

Low Cost Distributed Actuation in Large-Scale Ad Hoc Sensor-Actuator Networks

Yen-Ting Lin and Seapahn Megerian
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
yenting@cae.wisc.edu, megerian@ece.wisc.edu

Abstract—Distributed actuation is a major application of sensor networks that relies on the information provided by the sensors and the ability of the actuators to change the environment, to try to achieve a set of desired conditions. In large scale ad hoc distributed sensor-actuator networks, traditional centralized control solutions are undesirable due to limiting factors such as scaling, delays associated with collecting information, and power consumption. In order to solve distributed actuation problems in sensor networks, groups of sensors and actuators must first be matched (clustered) efficiently and then the problem posed and solved in localized manners.

In this paper, given the locations of a set of sensors and controllable sources of superposable phenomena (e.g. light and heat), we first formulate and optimally solve the distributed sensor-actuator problem as an instance of centralized quadratic programming. We then investigate several localized heuristics that can achieve near optimal results while trading off message exchanges and energy consumption for accuracy and latency in obtaining those results. We also discuss the clustering algorithm that we use as a heuristic for solving the matching problem associated with matching the sensors to the proper actuators. Light sensors and sources are used throughout the paper as a driving application.

I. INTRODUCTION

While sensor networks [1] enable our computers to learn about what is going on in our world, actuators allow us to actively influence, change, and control our environments in an attempt to achieve a set of desired conditions. For decades, sense-compute-actuate loops have been widely studied and applied in countless control and automation applications [5]. However, emerging applications of wireless distributed sensor-actuator systems, fueled by the flourishing wireless sensor networks research efforts, have opened new problems that need to be addressed. Before discussing the problems, let us briefly take a look at the application domain and the specific types of systems in our aim.

Low cost wireless sensor nodes used for environmental monitoring and control of offices, industrial complexes, public venues, and residential buildings have attracted a great deal of attention in the industry due to potentially large economic impacts. Such systems can be used to automate and customize these environments in ways that have not been practical or possible in the past. In addition to creating context aware surroundings [2], mundane systems such as heating, cooling, security, lighting, and air humidity and quality control can be brought under much tighter and judicious control in the

presence of sensors to provide feedback information and the actuators necessary to make the desired changes.

Although many existing technologies such as Bluetooth may not be suitable for low cost sensor networks applications [3], recently there have been a number of developments towards very low power and low cost devices (e.g. [4]) that can enable large scale deployments in many everyday environments. Eaton corporation [6] for example has announced a new product line designed to monitor and control indoor environments, due to be released in 2005. The “plug-and-use” sensor modules in the system rely on wireless technology developed by Ember Corporation [7], a member of the Zigbee Alliance [8]. The Zigbee industry consortium of semiconductor manufacturers, technology providers, and end-users are working to define a global specification for reliable, cost-effective, low power wireless applications based on the IEEE 802.15.4 standard. The extremely low cost Eaton sensor modules work on the unlicensed ISM bands and can have a communication range of 100 to 200m. They connect to a more sophisticated base station called “Keyfob” either directly or through mesh networks of sensor modules. Combining such a monitoring system with addressable actuators controlling different sources (e.g. [9]) to achieve a set of desired conditions is the main motivation behind our work.

For relatively small deployments, such as in small offices or residential houses, collecting the sensor information in a centralized location to perform the necessary calculations and then sending the commands to the appropriate actuators will not incur too much overhead. For such cases, the centralized approach presented in section III will be adequate. However, for larger deployments such as large office buildings, industrial complexes, and residences, a centralized approach may prove to be too costly in terms of energy consumption for transmitting the messages, and the latency in making the changes. For example, even a 500ms delay in turning on a light when a person walks in a room can be quite noticeable and even unacceptable for many users. Furthermore, controllable sources typically have a limited geographic extent, such as lights in a living room or an office cubicle, making the treatment of the distributed sensor-actuator problem in localized ways not only cost effective, but also intuitive.

