Apply Geometric Duality to Energy Efficient Non-Local Phenomenon Awareness using Sensor Networks

Jie Liu, Patrick Cheung, Leonidas Guibas and Feng Zhao[†]

ABSTRACT

A powerful concept to cope with resource limitations and information redundancy in wireless sensor networks is the use of collaboration groups to distill information within the network and to suppress unnecessary activities. When the phenomena to be monitored have large geographical extents, it is not obvious how to define these collaboration groups. This paper presents the application of geometric duality to form such groups for sensor selection and non-local phenomena tracking. Using a dual-space transformation, which maps a non-local phenomenon, *e.g.*, the edge of a half-plane shadow, to a single point in the dual space and maps locations of distributed sensor nodes to a set of lines that partitions the dual space, one can turn off majority of the sensors to achieve resource preservation without losing the detection and tracking accuracy. Since the group such defined may consist of nodes that are far away in the physical space, we propose a hierarchical architecture that uses a small number of computationally powerful nodes and a massive number of resource constrained motes. By taking advantage of the continuity of physical phenomena and the duality principle, we can greatly reduce power consumption in non-local phenomena tracking and extend the life time of the network.

1. INTRODUCTION

A wireless network of sensors can cover a large geographical region, and hence can be used to detect and track non-local phenomena which cannot be captured by any individual sensor. Because of its dense spatial sampling and multi-modality sensing, the network can assemble information from spatially diverse sources to improve signal/noise ratio. The redundancy in the network can ensure a certain degree of robustness against node failures. The network may be quickly

[†] Jie Liu and Patrick Cheung are with Palo Alto Research Center (PARC), 3333 Coyote Hill Rd, Palo Alto, CA 94304.
{jieliu, pcheung}@parc.com.
Leonidas Guibas is with Department of Computer Science, Stanford University, Stanford, CA, 94305. guibas@cs.stanford.edu
Feng Zhao is with Microsoft Research, One Microsoft Way, Redmond, WA 98052. zhao@microsoft.com

deployed for a particular application, and the ubiquity and low-cost nature of the MEMS micro-sensors can potentially give users unprecedented access to real-time situational information.

While the sensor data is local to each node, the information content to be extracted from the network can be global which must be obtained through collaboration among the nodes. Let us consider a scenario of tracking chemical plumes using ad hoc, just-in-time deployment of sensor nets:

> *The Valley Authority just declared a region-wide emergency: A large-scale hazardous chemical gas leak occurred ten minutes ago near the town of XYZ. The National Guard has been activated to evacuate nearby towns, and to close roads and bridges. To get a real-time situational assessment of the extent and movement of the gas release and aid the evacuation, a SWAT Team is called in. Three unmanned aerial vehicles* (*UAVs*) *are immediately launched from an open field 15 miles south of the accident site, each carrying 1,000 wireless chemical sensing nodes* (*see Figure* 1). *Upon flying over the vicinity of the accident site, the sensor nodes are released. The nodes self-organize into an ad hoc network once get to the ground and relay the tracking result back to a base station nearby: Where is the plume? How big is it? What is the shape? How fast is it moving?*

Figure 1. Tracking chemical plumes using ad hoc distributed sensors.

In this example, each sensor only has limited information such as whether certain chemical elements exist at the sensing spot, whereas the global information such as the shape of the plume and its motion need to be determined collaboratively by many sensors. In addition, because of limited resources (*e.g.*, battery power and communication bandwidth), such processing and communication must be carefully managed.

The ultimate way for reducing energy consumption of a sensor node is by turning it off. Modern wireless sensor hardware platforms usually have low-power sleeping modes, in which parts of the processor, sensors, and wireless communication circuits are turned off to preserve power. For example, in Berkeley/Crossbow MICA2 motes[2], one second of sleeping can save enough energy for sending more than 70 packets, or performing ~70K CPU instructions. Sensor nodes can be turned back to an active mode by timers or receiving wakeup packets using, for example, carrier detect circuits. Thus, the art of the design is to selectively put sensors into sleep without losing the application performance. This has traditionally been dealt with by adjusting the sampling and communication rate of the sensor nodes. In this article, we push this philosophy further. That is to use application-specific physical constraints to select nodes to be activated.

