Modern Applied Science
www.ccsenet.org/journal.html
Vol. 2, No. 4
July 2008

Distributed Network in Concave Operating Environment

Kam Mun Loong (Corresponding author)
Cooperative System Lab, Dept. of Mechanical Engineering, National University of Singapore
21 Lower Kent Ridge Road, Singapore 119077
Tel: 65-6874-002-4615 E-mail: g0500314@nus.edu.sg

Gerard Leng
Cooperative System Lab, Dept. of Mechanical Engineering, National University of Singapore
21 Lower Kent Ridge Road, Singapore 119077
Tel: 65-6516-6548 E-mail: mpelsb@nus.edu.sg

This work was supported in part by the Temasek Defense Science Institute under grant TDSI/03-001/1A.

Abstract

The connectivity of distributed network within buildings depends critically on the shape of the operating environment. We propose the mean blockage as a new measure for concave shapes. The formal definition and an analytical evaluation of this measure will be shown. An algorithm has been developed to facilitate the evaluation of this measure for complicated concave polygons. The inadequacies of a couple of existing shape measures will be discussed and contrasted with the mean blockage measure for a variety of concave polygons. A possible general relationship between the blockage measure and the number of distributed sensors required to maintain high connectivity probability in concave regions will be presented.

Keywords: Distributed networks, Blockage, Concave polygons, Concavity measure, Connectivity

1. Introduction

Distributed network [7] is concerned with deploying multiple sensors to operate and gather information in an unknown, cluttered and possibly hazardous environment to achieve a common goal. One of the key decisions to be made in implementing a distributed network for a particular application is deciding on the optimal number of sensors that should be deployed. Besides being an important issue to the sensor network research community, it is also central to the research of cooperative systems. For cooperative systems, teams of multiple robots (which can be treated as sensors) are deployed to complete a certain task. Researchers have proposed different solutions to the problem of determining the optimal number of nodes by considering different constraints. In [11], Mei et al looked at this problem from the perspectives of energy constraints, i.e. they examined the relationships between the optimal number of nodes required to serve random requests under energy constraints. Hayes [1] defined a cost function which relates the number of robots (sensors), time taken to complete the search task and the moving speed of each robot. After that, he optimized the number of robots required for a search task by determining the minimum point of the cost function.

We argue that for distributed networks, the number of nodes required for a particular application should be optimized by ensuring the network is formed with high connectivity probability within the operating environment, thus allowing the exchange of information among individual sensors of the network. The rationale is that the network can maximize the utilization of the information gathered by every node through information sharing. This optimization problem is central to the *ad-hoc networks* [3] research community. In the ad-hoc network connectivity problem domain, the interest is in finding out how easily connections can be established among the individual nodes of an ad-hoc network. On the other hand, for our problem, the array of sensor nodes with Random Direction model mobility [9], can be treated as an ad-hoc network operating within a given operating environment and we are interested in the optimal number of nodes required to maintain high connectivity probability.

While the potential applications of distributed sensor networks are vast, we are interested in small scale distributed networks which are deployed inside buildings to carry out surveillance or search. The sparse deployment of sensor network in constrained area is one of the applications of distributed sensor network, as noted in [5]. Operating inside buildings poses another challenge for distributed networks: the concave boundary. A distributed sensor networks will find it harder to maintain connectivity operating inside concave boundaries than convex ones, as shown in Figure 1.1.

Figure 1.1 demonstrates the definition of connectivity. Two randomly moving sensors are connected if and only if the distance between them, *D*, is smaller than their communication range (transmitting/receiving range), *R*. For simplicity, we shall assume that the communication range of each sensor is the same and the communication can only be established if the two sensors are within line-of-sight (LOS) of each other and that communication in a multi-hop manner (pass information to one or more intermediate sensors and have the information routed to the target) is assumed to be possible. This assumption will hold reasonably well for operation within an enclosed area (e.g. a housing unit of typical size 10m by 10m) and the technology which can support these types of communication will be the commercially available IEEE 802.11b wireless network and Bluetooth. The LOS of a sensor is the unobstructed view which the sensor can see/detect. A *fully connected network* is one in which there exists at least a path between any two sensors such that they can communicate to each other. The *connectivity probability* is the probability that the array of sensors can form a fully connected network given that each sensor is at a different position at any given time.

It is apparent that determining the optimal number of sensors to be deployed for a given search task is similar to the connectivity problem of an ad-hoc network. The connectivity probability of an ad-hoc network subjected to environment and sensory constraints is an active research field. Each node of the network is moving randomly inside an area and can either have a fixed transmitting range (connectivity probability is a function of distances between nodes) or that the connection between two nodes is fixed (determined by a probability function). The main research interest in the ad-hoc network domain is the conditions which ensure high network connectivity probability [2, 4, 8]. Bettstetter and Zangl [2] included the border effects in their study on ad-hoc networks inside a circular area. Ferrari and Tonguz [4] approached the connectivity problem from the viewpoint of the minimum number of neighbors required for each node. In [8], Santi studied the critical transmitting range for each node in an ad-hoc network under certain node mobility models. Despite the different approaches, the ad-hoc networks discussed in the literatures are concerned with convex areas (circle or rectangle) and the results are not directly applicable to our problem. Contrary to the focus on convex areas, for connectivity of distributed ad-hoc sensor networks within buildings, we are interested in *concave* regions.

In this paper, we will define a new measure of the concavity of a shape – the *blockage*. We argue that this new measure is a more natural choice in the study of connectivity probability for a distributed sensor network operating in a concave region (polygon). We will show how to evaluate the blockage analytically for a simple concave shape. Then, an algorithm will be provided to facilitate the computation of the blockage of a concave polygon. Results relating the blockage values and number of sensors required for high connectivity probability for various concave polygons will be shown. Comparisons with other shape measures will also be made to show the relationship of blockage measure with the connectivity probability of sensors inside concave areas.

2. Blockage – a new measure of concavity

Currently, there are many different shape measures studied in the literature [6]. However, for the connectivity problem inside a concave region, there is a more natural measure of concavity which has an intuitive physical meaning. Consider a mission in which sensors are deployed to search for a target inside a convex area. If the communication range *R* is larger than the maximum chord joining any two points of the boundary, the distributed sensor network will be connected at all time, i.e. connectivity probability is equal to 1. Thus, any number of sensors in the convex area will be able to maintain a fully connected network. On the other hand, if the same number of sensors are required to maintain connectivity in a concave area with exactly the same maximum chord length, the connectivity probability will be less than 1 (see Figure 2.1).

