CISTER
Research Center in
Real-Time & Embedded
Computing Systems

Conference Paper

XDense: A Dense Grid Sensor Network for Distributed Feature Extraction

João Loureiro
Raghu R.
Eduardo Tovar

CISTER-TR-150401

2015/05/15

XDense: A Dense Grid Sensor Network for Distributed Feature Extraction

João Loureiro, Raghu R., Eduardo Tovar

CISTER Research Center

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: joflo@isep.ipp.pt, raghu@isep.ipp.pt, emt@isep.ipp.pt

http://www.cister.isep.ipp.pt

Abstract

We propose XDense, a wired mesh grid sensor network architecture tailored for scenarios that benefit from thousands of sensors per square meter. XDense has a scalable network topology and protocol, customizable to application specifics, that enables complex feature extraction in realtime from the observed phenomena by exploiting the communication and distributed processing capabilities of such network topologies. We detail XDense's node and network architecture, protocols, and principles of operation. To demonstrate XDense's potentials, we evaluate it's response time, data traffic metrics and accuracy in the context of detecting fluid dynamic features.

XDense: A Dense Grid Sensor Network for Distributed Feature Extraction

João Loureiro, Raghuraman Rangarajan, Eduardo Tovar

CISTER/INESC-TEC, ISEP, Polytechnic Institute of Porto, Porto, Portugal

{joflo,raghu,emt}@isep.ipp.pt

Abstract. *We propose XDense, a wired mesh grid sensor network architecture tailored for scenarios that benefit from thousands of sensors per square meter. XDense has a scalable network topology and protocol, customizable to application specifics, that enables complex feature extraction in realtime from the observed phenomena by exploiting the communication and distributed processing capabilities of such network topologies. We detail XDense's node and network architecture, protocols, and principles of operation. To demonstrate XDense's potentials, we evaluate it's response time, data traffic metrics and accuracy in the context of detecting fluid dynamic features.*

1. Introduction

The advent of MEMS has enabled new applications to be developed that rely on dense deployments of sensors. In some applications, it has enabled deployments with very high spatial and temporal requirements. That is, resolutions as small as few micrometers of sensor inter space and sampling rates up to kHz. Application examples range from fluid mechanics for flow control in aircrafts [Kasagi et al. 2009], to artificial skins for robotics [Takei et al. 2010] and biomedical devices [Ohta et al. 2009]. For such dense deployments, which may be of thousands of nodes in a few square meters area, sensor network technology faces scalability issues in some key aspects as such as cost, communication time, processing time, power, and reliability. For example, collecting data from the entire network can lead to a data explosion. Processing all this data for feature extraction becomes costly not only in terms of communication, but also computationally. Extraction may also be difficult to achieve in real time, hence prohibiting its use in real time critical applications like closed-loop actuation which leads to tight requirements.

We present XDense as a mesh grid sensor network architecture, tailored to address the challenges of extremely dense sensor deployments. It enables efficient extraction of complex features of the observed phenomena without the need of collecting the data from each individual node centrally. Instead, it allows the user to program node's feature detection and extraction algorithms with respect to the application's objectives. The data is processed in the network in a distributed fashion, and only meaningful data is delivered to the sink, thus reducing transmissions towards it, minimizing congestions, and leading to faster response times.

XDense moves away from traditional contention prone wireless and wired sensor network that uses shared medium to communicate. It resembles more closely Network-on-Chip (NoC) architectures. This is especially true regarding aspects like network topology (mesh grid based on regular structures), routing schemes, timing properties, and on the distributed processing opportunities associated [Kumar et al. 2002]. On the other

hand, we believe that the key differentiating features of our architecture are: (a) the network is not on a single chip, but built on a larger surface that is physically attached, specific to each application scenario (b) the node count is much greater than that of NoC applications, and (c) the network does not deal with shared memory due to a larger coverage area (which imposes different restrictions and opportunities).

Roadmap We first conceptualized XDense with some preliminary simulation results in [Loureiro et al. 2013], and demonstrated its implementability in hardware [Loureiro et al. 2015]. In this paper, we first present in Section 2 some relevant related work, next we expand on that proof-of-concept by first reviewing the basics of the application requirements (Section 3), and then (Section 4) describing the architecture and protocol specifics of the proposed system. We then provide, in Section 5, evaluation results to validate our system by comparing our work with traditional approaches for feature detection. Finally in Section 6 presents the conclusions.

