Pergamon

Robotics & Computer-Integrated Manufacturing, Vol. 10, No. 5, pp. 377–389, 1993
Copyright © 1994 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0736-5845/93 $6.00+0.00

● *Paper*

GROUP TECHNOLOGY CLASSIFICATION AND CODING SYSTEM DESIGN: CONSIDERATIONS AND EXTENSION TO ROBOTIC END-OF-ARM-TOOLING

CHARLES T. MOSIER,* ROBERT A. RUBEN† and LAKSHMINARAYANA R. TALLURU‡

*Box 5790, School of Management, Clarkson University, Potsdam, NY 13699-5790, U.S.A.; †Operations Management Department, Graduate School of Business, Indiana University, Bloomington, IN 47403, U.S.A. and ‡School of Business, Pennsylvania State University at Erie, Erie, PA 16563-1400, U.S.A.

This paper discusses a number of the basic notions associated with the development and implementation of group technology oriented classification and coding systems. As well, an example system, designed to performed design retrieval and reference for a subset of tooling required for manufacture, is presented. In particular, this system was developed for retrieving and referencing the database of robotic end-of-arm-tooling (EOAT) designs.

The discussion includes the overall code development strategy, the template for the interactive graphical user interface, and the details of the proposed EOAT classification and code structure. The graphics interface and the overall retrieval system were implemented on the INTERGRAPH MicroStation environment. These modules were written in the C programming language, with prototype development accomplished on a microcomputer. Finally, examples are used to demonstrate the capabilities of the system, and discussion concerning implementation issues is included.

1. INTRODUCTION

The traditional drive for the development and implementation of group technology (GT) classification and coding systems in the engineering design and manufacturing arenas comes primarily from the current inability of existing systems to effectively and efficiently disseminate information concerning what is being designed (and ultimately manufactured) within a given firm. Any industrial design group has and will experience occasions where a given designer will be beginning a new design task, immediately after another designer, possibly in close proximity, has finished and "logged" a design which is very similar to that associated with the new design task. It is possible the new design could be efficiently accomplished by very minor modification to the existing design, an approach resulting in extremely high levels of savings, particularly in a computer-aided-design (CAD) environment.

The key is to address the balance between efficiency and effectiveness in this process of information dispersal. Given a CAD environment, the required information may be *effectively* transferred from designer to designer via the CAD database. The designer assigned to the new design task could peruse, via the tools available in the CAD system, the old designs in the database to find one similar to that which is required. This is clearly not efficient. Some design groups have hundreds of thousands of designs available in their database. In most cases it is simpler to begin the new design from scratch (or alternatively ask someone in the design group with extensive experience for potential old designs for reference).

The GT classification and coding is an approach directed at making this dissemination of information more efficient. The notion is to identify each entity design within the database using a multidigit number, embodying sufficient information concerning the nature of the entity, to make directed searching of the database viable, *and preferable to designing from scratch.*

The increased efficiency directly associated with the productivity of the corporate design function is usually the primary economic motivation associated with GT classification and coding system implementation, but it is not the only place for gain. Efficiency gained by reducing the design time is likewise reflected downstream, throughout all of the planning-oriented tasks required before ultimate manufacture of the design. This expansion requires extremely well connected databases as well as a classification and coding system developed in the language of engineering design *and* all other planning functions. Most often the ultimate form of the system is some compromise of the needs of these functions.

This paper discusses some of the facets of the theoretical basis for a GT classification and coding system design with consideration of its primary functional requirements, i.e. (a) basic retrieval, (b) broad-based usability by all interested parties within the firm, and (c) adaptability to sophisticated numerical pattern recognition approaches. As well, a prototype system for dissemination of information concerning robotic end-of-arm-tooling (EOAT) designs is presented. Section 2 presents the theoretical considerations. Section 3 presents the structure of the EOAT retrieval system, and Section 4 presents information concerning the form of the computerization of the system. Section 5 discusses a number of implementation issues and solutions, and Section 6 presents concluding remarks.

