Finding natural clusters having minimum description length

Richard S. Wallace *
NTT Human Interface Laboratories
Nippon Telegraph and Telephone Corporation
1-2356 Take, Yokosuka 238-03 Japan

Takeo Kanade
School of Computer Science
Carnegie Mellon
Pittsburgh, PA 15213 USA

April 9, 1990

Abstract

This paper summarizes a two-step procedure that finds natural clusters in geometric point data. The first step computes a hierarchical cluster tree minimizing an entropy objective function. The second step recursively explores the tree for a level clustering having minimum description length. Together, these two steps find natural clusters without requiring a user to specify and threshold parameters or "magic numbers". In particular, the method automatically determines the number of clusters in the input data. The first step exploits a new hierarchical clustering procedure called *Numerical Iterative Hierarchical Clustering (NIHC)*. The output of *NIHC* is a cluster tree. The second step in our procedure searches the tree for a level clustering having minimum description length (MDL). The MDL formulation, equivalent to maximizing the posterior probability, is suited to the clustering problem because it defines a natural prior distribution.

1 Approach

Cluster analysis is concerned with estimating the parameters of multiple probability distributions from data that contains superimposed samples of all those distributions. Consider a color TV picture of a rose bush in full bloom against a neutral background. Viewed in the space of red, green and blue intensities, the image pixel values form clouds of points, one each for the leaves, flowers and background. These clouds are random samples from complicated distributions that depend on factors such as the illumination conditions, the shape and reflectivity of the plant's surface and the noise introduced by the imaging system. Even after making assumptions about the form of the distributions sought, we are left with the difficult problem of deciding how many distributions there are, and how to partition the data to measure their parameters. This paper summarizes a two step procedure that finds natural clusters in data like the color TV image.

In order to be more specific we now introduce some notation. Our objective is to cluster a set of n vectors $\mathcal{S} = \{\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}\}$ from $\Re^d$. The output is a set of clusters $\mathcal{C} = \{u_0, \ldots, u_{k-1}\}$ representing the sets $\{\mathcal{S}(u_0), \ldots, \mathcal{S}(u_{k-1})\}$ such that $\bigcup_i \mathcal{S}(u_i) = \mathcal{S}$ and $\mathcal{S}(u_i) \cap \mathcal{S}(u_j) = \emptyset$ when $i \neq j$. In other words $\mathcal{C}$ *partitions* the set $\mathcal{S}$.

The hierarchical clustering step constructs a binary cluster tree t such that each node u in t represents a set $\mathcal{S}(u)$ of points. In particular if u is a leaf node then $\mathcal{S}(u) = \{\mathbf{x} \in \mathcal{S}\}$ and for any internal node u having children l and r, $\mathcal{S}(u) = \mathcal{S}(l) \cup \mathcal{S}(r)$. When u is a leaf node $l = r = \emptyset$. The root node t, also denoted t, represets $\mathcal{S}(t) = \mathcal{S}$. The tree t is oriented (the order of left and right subtrees is not important) and has $m = n-1$ internal nodes. The number of unique cluster trees t for an n point clustering problem is $(2m)!/(m!2^m)$. This number is larger than $m!$ for $m > 5$. The frontier of any binary subtree rooted at t has the form of a level clustering $\mathcal{C}$, so we let $\mathcal{C}(t)$ indicate a "pruned" tree representing a level clustering.

Hierarchical clustering attempts to minimize recursive objective functions having the form

$$E(u) = \begin{cases} 0 & \text{if } u = \emptyset \\ e(u) + E(l) + E(r) & \text{otherwise} \end{cases} \quad (1)$$

The node objective function $e(u)$ is often written as a

*The work reported here was compeleted at Carnegie Mellon University and was supported in part by the U.S. Defense Department Advanced Research Projects Agency (DOD) ARPA Order No. 7976 under contract F33615-87-C-1499 and monitored by: Avionics Laboratory, Air Force Wright Aeronautical Laboratories, Aeronautical Systems Division (AFSC), Wright-Patterson AFB, OH 45433-6453. Support was also provided through a Hughes Aircraft Company Fellowship. The views and conclusions contained in this article are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Department Advanced Research Projects Agency, Hughes Aircraft Company, the U.S. Government or Nippon Telgraph and Telephone Corporation.

distance function $d(l, r)$. Level clustering methods generally attempt to minimize objective functions defined either within clusters

$$E(\mathcal{C}) = \sum_{u \in \mathcal{C}} e(u) \tag{2}$$

for some within-cluster metric $e(u)$ or between them as

$$E(\mathcal{C}) = \sum_{u,v \in \mathcal{C}} d(u, v) \tag{3}$$

for some between-cluster metric $d(u, v)$, or as a combination of the two.