In a large-scale dense sensor network, it is sometimes desirable to process the information collected by the sensors within the network rather than sending raw data to a central server [12]. There are primarily two reasons. First is that the information collected by a sensor network is highly redundant. Considering that typical physical phenomena only have limited ranges of impact, most sensor data from a network contain no information about the phenomena of interest. Sending them out is simply a waste of resources. Secondly, given our current technologies on wireless sensor node design, wireless communication is still limited on bandwidth and expensive on power consumption. So, even for meaningful sensor data, it is desirable to summarize the sensor data locally before sending to the edge of the network.

The set of sensor nodes (or computing agents in general) that collaboratively process the data within the network is called a collaboration group. There are many ways to define collaboration groups, for example, based on geographical locations or data of interests [6]. For instance, let's consider a point signal source to be tracked at (x, y) and its radius of impact r (defined by certain signal-to-noise ratio), then it is intuitive to define a collaboration group consists of all nodes within the circle defined by (x, y) and r. Of course, since the true location of the signal source is unknown, one may have to estimate its location and maintain the group accordingly [7]. However, when the phenomenon is non-local, as in the case of the chemical plume, it is not so obvious how to find the minimum set of sensors that contribute to tracking the phenomenon.

This article reviews the dual-space transformation in computational geometry and applies it to the tracking of non-local phenomena [5]. We consider a dense sensor network so that the edge of a non-local phenomenon, modeled as shadow, can be piece-wise approximated by straight lines. We study edge detection and tracking problems for a 2-D continuous shadow over the senor field. A dual-space representation maps non-local line segments into a set of points in an appropriately parameterized configuration space. We then show how motion constraints from the target shape and dynamics can be

[2] Available from Crossbow Technology Inc., http://www.xbow.com

exploited to activate only those sensors relevant to the current configuration. This algorithm can serve as a building block in a scalable hierarchical architecture that overcomes the communication and computation limitations. In our experiment of a shadow tracking using 16 motes, we have observed that only 28% sensors on the average are awake at any given time.

2. DUAL-SPACE-BASED EDGE DETECTION

What does a dual-space representation buy us? The geometric duality described below allows us to map a seemingly non-local phenomenon (the position of the shadow edge), into a local attribute in the dual space. This allows the sensor nodes to be ordered according to how "close" they are relative to the frontier of the object motion and simplifies the sensor activation procedure. If the sensor activation algorithm were implemented in the primal space, without using the dual-space transformation, each sensor node will have to reason about its distance to the object edge relative to other sensor nodes and the motion of the object, a fairly complex geometric problem to solve. We consider a half-plane shadow in this section, and generalize it to other shapes later.

2.1 Dual-Space Transformation

Let us consider a line in a 2D space (called the *primal space*): $y = \alpha \cdot x + \beta$, which is uniquely defined by two parameters α and β . To represent this line through this pair of parameters, we can use the point $(-\alpha, \beta)$ in another 2D space (called the *dual space*)[3]. Similarly, a point in the primal space (a, b) uniquely defines a line in the dual space: $\varphi = a \cdot \theta + b$. This 1-to-1 mapping, as shown in Figure 2, is one form of a *dual-space transformation*[4] [1][8].

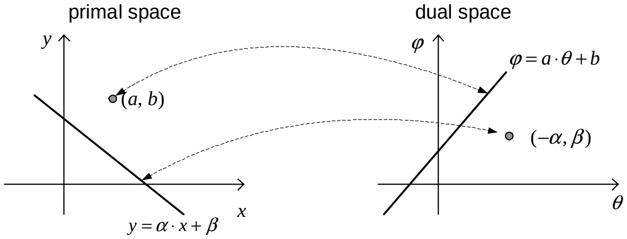


Figure 2. The mapping between the primal space and the dual space.

A dual-space transform has several useful properties, which follow immediately from the definition:

[3] We use $-\alpha$ instead of α in the dual space so that some properties are easy to derive later on.

