Published in the Proceedings of the International Symposium on Assembly and Task Planning

# Complete Indexing Strategies for Sparse Sensing Techniques

Aaron S. Wallack *
Cognex Corp.,
Natick, Massachusetts

John F. Canny †
Computer Science Division,
University of California, Berkeley

## Abstract

*In structured environments found in manufacturing, a few precise measurements are often sufficient to recognize and localize modeled objects. Modeled objects can be recognized and localized by enumerating interpretations of the sensed data in terms of model features, and then validating each hypothesized interpretation by computing the optimal pose estimate and associated error. The task of enumerating hypotheses is termed the correspondence problem, and it is difficult to perform efficiently because only a small fraction of possible interpretations are valid. Indexing is a general approach for solving the correspondence problem in which coordinates are distilled from the sensed data, and then these indexing coordinates are quantized to index a table entry containing the valid interpretations. Indexing table are central to indexing techniques; if the table is complete, i.e., it contains an entry for every valid combination of indexing coordinates and interpretations, then the indexing strategy is correct. For sparse sensing strategies, where each experiment only provides a few measurements, indexing table completeness is critical. The task of constructing complete indexing tables has previously been an open problem. In this paper, we describe a method for constructing complete indexing tables that problem which involves enumerating cells in an arrangement in configuration space.*

## 1 Introduction

Industrial manufacturing is mainly concerned with accuracy, speed, cost, and flexibility. Manufacturing environments often exhibit enough structure so that a few measurements are sufficient to perform complex tasks such as recognition and registration; furthermore strategies which rely on simple sensors can realize high performance in terms of speed, precision, size, computational requirements, and cost [12] due to the high performance capabilities of simple sensors.

*Supported by Fannie and John Hertz Fellowship and NSF FD93-19412

†Supported by NSF FD93-19412

Recently developed systems have proved the effectiveness of *sparse sensing techniques* in which only a few measurements are extracted in each experiment [3]. At first glance, simple devices seem incapable of providing flexibility, judicious use of simple devices can indeed provide both performance and flexibility. Light beam sensors (Figure 1) which detect the presence of an occluding object have been used to recognize and localize various objects.

Many manufacturing operations, such as assembly, machining, and parts feeding involve model based object recognition and localization. Model based object recognition involves determining which model from a set of candidate models best explains the sensed data, and model based localization involves determining the pose for a given object which best explains the sensed data.

Wallack *et al.* used a set of crossed light beam sensors (Figure 2) to recognize and localize generalized prismatic objects to 0.025 millimeters in 5 microseconds [14]. Paulos and Canny used the crossbeam sensor to infallibly perform peg in hole insertions under tight tolerances (0.025 millimeters) [10]. Wallack and Canny used a set of parallel light beam sensors (Figure 3) to recognize and localize generalized polyhedral objects to 0.025 millimeters in 0.1 seconds [13].

Many machine vision systems perform recognition and pose estimation in two steps: first interpret some sensed features as model features, and second, for each interpretation, estimating the object's pose. The former task is known in the machine vision literature as the model based correspondence problem.

Indexing is a general approach to the model-based correspondence problem in which indexing coordinates are distilled from the sensed data, and then quantized to index a table entry containing the correspondence information. For correctness, the indexing table should contain table entries for all valid combinations of quantized indexing coordinates and correspondence interpretations. For dense sensing applications, where each experiment produces inordinate amounts of data, indexing table completeness is a minor issue since only

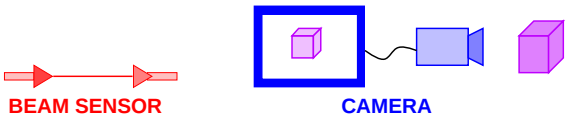


Figure 1: Light beam sensors, which only detect the absence or presence of a modulated signal, are much simpler devices than cameras.

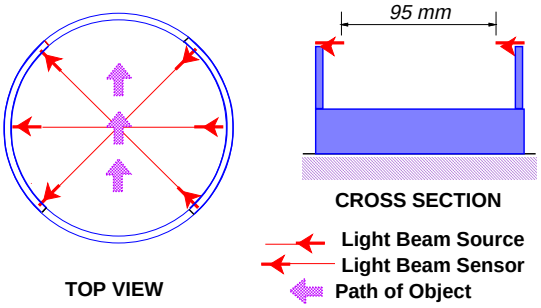


Figure 2: Crossbeam sensors can recognize and localize swept volume objects to 0.025 mm.

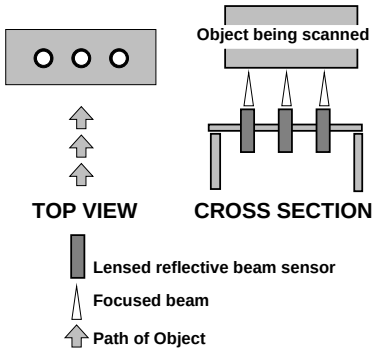


Figure 3: Scanning beam sensors can recognize and localize objects to 0.025 mm.

one of multiple cues needs to be correctly interpreted.