In our two-step procedure, which is a hybrid of hierarchical and level methods, there are naturally two objective functions. In the first step, the *NIHC* algorithm tries to minimize a recursive objective function (1). Although *NIHC* is defined for any function $e(u)$ computable from $\mathcal{S}(l)$ and $\mathcal{S}(r)$, we specifically focus on the the *Gaussian entropy* function $e(u) = \log |\mathbf{C}(u)|$,

where $\mathbf{C}(t)$ is the scatter matrix of the set $\mathcal{S}(t)$ and $|\cdot|$ is the matrix determinant. The name *Gaussian entropy* derives from the function's similarity to the relative entropy $H(u)$ of a d-variate normal having covariance $\mathbf{C}(u)$.

$$H(u) = \frac{1}{2} \log |\mathbf{C}(u)| + \frac{d}{2} \log 2\pi + \frac{1}{2}. \tag{4}$$

The second step in our procedure minimizes a level clustering objective function that differs in form from the standard within-cluster and between-cluster objective functions (3) and (2). The function measures the *description length* [2][3][5][6][7] [8][9] of the input set $\mathcal{S}$. Section 3 explains the details of our minimum description length (MDL) approach to deriving a level clustering from a hierarchical tree. But first, the next section introduces our procedure for hierarchical clustering.

2 The *NIHC* algorithm

The standard terminology for hierarchical clustering algorithms divides them into two categories: *agglomerative* algorithms build cluster trees through fusion from the bottom up, and *divisive* algorithms build them by splitting from the top down[4]. The term "iterative" in *NIHC* indicates that it is neither agglomerative nor divisive. Instead, *NIHC* starts with an arbitrary cluster tree and iteratively transforms it to reduce the value of the tree objective function. In ([10]) we show that the *NIHC* algorithm is superior to the standard agglomerative algorithm in several ways: it is no less efficient but reaches lower local minima of an entropy objective function, it consumes memory only linearly in the input size, it spends more time arranging clusters near the root of the tree, and *NIHC* finds better clusters than several versions of agglomeration in random data from Monte Carlo experiments and in the iris data set.

The basic transformation step of *NIHC* is a *grab* (figure 1). The grab operation was originally defined by Appel [1] to transform the binary trees he used to represent collections of point masses in an efficient solution to the n-body problem. The operation $u = grab(c, w, u)$ transforms the tree u in the following way. If either of the nodes c or w is the ancestor of the other, the grab is undefined. Otherwise, the grab transforms u so that c and w become siblings. Let p be c's parent and q be w's parent. Also let s stand for w's sibling. The grab replaces c in u with a new node q', which becomes the new parent of c and w. The node c is now an "only child", so it is "promoted" by deleting q and replacing it with s. The notation q and q' is intended to emphasize that the grab preserves the number of nodes in u.

The grab has the effect of rearranging the nested subsets represented by the nodes of a cluster tree. Let u be the first common ancestor of c and w. If v is a node along the path from p to u and v represents the set $\mathcal{S}(v)$ before the grab, then v represents the set $\mathcal{S}(v) \cup \mathcal{S}(w)$ after the grab. Similarly, if v is a node along the path from q's parent to u and v represents the set $\mathcal{S}(v)$ before the grab, then v represents $\mathcal{S}(v) - \mathcal{S}(w)$ after the grab. The important point here is that the grab rearranges only those sets along the paths back to u. Nodes elsewhere in the tree remain unaffected. This observation, called the *Grab property*, is key to writing *NIHC* .

Any cluster tree representing the point set $\mathcal{S}$ can be constructed from any other one by a sequence of grab operations. This property follows from the facts that $u = grab(w, c, grab(c, w, u))$ and that there always exits a sequence of grabs to construct a canonical tree from any tree representing $\mathcal{S}$. One choice of such a canonical tree is the tree that is a list (one child of each internal node is a leaf node) of the points sorted by their subscripts.