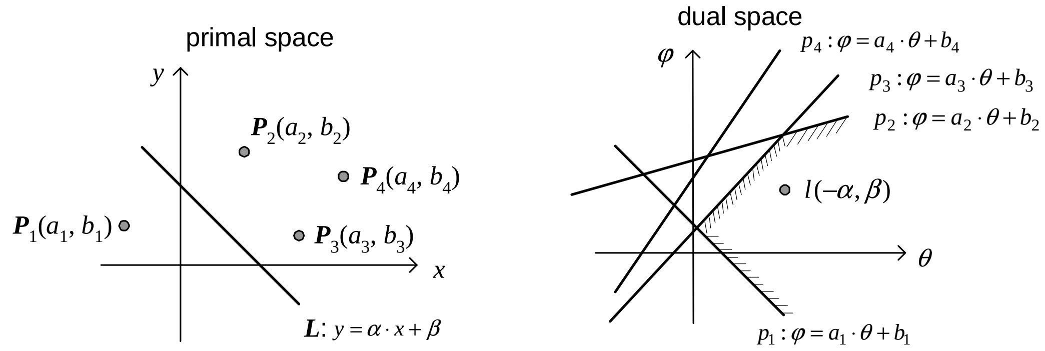
[4] It is also called *Hough Transformation* in some literatures.

A. If, in the primal space, a point (a,b) is *on* a line $y=\alpha\cdot x+\beta$, then, in the dual space, the corresponding line $\varphi=a\cdot\theta+b$ goes through the corresponding point $(-\alpha,\beta)$, and vise versa.

B. If, in the primal space, a point (a,b) is *above* a line $y=\alpha\cdot x+\beta$, i.e., $b>\alpha\cdot a+\beta$, then in the dual space, the corresponding line $\varphi=a\cdot\theta+b$ is *above* the corresponding point $(-\alpha,\beta)$, i.e. $\beta<-\alpha\cdot a+b$. Similar results hold for the *below* relation.

C. If, in the primal space, a line $y=\alpha\cdot x+\beta$ performs a continuous motion, including rotation and translation, the corresponding point $(-\alpha,\beta)$ performs a continuous motion in the dual space.

For example, consider a set of points {$\boldsymbol{P}_1$, …, $\boldsymbol{P}_4$} and one line $\boldsymbol{L}$, in the primal space, as shown in Figure 3(a), whose corresponding dual-space representations, {p_1, …, p_4} and l, are shown in Figure 3(b).


primal space
y
x
$\boldsymbol{P}_2(a_2, b_2)$
$\boldsymbol{P}_4(a_4, b_4)$
$\boldsymbol{P}_1(a_1, b_1)$
$\boldsymbol{P}_3(a_3, b_3)$
$\boldsymbol{L}$: $y=\alpha\cdot x+\beta$
dual space
φ
$p_4:\varphi=a_4\cdot\theta+b_4$
$p_3:\varphi=a_3\cdot\theta+b_3$
$p_2:\varphi=a_2\cdot\theta+b_2$
$l(-\alpha,\beta)$
θ
$p_1:\varphi=a_1\cdot\theta+b_1$

(a) (b)

Figure 3. A set of points and a line in the primal space and their representations in the dual space.

In Figure 3(b), the lines {p_1, …, p_4} define a *line arrangement* that partitions the dual space into a set of convex polygons, called *cells* [1], [8]. The boundaries of these cells are line segments lying on the lines {p_1, …, p_4}. Obviously, some cells are completely bounded, while others extend to infinity. The dual of a primal line $\boldsymbol{L}$ is a point l that must be contained in one of the cells (in this example, the shaded cell in Figure 3(b)), unless it is on a cell boundary. By abusing notations, let us use $l<p$ to denote that the point l is below the line p in the dual space; then, the shaded cell in Figure 3(b) contains all points l satisfying $l>p_1, l<p_2$, and $l<p_3$.

When the line $\boldsymbol{L}$ in the primal space moves, l moves in the dual space accordingly. As long as $\boldsymbol{L}$ does not rotate across the vertical direction or intersect any point in the primal space, l stays in the cell defined by the above set of constraints in the dual space. Furthermore, in the dual space, l can enter other cells only if it crosses one of the cell boundaries, including, conceptually, a boundary at infinity. In particular, as shown in Figure 3(b), *l* cannot intersect p_4, before it crosses one of the current cell boundaries. This observation is the key for our sensor selection scheme: *if {$\boldsymbol{P}_1$, …, $\boldsymbol{P}_4$} are the positions of four sensors and **L** is the boundary of a half-plane shadow, then $\boldsymbol{P}_4$ can be safely turned off as long as none of $\boldsymbol{P}_1$, $\boldsymbol{P}_2$ and $\boldsymbol{P}_3$ senses a transition.*

2.2 Shadow Edge Estimation and Sensor Selection

For a half-plane shadow, by using the dual space transformation, we can estimate its edge by solving the set of constraints imposed by the particular sensor readings. Using that information, we can further determine the group of sensors at the "frontier", i.e., the ones that may detect a transition next when the shadow moves. For ease of discussion, we use light sensors as a metaphor for the sensing model. Obviously, the mechanism applies to any sensing models that give binary readings through quantization. Let **0** represent a *dark* reading at a sensor, and **1** represent a *light* reading. Then at any time, the sensor field gives a vector of readings consists **0**'s and **1**'s. The goal is to identify the set of sensors that bounds the shadow edge, and thus estimate the shadow location and turn off the nodes that are irrelevant at this time.