The effect of the concave corner is shown in Figure 2.1b. Node A and B cannot establish connection because their line of sights are blocked by the concave corner. By now, one can see that the problem of connecting multiple randomly moving sensors in a concave area is essentially related to how large an area each sensor can cover within its line of sight (LOS). This close relation between the connectivity probability and the field of view of the points within a particular operational area is the main motivation for the new measure defined presently.

Consider a two dimensional operational space A (of total area A_T). Suppose a sensor is located at (x_o, y_o) and is within A. The *field of view* for the sensor is:

$$L(x_o, y_o) = \frac{\text{the area within the line of sight of the node at } (x_o, y_o)}{\text{total area}} \quad \textit{(Eq.2.1)}$$

The *blockage* is simply:

$$B(x_o, y_o) = \frac{\text{the area unobserved by the node at } (x_o, y_o)}{\text{total area}} = 1 - L(x_o, y_o) \qquad \textit{(Eq.2.2)}$$

The *mean blockage* is defined as "the mean of the blockage values for every location of the sensor inside the concave shape", normalized by the total area enclosed by the boundary, A_T .

$$\textit{Mean Blockage},\ \overline{B} = \frac{\int\limits_{x\in A}\int\limits_{y\in A} B(x,y)\,dxdy}{A_T} = 1 - \frac{\int\limits_{x\in A}\int\limits_{y\in A} L(x,y)\,dxdy}{A_T}$$

$$\textit{Variance of Blockage},\ Var(B) = \frac{\int\limits_{x\in A}\int\limits_{y\in A} \left[B(x,y) - \overline{B}\right]^2 dxdy}{A_T}$$

(Eq.2.3)

3. Analytical calculation of blockage for a simple concave shape

Before we discuss the algorithm to compute the mean blockage value of an arbitrary polygon, we shall calculate the mean blockage values for a simple concave shape, namely "L-shape", analytically (see Figure 3.1a). To evaluate the mean blockage of the L-shape boundary as shown in Figure 3.1, we first notice that the line connecting the concave corner and the vertex joining two sides of length 2 is the line of symmetry of this shape. Due to this symmetry, we need only to evaluate blockage for half the area.

In Figure 3.1b, we further divide the lower half of the L-shape into three sub-regions, namely **a**, **b** and **c**. **a**', **b**' and **c**' are the mirror images of **a**, **b** and **c** respectively. We will need to evaluate blockage for points in **a**, **b** and **c** only. We start by evaluating blockage for points inside sub-region **a** and **b**. As depicted in Figure 3.2, we use polar coordinate system with the origin (*O*) at the concave corner. The axes of the polar coordinates (θ=0), which are used for evaluating the blockage for region a and b, are assigned to be 90 degrees apart. For a point (r, θ) within region **a** and **b,** the area that this point can not see will be equal to:

$$\begin{aligned} A_a(r,\theta) &= \textit{area of } (OQ_1V_3V_4) \\ &= 1 - \frac{1}{2} L \square 1 \square \sin\theta \\ &= 1 - \frac{1}{2}\tan\theta, \text{ since } L = \frac{1}{\cos\theta} \end{aligned}$$

and

$$\begin{aligned} A_b(r,\theta) &= \textit{area of } (OQ_1V_3) \\ &= \frac{1}{2} L \square 1 \square \sin\theta \\ &= \frac{1}{2}\tan\theta, \text{ since } L = \frac{1}{\cos\theta} \end{aligned}$$

The sums of blockage values of all the points within region **a** and **b** are:

$$\begin{aligned} B_a &= \frac{1}{A_T}\int_0^{\frac{\pi}{4}}\int_0^{\frac{1}{\cos\theta}} A_a(r,\theta)\,rdrd\theta \\ &= \frac{1}{3}\int_0^{\frac{\pi}{4}}\int_0^{\frac{1}{\cos\theta}} \left(1 - \frac{1}{2}\tan\theta\right) rdrd\theta \\ &= \frac{1}{8} \end{aligned}$$

and

$$\begin{aligned} B_b &= \frac{1}{A_T}\int_0^{\frac{\pi}{4}}\int_0^{\frac{1}{\cos\theta}} A_b(r,\theta)\,rdrd\theta \\ &= \frac{1}{3}\int_0^{\frac{\pi}{4}}\int_0^{\frac{1}{\cos\theta}} \left(\frac{1}{2}\tan\theta\right) rdrd\theta \\ &= \frac{1}{24} \end{aligned}$$

For points in region **c**, the whole area is within their field of view, thus, the blockage values for these points are 0, i.e. B_c=0. The mean blockage values of L-shape can now be evaluated:

$$\overline{B} = \frac{\textit{Total blocked area}}{(\textit{Total area})^2} = \frac{\left(\frac{1}{8} + \frac{1}{24}\right)\times 2}{3} = \frac{1}{9}$$

4. Algorithms to compute blockage for a point inside a concave polygon

As seen from the definition of the blockage term, we need to determine the area which is within (or not within) the LOS of all the points inside the operational area. This is highly complicated, if not impossible, to be computed analytically. An algorithm to compute the blockage measure for a point inside a concave polygon is described in this section (see Algorithm 4.1 for pseudo-code). The *mean* blockage measure can be computed by applying this algorithm (Algorithm 4.1) to k (depending on the resolution) points covering the concave area. The inputs to the algorithm are the vertices of the concave polygon and the point *P* for which the blockage measure is to be computed. Without loss of generality, we assume the vertices of the boundary are given in the "correct sequence" and in the "anti-clockwise direction". This means that given a list of *n* vertices, $V = \{V_1, V_2 ..., V_n\} = \{(x_1, y_1), ..., (x_i, y_i), ..., (x_n, y_n)\}$, one can redraw the concave polygons simply by connecting the consecutive vertices together, i.e. connecting V_i and V_{i+1} for all $1 \le i \le n$. The "anti-clockwise direction" way of listing the vertices means that when the vertices are connected in sequence to form the edges of the polygon, the area of interest is always on the left of the edges. Equivalently, it means that the cross-product of the vectors $\overline{E_i}$ (a vector connecting vertex (i+1) from vertex i, $E_i = \overline{V_i V_{i+1}}$) and $\overline{E_{i+1}}$ will be positive if V_{i+1} is a convex corner.

Input: Vertices of Polygon, *V,* Coordinates of a point, *P* = (x, y).