The input to *NIHC* is an arbitrary cluster tree, such as a k-d tree or a cluster tree generated by another hierarchical algorithm. *NIHC* searches for grabs that reduce the value of the tree objective function. The grab property implies that value of the Gaussian entropy objective function $e(t) = \log |\mathbf{C}(t)|$ changes only at nodes v along the paths from p to u and s to u. *NIHC* computes the effect of a particular grab by calculating the net objective function change along these paths. The heart of *NIHC* is a routine *cluster* that searches each subtree for the best energy-reducing grabs. *NIHC* calls *cluster* iteratively until it can find no more entropy reducing grabs:

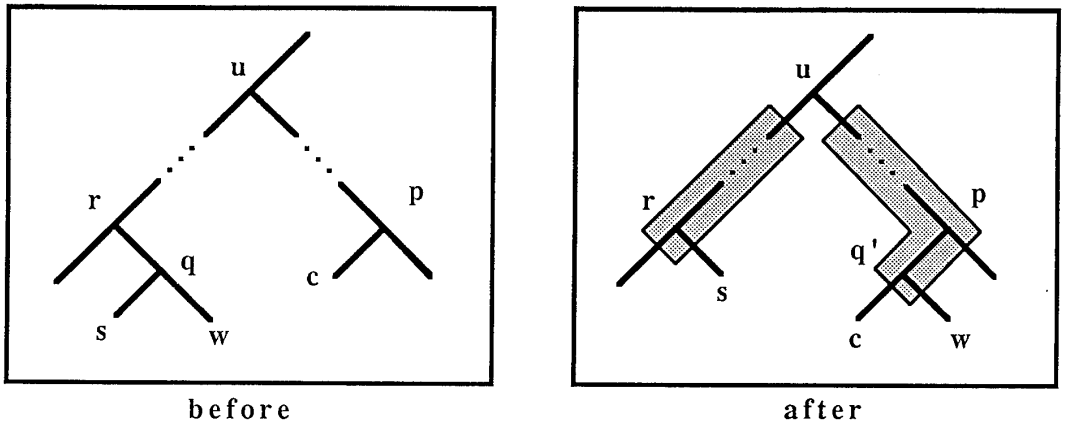

Figure 1: The grab operation makes nodes c and w siblings. The grab changes the tree energies only along the paths back to the first common ancestor of c and w

```
cluster (u)
begin
if u ≠ ∅ then
  begin
  cluster(u → left)
  cluster(u → right)
  find a pair of nodes c, w in u such that
      grab(c, w, u) reduces the objective function
      e(u) more than any other grab
  if c, w ≠ ∅ then grab(c, w, u)
  end
end
```

That *NIHC* eventually terminates is evident from the fact that there are finitely many trees, and these are ordered with respect to the objective function. The code hidden by the italic phrase *"find a pair of nodes..."* basically compares every node in the left subtree with every node in the right, looking for the best grab. Thus, *cluster* evaluates the objective function $O(n(u)^2)$ times in a subtree u. In the case of Gaussian entropy, each evaluation of the objective function takes $O(d^3)$ steps. In actual practice, we use a branch and bound procedure that drastically reduces the number of objective function evaluations. Because cluster evaluates the objective function $O(n(u)^2)$ times for each node u of a tree t, summing $n(u)^2$ over all nodes u in t tells the total complexity of *cluster*. In the worst case, when the tree t is a list, the total number of evaluations is $O(n(t)^3)$. If, on the other hand, the tree balanced (for any pair of siblings l and r, $n(l) = n(r)$), the total number of evaluations is just $O(n(t)^2)$.

If the choice of input tree is "reasonable", like a k-d tree or the output of another hierarchical algorithm, then *NIHC* generally converges after only a few iterations. However it remains an open problem to find a bound on the number of iterations. We do know that just *one* iteration of *NIHC* run on the output of the agglomerative algorithm, when agglomeration seeks to minimize the Gaussian entropy function, generates a tree in a lower entropy state. We do not yet know any theoretical bounds on the distance between the global optimum tree and the tree computed by *NIHC* with respect to the Gaussian entropy function, but experiments comparing *NIHC* with other heuristic methods having the same time complexity indicates that it reaches lower local minima.