For sparse sensing strategies, complete indexing tables are crucial because each experiment only produces a few indexing vectors, and thereby, incomplete indexing tables can result in misclassification. In this paper, we describe an approach for constructing complete indexing tables for sparse sensing strategies; this approach is applicable to dense sensing strategies as well [12]. The approach generates complete indexing table entries by enumerating cells in an arrangement [5] in configuration space.

### 1.1 Outline

The rest of the paper is organized as follows. In section 2 we define the correspondence problem, and in section 3, we describe indexing approaches to the correspondence problem. In section 4, we present the theoretical framework for our construction methodology. In section 5 we outline our methodology for constructing complete indexing tables. In section 6, we step through the construction of a complete indexing table for a system with one degree of freedom, and we outline indexing table construction for a system with two degrees of freedom in section 7; in section 7, we also describe a recognition system which uses this indexing table construction methodology. In section 8, we discuss implementation issues, and we conclude by summarizing the results and advantages of this technique.

## 2 Correspondence Problem

The model-based correspondence problem is the task of interpreting sensed features in terms of model features. We now present an example of a correspondence problem in order to explain the concept more fully. Consider the two models $(\Delta ABC, \Delta DEF)$ shown in Figure 4, and a bounding box sensor which perceives the smallest axis aligned rectangle containing a two-dimensional object in an arbitrary pose (Figure 5). The correspondence problem involves determining which model features correspond to the sensed features. For this system, the sensed features are the four sides of the rectangle. In other words, the correspondence problem involves mapping from sensed data to feature interpretation(s) (Figure 6).

Figure 7 depicts $(\Delta ABC)$ in an orientation for which the predicted height and width are consistent with the bounding box; in this case, the interpretation is $(B,B,C,C)$ which refers to the the right rectangle edge contacting vertex $B$, the top rectangle edge contacting vertex $B$, the left rectangle edge contacting vertex $C$ and the bottom rectangle edge contacting vertex $C$.

Most of the previous work in the correspondence problem has been done in the context of dense sensing strategies, where each experiment provides an inordinate amount of data, (camera based machine vision). Unfortunately few of these results are transferable to sparse sensing techniques. Most machine vision systems are feature-based, *i.e.,* they extract interesting features from the data (vertices, edges, moments, ...) to recognize and localize objects. Feature-based approaches are not well suited to sparse sensing techniques because there is a very low probability that interesting features will be observed.

## 3 Indexing

Indexing (Figure 8) is a general approach to the model-based correspondence problem in which index-

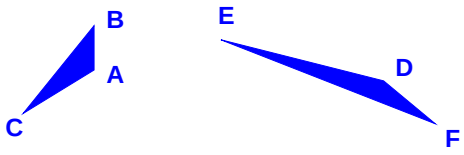


Figure 4: Two object models ($\Delta ABC, \Delta DEF$) used in our example of the correspondence problem

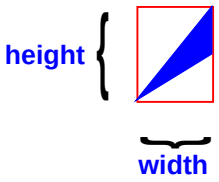


Figure 5: The bounding box sensor perceives the bounding box (smallest axis aligned rectangle containing an object) for an object at a given orientation

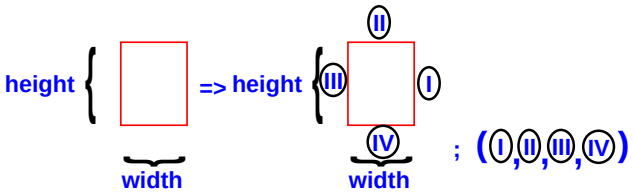


Figure 6: The correspondence problem involves determining the model features corresponding to the sensed features

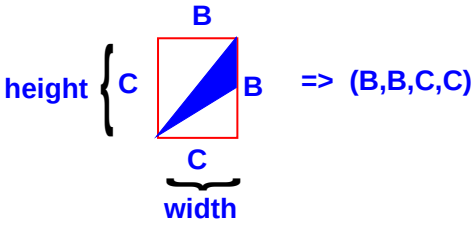


Figure 7: $(B, B, C, C)$ is a valid interpretation of the sensed data because the predicted and measured heights and widths agree.

ing coordinates are distilled from the sensed data, and these indexing coordinates are quantized to index a table entry containing the valid interpretations. Figure 8 depicts the indexing process for the bounding box sensor (described in section 2). Indexing tables lie at the heart of indexing techniques; if the table is complete, *i.e.*, it contains an entry for every valid combination of indexing coordinates and interpretations, then the indexing technique is correct.