Output: Blockage of (x, y) in the polygon, *B*(x, y).

```
 1: $V_C$ = **FindConcaveCorner**($V$)
 2: Initialize field of view of point $P$, ${}^{P}V_{FOV}$ = $V_C$
 3: **if** $P$ is within $V$, **then**
 4:     **for** each concave corner $V_{ci} \in V_C$ **do**
 5:         *Direction* = **FindBlockedDirection**($V$, $V_{ci}$, $P$)
 6:         **if** (*Direction* != 0) **then**
 7:             $Q_{ci}$ = **FindLOSIntersection**($V$, $V_{ci}$, $P$)
 8:             ${}^{ci}V_D$ = **FindDeleteVertices**($V$, $V_{ci}$, $P$)
 9:             ${}^{P}V_{FOV}$ = **UpdateFOV**(${}^{P}V_{FOV}$, $Q_{c1}$, ${}^{ci}V_D$)
10:         **end if**
11:     **end for**
12:     Calculate normalized field of view, $F$(x, y).
13:     Calculate blockage, $B$(x, y).
14:     Return $B$(x, y).
15: **end if**
```

Algorithm 4.1 Overview of algorithm to compute blockage of a point.

The algorithm starts by determining which of the vertices are "concave" (Algorithm 4.2). A concave vertex is one with an internal angle greater than 180º. As mentioned above, this can be done by checking the sign of the cross product of $\overline{E_{i-1}} \times \overline{E_i}$. This step is essential because the LOS of any point within a concave polygon will *only* be blocked by edges connecting to concave vertices. After this step, the set of *m* concave vertices $V_C = \{V_{c1}, V_{c2} ..., V_{cm}\}$ will have been determined (see Figure 4.1a).

Input: Vertices of Polygon, *V*.

Output: Vertices of all concave corners, V_C, and number of concave corners, *m*.

```
1: Initialize m = 0, $V_C$ = $\varnothing$, n = number of vertices
2: **for** $1 \le i \le n$ **do**
3:     Find the vector of linking vertices $V_{i-1}$ and $V_i$ ($\overline{E_{i-1}}$).
4:     Find the vector of linking vertices $V_i$ and $V_{i+1}$ ($\overline{E_i}$).
5:     **if** cross product of $\overline{E_{i-1}} \times \overline{E_i}$ > 0 **then**
6:         $V_C = V_C \cup \{V_i\}$, m=m+1
7:     **end if**
8: **end for**
9: Return $V_C$ and *m*.
```

Algorithm 4.2 FindConcaveCorner(*V*).

Input: Vertices of Polygon, *V*, a concave vertex, V_{ci}, and a point *P*.

```
Output: Direction of blocking of $V_{ci}$ on point P.
1: if $V_{ci}$ is within LOS then
2:     if $V_{ci-1}$ and $V_{ci-1}$ are visible to point P then
3:         return not_blocking
4:     else if $V_{ci-1}$ is not visible to point P then
5:         return clockwise_blocking
6:     else
7:         return anti-clockwise_blocking
8:     end if
9: else return not_blocking
```

Algorithm 4.3 FindBlockedDirection(*V*, V_{ci}, *P*).

The second step of the algorithm (Algorithm 4.3) determines which of the edges of those concave vertices determined in step one are blocking the LOS of point *P*. Depending on the position of *P* relative to each concave vertex, say V_{c1}, the concave vertex can either 1) have no blocking effect 2) block certain area from the LOS of P with the edge linking V_{c1-1} and V_{c1} 3) block certain area with edge connecting V_{c1} and V_{c1+1}. If the concave vertex V_{c1} is within the LOS of *P*, the algorithm will determine which type of blocking effect V_{c1} has on *P*. The algorithm will also then check which of the vertices, V_{c1-1} or V_{c1+1}, is not within LOS of point *P*. If both V_{c1-1} and V_{c1+1} are within LOS of *P*, then the concave vertex does not have blocking effect on *P* (case 1). If V_{c1+1} is not visible to *P*, then it can be concluded that edge $V_{c1-1}V_{c1}$ is blocking the view of *P* (case 2). Vice versa, if V_{c1-1} is not visible, it can be concluded that $V_{c1}V_{c1+1}$ is blocking the view of *P* (case 3). To facilitate our discussion, we shall refer to case 2 as "anti-clockwise" blocking and case 3 as "clockwise" blocking (see Figure 4.1b). On the other hand, if V_{c1} is not within LOS of *P*, we conclude that there is no blocking effect due to V_{c1}.

The next step is to determine which of the vertices from *V* need to be deleted as a result of the blockings by all the concave vertices V_C. In order to achieve this, connect *P* and V_{c1} to form a straight line PV_{c1}. Extend this line in the direction of $\overrightarrow{PV_{c1}}$ to a large enough extent, say to point *T*, such that the length of PT is larger than the maximum chord length of the concave polygon. Next, we find the intersection point(s) of PT with the edges of the boundary (V_1V_2, $V_2V_3, \ldots, V_nV_1$). Depending on the concave polygon, there can be many intersections. If this is the case, take the intersection point, say Q_{c1}, which is *farthest* away from *P* and is *within* LOS of *P*. Through this process, we will gain both the knowledge of point Q_{c1} and the boundary edge that PQ_{c1} intersects, say edge V_kV_{k+1} (see Figure 4.2a).

The previous two steps will be repeated for all the concave vertices in V_C. We can now figure out which vertices are not within LOS of *P* by using the results from the previous steps. For example, concave corner V_{c1} has "anti-clockwise" blocking effect and the intersection point Q_{c1} is found to be on edge V_kV_{k+1}. From these two pieces of information, we know that the vertices which are not within LOS are ${}^{c1}V_D = \{V_{c1+1}, V_{c1+2}, ..., V_k\}$. Thus, if a concave vertex, say V_{ci}, has no blocking effect on P, ${}^{ci}V_D = \varnothing$. To compute the field of view (FOV) of point P, we start by assuming that the FOV of *P* is the entire area enclosed by the concave polygon and denote it as a list of vertices (the same way as we described a concave polygon), ${}^{P}V_{FOV'} = \{V_1, V_2, ..., V_n\}$. Then, vertices in ${}^{ci}V_D$, $1 \le ci \le cm$, are deleted away and all the intersection points Q_{ci} are inserted (see Figure 4.2b). The blockage value is then computed using the Eq.2.2 of Section 2.