Using the dual space transform, each sensor defines a line in the dual space, and the edge of the shadow is a point. Thus, the problem is converted to determining the cells that are consistent with current sensor readings; these are the cells that may contain the dual of the shadow edge. Note that the constraints in the dual space are in forms of *above* and *below* relations. The same vector of sensor reading may yield two possible answers for the location of the shadow; i.e. the shadow is above its edge or the shadow is below its edge. For example, the two shadow locations, shown in Figure 4 (a) and (b), yield the same sensor readings, [**0**, **1**, **1**, **1**], on {$\boldsymbol{P}_1$, …, $\boldsymbol{P}_4$}.

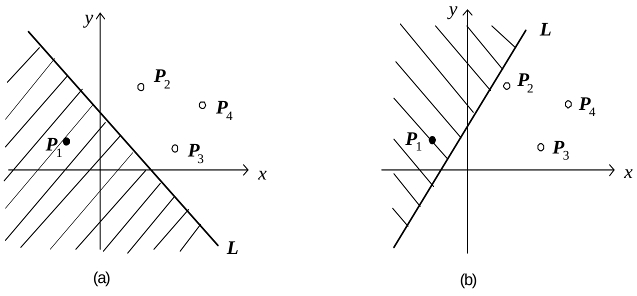


Figure 4. Two different configurations that yield the same sensor reading.

Moreover, in the dual space, situations in Figure 4 (a) and (b) have different representations. The representation of (a) is exactly the same as in Figure 3(b), while the representation of (b) is illustrated in Figure 5.

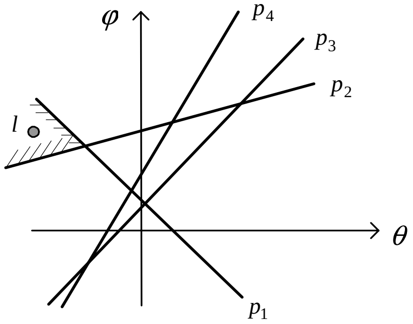


Figure 5. The dual space representation for the situation shown in Figure 4 (b).

The cells that are consistent with the set of sensor readings can be computed via linear programming over the results of a topological sweep. A topological sweep algorithm computes the segments created by the intersections of the set of lines, and their relative locations in terms of adjacency and direction. Using that arrangement, one may first assume that the shadow is above the line and use a linear programming algorithm to find the cell that is consistent with the set of sensor readings. A similar computation can be performed under a below assumption. Sometime, only one of the assumptions yields a feasible answer; sometimes both of them do. The details of the topological sweep and cell finding algorithms are out of the scope of this article. Interesting readers are referred to [2], [5], and [9]. But it is worth noticing that these are all centralized algorithms that require the location knowledge of all points.

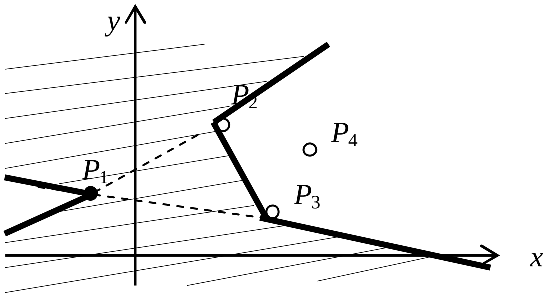


Figure 6. An estimation of the location of the shadow edge, subject to the resolution of the sensor field.