3 MDL clusters

The *NIHC* program generates a hierarchical cluster tree. In order to extract level clusters we search the tree for a level clustering having minimum description length. The expression for description length of the set $\mathcal{S}$ is $I(\mathcal{S}) = I(\mathcal{C}) + I(\mathcal{S}|\mathcal{C})$. The first term in the sum indicates the amount of information needed to represent the clustering $\mathcal{C}$. The second term indicates the amount of information needed to represent the input set $\mathcal{S}$ given $\mathcal{C}$. In our case the clustering $\mathcal{C}$ is the frontier of a pruned cluster tree t, and we exploit the hierarchical structure of t to obtain a recursive formulation of description length.

The second term $I(\mathcal{S}|\mathcal{C})$ is easiest to derive. Assuming independence, the quantity

$$\begin{aligned} I(\mathcal{S}|\mathcal{C}) &= -\log p(\mathcal{S}|\mathcal{C}) \\ &= -log \prod_{u\in\mathcal{C}} p(u) \prod_{\mathbf{x}\in\mathcal{S}(u)} p(\mathbf{x}|u) \\ &= -\sum_{u\in\mathcal{C}} \sum_{\mathbf{x}\in\mathcal{S}(u)} \log p(u)p(\mathbf{x}|u) \\ &= n(t)\log n(t) + \sum_{u\in\mathcal{C}} -n(u)\log n(u) \\ &\quad + \sum_{\mathbf{x}\in\mathcal{S}(u)} \log p(\mathbf{x}|u). \end{aligned}$$

The term $n(t)\log n(t)$ is constant for all trees representing $\mathcal{S}(t)$ and so may be dropped from $I(\mathcal{S}|\mathcal{C})$. As

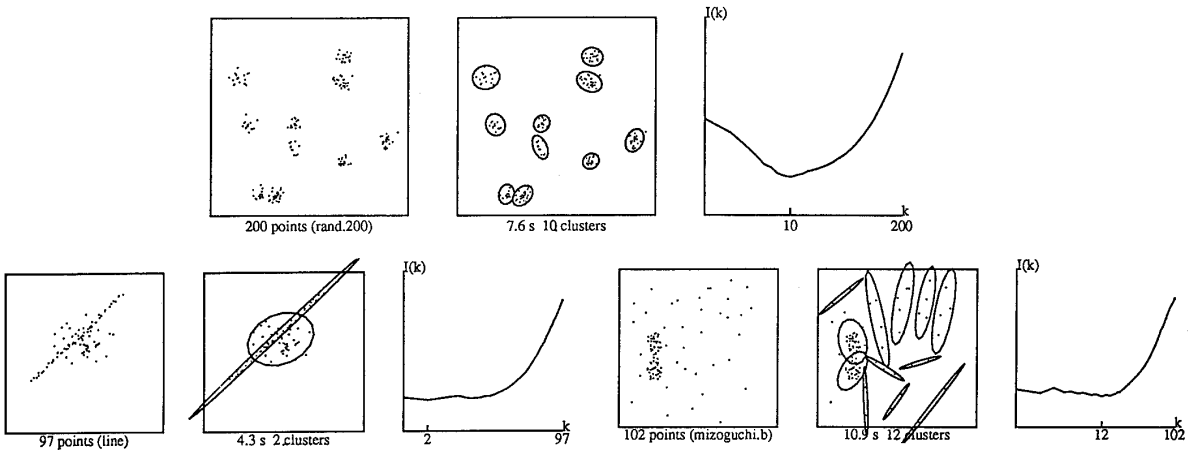

Figure 2: Finding the minimum description length clustering. The input (left) a set of 200 points selected randomly from each of 10 bivariate normals. The minimum description length (right) occurs where the number of clusters equals the number of classes in the input data. The detected clusters are shown in the center.

suming $p(\mathbf{x}|u)$ is normal with covariance $\mathbf{C}(u)$, the quantity

$$\sum_{\mathbf{x}\in\mathcal{S}(u)} \log p(\mathbf{x}|u) = (n(u)-1)H(u). \qquad (5)$$

where H(u) is given by equation (4).