Indexing relies on the assumption that the predicted data and the experimentally measured data both are quantized to the same indexing coordinates;

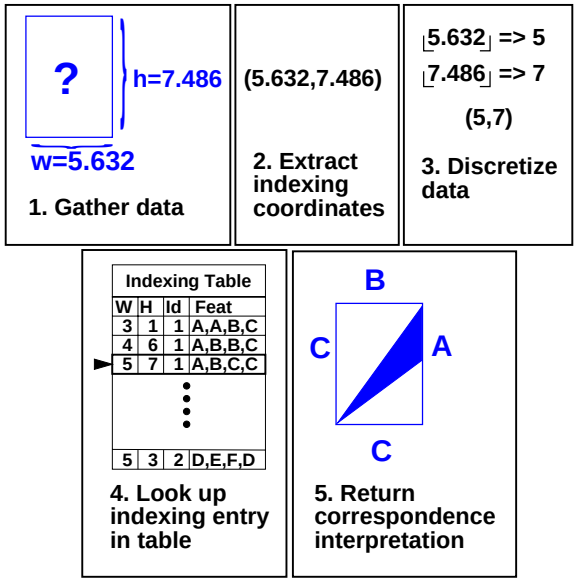


Figure 8: In indexing, the sensed data is discretized to index a table entry containing the associated feature correspondences.

this can be achieved by choosing the discretization resolution to be greater than the anticipated error. Indexing is practical for sparse sensing strategies because it can achieve real-time performance, and can exploit sensors' high precision via a fine table resolution.

The two main advantages of indexing are overconstrained data, and offline preprocessing. By utilizing overconstrained data, only a small subset of indexing coordinates are consistent with each interpretation, and, therefore, all of the indexing coordinates for different interpretations (and different models) can be combined into a single table. When a single table contains all of the entries for all of the interpretations, a single lookup can simultaneously compare the sensed data to all of the possible interpretations.

Indexed lookup tables have been used to recognize two-and-a-half-dimensional objects from two-dimensional images [1, 7, 8, 11], three-dimensional objects from two-dimensional images [4, 6], and two- or three-dimensional objects from two- or three-dimensional data [8]. Indexing has mainly been used in conjunction with invariants, where an invariant is a characteristic which remains constant irrespective of the object and camera pose [1, 2, 6, 7, 8, 11]. Invariants enable each interpretation to be characterized by a single indexing table entry.

Indexing is not limited to systems which exhibit invariants, but otherwise, indexing table construction is more difficult because multiple indexing ta-

ble entries must be enumerated for each interpretation [4, 12, 13, 14]. Many systems do not exhibit invariance: there are no generic invariants for generic point groups under perspective projection [2] or weak-perspective projection of generic three-dimensional data [4]. Indexing table completeness has been a minor issue because many indexing systems rely on invariants, and because image data usually contains multiple cues, of which only a single cue needs to be correctly interpreted. For this and other reasons, the task of enumerating all valid indexing table entries has remained an open problem [4].

# 4 Theoretical Framework

In this section, we define terms, view indexing from a geometric perspective, and analyze conventional indexing table construction strategies.

## 4.1 Definitions

**Definition 1** A *Configuration* $\vec{x}$ characterizes the observable state of a system, *i.e.,* the object's pose with respect to the camera. *Configuration space* (C) refers to the set of all configurations ($\vec{x} \in$ C). For example, under unscaled three-dimensional orthographic projection, configuration space can be parameterized by $(x, y, \theta, \phi, \psi)$; $z$ is not a configuration variable because it is unobservable. $k$ refers to the dimensionality of configuration space ($k = dim($C$)$).

**Definition 2** *Indexing coordinate* $y_i$ refers to a real-valued measurement distilled from the sensed data; indexing vector $\hat{y}$ refers to a tuple of $n$ indexing coordinates. *Indexing space* ($\mathbb{I}$) refers to the set of all possibly observed vectors ( $\hat{y} \in \mathbb{I}$, $n = dim(\mathbb{I})$).

**Definition 3** *Predicted indexing coordinate function* $\mathbb{X}_f^i$ predicts the $i^{th}$ indexing coordinate given the interpretation as a function of configuration. $\mathbb{X}_f^i$ provides a mapping from configuration space to indexing space: $\mathbb{X}_f^i$ : C $\rightarrow \mathbb{I}$. $\mathbb{X}$ is shorthand for the vector function which maps configurations to indexing coordinates, $\hat{y} = \mathbb{X}_f(\vec{x})$.

**Definition 4** An *Interpretation manifold* $\mathcal{M}_f$ for a given interpretation $f$, is the set of all predicted indexing coordinates: $\mathcal{M}_f \subseteq \mathbb{I}$. A model stratification $\mathcal{MS}_O$ refers to the union of all the interpretation manifolds consistent with interpretations from a given object model $O$ ($\mathcal{MS}_O = \bigcup_{f \in O} \mathcal{M}_f$).

**Definition 5** An *Indexing table entry* corresponds to a valid combination of predicted indexing coordinates and predicted correspondence interpretations.

