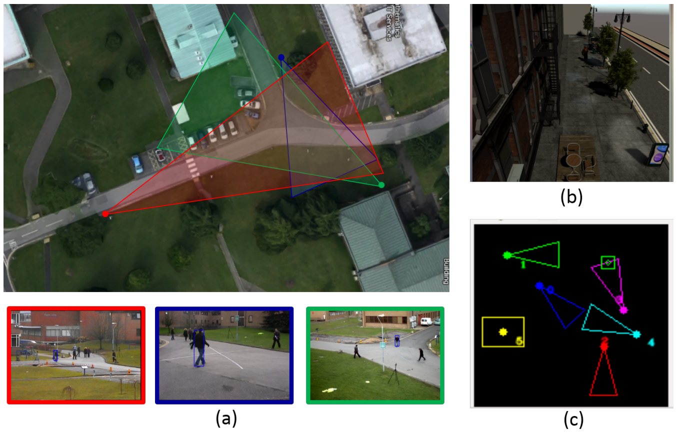


Figure 3: The three sensing options available in WiSEMnet++: (a) real-world video input or pre-recorded sequences (PETS 2009 dataset http://www.cvg.reading.ac.uk/PETS2009/); (b) streams from virtual 3D worlds; and (c) 2D synthetic data.

C. Communication

Unsynchronized and instantaneous inter-camera communication is enabled by `toNodeDirect` gates defined inside each `WiseNode` camera. This direct communication is useful for testing algorithms without considering the network.

The communication protocols and channels are implemented in the `WiseBaseComm` layer, which considers both ideal and realistic communication modes for data exchange. Buffer structures are defined to store the received data.

The *ideal communication* is an idealization of wired communications that helps develop collaborative algorithms while avoiding network and transceiver related problems, such as collisions due to simultaneous transmission by multiple cameras, when exchanging data. The `WiseDummyWirelessChannel` layer bypasses the communication protocol stack and enables a synchronized connectivity among cameras. The simulator also provides ideal communication conditions with instantaneous data exchanges without any packet losses or interferences.

The *realistic communication* is provided by the Castalia simulator that defines transceiver models (`Radio`), advanced channel models (`WirelessChannel`) and routing protocols for wireless sensor networks implemented in the `VirtualMac` layer. Realistic conditions should account for multiple factors such as the transceiver (radio) models; the communication protocol (e.g. MAC); interference and attenuation of the wireless channel; and the latencies of the camera modules.

D. Resource management

The `WiseBaseResource` layer models the resources and consumption associated to camera hardware which is key for resource-aware camera networks [6]. This layer also reports usage statistics to `WiseBaseApplication` for further reasoning. For example, a camera may re-allocate a task to other cameras to extend its lifetime.

WiSE-Mnet++ provides *capability descriptors* to model common hardware features, such as *frame rate* and *frame size* for sensing, *memory* and operating *frequency* for processing and available *bandwidth* and *power* modes for communication. These descriptors are loaded by the `WiseBaseResource` when initializing the simulation. New hardware features can be incorporated by extending this layer. To model *energy consumption*, each camera layer operates with a three-state model [7]. A specific state (active, sleep or idle) can be selected on demand (e.g. when the processor is requested to complete a task) or via designer-defined rules (e.g. by forcing a camera to sleep after a certain period of idleness). The power of the active state is approximated with an N-order polynomial model that accommodates existing non-linearities between resource usage and consumption. The power for the sleep and idle states are modeled as constants.

III. Camera network

Networked computer vision involves several cameras communicating with each other via single or multiple hops. WiSE-Mnet++ identifies the inter-camera links to enable the control of such networks (see Fig. 2b).

A. Network topology

WiSE-Mnet++ describes the network topology based on two types of neighborhood connectivity: vision and communication. The vision neighborhood defines cameras that share a portion of their FoV. The communication neighborhood determines cameras that can exchange messages with a single-hop communication. This neighborhood information can be manually introduced or automatically discovered. The WiseCameraAlgorithm can automatically compute the vision connectivity using external camera calibration data (i.e. camera location and orientation on a common coordinate system such as the ground-plane). The automatic discovery of communication connectivity relies on an iterative send-and-receive protocol performed in WiseBaseApplication. However, researchers can easily add more complex on-line approaches (e.g. task exchange patterns [8]) to discover and adapt the knowledge of the network topology during runtime.

B. Collaboration modes

The WiseCameraAlgorithm template supports two operation modes, asynchronous and synchronous, which can be selected in the initialization phase.

Asynchronous duty-cycled camera networks allow faster response times as cameras are always ready to collaborate and camera operations are not temporally coordinated. Hence, sensing acquires frames at a desired frame rate and the communication layer permanently listens to the channel for incoming data. Buffers are used for both sensing and communication as the data sensed or received may be processed with a delay. Processing is triggered when any of the buffers contains data.