Because we represent a clustering $\mathcal{C}$ by a pruned cluster tree t, our formulation of the description length $I(\mathcal{S}) = I(\mathcal{S}(t)) = I(t)$ is recursive.

$$I(t) = \begin{cases} -n(u)\log n(u) + (n(u)-1)H(u) \\ \text{if } u \text{ is a leaf node and} \\ h(u) + I(l) + I(r) \\ \text{otherwise.} \end{cases} \qquad (6)$$

The quantity $h(u)$, summed over the nodes u of the pruned tree t equals $I(\mathcal{C})$. Before giving its exact formulation, we digress to discuss our choice of cluster representations. One assumption we slipped in just above is that the distribution $p(\mathbf{x}|u)$ is normal. In fact the points $\mathcal{S}(u)$ need not be drawn from a normal distribution for the relation (5) to hold. The normal parameters $\mathbf{C}(u)$, the covariance matrix, and $\mathbf{m}(u)$, the mean vector, merely serve to specify a coordinate system centered at $\mathbf{m}(u)$ and directed parallel to the eigenvectors of $\mathbf{C}(u)$. In that coordinate system, the information needed to specify the points $\mathcal{S}(u)$ is given by (5). Without making any assumptions about the distributions of the $\mathcal{S}(u)$, we choose to represent the clusters by their means $\mathbf{m}(u)$ and covariances $\mathbf{C}(u)$.

Suppose we are given the sizes $n(v)$ of the leaf nodes v of a cluster tree. For each internal node u having children l and r, $n(u) = n(l) + n(r)$. Through recursive addition we can obtain the sizes of all the internal nodes. Thus, the size field is redundant information in the cluster tree data structure. What is perhaps not so obvious is that the mean vectors $\mathbf{m}(u)$ of all nodes u, except the root mean $\mathbf{m}(t)$, are also redundant information. All that is required are a bit vector $\mathbf{b}(u)$ at each internal node, all the covariance matrices, the leaf node sizes and the root mean. The application of the bit vector $\mathbf{b}(u)$ is explained below. We show the mean vector of the child nodes can be derived from the mean of the parent, given the sizes and covariances of the three nodes.

For each internal node u having children l and r the relation between their means is

$$\mathbf{m}(u) = q\mathbf{m}(l) + (1-q)\mathbf{m}(r) \qquad (7)$$

where $q = n(l)/n(u)$. Similarly the covariance matrix

$$\mathbf{C}(u) = q\mathbf{C}(l) + (1-q)\mathbf{C}(r) + q(1-q)\mathbf{v}\mathbf{v}^T \qquad (8)$$

where $\mathbf{v} = \mathbf{m}(l) - \mathbf{m}(r)$. From (7) and (8) we can derive the expression

$$(1-q)/q\mathbf{C}(u) - (1-q)\mathbf{C}(l) - (1-q)^2/q\mathbf{C}(r) = \mathbf{n}\mathbf{n}^T. \qquad (9)$$

where $\mathbf{n} = \mathbf{m}(l) - \mathbf{m}(u)$. This expression enables us to solve directly for the *magnitude* of the components of $\mathbf{n}$, but not their sign. Now we need the assumption above that the bit vector $\mathbf{b}(u)$ of *sign* information is available at each parent node u. This bit vector disambiguates the sign of the terms in $\mathbf{n}$. Having obtained the value of $\mathbf{n}(l)$, we can derive $\mathbf{m}(l)$ and $\mathbf{m}(r)$.

The description length of the root mean $\mathbf{m}(t)$ is constant over all trees representing $\mathcal{S}(t)$, so we can ignore this term when calculating $I(\mathcal{C})$. Also constant over all trees is the information needed to represet the root cluster. What does vary with the tree are the description length of the covariance matrices, so we rewrite the recursive expression for cluster description length

$$I(t) = \begin{cases} -n(u)\log n(u) + (n(u)-1)H(u) \\ \text{if } u \text{ is a leaf node and} \\ d - \log p(\mathbf{C}(l), \mathbf{C}(r)|\mathbf{C}(u)) + I(l) + I(r) \\ \text{otherwise.} \end{cases} \qquad (10)$$

The term d captures the number of bits needed for the bit vector u. The quantity $-\log p(\mathbf{C}(l), \mathbf{C}(r)|\mathbf{C}(u))$ measures the information gained at the node u when expanding the covariance matrix $\mathbf{C}(u)$ into its child components $\mathbf{C}(l)$ and $\mathbf{C}(r)$. We made an assumption in order to approximate $p(\mathbf{C}(l), \mathbf{C}(r)|\mathbf{C}(u))$. We assume that $p(\mathbf{C}(l), \mathbf{C}(r)|\mathbf{C}(u))$ depends only on the "volume" of the sets $\mathcal{S}(l)$ and $\mathcal{S}(r)$, expressed by $|\mathbf{C}(l)|$ and $|\mathbf{C}(r)|$, relative to the parent volume $|\mathbf{C}(u)|$. Thus we write $p(\mathbf{C}(l), \mathbf{C}(r)|\mathbf{C}(u))$ as the bivariate distribution $p(|\mathbf{C}(l)|/|\mathbf{C}(u)|, |\mathbf{C}(r)|/|\mathbf{C}(u)|)$ defined over the unit square. (Strictly speaking it is possible for one of $|\mathbf{C}(l)|$ or $|\mathbf{C}(r)|$ to be larger than $|\mathbf{C}(u)|$, but this has never happened in practice).