Once we find the cells that satisfy the sensor reading constraints, the corners of the cells, corresponding to several lines in the primal space, define the “extreme” positions that the edge of the shadow could be in. Each pair of lines, intersecting at a corner point, together with the corresponding constraints on the lines, gives a wedge in the primal space. For example, in Figure 3(b), the corner of the intersection of p_1 and p_3 together with the relations that the cell is above p_1 and below p_3 defines a wedge that contains all lines that are above P_1 and below P_3 in the primal space. Similarly, the intersection of p_2 and p_3 and the fact that the cell is below p_2 and below p_3, gives a wedge that contains all lines that are below P_2 and P_3. For each cell, the intersection of these wedges is the estimate of the shadow edge. That is, the edge of the shadow must be within that wedge under a certain assumption, *e.g.* dark means below. Of course, if there are two consistent cells in the dual space, the union of each shadow edge estimate gives the overall answer. For example, mapping back the cells in Figure 3(b) and Figure 5, we get a wedge that is shown in Figure 6. In general, the size of the cells in the dual space dictates the "freedom" of the edge in the primal space. Thus, the smaller the cells are, the more accurate estimation one can get.

Furthermore, at any time, the dual of the edge of a shadow can possibly be in at most two cells. The sensors corresponding to the boundaries of these cells are the sensors at the "frontier," in the sense that no matter how the shadow moves, it must cross one (or more) of these sensors before crossing any other sensors. So, only those sensors corresponding to the lines bounding the cell(s) need be kept active, therefore brings the energy savings. It can be shown that in any line arrangement the expected number of lines bounding an "average" cell is at most four [1] (in the model where all cells are equally likely), independent of the overall number of sensors present. Thus, we can expect the number of sensors that need to be active at any moment to be very small. In a dense sensor field this may lead to substantial energy savings over time without losing the tracking quality.

2.3 Distributed Sensor Management

The method described above on finding the cells in the dual space is static, and could be applied without knowing the motion history of the shadow. If we take advantage of the fact that the motion of a shadow is continuous, so that the dual of its edge can only move from one cell to an *adjacent* cell, then the linear programming part of the computation does not have to be performed after the system is properly initialized. For example, if the cell {C, G, H} in Figure 7 contained the dual of the shadow edge, and sensor p_2 just flipped its reading, then it is clear that {H, K, I} is the new cell. If the line arrangements are precomputed and stored in the sensor nodes, then finding the new cell is simply a table lookup.

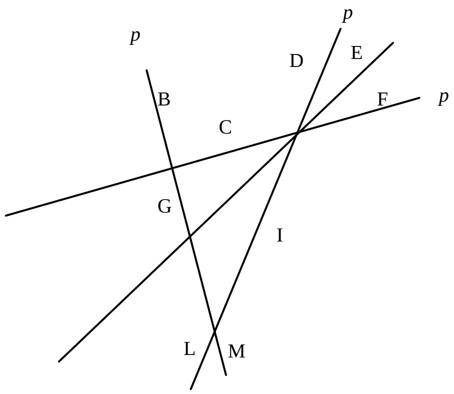


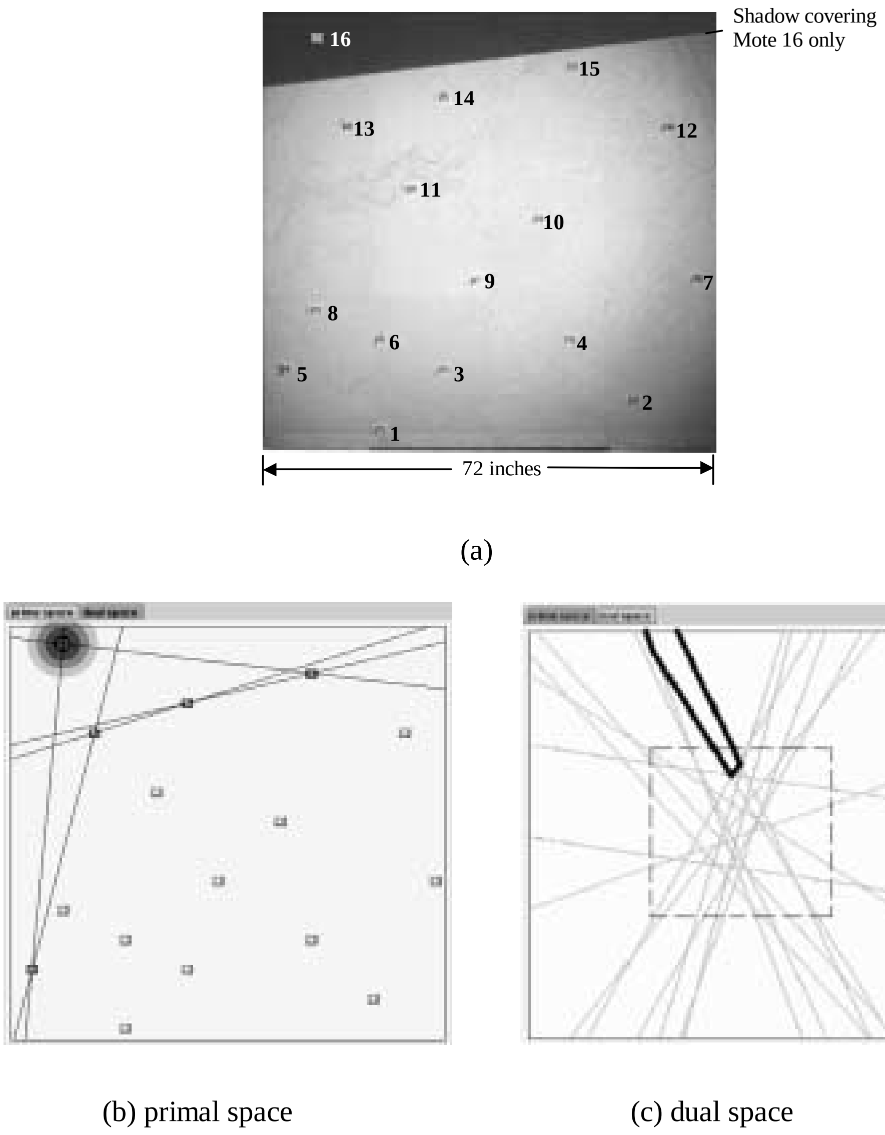
Figure 7. A line arrangement of four lines.