5. Blockage values and connectivity probability (random search)

The algorithm has been tested on several concave shapes (Figure 5.1) and the blockage values are tabulated in Table 5.1. To determine the mean blockage value of a concave shape, the algorithm discussed in Section 4 is applied together with some standard algorithms which are used to solve this type of double integral problem [10]. Showing also in Table 5.1 are the compactness and bending energy measures of the various concave shapes. The definitions of compactness and bending energy [6] are:

$$Compactness = \frac{4\pi \cdot Area}{(Perimeter)^2}, \qquad Bending\ energy = \frac{1}{n}\sum_{i=0}^{n} k^2(i)$$

, where $k(i)$ is the discrete curvature. The chain codes for all of the concave shapes are given in the Appendix, together with the coordinates of the vertices for the "3-room house floor plan", which is an actual floor plan for a typical housing unit in Singapore. The mean blockage value for the "L1 shape" is very close to the analytical value, 0.111, we have determined in section 3. Also, the analytical blockage value for T-shape (0.125) is very close to the simulated result (0.123). The results listed in Table 5.1 are in ascending order of the mean blockage values and are plotted in Figure 5.2.

Recall that the motivation of inventing blockage measure is to come out with an efficient measure to describe the connectivity probability of the concave operational area. In other words, it is desirable that by evaluating the blockage values of a concave shape, we will know roughly how difficult (or easy) it is for an array of sensors to maintain connection with each other. Thus, we investigate the number of sensors required (*N*) to achieve connectivity probability of greater than 90% and 95% when they are deployed to carry out a *purely random search task*. The purely random search task is one in which all the sensors' movements are random and are independent to the movements of all the other sensors (except for collision avoidance). This is a well established way of searching and works well when there is little prior information about the search task or the reliabilities of the sensors are low.

```
Input: Vertices of Polygon, V, a concave vertex, $V_{ci}$.
Output: $N_{95\%}$ $N_{95\%}$.
 1: Initialize N = 2, flag = FALSE.
 2: Do
 3:     for sample = 1 to NSAMPLE
 4:         Generate N points randomly (with uniform distribution) inside polygon V.
 5:         if (N points are connected)
 6:             Count = Count +1
 7:         end if
 8:     end for
 9:     if (Count > 0.95*NSAMPLE)
10:         $N_{95\%}$ = N
11:         flag = TRUE
12:     end if
13:     N = N + 1
14: while (flag = FALSE)
```

Algorithm 5.1 Compute$N_{95\%}$(*V*, V_{ci}, NSAMPLE).

The random movements of the sensor nodes are simulated as a random point process [5]. In a single trial, a set of N points are generated uniformly and randomly inside a concave environment of interest and the connectivity of these N points is determined. If the N points are connected, the trial is considered to be "successful". The connectivity probability of N sensor nodes moving randomly inside a concave environment is approximated by running NSAMPLE trials and counting the number of trials which are "successful". For a particular concave shape, this set of simulation is run for $2 \le N \le N_{95\%}$, where $N_{95\%}$ is the number of sensor nodes required to maintain a fully connected network with 95% probability inside the concave shape. Depending on the complexity of the concave shapes, $N_{95\%}$ varies and one can only discover it by increasing N until the connectivity probability is close to 95%. The pseudo code for this iterative process is summarized in Algorithm 5.1 and the results are shown in Table 5.1.

Figure 5.2 shows the relationship between *N* and the blockage values for various concave polygons. The behaviors of the relationships between *N* and blockage are expected. As blockage tends to zero, which corresponds to a convex shape, *N* should tend to 2 (1 will be meaningless for connectivity). On the other extreme, as blockage tends to 1, which corresponds to a concave shape so complex that it is equivalent to the case where all sensors have zero communication range, the *N* value will tend to infinity.

Having shown the usefulness of blockage, a rule of thumb can be inferred on the number of sensors required to maintain high connectivity probability (90% or 95%) when carrying out random search task. Figure 5.3 is the plot of the same data as in Figure 5.2, but in natural-logarithm scale, i.e. $\ln(N)$ vs $-\ln(1-B)$. As seen from Figure 5.3, there exists a similar third order polynomial relationship for both 90% and 95% connectivity. In addition, this polynomial relationship between $\ln(N)$ and $-\ln(1-B)$ has the desired behavior that N tends to infinity when mean blockage value tends to 1. For the 95% connectivity curve, the model also passes through N=2 when mean blockage value tends to 0. This relationship should provide a fast and good estimate for real operation considerations.

Another observation which can be made from this study is the limitation of random search strategy for distributed sensor networks operating in concave regions. The number of sensors required to maintain high connectivity inside a 3-room housing unit is around 75 (Table 5.1), which, in real operation, may not be as feasible. Thus, as a rule of thumb, if the mean blockage value of a particular concave region of interest is higher than 0.5 (which requires more than 20 sensors to maintain 95% connectivity), it is generally *not feasible* to employ the random search strategy.

On the other hand, compactness and bending energy fail to capture this elegant and simple relationship (see Figure 5.4). For example, "Z1", "Z2" and "+" have the same compactness values measures but the number of sensors required for full connectivity in these boundaries are different. Similarly, bending energy does not perform well if one is to relate it to the distributed sensor network connectivity (check shape "L2", "Z1", "Z2"). In short, mean blockage has been shown to be more appropriate in the study of network connectivity for concave shapes.

References

Adam T. Hayes (2002), "How Many Robots? Group Size and Efficiency in Collective Search Task", *Proceedings of the 6th International Symposium on Distributed Autonomous Sensoric Systems*. 289.

C. Bettstetter, J. Zangl (2002), "How to Achieve a Connected Ad Hoc Network with Homogeneous Range Assignment: An Analytical Study with Consideration of Border Effects". *Mobile and Wireless Communications Network*. 125 - 129.

E. Royer, and C. K. Toh (1999), "A review of current routing protocols for ad hoc mobile wireless networks", *IEEE Personal Communications*. 6(2), 46-55.

G. Ferrari, O. K. Tonguz (2004), "Minimum Number of Neighbors for Fully Connected Uniform Ad Hoc Wireless Networks". *IEEE International Conference on Communications*.

K. Rőmer, F. Mattern (2004), "The Design Space of Wireless Sensor Networks". *IEEE Wireless Communications*, Dec.

L. Fontoura Costa, & R. M. Cesar Junior, "*Shape Analysis and Classification: Theory and Practice*".

Neha Jain and Dharma P. Agrawal (2005), "Current Trends in Wireless Sensor Network Design", *International Journal of Distributed Sensor Networks*, 1: 101–122.