### 4.1.1 Geometric View of Correspondence Problem

Viewed geometrically, the correspondence problem can be defined as: *determine the interpretation manifold(s)* $\mathcal{M}_f$ *nearby a given indexing coordinate.* For the bounding box sensor and two-dimensional geometric models (section 2), the orientation, $\theta$, was the sole degree of freedom: $\vec{x} = (\theta); C = S^1$. The only measured values were width and height and they were used as the indexing coordinates: $\hat{y} =$ (width, height $); \mathbb{I} \subseteq R^2$. For each interpretation $f$, the function $\mathbb{X}_f^0(\theta)$ predicts the bounding box width as a function of theta, and the function $\mathbb{X}_f^1(\theta)$ predicts the bounding box height as a function of theta.

Figure 9 depicts the two model stratifications for the two object models ($\Delta ABC, \Delta DEF$) from Figure 4. In Figure 9, each interpretation manifold is colored differently. Figure 10 graphically depicts the correspondence problem as the task of determining the interpretation manifold closest to the given indexing vector.

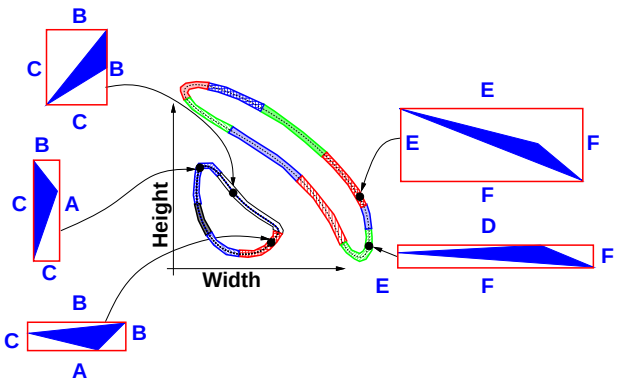


Figure 9: Two one-dimensional model stratifications for the two objects $\Delta ABC, \Delta DEF$ immersed in a two-dimensional indexing space.

### 4.1.2 Geometric View of Quantizing Indexing Coordinates

In indexing techniques, the indexing coordinates are quantized to form integral indices to be used to index (hash) table entries in an indexing table. Quantizing the indexing coordinates partitions the indexing space into equivalence classes of discretization hypercubes, where all the points in each discretization hypercube quantize to the same integral coordinates.

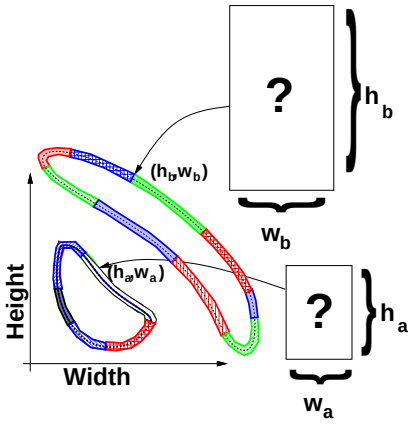


Figure 10: Objects are identified by finding the interpretation which best matches the measured indexing coordinate.

### 4.2 Conventional Indexing Table Construction Methods: Configuration Space Sampling

In this section, we discuss and analyze conventional methods for constructing indexing tables which include regular and random configuration space sampling. In these configuration space sampling approaches, index table entries are predicted for sample configurations, and these predicted entries form the indexing table. Clemens and Jacobs [4] used regular configuration space sampling to generate indexing tables to identify polyhedral objects from image data. Regular sampling has the drawback that a poor choice of sampling can produce undesirable results (*i.e.,* sampling $sin(x)$ at intervals of $\frac{\pi}{2}$). Random sampling, on the other hand, cannot suffer from a poor sampling choice. Indexing tables can be constructed via random sampling in two ways: sampling a fixed number of configurations, or sampling as many configurations as necessary to achieve desired coverage (which can be determined via Monte Carlo techniques). Indexing table coverage can be quantitatively defined in terms of the probability that an indexing table entry for randomly chosen configuration will be found in the indexing table.

### 4.3 Quantitative Indexing Table Coverage Analysis for Conventional Approaches

In this section, we analyze the indexing table coverage achieved by randomly sampling a fixed number of configurations, and we use this analysis to predict the number of samples necessary for achieving a desired coverage $\kappa$. Assume each table entry $t_i$ has probability $p_{t_i}$ of being observed by a randomly sampled configuration. In order for $t_i$ to be absent from the indexing table, $t_i$ needs to be missed by all $n$ random samples. The probability that all $n$ trials will miss indexing table entry $t_i$ is $(1-p_{t_i})^n$, which can be approximated by $e^{-p_{t_i}n}$ (equation (1)).

$$(1-p_{t_i})^n = (1-\frac{np_{t_i}}{n})^n; \lim_{n\to\infty}(1-\frac{np_{t_i}}{n})^n = e^{-p_{t_i}n} \quad (1)$$

$$\frac{d\, p_{t_i}e^{-p_{t_i}n}}{dp_{t_i}} = (1-np_{t_i})e^{-p_{t_i}n} = 0 \Rightarrow p_{t_i} = \frac{1}{n} \quad (2)$$