This process is so simple that it amendable for a distributed implementation on tiny sensors with very limited memory and processing power. After giving each cell a unique ID, each sensor node only needs to remember those dual space cells that are incident on the line representing it. These cells correspond to the concept of the *zone* [1] of a line in a line arrangement, and it is known that the storage required is only linear in the total number of lines (nodes). The network can be initialized by having all sensors agree on the same cell ID. The sensors nodes that know nothing about that cell can go to the sleep mode. When one of the sensors on guard notices a flip of sensor reading, it wakes up sensor nodes that are new in the new cell, and broadcasts to all awaken sensors with the new cell ID. The sensor nodes forming the old cell but are no longer in the new cell can turn itself into the sleep mode.

2.4 *A Lab Experiment*

We have built an experiment testbed [5] to validate the shadow tracking algorithms and to demonstrate the benefits of sensor management using a network of Rene motes with light sensors [3]. The Rene motes are an older generation of Berkeley motes that have an 8-bit Atmel AT90LS8535 processor running at 4 MHz, 8Kbytes of flash memory, and 512Bytes of SRAM. One of the A/D channels is connected to a light sensor. The rate of sensor reading is programmed to be 8Hz. RF communications between motes are carried on the 916 MHz band. Data are communicated at up to 10K bits per second.

The experiment is performed on a vertical 6 foot by 6 foot board to allow an overhead viewgraph projector to illuminate the entire platform. Sixteen motes are mounted on the board at randomized but known location. Figure 8 (a) shows the photograph of the board and the motes, which are numbered from 1 to 16.



Shadow covering Mote 16 only

(a)

(b) primal space

(c) dual space

Figure 8. A testbed contains 16 motes mounted on a board. A half-plane shadow is cast onto the board. (b) shows the detection and estimation of the shadow edge. (c) shows the dual-space representation of the corresponding cell. Motes 16, 15, 14, 13, and 5 are the current frontiers for detection.

Figure 8 (a) and (b) show a scenario where the shadow covers mote 16. Five of the motes are elected as the frontier of the shadow tracking, which is obtained from the boundaries of the cell in the dual space (shown in Figure 8 (c)). In Figure 8 (b), the lines connecting the frontier motes depict the extreme position and orientation of the edge of the shadow. In other words, the shadow's edge must lie within the bounds of all the five lines.

In this particular setup, the 16 lines create a total of 102 cells, which cover all possible positions of the shadow edge. The number of boundaries for each cell indicates the number of motes that need to be activated when the dual of the shadow edge falls in that cell. In almost all cases (>97%), only 3 to 5 out of the 16 nodes need to be active at the same time. In other words, less than 30% motes are active at any time on average. The rest can be put into sleep to preserve power.