P. Santi (2005), "The Critical Transmitting Range for Connectivity in Mobile Ad Hoc Networks". *IEEE Transactions on Mobile Computing*. Vol 4, 310-317.

T. K. Madsen, Frank H. P. Fitzek and Ramjee Prasad (2005), "Connectivity Probability of Wireless Ad Hoc Networks: Definition, Evaluation, Comparison". *Special Issue of the International Journal on Wireless Personal Communications*.

W. H. Press, B. P. Flannery, S. A. Teukolsky, & W. T. Vetterling, "*Numerical Recipes in C*", Cambridge University Press.

Yongguo Mei, Yung-Hsiang Lu, Y. Charlie Hu, and C.S. George Lee (2004), "Determining the Fleet Size of Mobile Robots with Energy Limitations". *Proceedings of Intelligent Sensors & Systems* 1420.

Table 5.1 Compactness, bending energy, blockage measure of the concave shapes in Figure 5.1.

Shape	Compactness	Bending Energy	Mean Blockage	N (for 90%)	N (for 95%)
L1	0.5890	3.000	0.110	3	4
L2	0.5236	2.667	0.111	3	4
T	0.5026	3.200	0.123	3	4
+	0.4363	4.000	0.161	4	5
X1	0.4363	3.333	0.215	5	6
Z1	0.4363	2.667	0.219	5	6
Z2	0.4363	2.667	0.278	7	9
U	0.4363	2.667	0.287	7	9
F1	0.3847	2.857	0.319	8	10
F2	0.2827	2.800	0.367	11	14
X2	0.2827	2.800	0.404	12	14
X3	0.3436	2.500	0.432	12	15
X4	0.3699	4.400	0.462	14	16
E	0.2400	2.000	0.534	20	23
C	0.1841	1.500	0.611	31	37
G	0.1739	1.412	0.641	34	40
3-Room	0.1758	0.1729	0.679	62	75

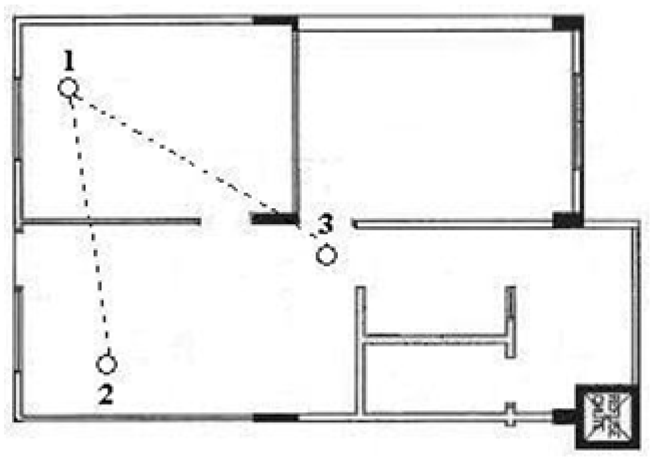
The last two columns show the minimum number of sensors required to get connectivity probability of 90% and 95%.

Appendix: Chain codes of concave shapes studied

The details about chain codes can be found in [11]. The chain codes of all the concave shapes are listed in Table A. The x-y coordinates of the vertices for the 3-room flat are included in Table A as well.

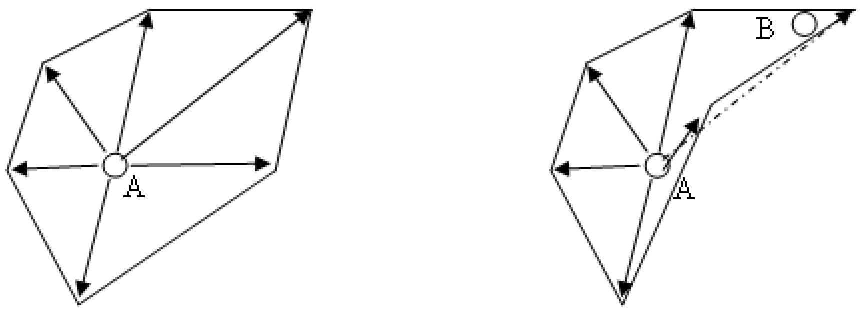
Shape	Absolute Chain Codes
L1	0, 0, 2, 4, 6, 6
L2	0, 0, 0, 2, 4, 2, 4, 2, 4, 6, 6, 6
Z1	0, 0, 2, 4, 2, 2, 4, 4, 6, 0, 6, 6
Z2	0, 0, 0, 2, 4, 4, 2, 4, 4, 6, 0, 6
F1	0, 2, 2, 0, 2, 4, 2, 0, 2, 4, 4, 6, 6, 4, 4, 6, 0, 0, 6, 6
F2	0, 0, 0, 2, 4, 2, 0, 2, 4, 4, 6, 6, 4, 6
T	0, 2, 0, 2, 4, 4, 4, 6, 0, 6
U	0, 0, 0, 2, 2, 4, 6, 4, 6, 4, 6, 6
+	0, 2, 0, 2, 4, 2, 4, 6, 4, 6, 0, 6
C	0, 0, 0, 0, 0, 2, 2, 4, 6, 4, 4, 4, 2, 2, 2, 0, 0, 0, 6, 0, 2, 2, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6
X1	0, 2, 0, 2, 4, 2, 4, 4, 6, 0, 6, 6
X2	0, 2, 4, 1, 3, 4, 6, 0, 5, 7
X3	0, 2, 2, 0, 2, 4, 4, 4, 2, 2, 4, 6, 6, 4, 6, 0, 0, 0, 6, 6
X4	0, 0, 0, 2, 2, 0, 2, 4, 4, 6, 6, 4, 2, 4, 6, 6
E	0, 0, 0, 2, 4, 4, 2, 0, 0, 2, 4, 4, 2, 0, 0, 2, 4, 4, 4, 6, 6, 6, 6, 6
G	0, 0, 0, 0, 0, 2, 2, 2, 4, 4, 6, 0, 6, 4, 4, 4, 2, 2, 2, 0, 0, 0, 0, 2, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6