After constructing the indexing table, consider the probability that an upcoming table entry $t_i$ will be absent from the indexing table. This probability is approximately $\sum_{t_i} p_{t_i}e^{-p_{t_i}n}$, and this sum is maximized when the individual components of the sum are maximized. This occurs when $p_{t_i} = \frac{1}{n}$ (equation (2), $\frac{dp_{t_i}e^{-p_{t_i}n}}{dp_{t_i}}|_{p_{t_i}=1/n} = 0$). When $p_{t_i} = \frac{1}{n}$, the sum probability $\sum_{t_i} p_{t_i}e^{-p_{t_i}n}$ reduces to $\frac{|\mathcal{T}|}{ne}$, where $|\mathcal{T}|$ refers to the number of table entries in the complete table. Therefore, in the worst case, $\frac{|\mathcal{T}|}{(1-\kappa)e}$ random samples are necessary to achieve $\kappa$ coverage, *i.e.,* $\approx 100e|\mathcal{T}|$ entries are necessary to achieve 99% coverage.

## 5 Complete Indexing Table Construction Methodology

In this section, we describe an efficient method for constructing complete indexing tables. This method involves two reductions: first, the problem of constructing a complete indexing table is reduced to the problem of constructing a complete regular cell (discretization hypercube) covering of the interpretation manifolds, and, second the problem of constructing complete cell coverings of the interpretation manifolds is reduced to the problem of enumerating cells in an arrangement in configuration space.

**Lemma 1 A complete indexing table corresponds to the union of cell coverings of all of the interpretation manifolds**

Lemma 1 is based upon the observation that each interpretation is only consistent with the discretization hypercubes intersecting the interpretation manifold. Recall that every indexing table entry corresponds to a unique combination of indexing coordinates and interpretations. Therefore, we can enumerate all the valid indexing table entries by enumerating a cell covering of the interpretation manifolds.

**Lemma 2 The cell coverings of the interpretation manifolds can correspond to the cells in the arrangement in configuration space of the projections of indexing space boundaries provides a complete cell covering**

Lemma 2 allows us to perform the cell enumeration in configuration space rather than higher-dimensional indexing space. The intersection of interpretation manifolds and discretization hypercubes separate the model stratification into patches where every indexing table entry corresponds to a finite number of these patches. Viewed on the model stratification these patches are separated by two types of boundaries: discretization boundaries, the intersections of the discretization hypercubes and the model stratification, and interpretation boundaries, intersections of interpretation manifolds. These indexing space boundaries can be projected down onto configuration space. Thereby, we can enumerate the cells in the arrangement in configuration space rather than indexing space.

### 5.1 Discretization Boundary Curves

Discretization boundary curves (DB-curves) in C-space characterize the configurations $\{\vec{x}\}$ for which a predicted indexing coordinate straddles two discretization regions. Assuming we round numbers down, *i.e.,* $\lfloor 2.999 \rfloor \rightarrow 2$ and $\lfloor 3.001 \rfloor \rightarrow 3$, $y_i = 3.0$ (in general $y_i = k\delta | k \in \mathbb{Z}$) forms a discretization boundary. DB-curves are of the form $\mathbb{X}^i(\vec{x}) = 3.0$ ($\mathbb{X}^i(\vec{x}) = k\delta$), and DB-curves can be implicitly defined by $Q(\vec{x}) = (\mathbb{X}^i)^{-1}(3.0)$, or in general $Q(\vec{x}) = (\mathbb{X}^i)^{-1}(k\delta)$.

### 5.2 Correspondence Boundary Curves

Correspondence boundary curves (CB-curves) characterize the configurations $\{\vec{x}\}$ which straddle two interpretation regions $\vec{x} \in \mathcal{M}_f \cap \mathcal{M}_{f'}$. CB-curves can be defined implicitly as $R(\vec{x}) = 0$ for some function $R$.

### 5.3 Intersecting Boundary Curves

If these DB-curves and CB-curves are defined implicitly as the roots of equations, then we can compute the intersections of these curves by finding simultaneous solutions to multivariate systems of equations. Furthermore, if these curves are defined algebraically, then we can use a combination of classical algebraic methods and numerical techniques to solve these non-linear multivariate systems exactly in constant time [9].

## 6 One Degree of Freedom Systems

Figure 11 shows a complete regular cell covering and associated indexing table entries of the model stratification for $\Delta ABC$. Indexing table entries are attributable to either different discretizations or different interpretations. We can enumerate all the indexing table entries by enumerating at least one witness configuration for each indexing table entry. The witnesses can be the vertices of the cells corresponding to each indexing table entry; *i.e.,* intersections of the model stratification and the hyperplanes defining the discretization hypercube boundaries, and boundary points on the interpretation manifolds (Figure 12). Notice that each indexing table entry corresponds to a patch of the model stratification which corresponds to a patch of configuration space. In configuration space, these patches are bounded by projections of the indexing space boundaries (Figure 13). These patches all correspond to cells in the arrangement formed by the projected boundary curves.

## 7 Multiple Degree of Freedom Systems

Our complete indexing table methodology generalizes to configuration spaces with arbitrary degrees of freedom. In this section, we describe an application of our approach for a system with two degrees of freedom. Consider a scanning beam sensor system, where $b$ parallel beam sensors move with respect to an object (Figure 14).