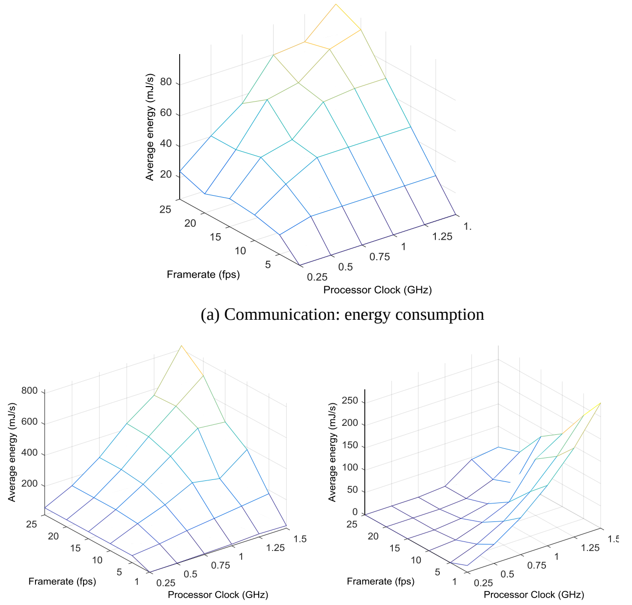
In the *synchronous* mode, cameras iteratively perform sequential sensing, processing and communication. No buffering is required as each operation starts after the previous one finishes. The speed of the execution pipeline is therefore determined by the slowest operation of the pipeline, thus potentially limiting the responsiveness of the SCN during collaboration.

IV. Case studies

We illustrate the advantages of WiseMNet++ in two important SCN applications, namely person re-identification and distributed tracking. As specific model for smart camera hardware, we use the ARM-A9 processor (0.5-1.5 GHz), the B3 image sensor (10-24MHz) and the C2420 radio (250kbps) [7]. The simulations are performed on a PIV-3.1GHz, 4GB RAM.

A. People descriptors (in-node processing)

Let us profile the energy consumption of a *detect-describe-transmit* task for people re-identification [9] when varying the sensing frame rate and the processing clock frequency. Each camera detects people within its FoV and generates visual descriptors of their appearance. For each frame, people are described by a vector including synchronization data (timestamp), the number of detections, normalized RGB histograms (3 channels, 16 bins/channel and 256 levels/bin) and spatial descriptors (center coordinates, width and height of the bounding box). Each detection generates a 6600-bit packet, which is compressed using Huffman encoding. The sub-layer WiseCameraApplication is customized to implement the described functionality and the camera employs video files using the WiseVideoFile extension.



(a) Communication: energy consumption

(b) Processing: active energy consumption

(c) Processing: idle energy consumption

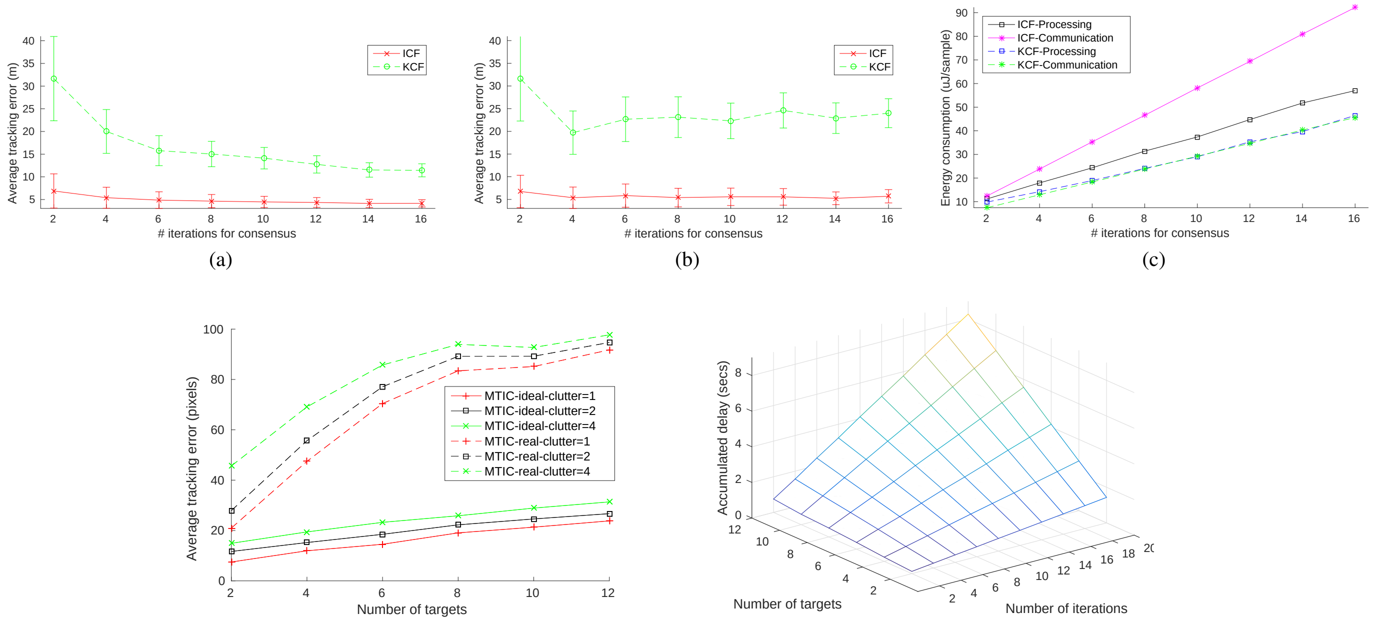
Figure 4: Energy consumption of the *detect-describe-transmit* task for the (a) communication module; (b) processing module (active state) and (c) processing module (idle state). Note that for high processor clocks and low frame rates the consumption of the idle and active states are comparable.