3. A TWO-TIER ARCHITECTURE

There are still two potential difficulties for the above sensor management scheme: 1) obtaining the cell configuration is computationally expensive and the algorithm is centralized; and 2) sensor nodes that forming a particular cell in the dual space may be far away in the primal space. There may not be a direct communication link to wake up a remote sensor node if all not-on-guard nodes are sleeping. To overcome these limitations, we present a two-tier sensor management architecture over a virtual grid. The ideas of two-tier architectures and virtual grids have been shown helpful for scalable network services such as *ad hoc* routing [10] [4] and data dissemination [11]. Here we apply similar ideas to sensor management.

The two-tier hierarchical architecture consists of a large number of feather-weight sensor motes and a small number of more powerful (in terms of computation) nodes serving as cluster heads[5]. In order to maximize the life time of the network, both motes and the cluster heads may have a deep sleep mode that consumes almost no power. We assume that they can also be waked up wirelessly via carrier sensing. As shown in Figure 9, the sensor field is virtually divided into a regular grid. Each grid square (whose shape may not be exactly a square) has one cluster head (shown as a star) and a set of motes (shown as the circles) deployed in an *ad hoc* fashion. Both cluster heads and motes have wireless communication capabilities. In addition to communicating with the motes in its square, a cluster head can also communicate with other cluster heads in adjacent squares to create a mesh network topology (depicted as the hashed lines in Figure 9). The size of the grid squares is small enough that a broadcast from one mote can be heard by all the motes (and the cluster head) in the square, and it is big enough to minimize the total number of squares in the field. The cluster heads, which may not be equipped with any sensor, can be placed arbitrarily in the grid. We also assume that all motes are localized, presumably with the help of the cluster heads.

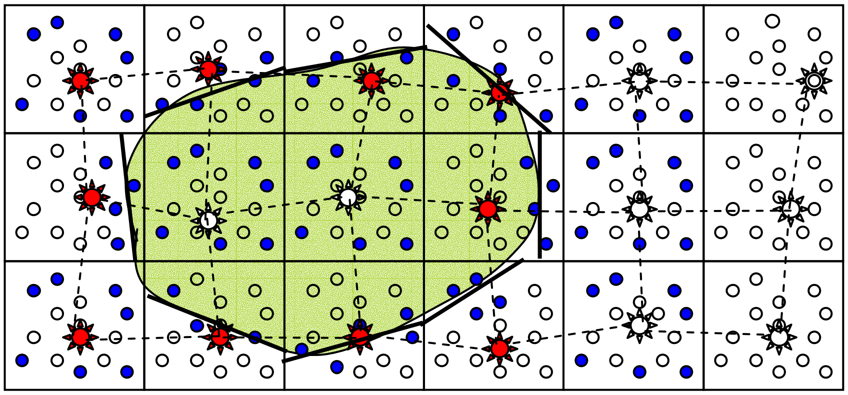


Figure 9. Using a hierarchical architecture to track non-local phenomena.

Suppose that the boundary of the physical phenomena to be detected and tracked, for example, a chemical or biological cloud, is smooth such that it can be approximated as straight lines in each grid square. Then, a dual-space-based detection, tracking, and sensor management scheme can be applied as following:

- **Intra-cluster initialization**: Once deployed, the motes send the cluster head their locations and current sensor readings. Using that information, the cluster head can perform a topological sweep algorithm and compute all cells in the dual space and the current cell(s). This configuration is then sent back to the motes in the form of a table.
- **Inter-cluster initialization:** A grid square is called *covered* by the shadow if all its sensors have dark readings; it is called *uncovered* if all sensors have light readings; and it is called *partially covered* otherwise. The coverage properties are sent by the cluster heads to their direct neighbor cluster heads. For partially covered squares, their cluster heads also send the current estimate of the shadow edge. Upon receiving these messages, a cluster head decides whether to activate its cluster using the *grid activation criteria*:
    - **A**: A partially covered square is always *active* in the sense that the frontier motes continuously sense the shadow.
    - **B**: A covered or uncovered square is active only if at least one of its neighbors estimates that the edge of the shadow possibly intersects the grid boundary between them.

  For an inactive square, all nodes turn to a sleep mode.