Solving the general distributed sensor-actuator network control problem not only encompasses topics from traditional control theory, but also problems of phenomena modeling,

query processing, and other sensor networks related management tasks such as location and orientation discovery, routing, topology management, and security. In this paper, we proposed a distributed computation algorithm which can be applied for general control problems, and choose lighting control as our primary driving application for purposes of environment automation and energy conservation. In section II we discuss our assumptions and the types of sources and sensors in our system. Essentially, for our lighting control example, we are given light intensity requirements at each sensor and the goal is, given the current measurements obtained by the sensors, to actuate the controllable light sources in such a way that the requirements are satisfied according to a convex cost function.

A. Paper Organization

The remainder of this paper is organized as follows. The next section contains the technical preliminaries, notation, the relevant background information, and assumptions used in this paper. Section III contains the problem formulation and optimal quadratic programming solution to the problem at hand. We use this solution as the base-line for quality comparisons. In section IV we discuss four localized heuristic methods for solving the problem: (i) non-overlapping clusters without historical data (ii) non-overlapping clusters with historical data (iii) overlapping clusters with historical data, and (iv) multiple source clusters. In section V we present several simulation results and discuss the advantages, disadvantages, and performance of each method, followed by the concluding remarks in section VI.

II. TECHNICAL PRELIMINARIES

A. Environmental Assumptions

We assume we have n sensors $S_1 \dots S_n$ and k controllable sources $L_1 \dots L_k$ connected to actuators. A source L_j is said to be controllable by sensor S_i if the effect of L_j on the environment can be measured by S_i and the distance between them is less than a predefined *controllable range* d (e.g. communication range). Each source L_j can be controlled by one or a group of sensors. A group of sensors controlling a source form a cluster. Thus, all sensors can be grouped into m clusters $C_1 \dots C_m$, with each cluster containing one controllable source and at least one sensor. Unless specified otherwise, we assume that sources are not shared between clusters.

At any given time, each sensor S_i reports a measurement M_i which is assumed to be a scalar value. All the sources in the network, including controllable and uncontrollable sources, can contribute to the measurement. Generally, the contribution of L_j to S_i is proportional to $|\overrightarrow{S_i L_j}|^{-k}$, where k is a positive value depending on the type of source. That is, the longer distance between the sensor and source, the smaller the source contributes to that sensor's measurement. In our work, we assume we are given a parametric model relating the measurements of the sensors as a combination of source intensities and other environmental variables such as shadows and reflections for light sensing and convection currents in a heat field. More specifically, we assume that the model

relating sensor measurements and sources can be posed as the optimization of an objective function plus constraints (for example, each source intensity can not be negative and has a maximum value). A typical model, such as for light sensing, is the superposition model where source intensities impact sensor measurements according to a linear coefficient. The specific details regarding our light sensing and control example is presented in the following subsection.

B. Learning Phase

Our system relies on an explicit learning phase to learn the relationships between sources and sensors. As mentioned above, this is typically done according to a parametric model, the coefficients of which are learned during this phase of the algorithm. In our implementation for light sensing, we determine the coefficient relating each controllable source to each sensor by altering one source at a time and recording the measurements. We assume that all other environmental conditions do not change in this period, except for the source being altered. In the simulations, we explicitly calculate this coefficient. In practice, we expect to have a combination of controllable and uncontrollable sources (such as sunlight) which may require an extended learning phase to accurately learn the coefficients, account for noise, and to model other changes in the environment.

Here, we have chosen lighting control as our example application, and thus, model each measurement M_i as a linear combination of each source's intensity at the sensor location $\langle S_{x_i}, S_{y_i}, S_{z_i} \rangle$:

$$M_i = \sum_{j=1}^k a_{ij} I_j$$

where I_j is the intensity of the source L_j and a_{ij} is the coefficient relating the measurement S_i to the intensity at the source L_j . This coefficient is influenced by three major factors (as shown in Figure 1):

- The distance between S_i and L_j ($|\overrightarrow{S_i L_j}|$)
- The angle θ of $\overrightarrow{S_i L_j}$ and the normal vector of the sensor plane $\overrightarrow{N_i}$
- Other environmental conditions such as obstacles, reflections, shadows, etc.

Also, since we are dealing with light sources, the delay between a source being actuated and its impact on sensor measurements is assumed to be negligible and thus, ignored. For other sources where this delay must be accounted for, at each step of the algorithm we assume the system can reach an equilibrium state before proceeding to the next step.