Fig. 4 reports the results for the *AVSS07_AB_eval* sequence (http://www.avss2007.org). Fig. 4a shows the energy consumption of the communication layer as a function of the camera hardware capabilities. High frame rates and high processor speeds lead to an energy consumption that is only one order of magnitude smaller than that of processing. Moreover, Fig. 4b and Fig. 4c show the energy consumption rate for the active and idle states of the processing module. The energy required for processing depends on both the frame rate and clock frequency. The consumption ranges from 25mW (0.25GHz) to 870mW (1.5GHz) when combining the idle and active states. As we increase the clock frequency, frames are processed faster and the associated cost increases. The energy of the idle state is only relevant when the processor is not loaded (1-5 fps) and operates at high frequencies (0.75-1.5GHz), being comparable to the active energy. This interestingly shows that, differently from the current beliefs, the idle power must be considered when measuring power consumption.

B. Distributed tracking (in-network processing)

Let a wireless camera network with eight cameras cover a $500m \times 500m$ area. Cameras get measurements at $4Hz$ (i.e. sampling time of $0.25s$) and have a communication range of $250m$. Targets move during $40s$.

Let us consider a distributed fusion task, with cameras exchanging data without the coordination of a task leader. We apply *consensus-based approaches* to distributively achieve an average over a quantity among the network nodes. Consensus is an iterative scheme where nodes share the data and then compute the mean of the received quantities. We



(a) (b) (c)

(d) (e)

Figure 5: Consensus-based distributed tracking under different network conditions. (Top row): The single-target trackers ICF and KCF under (a) ideal and (b) realistic wireless communication channels. The error decrease visible under ideal conditions is not maintained under realistic networks due to processing and transmission delays. (c) The average energy consumption for all cameras. (Bottom row): The multi-target tracker MTIC for various clutter levels and network conditions. (d) The tracking error depends on the delay in real networks. (e) Delay in processing. Note that for 2 targets and 6 iterations the delay in processing one sample exceeds the sampling rate (0.25s).

perform consensus-based single and multiple target tracking and measure the accuracy, the energy consumption and the delay associated to processing in ideal and realistic network conditions over 200 independent runs. We adapt the sub-layer `WiseCameraPeriodicTracker` to perform consensus and use the `WiseMovingTarget` extension for sensing moving targets within the FoV of cameras.

For single target tracking, we compare two consensus-based approaches: the Kalman-Consensus Filter (KCF) [10] and the Information-Consensus Filter (ICF) [11]. Each camera runs a KCF or ICF whose output is broadcast to all neighboring cameras, which apply consensus to estimate the target state (e.g. its position on the ground-plane).

Under ideal network conditions, as expected the tracking error decreases when increasing number of iterations as the estimation error of each camera is diffused over the other cameras (see Fig. 5a). KCF performs a *blind average* of the target state and therefore accumulates errors of cameras far away from the target. ICF outperforms KCF by sharing prior information about the absence of measurements when the targets are outside the FoV of cameras.

Under realistic conditions, the tracking error for ICF and KCF does not decrease when increasing the number of iterations (Fig. 5b). This is due to the accumulated delay for the iterations, as the transmission and reception of packets does not occur instantly, even for the small packets of ICF ($36bytes$) and KCF ($18bytes$).

The improvement of ICF in ideal conditions comes at an extra cost for processing and communication. ICF requires more than twice the energy of KCF for all iterations (Fig. 5c). Note that while research on smart cameras has traditionally considered communication costs negligible compared to that of processing, Fig. 5c shows equal costs for KCF whereas for ICF the cost for communication is greater than that for processing.