2. Related work

Our work has similarities to boundary estimation and tracking techniques using collaborative sensors found in literature, and a good review of such techniques can be found in [Srinivasan et al. 2012]. Many of such techniques often construct an hierarchical structure, and apply data aggregation functions as the data travels up the hierarchy. Other works (e.g. [Li and Liu 2010]), employ techniques that, similarly to us, rely on local communication between neighbours to detect contours, however were projected for shared medium link layers, and opportunities are limited due to concurrency issues.

Wireless sensor networks (WSN) are suitable for large scale deployments, and effort have gone towards minimizing communication by distributively processing data. For example, in [Luo et al. 2009], the authors explore distributed data compression. Although wireless sensor nodes exceed in complexity, power constraints, size, latency and overhead, and therefore does not meet the application temporal requirements.

Closer to our architecture, a multi modal sensor network was proposed in [Lifton et al. 2002]. The authors present a sensor network with an embedded processor dedicated to each sensor node. Node communicate using an infrared transceiver, with its surrounding neighbors, using a single full-duplex serial. The authors presents a scalable sensor network deployment and propose solutions for data management, localization, data aggregation and routing. But, due to link contentions and collisions, in other words, cost of communication, their research leans more towards WSNs, which are not suitable for the applications we are focused on. The concept was extended, and an alternative topologies were presented in [Paradiso et al. 2004]. Although the network is a wired grid, master-slave communication is utilized, decreasing distributed processing opportunities due to shared links. The number of slaves is also limited, which is not scalable.

3. Dense sensor networks in fluid dynamics

An important objective in fluid dynamics is to characterize airflows and determine its laminar and turbulent characteristics. For example, in an aircraft, a high speed airflow over a wing can present both laminar and turbulent characteristics at the same time at different points of it. Turbulence can be highly undesirable on aircrafts since it increases drag and noise, and consequently fuel expended [Blake 2012].

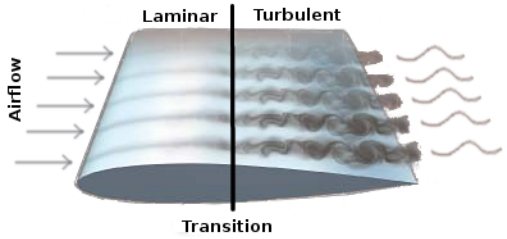


Figure 1. Example scenario: Air flow over a wing exhibiting separation phenomena by switching from laminar to turbulent.

Figure 1 shows an airflow phenomena over a wing surface and illustrates the transition from laminar to turbulent flow. This transition separates laminar flow (which has a more homogeneous speed profile distribution) from turbulent flows (which is composed of coherent structures, such as vortices of chaotic evolution) [Robinson 1991]. An increase in the turbulent region consequently increases drag on the surface of the wing.

There are various techniques available, and being developed, that allow to study a flows' properties by extracting profile data such as speed and temperature distribution [LaRue 1974]. For such profiles, large deployments of sensors may require sensors' inter-space to be smaller than that of the spatial granularity of the observed phenomena (for example, of 100 μm or less) and have high sampling rates (in excess of 10 kHz) [Buder et al. 2008]. Kasagi and others [Kasagi et al. 2009] surveyed micro flow sensors that attempt to meet such requirements. However, despite the effort gone towards sensor development, to the best of our knowledge, no interconnection solutions are found in the literature. Validations are usually done by individually connecting each sensor to channels of an analogue-to-digital converter (ADC) [Buder et al. 2008, Bruinink et al. 2009]. However, due to the dynamic and complex nature of the data in the above discussed approaches, data-processing is usually done offline and is limited to laboratory controlled conditions. This makes their use in real applications difficult.

Based on the above discussion, we state the following requirements of such a system, as being of interest to us, in sensing such phenomena in real time: (a) Efficient data extraction: The network infrastructure should allow efficient extraction of complex information about the phenomena. This should be done without the need of centralized data collection or processing. (b) Real time behaviour: Along with efficiency, the network infrastructure should also be able to respond in a timely manner such that actions can be taken based on the extracted data.