C. Cluster Formation

The underlying problem of assigning sources to sensor clusters is solving the the well known corresponding weighted bi-partite matching [10]. The sources and sensors can be thought of as a bi-partite graph with edges connecting each sensor (or set of sensors) to each source. Each edge is weighted according to a cost function that for example could be based

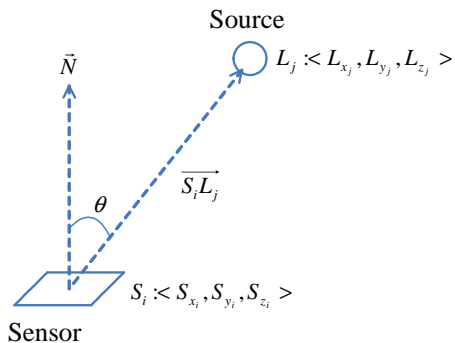


Fig. 1. Sensor and Source Notation

on the distance from the sensor to the source. Although for the centralized case we directly solve for the optimal matching, for the localized heuristics we have to address the distributed clustering and sensor to actuator matching problem as well.

The cluster formation mechanism can be incorporated with an underlying self-configuring protocol such as ASCENT [11] and SPAN [12]. When a sensor activates, it listens to collect current cluster information in the network. If there are already formed clusters, it chooses the cluster whose corresponding source has the shortest distance with itself by sending a join request to that cluster. If the cluster is full (each cluster has its capacity limitation to limit the traffic within the cluster), the sensor will choose the next best cluster in a greedy fashion. This will repeat until the sensor successfully joins one cluster. If a cluster is not found and there are controllable sources in the environment, the sensor will start a new cluster by becoming the primary master controller to the source(s).

III. CENTRALIZED FORMULATION AND SOLUTION

Consider the case where $m = 1$, i.e. all the sensors and lights belong to a single cluster. Here, the problem can be solved by a centralized controller who is responsible for collecting information from all sensors to compute the solution. This controller can be a special device or an elected sensor node. As we stated above, it is assumed that there are n sensors S_i , $1 \leq i \leq n$, and k sources L_j , $1 \leq j \leq k$. For each S_i we have a physical measurement M_i and *required intensity* R_i . The goal is to find a set of source intensities L such that the resulting physical measurements M_i satisfy the required intensities R_i , by minimizing the error function $F(E)$. The problem can thus be formulated as:

$$\arg \text{Min}_L F(E)$$

$$F(E) = f(E_1, E_2, \dots, E_n)$$

$$\forall i : E_i = |M_i - R_i|$$

$$\forall j : 0 \leq L_j \leq \text{max value}$$

The physical constraint that the intensity of each source must be located between zero and its maximum value is applied. In many cases, the result from this centralized approach is guaranteed to be optimal and can be solved in one step.

However, clearly this comes at the cost of every sensor having to send its measurements to the central controller or flood across the entire network. The controller must then send the source assignments to each controllable source.

Consider the light example we mentioned in the previous section where M_i is a linear combination of L_j . The problem in this case is an instance of linear programming (LP) if the error function is defined as the sum of error ($F(E) = \sum E_i$). If the error function is posed as the Mean Square Error (MSE, $\frac{1}{n} \sum E_i^2$), then it is a standard form of quadratic programming [13]. We will use the later definition of error function in this paper and use this result for this centralized approach as the baseline to evaluate our distributed approaches.

IV. DISTRIBUTED ALGORITHMS

Although the centralized approach in many cases is guaranteed to find the optimum solution, the power consumption and the latency of transmitting the necessary information to the centralized controller and back to the actuators may be too large. In many control applications, latency and power constraints are far more important issues than global optimality of the final solution. As a side note, since our application is targeted at the extremes of low cost operation, the available memory and processing power at each node will be very limited, making the centralized optimization approach impractical.