2. SOME BASIC CONSIDERATIONS OF GT CLASSIFICATION AND CODING SYSTEM DEVELOPMENT

For detailed surveys of GT and associated benefits see Mosier and Taube,[9] and Stylinger and Melkanoff.[11] What is severely lacking in the literature is the extension of the GT classification philosophy beyond the piece part level to other manufactured entity populations. Only a handful of papers (e.g. Refs 1, 7, and 8) describe such extensions. A recent paper,[2] discussing the relationship of GT classification concepts to more general database abstractions, provides some theoretical justification for extending GT classification and coding application beyond the problem domain of piece parts.

Styslinger and Melkanoff[11] propose distinct definitions for *classification* and *coding*, terms which are frequently used synonymously. In particular, they define classification as *a technique to organize related data into logical and systematic order using a specific set of criteria that categorizes similar items together.* Further, they define coding as *the arbitrary assignment of one or more symbols to a part, which when deciphered communicates specific meaning or intelligence.* The result is a code which can be used to identify an entity (or set of entities) according to criteria which are of particular interest. Development of the code for a particular entity, based upon numericallly describing the level of each attributes which best "measures" that entity, is not unlike a set of rank order *statistics* describing the subject of an experiment, i.e. information is aggregated so meaningful comparisons can be made, but a significant amount of information is lost.

Development efforts in the area of GT classification and coding of piece parts are extensive, ranging from experience based, e.g. the OPITZ system,[10] to systems custom tailored to the user's particular needs, e.g. MIClass.[4] A number of GT classification and coding principles have been proposed (Refs 4–6 and 10) for individual pieceparts, and can be used as the basis for classification and code system development. The following sections are an expanded discussion of each of these principles.

2.1. *Completeness of the system*

The format of a classification and coding system must be broad enough to include all entities in the defined population. It should be sufficiently flexible to allow for updating to include new items not currently in the population. The operational form of this requirement is not to require that *all* entities (parts, assemblies, components, or end-items) are necessarily coded within a single classification and coding system. The notion of a "defined" population prohibits this interpretation of completeness. The entity subpopulation being addressed by the system is determined by the code developer, and may be defined as any subset of the corporate design population.

The intrinsic meaning of completeness in this context is in reference to the designated subpopulation. This subpopulation is to be "viewed" via a "lens" (or rose-colored glasses, as the reader prefers) in the form of a GT classification and coding system; thus the subpopulation must be presented in its entirety. This is a case where adhering to Pareto's law, i.e. focusing efforts on the 80% of a population where benefit potential is high, is totally inappropriate. The most basic tenet of GT is to search for patterns in a population's attributes, and, subsequently, capitalize on these patterns. Without consideration of the whole population, the analyst's view of the patterns will most assuredly be incomplete.

This leads to the first example of the most basic difficulty associated with the philosophy *and* implementation of classification and coding systems. To sensibly define the subpopulation of interest, one must make *a priori* judgements concerning entity similarities and differences. This is just the tip of the iceberg; this "chicken or egg" problem tends to occur numerous times throughout the development process.

2.2. *Specification consistency*

The intent of any classification and coding system is to bring together entities in terms of their similarities and segregate them by their essential differences. The format used must be unambiguous and use clearly defined parameters so that items are consistently classified to the same place within the classification structure. One may consider a GT coding and classification structure as a subjective multivalued function, i.e. given a population of entities under consideration, denoted as $\mathscr{P}$, where a population entity may be generally denoted as p (thus $p \in \mathscr{P}$), then

$$f\colon \mathscr{P} \mapsto I_1 \times I_2 \times I_3 \times \ldots I_n \qquad (1)$$

where

$$f(p) = i_1, i_2, i_3, \ldots, i_n \qquad (2)$$

where I_l is the indexing set of levels of the lth code digit, and n is the number of "digits" within code structure. It is the requirement that f be a function, i.e. $f(p)$ is unique, which is basic to this notion of consistency.

2.3. *Accurate discrimination*

When a user seeks a specific group or family, there must be enough entities retrieved to provide reasonable selection. On the other hand, there should not be so many items such that the final search and retrieval time becomes excessive and onerous to the user.