4. Proposed Architecture

The selection of a good topology is a job of fitting network requirements to available technology. A trade-off between cost and performance should always be considered when specifying many aspects, including connection density and length, and the number of GPIOs utilized for communication and signaling. Considering the requisites from Section 3, with the considerations above, we present our architecture in more detail in the following subsections.

4.1. Architecture

Network The architecture consists of a 2D mesh network of sensor nodes and sinks, with point-to-point connections with up to four neighboring nodes, physically located in

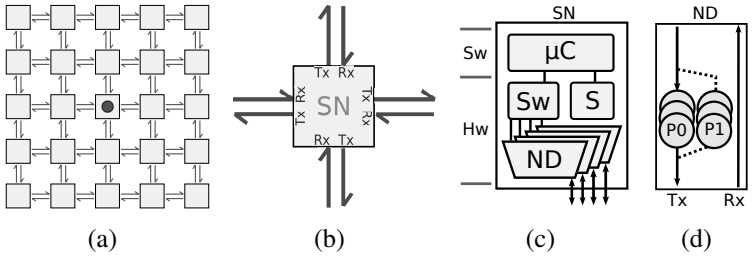


Figure 2. (a) A 5×5 network with a sink in the center; (b) The node's pinout; (c) Node's model architecture: a software layer, which is the application, and the hardware layer, which includes the switch (Sw), the net-device (NE) and the sensor (S); (d) The net-device architecture, with two queues with different priorities in the output channel (Tx), and the input channel on the right (Rx)

four directions (north, south, east, west). Figure 2(a) shows an example scenario of a 5×5 network (24 nodes and one sink) with the sink located in the center. Figure 2(b) shows the node's layout. Figures 2(c) and 2(d) shows in detail the internal architecture of the sensor node. Any set of nodes is a potential communication link from any node to the sink, enabling fault tolerant protocols. Multiple sinks are supported, and any node can be configured to be a sink while deploying and setting up and the network. Each node requires at least one sink in its address space to operate, but multiple sinks can be deployed to increase number addressable nodes (and network size), or to increase redundancy on data collection.

Node: Figure 2(b) shows the main functional blocks of a sensor node (*SN*). Each node can be seen as a system on chip (SoC), with dedicated hardware peripherals and a CPU. The Switch (*Sw*) and the Net-Devices (*NE*) are responsible for communication on the network. The Sensor (*S*) element is the interface with the physical world, and they are the sources of data. These peripherals are interconnected by the software layer, or application layer, which handles the communication protocol and the application specific algorithms.

The network is homogeneous, apart from the sinks, whose sensor interface gives place to an external connection to another communication link. This extra link can be, for example, to a wireless connection to a supervisory system, or even more, to a local actuator for closed loop distributed actuation. Detailed information of each peripheral is provided in the sequence.

Net-device: At the bottom layer in the node's architecture (Figure 2(c)), Net-Devices (*NE*) are the node's hardware peripherals responsible for connecting two distinct nodes through the channel interface. Each node contains four *NEs* that connects them to their four immediate neighbors in the grid. It consists of a full-duplex serial port, with two output queues with different priorities, with fixed sizes. It's design is in Figure 2(d). Different queues are used for different protocols to minimize interference of one over the other. No input queues are used, since packets are processed immediately after being received.

Switch: The switch (Sw) is the interface between *NE* and the application layer. It connects n *NEs* to the application and allows individual or parallel access to any *NE*. The switch works on the basis of analysing the packet headers, and is able to not only send packets from application to *NE*, but also it is able to store and forward packets among *NEs*

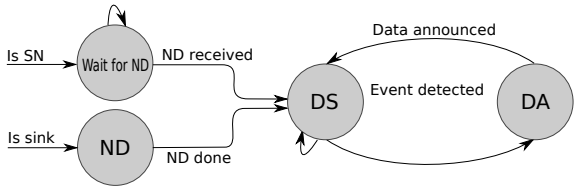


Figure 3. Operational states of the network

without interference of application layer. This should be implemented in hardware, and allows for connection establishment and message forwarding, for better response times. The packets have fixed size for the ease of in-hardware processing.

Sensor: Another of the node's peripheral are the sensors, connected to the Microcontroller (μC) through an analogue-to-digital interface. It is the network data input located at each node in the network. Each node can contain one or more sensors of any nature, according to the phenomena to be observed.