We estimated $p(|\mathbf{C}(l)|/|\mathbf{C}(u)|, |\mathbf{C}(r)|/|\mathbf{C}(u)|)$ through a simple learning procedure. For example, we collected a sample of 50 typical two-dimensional clustering problems ranging in size from 24 to 417 points. *NIHC* generated cluster trees for each of them. Then, for each internal node u in each of the resulting trees, we calculated $|\mathbf{C}(l)|/|\mathbf{C}(u)|$ and $|\mathbf{C}(r)|/|\mathbf{C}(u)|$. Our program accumulated these values in a 2-d histogram serving to approximate the density function. Since the trees are oriented, the histogram is symmetric. There is a significant ridge along the diagonal line where $|\mathbf{C}(l)| = |\mathbf{C}(r)|$. This ridge was due to our choice of representations for the leaf nodes. Rather than calculate the $\mathbf{C}(u)$ at nodes where $\mathcal{S}(u)$ was nondegenerate, we assigned a small nonzero value to $\mathbf{C}(u)$ at the leaf nodes. Since all the leaf nodes were assigned the same small variance, half the internal nodes have the property $|\mathbf{C}(l)| = |\mathbf{C}(r)|$.

Our method to apply (10) is a simple greedy algorithm. Call the set of leaf nodes of a pruned cluster tree t the "frontier set". Initially the frontier set consists of just the root node t. At each step, the greedy algorithm computes $I(t)$ by treating the nodes in the frontier set as the leaf nodes in (10). Then, it splits that node from the frontier set which yields the biggest decrease (or smallest increase) in (10). The split node is replaced in the frontier set by its children. The algorithm stops when the frontier set contains all $n(t)$ leaf nodes. At each step k the frontier set contains k clusters, so we can plot a graph like the one on the right in figure 2. The minimum occurs where k equals the number of classes in the original data.

4 Conclusion

The MDL formulation of the clustering problem eliminates all free parameters. This paper has illustrated an example of MDL clustering that takes advantage of a minimum entropy cluster tree to formulate the description length of a level clustering. The cluster trees generated by our hierarchical procedure, *NIHC* , are coordinate frame invariant and take into account covariance relationships between the variables. Also, *NIHC* can separate clouds of points that overalp. These properties make *NIHC* suitable for clustering applications where the clusters are not compact and well-separated. Although our minimization procedure is heuristic, it reaches lower local minima than other procedures having no better time complexity. Our experiments indicate that the trees generated by *NIHC* contain natural level clusters, and that these clusters can be extracted by a simple greedy procedure that minimizes description length.

References

[1] Andrew Appel. An efficient solution for many-body simulation (or, cray performance from a vax). Technical Report CMU-CS-TR-83-118, Carnegie Mellon, 1983.

[2] D.M. Boulton and C.S. Wallace. An information measure for single-link classification. *The Computer Journal*, Vol. 18, No. 3,, 1975.

[3] Peter Cheeseman. private communication.

[4] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, 1973.

[5] Yvan Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, Vol. 3, No. 1,, 1989.

[6] J. Rissanen. A universal prior for integers and estimation by description length. *Annals of Statistical Sciences*, Vol. 11, No. 2, pp. 416–31, 1983.

[7] J. Rissanen. *Minimum description length principles*, volume 5, pp. 523–7. Wiley, 1987.

[8] Arthur C. Sanderson and Nigel J. Foster. Attributed image matching using a minimum representation size criterion. In *Proceedings of 1989 Conference on Robotics and Automation*, pp. 360–5, 1989.

[9] C.S. Wallace and D.M. Boulton. An information measure for classification. *The Computer Journal*, Vol. 11,, 1968.

[10] Richard S. Wallace. *Finding Natural Clusters through Entropy Minimization.* PhD thesis, Carnegie Mellon, 1989.