Shape	Vert.	x	y	Vert.	x	y	Vert.	x	y
3-Room	1	0	0	12	9.1	17.1	23	7.2	11.8
	2	7	0	13	9.1	17.3	24	7	11.8
	3	7	6.4	14	11.6	17.3	25	7	19.6
	4	7.2	6.4	15	11.6	17.1	26	0	19.6
	5	7.2	0	16	11.1	17.1	27	0	9.8
	6	13.6	0	17	11.1	12.1	28	7.2	9.8
	7	13.6	11.9	18	13.6	12.1	29	7.2	8.2
	8	9.1	11.9	19	13.6	19.6	30	7	8.2
	9	9.1	12.1	20	12.7	19.6	31	7	9.6
	10	10.7	12.1	21	12.7	21.6	32	0	9.6
	11	10.7	17.1	22	7.2	21.6			



Three-sensor operation in a house

Figure 1.1 The line-of-sight of sensor 1 is blocked by the walls of a bedroom, thus rendering it unable to communicate with sensor 2 and sensor 3.



a. Convex polygon b. Concave polygon

Figure 2.1 a) The field of view of a sensor inside a convex region will be the whole convex area if the sensing range is at least the maximum chord. b) The field of view of a sensor can be less than the whole concave area even if its sensing range is larger than the maximum chord of the boundary (sensor A and B are not within line of sight of each other).

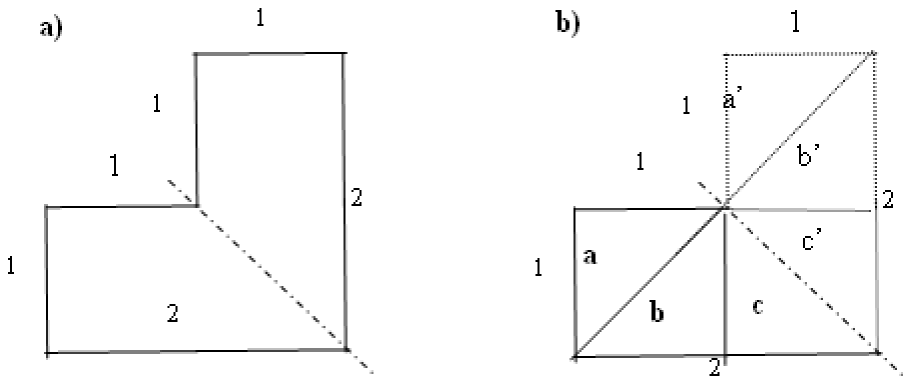


Figure 3.1 a) The simple concave shapes: L-shape. The edge lengths and symmetry lines (dotted lines) are shown in the figure as well. b) Sub-regions defined for points having similar FOV, to facilitate the evaluation of blockage values.

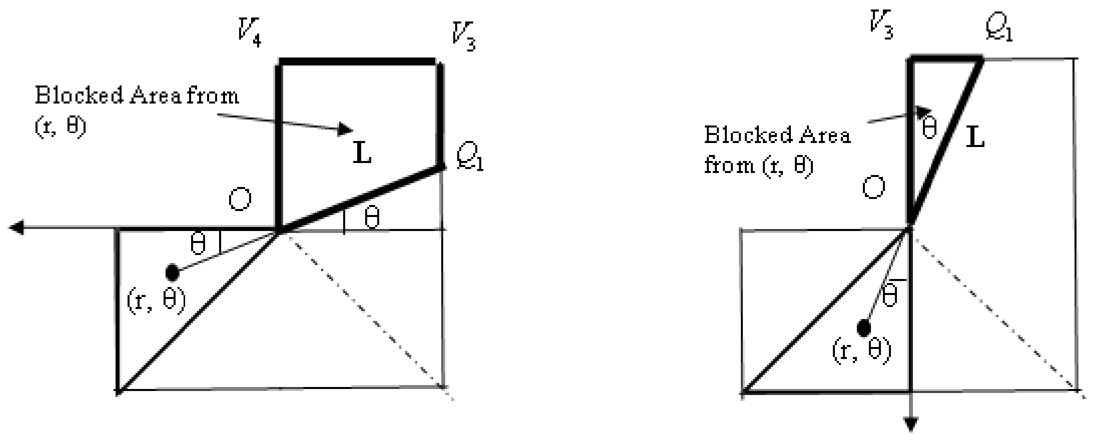
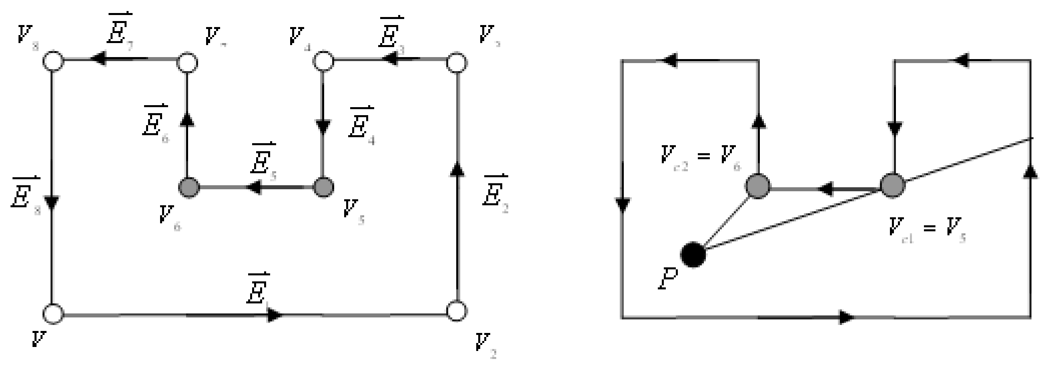


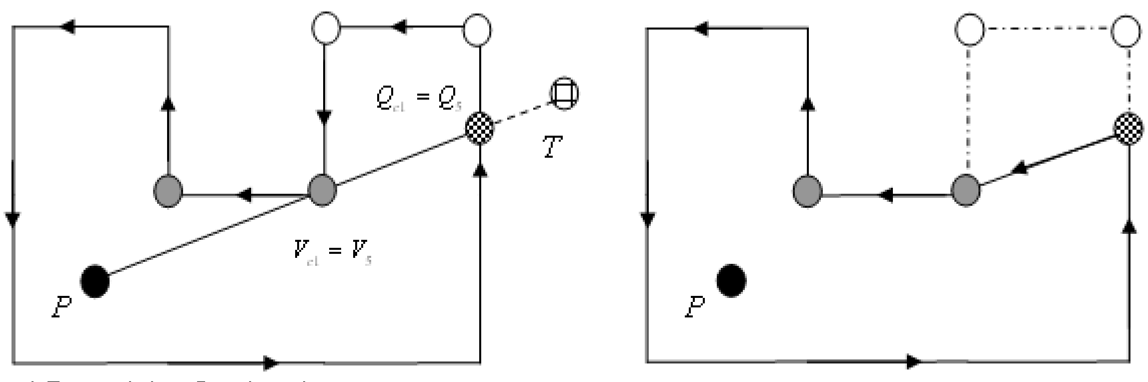
Figure 3.2 Evaluation of blockage for points in regions a and b (of L shape). The arrows show the θ=0 axes of the polar coordinates.