Microcontroller: This where the user's applications run. The micro controller also executes the communication, routing and the operational protocols discussed in the next section. In general, the application layer is responsible for reading from the sensor and deciding what to do with that information, and when, how and to whom to transmit its data through the net-devices. Another basic functionality is to enable the user to program all nodes in the network through a sink by broadcasting the new applications to this layer.

4.2. Protocol

Our protocol is designed using mainly three operational states. Initially, nodes wait until they receive a network discovery packet by the sink. After that, (i) each node collects and distributes sensor's data within its neighborhood, (ii) processes the collected data, and depending on computed results and event detection algorithm policies, (iii) send its results to the closest known sink, in order to announce its findings. These operational states are named: *Network discovery*, *local data-sharing*, and *remote data-announcement* respectively, and from now on referred simply as *ND*, *DS* and *DA*. Figure 3 is a diagram of the principles of operation.

Network discovery: Network discovery (*ND*) is executed once at initialization of the network by the sinks, which sends an *ND* packet carrying the system's settings about sink location (or packet origin), baudrate, sampling rate, and the algorithms used for data processing. Nodes will wait for at least until they get an announcement from at least one sink, which is then added to the know sinks list.

Local data-sharing: After receiving the announcement packet from the sink, the nodes continuously sense the environment for phenomena of interest. They communicate these sensed values with their neighborhood, of size defined by parameter n_{hops}. The neighborhood is a system defined parameter which plays an important role in the system operation. That is, all nodes initially send their values in all four directions, which is then stored and forwarded by the immediate neighboring nodes up to the n_{hops} neighbor.

The value of n_{hops} is selected according to the expected characteristics of the phenomena to observe, in such a way that, each node will send and receive data from up to N nodes around it. The greater the value of n_{hops}, the larger is the neighborhood N.

(a) Original image (b) Binarization and contour tracing (c) Contour smoothing

Figure 4. Process steps for boundary computation described in [Westerweel et al. 2009].

This allows more detailed detections to be made but at the cost of increased overhead and latency. Using n_{hops}, the neighborhood size N is computed as:

$$N = 4 \times \sum_{i=1}^{n_{hops}} i = 2 \times n_{hops} \times (n_{hops} + 1)$$

If the node computes data of interest, it immediately switches to the remote data-announcement (*DA*) state and communicates to the sink its detection. Packets exchanged at the *DS* phase are queued in the lower priority queue *P0*.

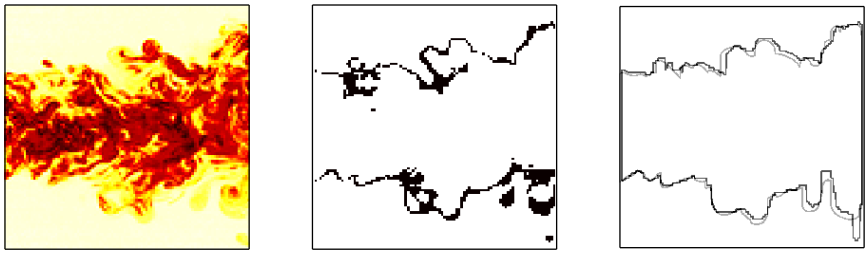
Remote data-announcement: On switching to the *DS* state, a node forwards the packet with information detected towards the closest sink. In turn, the sink receives data from different origins, allowing it to reconstruct the observed phenomena with increasing accuracy and coverage (after each *DS* reception). Packets exchanged during this state are queued in the higher priority queue (*p1*).

5. Evaluation

In [Westerweel et al. 2009] the authors use a camera setup to indirectly measure the speed distribution of an air-jet flow on free air by applying tracers on the flow. They apply image processing techniques to detect such bounding interfaces on captured images. These processing techniques are applied sequentially on an image as follows: First, the image is binarized using a fixed threshold, then all the contours are traced, and finally, only the contour with the greatest area is chosen, with its indentations removed. This process is shown in Figures 4 (a) to (c).