The need to meaningfully segregate entities significantly impacts classification and coding system design and development. The most basic tenet of GT is the existence of entity *families*—the most natural form of which is all $p' \in \mathcal{P}$ such that $f(p) = f(p')$ (this is feasible since f is a *subjective* function as mentioned in the previous section). This level of family specification may be too restrictive for searching and retrieving from the population by specifying a unique element in the range of f, i.e. for a given $i_1, i_2, i_3, \ldots, i_n \in I_1 \times I_2 \times I_3 \times \ldots I_n$, there may be no entity p within the domain such that $f(p) = i_1, i_2, i_3, \ldots, i_n$. It is better to specify $i_1 \pm \varepsilon_1$, $i_2 \pm \varepsilon_2, i_3 \pm \varepsilon_3, \ldots, i_n \pm \varepsilon_n$ in a controlled manner, thus "zeroing-in" on a reasonable set of entities to consider.

The classification and coding system must be carefully designed to enable a process of "natural selection" via this manipulation of ε_i. For example, if $i_1 = 3$ and $\varepsilon_1 = 1$, the system will restrict its retrieval to those entities where, minimally, $2 \leq i_1 \leq 4$. The implication is that entities with $i_1 = 2$, 3, or 4 are, to some degree, "close" to each other. Due to the intrinsically non-numeric nature of many of the criteria this can be a difficult task indeed.

For example, consider an attribute which is **internal features of a gear**, possibly including: (a) a key-way, denoted by **K**, (b) a drilled and tapped hole for a locking screw or bolt, denoted by **H**, and (c) internal threads, denoted by **T**. If **K** is coded with $i_1 = 2$, **H** is coded with $i_1 = 3$, and **T** is coded with $i_1 = 4$, the search restriction mentioned above, i.e. $2 \leq i_1 \leq 4$ will include designs with all of these features. As well, the system tacitly assumes that **K** is "closer" to **H** than it is to **T**. Thus, for each digit within the code, the context for determining the level assignment must be studied very carefully. In the case of **K**, **H**, and **T**, it may be that the changeover required to manufacture each of the features may provide the best estimate of the closeness between levels which is required. The unfortunate fact is that this facet of the classification and coding system is often prohibitively difficult or impossible to attain for all digits.

This notion may be expanded to encompass accuracy in the sense of the population distribution. For example, often an attribute is continuous, as with the length of an entity, say ranging from 1 to 1000 mm. Ten levels can be easily assigned to the code digit associated with this attribute. One option is to use the following breakdown of the associated number of entities (with the specified frequency of occurrence):

Length (l)	Code level	Frequency of occurrence
$l \leq 10$	0	347
$10 < l \leq < 20$	1	640
$20 < l \leq < 30$	2	12
$30 < l \leq < 40$	3	0
$40 < l \leq < 50$	4	0
$50 < l \leq < 60$	5	0
$60 < l \leq < 70$	6	0
$70 < l \leq < 80$	7	0
$80 < l \leq < 90$	8	0
$90 < l$	9	1

With a simple adjustment to the code structure we have the following:

Length (l)	Code level	Frequency of occurrence
$l \leq 7$	0	90
$7 < l \leq < 8$	1	66
$8 < l \leq < 9$	2	87
$9 < l \leq < 10$	3	105
$10 < l \leq < 11$	4	114
$11 < l \leq < 12$	5	113
$12 < l \leq < 13$	6	106
$13 < l \leq < 14$	7	91
$14 < l \leq < 15$	8	72
$15 < l$	9	156

Clearly, this represents a more useful breakdown.

2.4. *Corporate consistency*

A classification and coding system should be directed at the ultimate objectives of the user. This principle reflects the need to direct the selection of digit criteria to selecting "salient" features or attributes, where salient is directly linked to the needs of the user and the environment of the user. Overall this tends to impede the broad-based corporate application of GT codes. Historically codes have been developed for use in the manufacturing context, i.e. with attributes such as the *internal features of a gear*, as discussed in the previous section, playing a primary role (e.g. Opitz[10]). If a purchasing agent was attempting to search a design database for reference designs to be used as the basis for predicting the cost of externally sourcing a new design, his or her attributes of interest may be very different than those of shop personnel. Further, his or her notion of "closeness" of the attribute levels which define each code digit would reflect aspects of shop configuration and operating constraints associated with a target vendor, which may be significantly different from those in-house.

2.5. *User friendliness*

The requirement to be *user-friendly* goes beyond the traditional notion of being convenient for the user. When a particular design is coded and buried in the database for future retrieval, the code structure provides the only quick link back to the design. This is not unlike the notion of misnumbering a book being placed in a library. None of the tools for searching the database will retrieve such misnumbered books; thus, the book is effectively lost to the potential user. Thus, the portion of the system used for determining the specific code for a "new" entity must be foolproof to the extent feasible.