[5] In fact, with sufficient computational power, the sensing nodes and the cluster heads can be of the same kind, and their roles can be dynamically assigned

- **Tracking**: An active square performs a tracking and sensor selection scheme using the algorithm described in section 2.3. Motes that are not at the frontier of tracking go to the sleep mode. Once the current dual-space cell in a square changes, its cluster head is waked up by the mote that sense the change. The cluster head tells all direct neighboring cluster heads about its new estimate of the shadow edge. The receiving cluster head then determines whether it needs to activate its cluster by the grid activation criteria.

By using this scheme, only the sensors that are absolutely necessary in detecting and tracking the non-local phenomena are active. Note that one of the assumptions we made in the system is that the edge of the shadow can be piece-wise approximated as straight lines in each grid square. While the straight line approximation assumption is in general true, the extent of the lines may not be aligned with the boundaries of the grid squares. A future work is to dynamically create the clusters to adapt to complex shapes.

4. CONCLUSION

In large-scale dense sensor networks, the scalability requirements suggest we organize sensor nodes and computation into collaboration groups for in-network information processing. This article reviews the dual-space transformation principle and applies it to the tracking the boundary of a non-local shadow. By converting non-local phenomena into localized representations and solves the problem in an appropriate configuration space, the sensor nodes at the frontier can be easily identified. Thus other nodes can be safely switched to a power saving mode. To scale up this approach, we proposed a hierarchical heterogeneous network with more powerful nodes performing dual-space transformation and long range communication and tiny motes performing fine-grained edge detection and tracking.

ACKNOWLEDGEMENT

This work is supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract number F30602-00-C-0139 through the Sensor Information Technology Program. The authors would also like to thank Olaf A. Hall-Holt for helping on the topological sweep software, and thank Jim Reich and Juan Liu for inspiring discussions during this work.

after deployment to improve robustness and to share load.

5. REFERENCES

[1] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Springer-Verlag, Berlin, 1997.

[2] H. Edelsbrunner and Leonidas J. Guibas, “Topologically sweeping an arrangement,” *J. Comput. Syst. Sci.*, vol. 38, 1989, pp. 165-194.

[3] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, Kristofer Pister. “System architecture directions for network sensors,” in Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX) Cambridge, MA, Nov. 2000.

[4] P. Krishna, N. Vaidya, M. Chatterjee, and D. Pradhan, “A cluster-based approach for routing in dynamic networks,” ACM SIGCOMM *Computer Communication Review*, pages 49--65, April 1997.

[5] Jie Liu, Patrick Cheung, Leonidas Guibas, and Feng Zhao, “A Dual-Space Approach to Tracking and Sensor Management in Wireless Sensor Networks,” The 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002), Atlanta, GA, Sept. 28, 2002, pp. 131-139.

[6] Jie Liu, Maurice Chu, Juan Liu, James Reich, and Feng Zhao, “State-Centric Programming for Sensor-Actuator Network Systems,” IEEE Pervasive Computing, vol. 2, No. 4, October-December, 2003, pp.50-62.

[7] Juan Liu, Jie Liu, James Reich, Patrick Cheung, and Feng Zhao, “Distributed Group Management for Track Initiation and Maintenance in Target Localization Applications,” The 2nd International Workshop on Information Processing in Sensor Networks (IPSN’03), Palo Alto, CA, April 2003, Lecture Notes in Computer Science (LNCS 2634), Springer-Verlag Heidelberg, pp. 113-128.

[8] J. O'Rourke, *Computational Geometry in C*, 2nd Ed. Cambridge University Press, 1998

[9] Eynat Rafalin, Diane Souvaine, and Ileana Streinu, “Topological sweep in degenerate cases,” in Proceedings of the 4th Workshop on Algorithm Engineering and Experiments (ALENEX’02), San Francisco, CA, January, 2002.

[10] Ya Xu, John Heidemann, and Deborah Estrin, “Geography-informed Energy Conservation for Ad Hoc Routing,” In Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001), Rome, Italy, pp. 70-84.

[11] Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, Lixia Zhang, “A Two-tier Data Dissemination Model for Large-scale Wireless Sensor Networks, ” The Eighth Annual International Conference on Mobile Computing and Networking (MibiCom 2002), Atlanta, GA, pp. 146-159.

[12] Feng Zhao, Jie Liu, Juan Liu, Leonidas Guibas, and James Reich, “Collaborative Signal and Information Processing: An Information Directed Approach”, Proceedings of the IEEE, vol. 91. No. 8, August, 2003, pp. 1199-1209.