### 7.1 Normalization

For generic systems, the size of the indexing tables, *i.e.,* the number of entries, is $O((\frac{n}{\delta})^k)$ where $n$ is the number of indexing coordinates, $\delta$ is the discretization resolution, and $k$ is the dimensionality of the configuration space. Since the table size grows exponentially in the size of the configuration space, it is desirable to parameterize configuration space in terms of the fewest variables. It turns out that in some cases, some of the degrees of freedom are irrelevant with respect to the correspondence interpretation, and can be ignored. The bounding box sensor (sections 2,4.2) measured height and width which were independent of position, and only depended upon the orientation $\theta$. Had we instead observed four absolute measurements, $x_{min}, x_{max}, y_{min}, y_{max}$, the indexing coordinates would not have been independent of $x, y$ and we would have had to deal with a three dimensional configuration space.

The scanning sensor generates scanline endpoints which characterize the robot configurations where the beams first become occluded or last reconnect. Assume without loss of generality that the object travels along the $x$ direction; notice that the object's $x$ position along the path does not provide any correspondence relevant information (Figure 15). Therefore we are free to insist that particular scanline endpoint occurs at $x = 0$. This leaves only $2b - 1$ indexing coordinates which depend only upon $y$ and $\theta$.

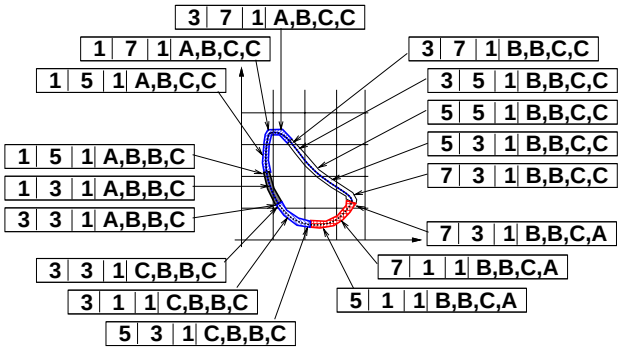


Figure 11: A complete cell covering (associated indexing table entries) for $\Delta ABC$.

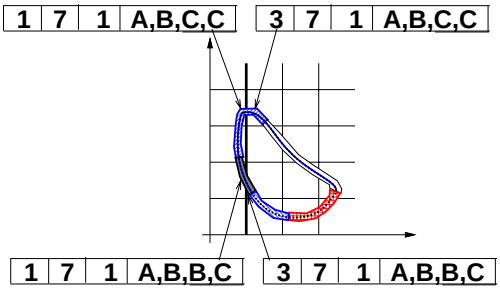


Figure 12: For systems with a single degree of freedom, we can enumerate all indexing table entries by enumerating witnesses at every configuration where the discretized coordinates or interpretations change.

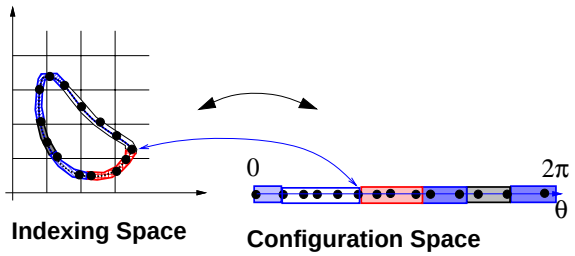


Figure 13: We can also view the patches corresponding to the individual indexing table entries in terms of cells in configuration space.

### 7.2 Methodology

Again, since each indexing table entry corresponds to a valid combination of quantized indexing coordinates and interpretation, we can partition the model stratification in terms of the discretization hypercubes and correspondence interpretation. In this case, each indexing table entry corresponds to a finite number of patches of the model stratification. Rather than enumerate these patches in indexing space, we can enumerate all of these patches in the lower-dimensional

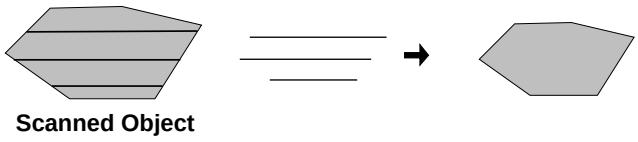


Figure 14: Scanning sensing is a system with two degrees of freedom for which we can construct complete indexing tables.

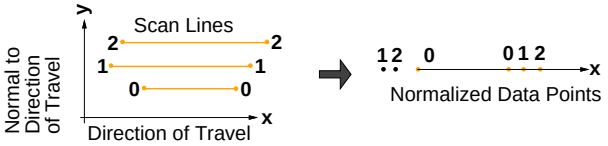


Figure 15: The $2b-1$ normalized scanline endpoints (produced by insisting that a particular scanline endpoint occurs at $x = 0$) can be used as indexing coordinates.

configuration space. This is accomplished by projecting the intersections of the discretization hyperplanes (bounding the discretization hypercubes) down onto DB-curves in configuration space, and by projecting the correspondence boundaries down onto CB-curves in configuration space. Each cell in the arrangement bounded by the DB-curves and CB-curves corresponds to a patch on the model stratification. By enumerating the cells in the arrangement defined by these DB- and CB- curves, we enumerate all of the patches of the model stratification, and therefore enumerate all of the indexing table entries.