The main idea in the distributed approach is to divide all sensors into smaller clusters and solve the problem locally. Instead of doing global optimization, in distributed approach a cluster does local optimizations and try to approach the final solution iteratively. Here, we present and analyze four scenarios, the relative performances of which is analyzed in the next section. The four scenarios are:

- 1) Non-overlapping clusters without historical data: Here the clusters are formed as disjoint sets of sensors.
- 2) Non-overlapping clusters with historical data: An adaptive version of the above.
- 3) Overlapping clusters with historical data: Here clusters are allowed to have overlaps. The shared sensors serve as a means of data exchange between the clusters.
- 4) Multiple source clusters: In this scenario we investigate the presence of multiple sources and their simultaneous optimization at each iteration step.

In order to make the following discussions concrete and brief, we focus on the light sensing and control application as posed above.

A. Scenario 1: Non-overlapping clusters without historical data

In this scenario all sensors are divided into k clusters; each cluster contains exactly one designated source and each sensor belongs to exactly one cluster. Sensor and source information are only transmitted within the cluster. In each iteration, the control sensor attempts to optimize the objective function by computing the suitable intensity for its designated source (treat the contribution from other source as constant), based on

the local information from all its members. Compare to the centralized approach, the number of variables needed to be optimized here is only one and the objective function is much simpler.

Similar to the centralized approach, the resulting source intensities may be negative in some cases. If the non-negative constraint is introduced into the system, the optimal local answer is to let the light intensity to be zero. However, setting intensity to zero will sometimes cause the result of the overall optimization to diverge. To address this, in such cases we randomly choose a value between zero and the intensity at the previous iteration. In this scenario, the information required for the algorithm is little and the computation is quite simple, at the cost of solution optimality.

B. Scenario II: Non-overlapping clusters with historical data

In Scenario I, a simpler function and minimal memory is used. However, it may result in an over- increment or decrement in source intensities between iterations. Let us demonstrate this by considering a simple topology with only two lights and two sensors as shown in Figure 2(a). If after one iteration both sensors' measurements are larger than their requirements (the I_1 and I_2 of iteration 2 in Figure 2(b)), they will both try to decrease the intensity of their designated source in the next iteration. Since the control sensor treats intensity contribution from other lights as a constant, the decrement for both lights will be larger than they should be. Conversely, if the measurement for both sensors are less than they require (the I_1 and I_2 of iteration 3 in Figure 2(b)), over-increment at the next iteration will occur. Also, as shown in Figure 2(c), when one measurement is less than required and the other larger, the amount of increment/decrement will be less than it should be.

As a result, Scenario I may need more iterations to converge to the final solution. To address this, we use the following adaptivity rules for adjustment:

- If the intensity change of a designated light has the same sign bit as the change of the other lights, the increment/decrement for the next iteration should be reduced.
- If the intensity change of a designated light has different sign bit as the change of other lights, the increment/decrement for the next iteration should be increased.

Based on these rules the algorithm in Scenario I can be modified as shown in the pseudo code in figure 3.

The *AdjustCoefficient* here is a positive constant to adjust the increment/decrement value. In this scenario, the speed of convergence is increased and the trade-off is that additional storage for the state of the previous iteration is required, and the computation is slightly more complicated. As the example given above, the result of Scenario II with *AdjustCoefficient* equals to 0.5 is shown as the *modifiedI₁* and *modifiedI₂* in figure 2(b) and 2(c).

C. Scenario III: Overlapping clusters with historical data

In the two scenarios discussed above, each sensor belongs to exactly one cluster. That is, the requirement of one sensor will

Assume S_i is the control sensor in the cluster, $I(j)$ and $C_i(j)$ denote the value in iteration j .

```

for each cluster  $C_k, k = 1 \dots m$  {
  compute  $I(j)$ ;
  increment =  $I(j) - I(j - 1)$ ;
   $I(j) = I(j) - \text{sign}(I(j) - I(j - 1)) \cdot \text{sign}(C_i(j) - C_i(j - 1)) \cdot \text{increment} \cdot \text{AdjustCoefficient}$ ;

  if  $I(j) < 0$  {
     $I(j) = I(j - 1) \cdot \text{rand}()$ ;
  }
}

```

Fig. 3. Pseudo-code for Scenario II

be manipulated in only one cluster. In certain cases, sensors may be allowed to join more than one cluster. For example, if sensor S_i belongs to cluster C_1 and C_2 (with corresponding sources L_1 and L_2), it will send its local information to the two clusters respectively. Then its requirement can be considered in two clusters. This scenario will result in a larger number of exchanged messages but may increase the accuracy of the final solution (reduce the mean square error as in our lighting example).