This image processing approach for feature detection is commonly used, and considering that our feature detection goals are the same, we use this approach as a reference to compare our results. We do that as follows: first, we use the same flow image from [Westerweel et al. 2009] as an input to our XDense simulation (this is, the data is superimposed in the simulation of our sensor network grid, and is sampled by each node's sensor). We detect the separation layer, or envelope, which is then compared with the envelope previously computed by processing the same snapshot, and applying the processing steps from [Westerweel et al. 2009].

We utilize an adaptation of the Sobel operator for edge detection [Vincent and Folorunso 2009], widely used the image processing domain. Using this algorithm, nodes performs a 2-D spatial gradient measurement on it's neighboring data, and so emphasizes regions of high spatial frequency that correspond to edges.



(a) Original data (b) Envelope detection (c) Contour smoothing

Figure 5. Processing steps of our network. (a) Is the phenomena as seem by our network, after downsampling the full resolution image to 101×101 pixels. (b) is the envelope detection by the SN with $n_{hops} = 3$, and (c) is the contour smoothing post processing done by the sink in black, compared to 4(c) in gray.

To perform the evaluation, we implemented our network using the Network Simulator 3 (NS-3) that implements all the abstraction of our architecture, for a matrix of sensor nodes with arbitrary parameters. The density of the deployment is chosen according to scales of the phenomena (turbulent structures) that we want to observe. The size of the network is chosen accordingly to the area to be observed. In this case, our region of interest is covered by a network with size $101 \times 101 - 1 = 10200$ SN, and one sink on the center. Our communication protocol uses fixed packet sizes of 7 bytes, and baudrate of 10 Mbps. We vary the neighborhood size of n_{hops} from 1 to 7. Figure 5 shows the results of our simulation, depicting the original snapshot(Figure 5(a)), the events sensed by our setup (Figure 5(b)), and the resulting bounding interface (Figure 5(c)) superimposed with the original output from Figure 4(c). Comparing the two results gives us a measure of the accuracy of our computation.

Using this process, we evaluate the behavior of our system under varying neighborhood size (n_{hops}). We are interested in observing: (i) the accuracy of our bounding interface detection; (ii) the total number of transmissions and (iii) the total time needed to acquire this bounding interface. We then compare it to centralized data acquisition approach.

The CDF in Figure 6 gives a measure of the effect of dimensioning different system parameters (density, neighborhood size). Figure 6 shows that for $2 < n_{hops} < 4$, more that 80% of the snapshot has an error under 3%. Moreover, increasing neighborhood size does not necessarily implies in greater accuracy, but in smaller number of detections. The other way around, the minimum neighborhood size ($n_{hops} = 1$) presents noisy detections, and an intermediate values of n_{hops} might provide the best trade-off.

Figure 7(a) shows the trade-off between the maximum end-to-end delay in terms of transmission time slots (TTS), and mean square error (MSE) of the envelope found. This gives an idea of how responsiveness varies for the different values of n_{hops}, and the impact on the accuracy. With $n_{hops} = 3$ MSE is minimum, with a minimum cost in time, when compared to $n_{hops} = 2$ which presents the best response time, but with higher MSE. Figure 7(b) presents the number of transmissions, therefore providing a picture of the actual load on the network. It shows that, with increase in neighborhood size, the number of global transmissions (*DA*) goes down (although the local transmissions (*DS*) increases accordingly). This in turn affects the total transmissions time (as can be seen

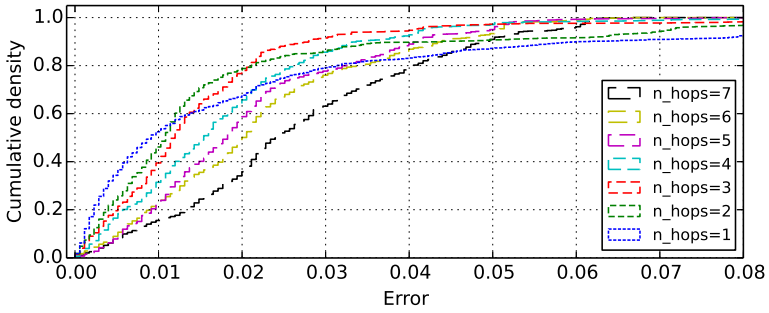


Figure 6. Cumulative density function for different values of n_{hops}, of the error between the reference envelope, and the detected envelope.

back in Figure 7(a)).