As well, the use of the system for design retrieval has proven to depend heavily upon its operational convenience. The best paradigm for use in the new product design area is that use is naturally motivated, rather than mandated by managerial edict. An option which has been demonstrated[8] to be successful is the use of a computerized graphics user interface (GUI).

2.6. *Adaptable to numerical analysis*

In addition to the previous principles of classification and coding system design and implementation, the GT classification and coding system *must be adaptable to numerical analysis techniques geared at determining meaningful partitions of the population*. In particular, it must be adaptable to future analysis via similarity comparisons, measures, and grouping algorithms. To enable the use of the coding structure for descriptive and prescriptive analysis of the engineering design database it is absolutely necessary that the coding structure be, to the extent feasible, constructed in a manner which will facilitate numerical analysis. The incorporation of two specific properties within the code structure greatly supports this adaptability to numerical analysis. These properties are termed (in this paper) *ordered* and *comparable*, and are discussed in the following.

The property termed *ordered* implies the existence of an *order* relation upon the code. To illustrate, consider an entity, designated as j_1, where the level assigned to its lth digit is denoted as $i_l(j_1)$, and the natural expansion to two more entities (denoted as j_2 and j_3), with lth digit assignment denoted as $i_l(j_2)$ and $i_l(j_3)$, respectively. An ordering assumes that (for example) if $i_l(j_1) \neq i_l(j_2)$, then $i_l(j_1) \prec i_l(j_2)$ or $i_l(j_1) \succ i_l(j_2)$. To further elaborate, we consider an example which does not include this property. Consider the number of EOAT or gripper orientation changes required for the insertion of a particular part. If we denote the required orientation changes by the following symbols, change in pitch $\rightarrow P$, change in yaw $\rightarrow Y$, and change in roll $\rightarrow R$, it is possible to construct the following code assignment level structure:

Rotation	$\emptyset$	P	R	Y	$P \cap R$	$P \cap Y$	$R \cap Y$	$P \cap Y \cap R$
Code level	0	1	2	3	4	5	6	7

where, for example, an insertion requiring changes in gripper pitch **and** roll orientation would be indicated by $P \cap R$. Since we cannot, in general, say that P is better than R, i.e. $P \prec R$, or vice versa, this structure is not ordered. Logically, in the terms of the notation of our example:

$$\emptyset \prec P = R = Y \prec P \cap R = P \cap Y = R \cap Y \prec P \cap Y \cap R \quad (3)$$

but this implies that, in terms of our code levels for this digit:

$$0 \prec 1 = 2 = 3 \prec 4 = 5 = 6 < 7. \quad (4)$$

Note the potential for inconsistency between a naive user perception of the order relationship between 4 and 5, i.e. numerically 4 is clearly less than 5, and the reality of this case, in terms of the attribute measured by this digit: 4 is equal to 5. The Y, P, R example does satisfy a somewhat weaker requirement, termed a *partial ordering*, where $i_l(j_1) \neq i_l(j_2)$, implies $i_l(j_1) \preceq i_l(j_2)$ or $i_l(j_1) \succeq i_l(j_2)$. Also, notice that, with an arbitrary assumption concerning the preferability of each of P, Y, or R, say $\emptyset \prec P \prec R \prec Y$, and with the imposition of a transitivity assumption, i.e. $P \prec R$ and $R \prec Y$ implies $P \prec Y$, we have an ordered structure. Finally, we note that the existence of an order or even a partial order is not always attainable, even with arbitrary assumptions, but to the extent possible the code structure should have levels which possess these properties.

The property termed *comparable* implies the existence of a metric which makes sense in terms of measuring how close two entities are to each other. In terms of the second example breakdown in Section 2.3, such a measure is simple to construct. For example, if the code level of 1, i.e. $(7 < l \leq 8)$, is compared with a code level of 7, i.e. $(13 < l \leq 14)$, their difference may be specified in terms of the physical attributes they measure, that is

$$\left| \frac{7+8}{2} - \frac{13+14}{2} \right| = |13.5 - 7.5| = 6$$

which happens to be identical to their code level difference, i.e.

$$|1 - 7| = 6.$$

Notice that this breaks down when one considers the extreme levels, i.e. 0 and 9. Clearly their "distance" is greater than $|0 - 9| = 9$.