D. Scenario IV: Clustering with multiple lights

In the scenarios described above, only one source was allowed in a cluster. Thus the optimization in each cluster is shooting for just one variable. Considering our light example, the equation for MSE is a quadratic polynomial with one variable. Such kind of polynomial has only one minimal value (minimum). If the minimum occurs at $I < 0$, then the minimal will only occur at $I = 0$ with the addition of the non-negative constraint. The trick described in Scenario I reduces the probability of divergence but may not helpful in improving the accuracy or speeding up convergence.

In this section we consider another scenario which allows multiple sources in a cluster. Assuming that there are p sensors and q sources in a cluster, the MSE equation in a cluster becomes a quadratic polynomial but with multiple variables. Even if the minimum occurs at any $I_j < 0$, there may still be other minimal values with all I_j satisfying the non-negative constraint. Similar to scenario III, lights could be overlapped between clusters. However, this may lead to the control contention between clusters and thus we only consider the non-overlapping sources case in the our simulations.

V. SIMULATION RESULTS

To evaluate the performance among the proposed scenarios, we designed a simulation environment with 4 lights and 16 sensors. Each light or sensor is randomly placed in a 3D space with the coordinate $\langle x, y, z \rangle$ where $(0 \leq x \leq 1000, 0 \leq y \leq 1000, 500 \leq z \leq 1000)$ for each light and $(0 \leq x \leq 1000, 0 \leq y \leq 1000, 0 \leq z \leq 500)$ for each sensor. The lights

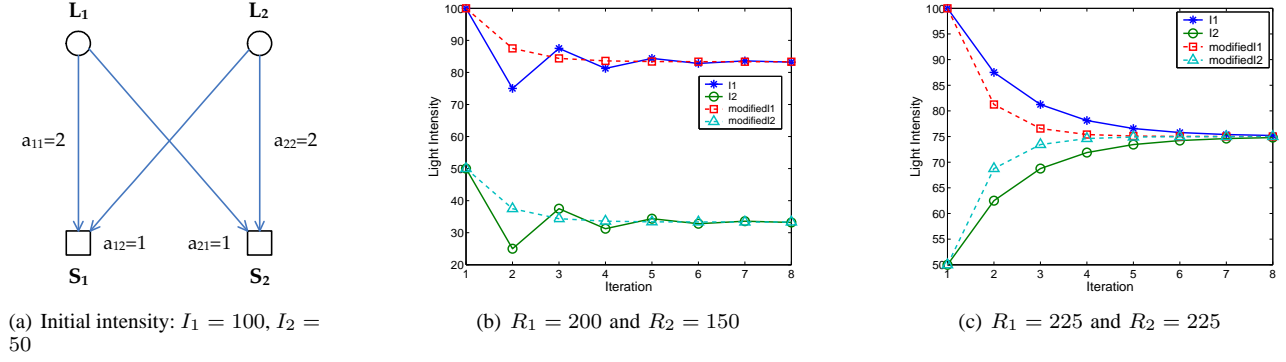


Fig. 2. Simple example: 2 lights and 2 sensors control problem

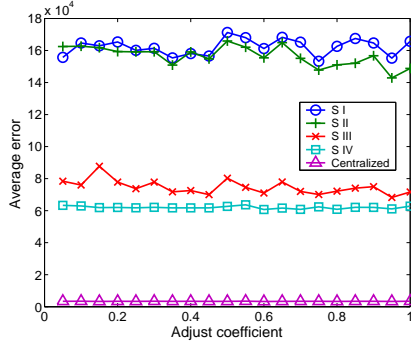


Fig. 4. MSE for different scenarios

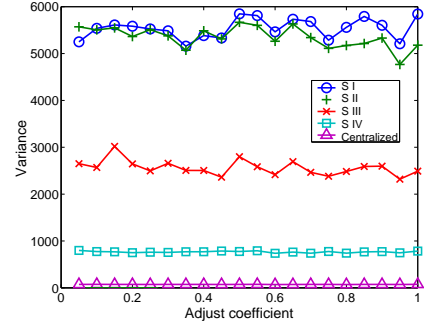


Fig. 5. Variance for different scenarios

are point sources and the coefficient is computed by

$$a_{ij} = \frac{\overrightarrow{S_i L_j} \cdot \overrightarrow{N_i}}{|\overrightarrow{S_i L_j}|^2} \cdot I_j$$