Again, we only need to enumerate witness configurations for each of these cells, and these witness configurations can correspond to the $k$-wise intersections of curves. Figure 16 shows the correspondence boundaries and the discretization boundaries in indexing space, and Figure 17 shows the correspondence boundaries and the discretization boundaries in configuration space. For scanning sensors, the correspondence boundary curves characterize configurations where a scan line contacts two features at the coincident vertex. Discretization boundaries characterize configurations for which $x_i = k\delta | k \in \mathbb{Z}$, where $\delta$ is the discretization.

### 7.3 Experimental Performance

The indexing strategy discussed in this paper has been used to recognize and localize objects. Wallack *et al.* used the indexing construction method described in this report to recognize generalized polyhedral objects from scanning sensor data [12, 13, 15]. The scanning beam sensor apparatus consisted of six lensed beam sensors. Each experiment produced at most 11

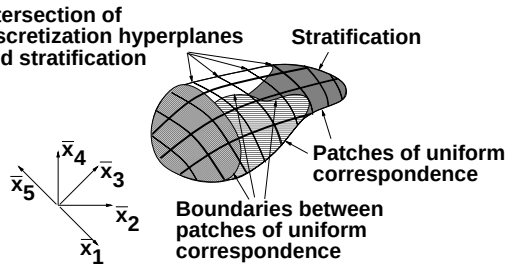


Figure 16: For the two dimensional model stratification parameterized by $(\theta, y)$ in indexing space, each indexing table entry corresponds to either different discretized indexing coordinates or different correspondence interpretations. Additionally, each different indexing table entry corresponds to at least one patch on the model stratification.

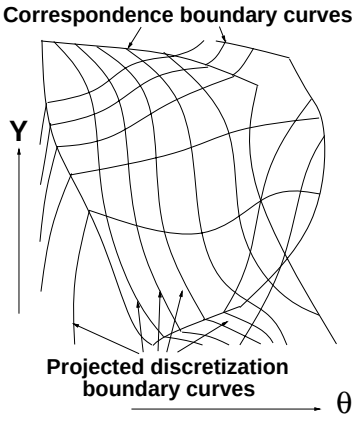


Figure 17: We can project the boundaries down onto configuration space where the cells defined by the arrangement of DB-curves and CB-curves correspond to all the indexing table entries.

indexing coordinates. We constructed indexing tables with relatively fine discretizations (1 mm) relative to the sensor's precision of 0.02 mm, and object sizes of $20 - 50$ mm. For these discretizations, the indexing table returned, on average, a handful ($\approx 2$) of plausible interpretations. Each hypothesized interpretation was then verified via pose estimation. The system achieved 100% recognition performance in 500 trials involving ten different objects, and achieved 0.025 mm positional and $0.1^o$ orientational repeatability.

# 8 Implementation Issues

In this section, we discuss three important implementation issues: sweep algorithms, verifying completeness of indexing tables, *i.e.,* correctness of the implementation, and debugging the implementation. It turns out that the latter two issues are related, in that absent indexing table entries determined in the verification step highlight fallacies and provide leads for debugging purposes.

## 8.1 Enumerating Cells in an Arrangement via Sweep Algorithms

In this section we describe sweep algorithms which are classical approaches for solving computational geometry problems (*i.e.,* enumerating the cells defined by an arrangement of curves). In sweep algorithms, a representation of a cross-section of the space is maintained as the cross-sectional plane is swept across the space. For systems which depend only upon local interactions, the sweep algorithm only needs to consider curves which are adjacent in the cross-section. Thereby, sweep algorithms reduce $d$-dimensional problems to a finite number of $d-1$-dimensional subproblems; this dimensional reduction can be recursively applied to end up with low dimensional subproblems.

Depending upon the application, sweep algorithms may be extremely sensitive to numerical imprecision, because the representation of the cross-section can become inconsistent or incorrect. This is an extremely important issue for our applications, since we may rely on numerical techniques to compute intersections of implicitly defined curves [9].

For constructing complete indexing tables, we only need to enumerate witnesses, *i.e.,* configurations at $k$-wise intersections of curves. Therefore, we can use a simpler algorithm (Figure 18) which is less sensitive to numerical imprecision than a sweep algorithm. This approach involves partitioning configuration space into hypercubes and marking every hypercube which contacts each curve. After marking all such hypercubes, we only need to consider tuples of curves which contact the same configuration space hypercube. Furthermore, we can use the information provided in each hypercube to facilitate prediction of indexing table entries for arbitrary configurations. If the hypercube does not contact any discretization curves for the $i^{th}$ coordinate, then we know that value of the indexing coordinate for every configuration within the hypercube, and therefore we do not have to recompute it. Similarly, if the hypercube does not contact any correspondence boundary curves for the $j^{th}$ sensed feature, then we know the interpretation of all configurations within the hypercube.