The form of this metric may potentially be much more complex. Consider the pitch (P), roll (R), and yaw (Y) example from above. If we define our metric (generically denoted as $\|\cdot\|$) as the count of requirements *not* in common, we have the following "differences:"

	P	R	Y	$P \cap R$	$P \cap Y$	$R \cap Y$	$P \cap Y \cap R$
$\emptyset$	1	1	1	2	2	2	3
P		2	2	1	1	3	2
R			2	1	3	1	2
Y				3	1	1	2
$P \cap R$					2	2	1
$P \cap Y$						2	1
$R \cap Y$							1

For example, consider $A = P \cap R$ and $B = P \cap Y \cap R$. A indicates a need for a change in gripper pitch (P) and roll (R), and B indicates a need for all three gripper orientation changes. Thus, A and B match on the need for change in pitch (P) and roll (R), but do not match on the need for change in yaw (Y), thus our "difference" measure is $\|A - B\| = \|P \cap R - P \cap Y \cap R\| = 1$ because only one action, Y, of those required for either A or B is not required in both.

Another alternative might focus upon "similarity" rather than distance, i.e. if, with no loss of generality, we let A be one condition and B be the other, then let

$$\|A - B\| = \frac{\text{the number of actions which match}}{\text{the number of potential matches}}.$$

Again, if we let $A = P \cap R$ and $B = P \cap Y \cap R$, we have

$$\|A-B\| = \|P\cap R - P\cap Y\cap R\| = \tfrac{2}{3}.$$

Alternatively:

$$\|P - Y\cap R\| = \tfrac{0}{2} = 0.$$

In this section we discuss what we perceive to be a principle basic to the notion of constructing GT classification and coding systems for use beyond simple search and retrieval. Experience has proven that the attainment of both of these properties may not be feasible in a given application, but they are desirable. Also notice that they are not interrelated. Clearly the pitch, roll, and yaw does not (without the imposed assumption) have an ordering. Alternatively we see above two potential candidates for measuring similarity (or the antithetical distance) between two levels. Alternatively the example concerning the length dimension clearly possesses ordering, but comparability becomes extremely difficult to assess for the extreme level specifications.

3. PROPOSED GT CODE FOR EOAT

The GT code structure developed for this project is an 11-digit numeric description embodying the salient characteristics of robotic EOAT. Each code digit is used to designate a single "dimension," and the numerical value of the digit is used to designate the "level" associated with its given dimension. Thus we have a code description involving 11 dimensions, with the potential for 10 levels within each dimension. The details of each dimension and their levels are included in the following. In this discussion each digit is referenced to a representation of the GUI screen used in the system. These "screens" are described further in Section 4.

Digit 1: class specification. This digit indicates the subclass of the population of interest. The inclusion of this digit was to enable the expansion of the current system (i.e. directed at classifying and coding EOAT) to the population of entities broadly described here as "portable tooling." Since these entities typically represent significant cost expenditures for a firm and they are typically designed and manufactured in "ones and twos," often for custom applications, coding and retrieval of these designs hold great potential for cost reduction. To date, only the code for robotic EOAT, i.e. grippers, has been developed (see Fig. 1).

The rest of the code attempts a gradual progression from a part focus to an EOAT focus. The notion was that product design engineers (as well as the EOAT design engineers) could make good use of the system, possibly modifying assembly designs in minor ways which would reflect major reductions in the cost of assembly. In the next three digits, we see a portion of that progression.

Digit 2: part geometry. This digit indicates all possible geometric forms of the part requiring insertion. The levels of this dimension are fairly intuitive (see Fig. 2).

Digit 3: part contact configuration. This digit categorizes the EOAT by the method used to make contact with the part. Again, the levels of this dimension are fairly intuitive, requiring only a minor level of familiarity with basic notions associated with the use of robots (see Fig. 3).