during the setup stage. In Scenario I and II, each cluster contains one light and four sensors while no sensor is allowed to belong to more than one cluster. For scenario III, we allow one of the sensors in each cluster to be shared by another cluster. For scenario IV, each cluster contains two lights and eight sensors, neither of which is shared between clusters. With the exception of the controllable light sources, we assume there are no other dynamic events during the simulation. Consequently, clusters are settled before computation starts, and the desired goal for each sensor does not change before computation completes.

In the following subsections we evaluate each scenario by its performance on accuracy, speed of convergence and energy consumption. The figures in the following sections show the simulation result for each criterion versus the *AdjustCoefficient* parameter. Since *AdjustCoefficient* is only used in Scenario II and III, the value for other scenarios are kept constant while *AdjustCoefficient* changes.

A. Accuracy

Figure 4 shows the result of the MSE for all proposed scenarios. As expected, the centralized approach has the best MSE among all cases. Scenario IV has the second best MSE and scenario III performs almost the same. Comparing with

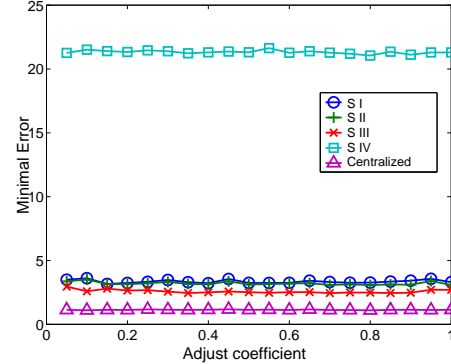


Fig. 6. Minimal error for different scenarios

these three scenarios, scenario I and II have clearly worse MSE and the most suboptimal solutions. Again as expected, we can conclude that the more lights and sensors considered with each cluster, the more accurate our results will be, at the cost of communication power consumption. Also, the value of *AdjustCoefficient* doesn't affect the result of MSE.

In addition to MSE, the analysis of E_i for each sensor is shown in Figure 5 and 6. The centralized approach and scenario IV has lower variance than the others thus they are "fair" scheme for every sensor. However, though scenario I, II and III have higher variance, their minimal error among all sensors are much less than that of scenario IV. So these three scenarios may be useful for some applications which some set of sensors have higher priority than others.

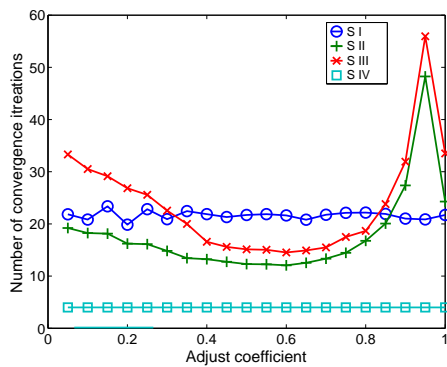


Fig. 7. Convergence speed for different scenarios

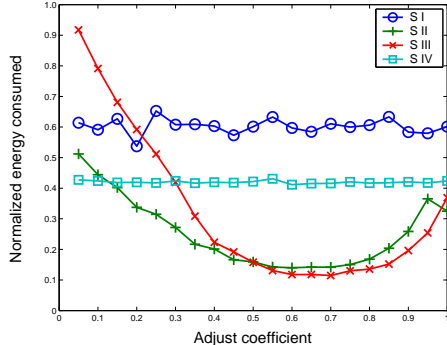


Fig. 8. Energy consumed for different scenarios

B. Convergence speed

Except for the centralized approach, all the four distributed scenarios need to approach their final solution iteratively. Thus the number of iteration they need to get the final solution is also an important criterion in our evaluation. By checking the result of each simulation, before reaching the final solution, there are several iterations during which the difference between iterations is small. Thus we consider the number of iterations required to enter a steady state N instead of the number of iterations needed to reach the final solution. This N is defined as the minimal positive integer N so that

$$\frac{|solution(iteration\ i) - (final\ solution)|}{(final\ solution)} \leq 0.05$$

for all $i \geq N$. The result is shown in Figure 7.