In another simulation, considering that centralized data acquisition and processing is utilized ($n_{hops} = 0$), we obtained the time required for all the nodes to send their data to the sink, which can be computed as $(101 \times 101 - 1)/4 * t_{pck} = 2550$ TTS. Which correspond the number of nodes in each quadrant, times the packet duration. The same results would be achieved if any kind of shared medium with ideal TDMA was utilized to interconnect one quarter of the nodes located at one quadrant.

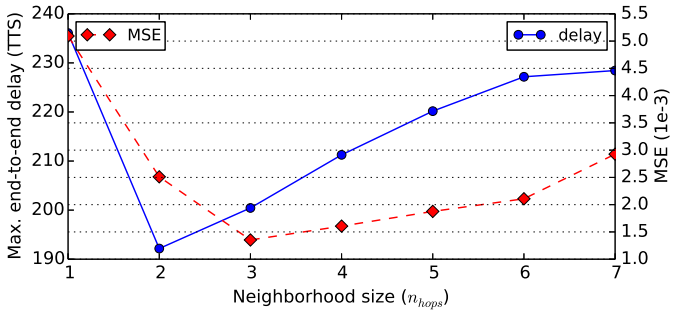
Maximum end-to-end delay dropped in the order of magnitude of 10, when distributed data processing is utilized for complex data extraction, compared to simple centralized raw data extraction.

6. Conclusion

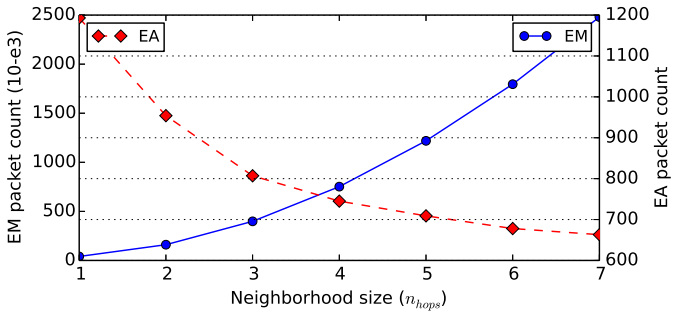
Dense sensor networks, like our proposed XDense, are going to be increasingly used in application scenarios which require realtime data. Combined with novel feature detection techniques systems can then recreate the phenomena, present in the application scenarios, and use this information for actuations. We evaluated the XDense architecture and protocols work with performance metrics on accuracy, timeliness and network usage and showed the tradeoffs that have to be made among the parameters (such as, larger neighborhood leads to increased accuracy but also increased traffic). It is important to understand the underlying issues that affect the performance of XDense. One issue is that of dimensioning XDense based on the requirements imposed by the application scenario. We showed that dimensioning issues such as node density and neighborhood size affect the performance of the system and depend closely on the application scenario.

Another issue are the feature detection techniques which strongly affects the performance. We used one base technique for evaluation and we intend to investigate further techniques for comparative analysis. It is important to note that the simplicity of the algorithms utilized, allow for practical construction of such networks using currently technology, as demonstrated earlier in [Loureiro et al. 2015]. For example, with COTS microcontrollers [1] ,or a SoC solution could be developed for it, to be low cost.

[1] Atmel ATSAM4N8A [Atmel] is a low powered Cortex-M4-based microcontroller, that runs at up to 100MHz and features 512KB of Flash and 64KB of SRAM. Peripherals include five USARTs, for fast serial communication, as well as an 8-channel 12-bit ADC, More than 8 DMA channels allows fast receiving and transmitting at each port.



(a)



(b)

Figure 7. (a) Trade-off between mean square error (MSE) and maximum end-to-end delay (TTS) for different values of n_{hops}, and (b) is the total number of transmissions for the different protocols, for the same values of n_{hops}

Acknowledgments

This work was partially supported by National Funds through FCT/MEC (Portuguese Foundation for Science and Technology) and when applicable, co-financed by ERDF (European Regional Development Fund) under the PT2020 Partnership, within project UID/CEC/04234/2013 (CISTER Research Centre); also by FCT/MEC and ERDF through COMPETE (Operational Programme 'Thematic Factors of Competitiveness'), within project FCOMP-01-0124-FEDER-020312 (SMARTSKIN); by FCT/MEC and the EU ARTEMIS JU within project ARTEMIS/0004/2013 - JU grant nr. 621353 (DEWI, www.dewi-project.eu); by the government of Brazil through CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico.