Digit 4: Gripper–part adhesion method. This digit categorizes the method whereby the gripper–part

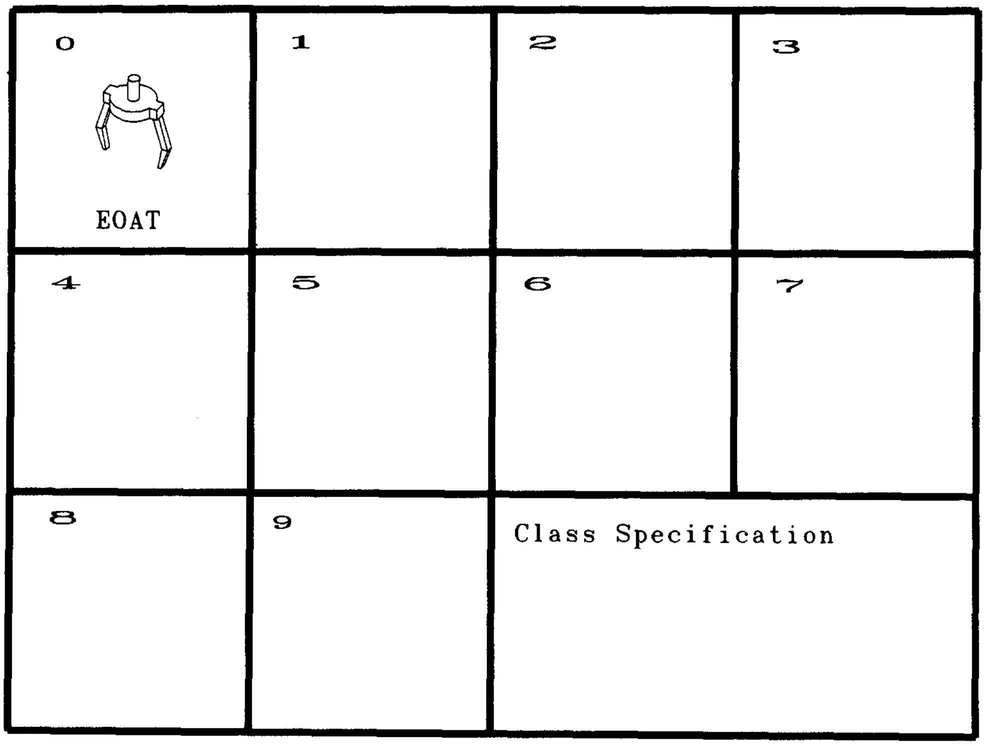


Fig. 1. GUI screen for selecting the class of tooling to code.

contact is *maintained* during the lift, linear and rotational motion, and insertion processes (see Fig. 4).

The next five digits describe information more closely associated with the robotic assembly environment, and its relationship with the general shape orientation of the assembly. Coding of these digits requires more detailed knowledge of the automated assembly setting as well as more knowledge of robotic assembly in general.

Digit 5: lateral movement required for insertion. This digit categorizes the distance that the part must be moved from the part feeder device to the site for insertion (see Fig. 5).

Digit 6: rotation required for insertion. This digit identifies the rotation required by the part for