Since scenario IV controls multiple lights at the same time, its speed for convergence is the fastest (about 4 iterations). By choosing proper *AdjustCoefficient* (0.5 ~ 0.6), scenario II and III converge faster than scenario I by 6 ~ 9 iterations. It proves that considering historical record does help. Besides, some sensors in scenario III participate in multiple clusters, its converging speed is slightly slower than scenario II.

C. Energy consumed

Sharp changes in intensity will typically cause large energy diffusion at the sources. Also, in case of lighting control for example, such changes are undesirable from the users' perspective as well. Thus we treat total energy consumed (changes at the source) as another criterion. Note that this is different from the power consumption related to communication and

processing by the nodes. Assuming there are k lights and the number of iteration needed to convergence is N , energy consumed is defined as

$$energy = \sum_{i=2}^N \sum_{j=1}^k (I_j(i) - I_j(i-1))$$

and the result is shown in Figure 8. As the feature of scenario II and III the historical record is referred to adjust the increment/decrement for each iteration, the probability of sharp changes in intensity is reduced. Thus their energy consumed is less than that of scenario I. By choosing proper *AdjustCoefficient*, their performance in this category is even better than scenario IV.

VI. CONCLUSIONS

Distributed sensing and actuation in the context of wireless sensor networks hold huge potentials for progress in automation, control, energy conservation, and smart spaces. At the same time, they pose a number of difficult new challenges that are crucial to tackle in order to design and deploy practical systems. In this paper we focused on environment control using networks of sensors and actuators targeted at the ultra low cost, low bandwidth, and low power domains exemplified by the Eaton home control devices, the IEEE 802.15.4 standard, and the efforts of the Zigbee Alliance.

We formulated the light control problem as a centralized instance of quadratic programming and proposed several distributed heuristics to solve it. The algorithm also relies on a learning phase to learn the effects of each source on each sensor and a clustering technique to organize the distributed computation model. Simulation were used to explore several tradeoffs and analyze the performances of each scenario.

REFERENCES

- [1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. IEEE Communication Magazine, August, 2002.
- [2] S. Meyer and A. Rakotonirainy. "A Survey of Research on Context-Aware Homes." Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, Adelaide, Australia, 2003.
- [3] M. Leopold, M.B. Dydensborg, and P. Bonnet. "Bluetooth and Sensor Networks: A Reality Check." The First ACM Conference on Embedded Networked Sensor Systems (Sensys03), Los Angeles, CA, USA. November, 2003.
- [4] J.M. Rabaey, J. Ammer, t. Karalar, S. Li, B. Otis, M. Sheets, and T. Tuah. "PicoRadios for Wireless Sensor Networks: The Next Challenge in Ultra-Low Power Design." ISSCC 2002.
- [5] Robert F. Stengel. Optimal Control and Estimation. Dover Publications; Rei edition, September 1, 1994, ISBN: 0486682005.
- [6] Eaton Corporation. "Monitoring your home from afar," Boston Globe, www.boston.com/business/gallery/InternetofThings, 2004.
- [7] Ember Corporation. www.ember.com
- [8] Zigbee Alliance. www.zigbee.org
- [9] X10 Corporation. www.x10.com
- [10] C. Papadimitriou, K. Steiglitz. Combinatorial Optimization : Algorithms and Complexity, Dover Publications; Unabridged edition, July 7, 1998, ISBN: 0486402584.
- [11] A. Cerpa and D. Estrin. "ASCENT: Adaptive Self-Configuring sSensor Networks Topologies." INFOCOM 2002. Volume: 3, 23-27 June 2002.
- [12] B. Chen et. al. "Span: an Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks." ACM Wireless Networks Journal, Vol 8, No 5, Sep 2002.
- [13] R. Fletcher, Practical Methods of Optimization, 2nd ed., John Wiley & Sons, Inc., New York, 1987.