References

Atmel. http://www.atmel.com/devices/sam4n8a.

Blake, W. K. (2012). *Mechanics of Flow-Induced Sound and Vibration V2: Complex Flow-Structure Interactions*, volume 2. Elsevier.

Bruinink, C., Jaganatharaja, R., de Boer, M., Berenschot, E., Kolster, M., Lammerink, T., Wiegerink, R., and Krijnen, G. (2009). Advancements in technology and design of biomimetic flow-sensor arrays. In *Micro Electro Mechanical Systems, 2009. MEMS 2009. IEEE 22nd International Conference on*, pages 152–155.

Buder, U., Petz, R., Kittel, M., Nitsche, W., and Obermeier, E. (2008). Aeromems polyimide based wall double hot-wire sensors for flow separation detection. *Sensors and Actuators A: Physical*, 142(1):130–137.

Kasagi, N., Suzuki, Y., and Fukagata, K. (2009). Microelectromechanical systems-based feedback control of turbulence for skin friction reduction. *Annual review of fluid mechanics*, 41:231–251.

Kumar, S., Jantsch, A., Soininen, J.-P., Forsell, M., Millberg, M., Oberg, J., Tiensyrja, K., and Hemani, A. (2002). A network on chip architecture and design methodology. In *VLSI, 2002. Proceedings. IEEE Computer Society Annual Symposium on*, pages 105–112. IEEE.

LaRue, J. C. (1974). Detection of the turbulent-nonturbulent interface in slightly heated turbulent shear flows. *Physics of Fluids (1958-1988)*, 17(8):1513–1517.

Li, M. and Liu, Y. (2010). Iso-map: Energy-efficient contour mapping in wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 22(5):699–710.

Lifton, J., Seetharam, D., Broxton, M., and Paradiso, J. (2002). Pushpin computing system overview: A platform for distributed, embedded, ubiquitous sensor networks. In *Pervasive Computing*, pages 139–151. Springer.

Loureiro, J., Gupta, V., Pereira, N., Tovar, E., and Rangarajan, R. (2013). Xdense: A sensor network for extreme dense sensing. *Proceedings of the Work-In-Progress Session at the 2013 IEEE Real-Time Systems Symposium - RTSS*, pages 19–20.

Loureiro, J., Rangarajan, R., and Tovar, E. (2015). Demo abstract: Towards the development of xdense, a sensor network for dense sensing. *12th European Conference on Wireless Sensor Networks - EWSN*, page 23.

Luo, C., Wu, F., Sun, J., and Chen, C. W. (2009). Compressive data gathering for large-scale wireless sensor networks. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 145–156. ACM.

Ohta, J., Tokuda, T., Sasagawa, K., and Noda, T. (2009). Implantable cmos biomedical devices. *Sensors (Basel, Switzerland)*, 9(11):9073.

Paradiso, J. A., Lifton, J., and Broxton, M. (2004). Sensate media-multimodal electronic skins as dense sensor networks. *BT Technology Journal*, 22(4):32–44.

Robinson, S. K. (1991). Coherent motions in the turbulent boundary layer. *Annual Review of Fluid Mechanics*, 23(1):601–639.

Srinivasan, S., Dattagupta, S., Kulkarni, P., and Ramamritham, K. (2012). A survey of sensory data boundary estimation, covering and tracking techniques using collaborating sensors. *Pervasive and Mobile Computing*, 8(3):358–375.

Takei, K., Takahashi, T., Ho, J. C., Ko, H., Gillies, A. G., Leu, P. W., Fearing, R. S., and Javey, A. (2010). Nanowire active-matrix circuitry for low-voltage macroscale artificial skin. *Nature materials*, 9(10):821–826.

Vincent, O. and Folorunso, O. (2009). A descriptive algorithm for sobel image edge detection. In *Proceedings of Informing Science & IT Education Conference (InSITE)*, pages 97–107.

Westerweel, J., Fukushima, C., Pedersen, J. M., and Hunt, J. (2009). Momentum and scalar transport at the turbulent/non-turbulent interface of a jet. *Journal of Fluid Mechanics*, 631:199–230.