a) Determining concave vertices

b) Determining blocking direction

Figure 4.1 a) The "anti-clockwise" order of listing the vertices of a concave "U" shape, $V = \{V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8\}$. The grey dots are concave vertices determined by checking the signs of the cross-products of $\overrightarrow{E_{i-1}} \times \overrightarrow{E_i}$ for all vertices and V_C is found to be $V_C = \{V_5, V_6\}$. b) Concave vertex V_5 has a "clockwise" blocking effect on point P (V_4 is not visible to P) while concave vertex V_6 has no blocking effect on P (both $V_{[illegible]}$ and $V_{[illegible]}$ are visible to P).



a) Determining Q and vertices not within line of sight

b) Field of view of P

Figure 4.2 a) Point Q_5 is found to be intersecting edge V_2V_3. Thus, vertices which are not visible to P are ${}^5V_D = \{V_3, V_4\}$. b) The field of view of P is determined by deleting vertices in ${}^5V_D$ from V and insert Q_5 after vertex V_2.

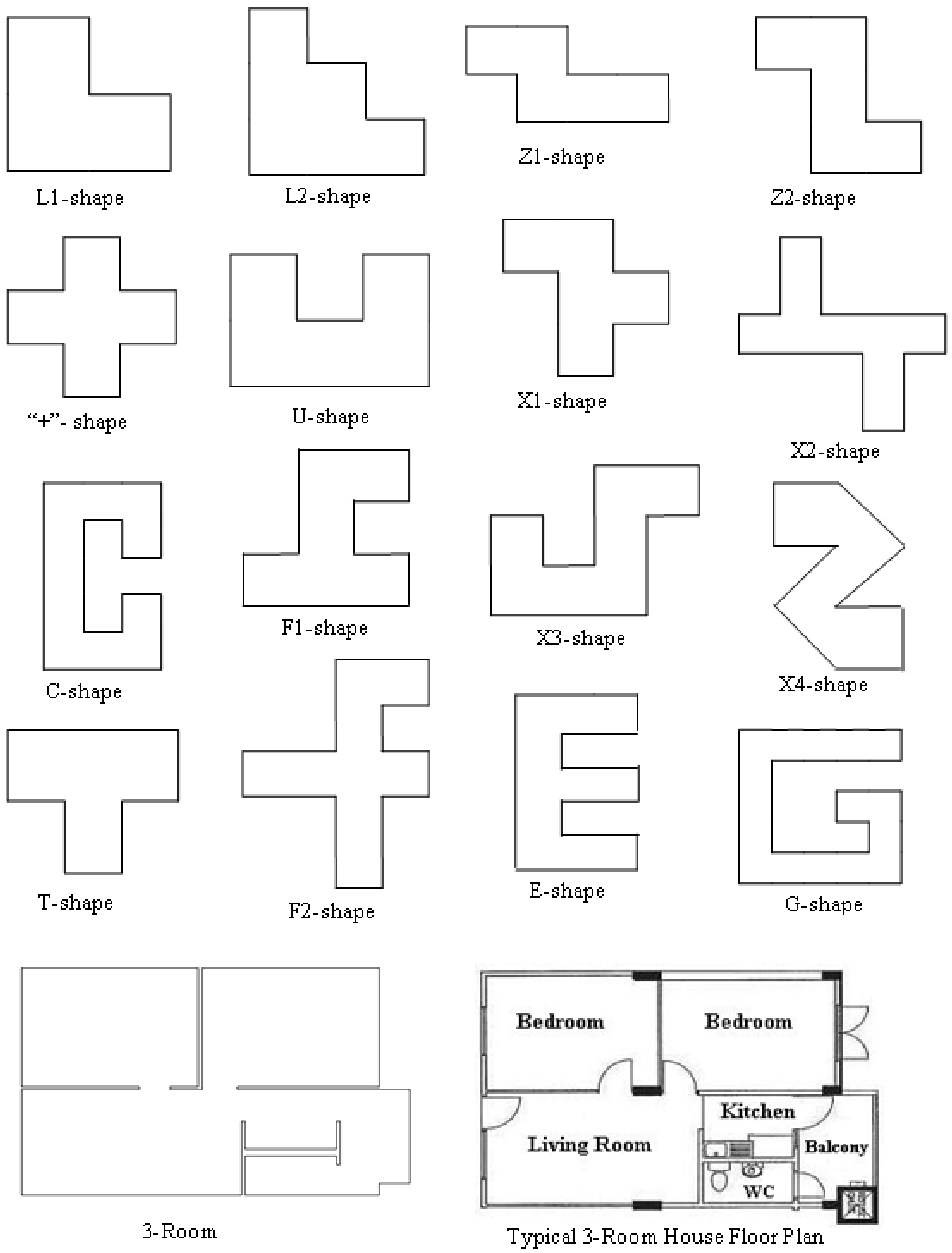


Figure 5.1 Concave shapes whose mean blockage values were computed by using the algorithm introduced in Section 4. The chain codes for all the shapes (vertices are given for 3-room house floor plan) are provided in the Appendix. Note that the polygon "3-Room" is modeled using the floor plan of a typical housing unit in Singapore.