## 8.2 Verifying Indexing Table Completeness

For the scanning beam sensor, we used random sampling to test the correctness of our technique, *i.e.,* the completeness of our indexing tables. We synthe-

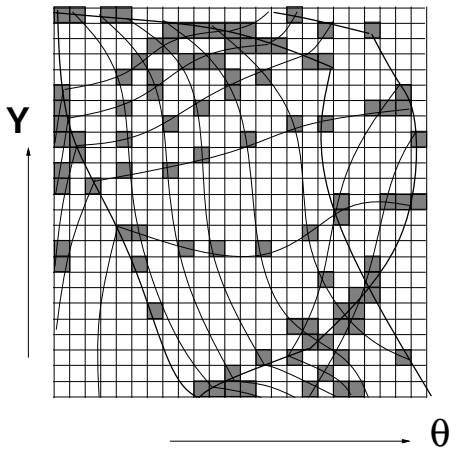


Figure 18: Robust hypercube based method for conservatively estimating combinations of curves which may intersect. The hypercube method also facilitates prediction of indexing table entries for arbitrary configurations.

sized hundreds of random configurations, predicted indexing table entries for these configurations, and then verified that the indexing table entries were indeed in the indexing table.

### 8.3 Debugging Issues

If we run across a valid indexing table entry which is absent from the indexing table, the associated configuration is an important lead towards determining why the table entry failed to be enumerated. Our implementation is supposed to enumerate witnesses at every zero-dimensional intersection of $k$ configuration space curves. If we can find a zero-dimensional intersection and associated indexing entry which was supposed to be (but not) enumerated, then we can focus specifically on the associated curves. Starting with the associated configuration, we can perform a fine grained constrained search to locate an intersection configuration with the same indexing table entry characteristics as the initial configuration.

## 9 Conclusion

In this paper we presented a general indexing strategy for constructing complete indexing tables for sparse sensing systems. Indexing is a method for solving the correspondence problem in which coordinates are distilled from the data, and these coordinates are discretized to index a table entry containing the correspondence information. Our construction technique involves enumerating cells in an arrangement in configuration space. This approach has been used to construct complete indexing tables to recognize and localize objects from scanning sensor data in constant time.

**Acknowledgements**

The authors wish to acknowledge: Dina Berkowitz for her assistance, Prof. Dinesh Manocha for valuable help in producing this paper, Prof. Raimund Seidel, Prof. David Forsyth, Francesca Barrientos, and Eric Paulos for helpful criticisms and suggestions.

## References

[1] Nicholas Ayache and Oliver D. Faugeras. Hyper: A new approach for the recognition and positioning of two-dimensional objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54, 1986.

[2] J. Brian Burns, Richard S. Weiss, and Edward M. Riseman. View variation of point-set and line-segment features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):51–68, 1993.

[3] John Canny and Ken Goldberg. RISC robotics. In *IEEE International Conference on Robotics and Automation*, 1994.

[4] D. T. Clemens and D. W. Jacobs. Space and time bounds on indexing 3-d models from 2-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1007–1017, 1991.

[5] Herbert Edelsbrunner. *Algorithms in combinatorial geometry*. EATCS monographs on theoretical computer science. Springer-Verlag, 1987.

[6] D. Forsyth, L. Mundy, A. Zisserman, A. Heller, and C. Rothwell. Invariant descriptors for 3-d object recognition and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:971–991, Oct 1991.

[7] Alan Kalvin, Edith Schonberg, Jacob T. Schwartz, and Micha Sharir. Two-dimensional model-based boundary matching using footprints. *International Journal of Robotics Research*, 5(4):38–55, 1986.

[8] Yehezkel Lamdan, Jacob T. Schwartz, and Haim J. Wolfson. Object recognition by affine invariant matching. *International Conference of Computer Vision*, 1988.

[9] Dinesh Manocha. *Algebraic and Numeric Techniques for Modeling and Robotics*. PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1992.

[10] Eric Paulos and John Canny. Accurate insertion strategies using simple optical sensors. In *IEEE International Conference on Robotics and Automation*, pages 1656–1662, May 1994.

[11] Jacob T. Schwartz and Micha Sharir. Identification of partially obscured objects in three dimensions by matching noisy characteristic curves. *International Journal of Robotics Research*, 6(2):29–44, 1987.

[12] Aaron Wallack. *Algorithms and Techniques for Manufacturing*. PhD thesis, University of California at Berkeley, 1995.

[13] Aaron Wallack and John Canny. Object recognition and localization from scanning beam sensors. In *IEEE International Conference on Robotics and Automation*, 1995.

[14] Aaron Wallack, John Canny, and Dinesh Manocha. Object localization using crossbeam sensing. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 692–699, May 1993.

[15] Aaron Wallack and Dinesh Manocha. Robust algorithms for object localization. Technical report, University of North Carolina, Chapel Hill, 1994.