For multi-target tracking (MTT), we analyze the MTIC filter [12], which extends ICF to multiple targets. Network parameters, such as the MAC synchronization window, are configured to the setting that provides the fastest communication without error, which depends on the maximum number of targets (12) for the test conditions. With WiSE-Mnet++ we can explore two key factors affecting the MTT performance, namely measurements with clutter and network delay.

Fig. 5d shows the tracking error for MTIC for various clutter levels in ideal and realistic communication conditions. As the number of targets grows, it takes longer to exchange target states thus producing a delay that increases the tracking error (Fig. 5e). After the 6th iteration for two targets, the accumulated delay is greater than $0.25s$ (i.e. the sampling frequency) and therefore cameras miss target measurements. This latency to process each sample increases the final error of the estimation, regardless of the number of consensus iterations. Considering Fig. 5d and Fig. 5e, MTIC is more affected by network delays than by clutter, whose comparison is not usually performed when reporting tracking results [12].

V. Conclusions

The success of smart-camera networks depends on the supply of simulation environments that ease the development of distributed computer vision algorithms under realistic operational conditions. WiSE-Mnet++ is a holistic simulator that abstracts the key functions of camera networks and models the main operations whole accounting for hardware capabilities, the complexities of visual data and their associated high data-rate communication. WiSE-Mnet++ offers tools that help identify shortcomings and bottlenecks when designing or adopting algorithms for real smart-camera networks that might not be identified beforehand. WiSE-Mnet++ is extensible, flexible and ready to incorporate new features at algorithm, network and hardware levels.

References

[1] H. Aghajan and A. Cavallaro, *Multi-camera networks: principles and applications.* Academic press, 2009.

[2] M. Reisslein, B. Rinner, and A. Roy-Chowdhury, "Smart camera networks [guest editors' introduction]," *IEEE Computer*, vol. 47, no. 5, pp. 23–25, May 2014.

[3] C. Nastasi and A. Cavallaro, "WiSE-MNet: an experimental environment for wireless multimedia sensor networks," *Sensor Signal Processing for Defence (SSPD)*, pp. 34–34, 2011.

[4] J. Schlessman and M. Wolf, "Tailoring Design for Embedded Computer Vision Applications," *IEEE Computer*, vol. 48, no. 5, pp. 58–62, May 2015.

[5] B. Bhanu, B. Lovell, A. Prati, and F. Qureshi, "Guest editorial special issue on distributed smart sensing for mobile vision," *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2631–2631, May 2015.

[6] C. Piciarelli, L. Esterle, A. Khan, B. Rinner, and G. Foresti, "Dynamic reconfiguration in camera networks: a short survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 26, pp. 965–977, May 2015.

[7] J. C. SanMiguel and A. Cavallaro, "Energy consumption models for smart-camera networks," *IEEE Trans. Circuits Syst. Video Technol.*, 2016 (in press), http://dx.doi.org/10.1109/TCSVT.2016.2593598.

[8] P. Lewis, L. Esterle, A. Chandra, B. Rinner, J. Torresen, and X. Yao, "Static, dynamic, and adaptive heterogeneity in distributed smart camera networks," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 2, pp. 1–30, Jun. 2015.

[9] R. Mazzon, S. Tahir, and A. Cavallaro, "Person re-identification in crowd," *Pattern Recogn. Lett.*, vol. 33, no. 14, pp. 1828–1837, 2012.

[10] R. Olfati-Saber, J. A. Fax, and Murray R.M., "Consensus and Cooperation in Networked Multi-Agent Syst." *IEEE Proc.*, vol. 95, no. 1, pp. 215–233, 2007.

[11] A. Kamal, J. Farrell, and A. Roy-Chowdhury, "Information weighted consensus filters and their application in distributed camera networks," *IEEE Trans. Automat. Control*, vol. 58, no. 12, pp. 3112–3125, Dec. 2013.

[12] A. Kamal, J. Bappy, J. Farrell, and A. Roy-Chowdhury, "Distributed multi-target tracking and data association in vision networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 7, no. 38, pp. 1397–410, Jul. 2016.

Juan C. SanMiguel is an assistant professor at the University Autonóma of Madrid. His research interests include multicamera activity understanding. SanMiguel received a PhD in electrical engineering from the University Autonoma of Madrid. He is a member of IEEE. Contact him at juancarlos.sanmiguel@uam.es

Andrea Cavallaro is a professor of multimedia signal processing and director of the Centre for Intelligent Sensing at Queen Mary University of London. His research interests include smart camera networks and behavior recognition. Cavallaro received a PhD in electrical engineering from the Swiss Federal Institute of Technology (EPFL), Lausanne. He is a member of IEEE. Contact him at a.cavallaro@qmul.ac.uk.