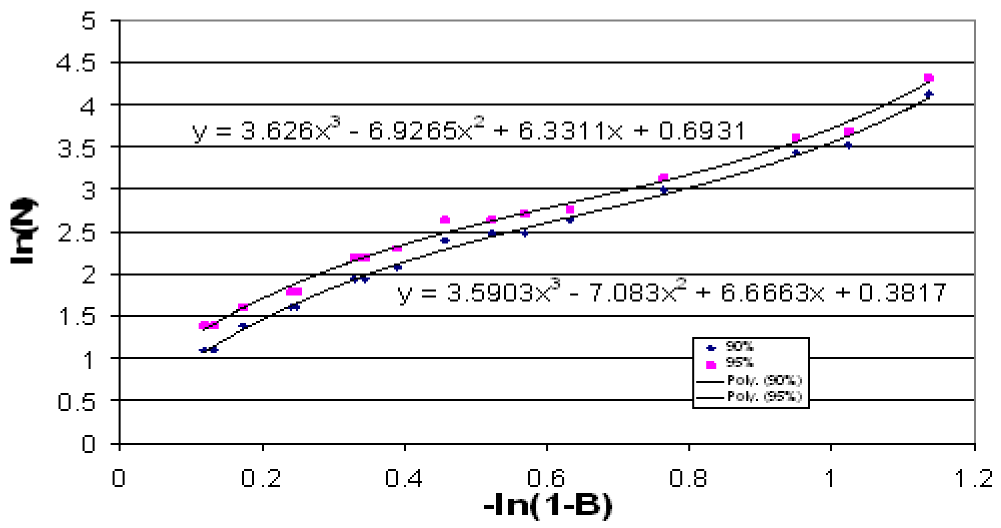
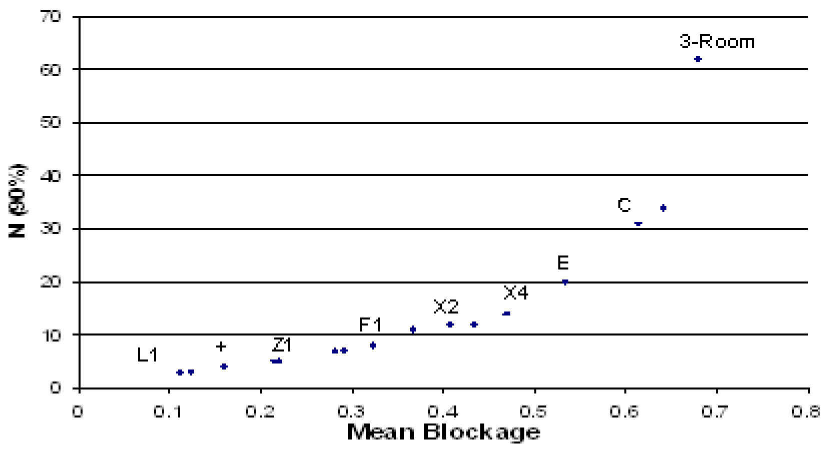


Figure 5.3 The plot of $\ln(N)$ against $-\ln(1-B)$, where B is the mean blockage measure.



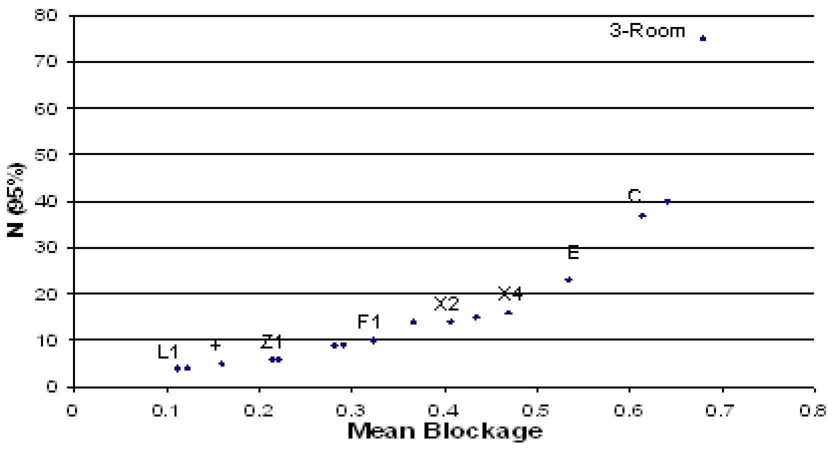


Figure 5.2 The plots of N against the blockage measure (for 90% and 95% connectivity probability).

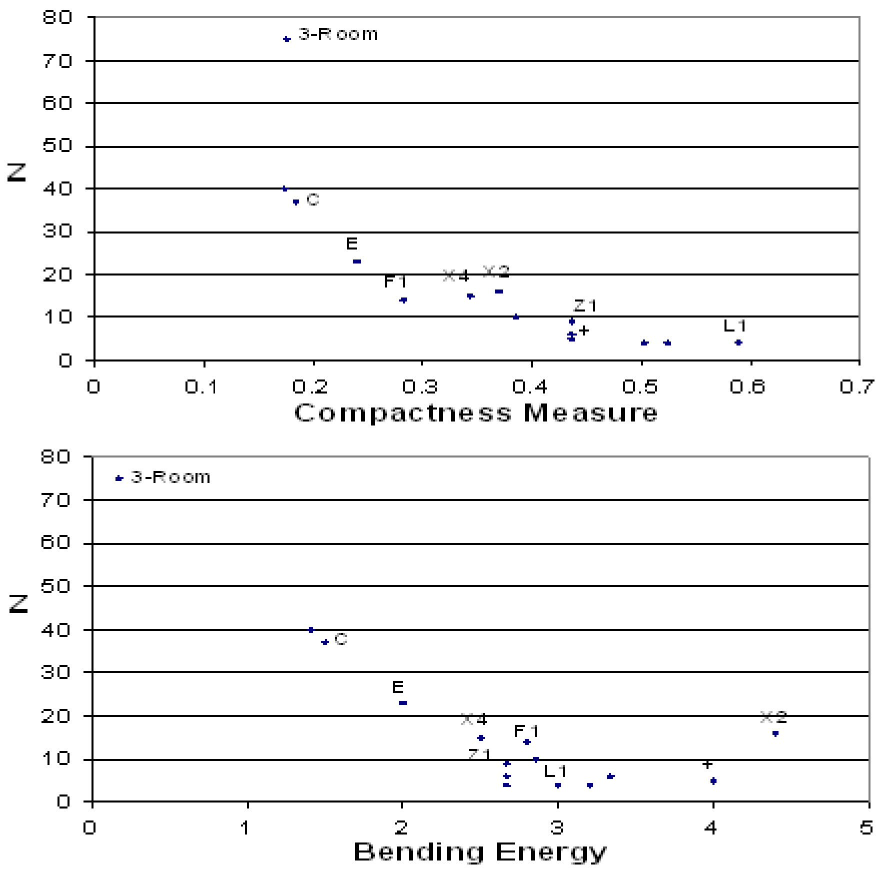


Figure 5.4 The plots of N versus compactness measure and bending energy.