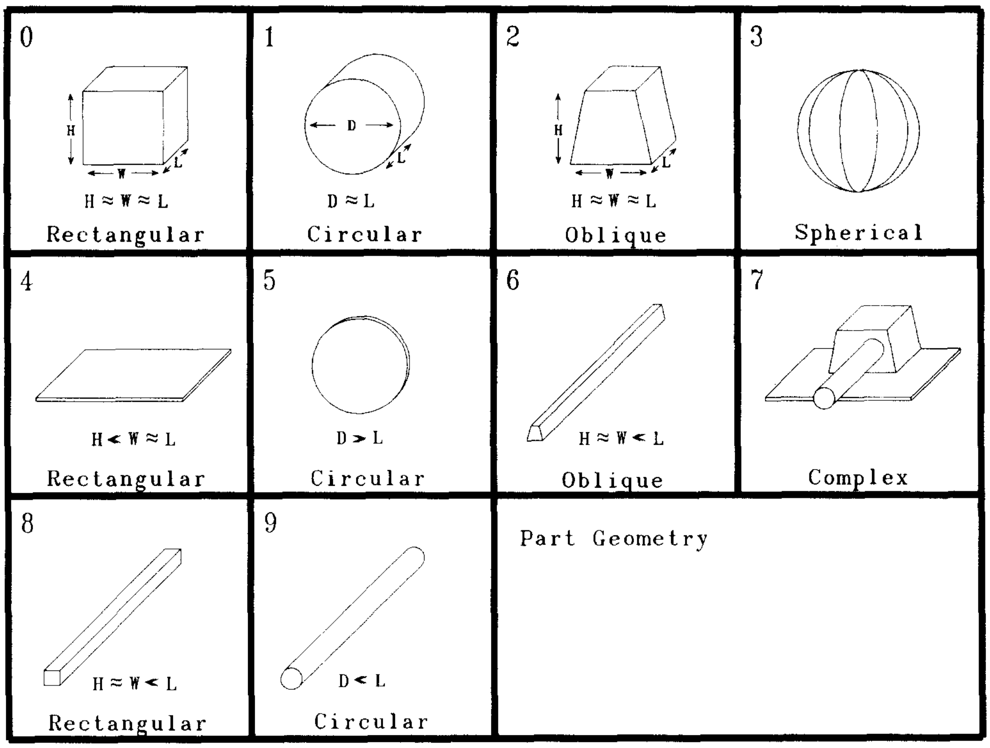


Fig. 2. GUI screen for selecting the geometry of the part to be robotically inserted.

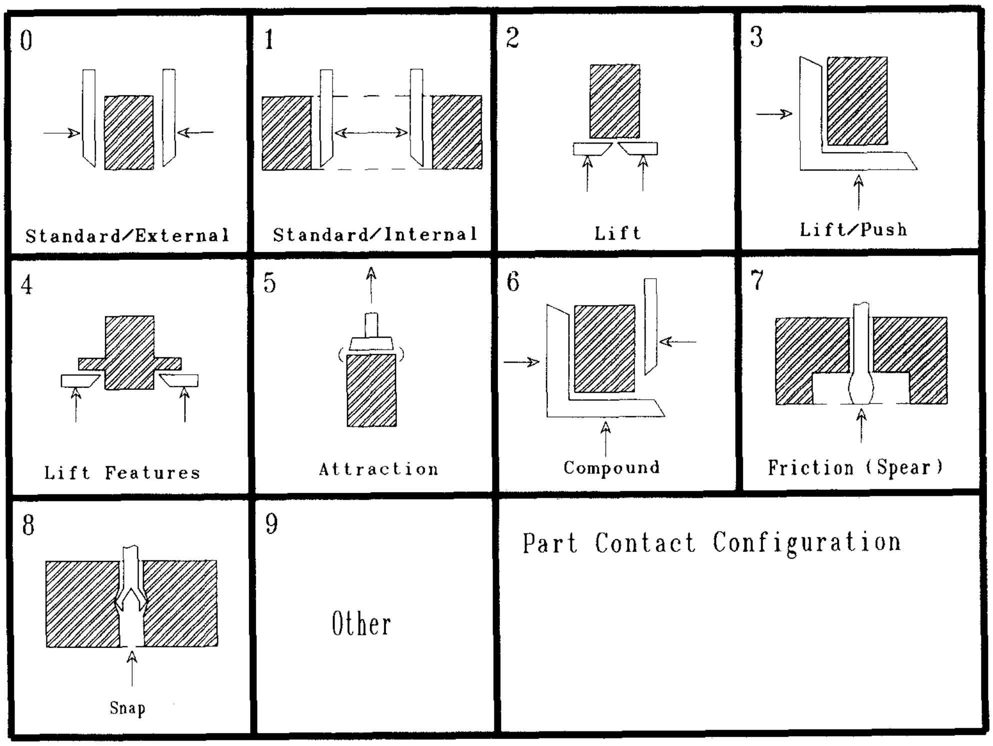


Fig. 3. GUI screen for selecting the part contact configuration.

insertion. Notice that this and the next three digits have a certain degree of redundancy (see Fig. 6).

Digits 7–9: yaw, pitch, and roll rotation required for insertion. These three digits categorize the degree of yaw, pitch, and roll rotation required to change the part from its feed orientation to its required insertion orientation, respectively (see Figs 7–9).

The next two digits are categorizations of attributes of the EOAT independent of the part being inserted. They reflect information concerning the features and attributes directly associated with those EOAT in the design database.

Digit 10: actuator power. This digit categorizes the

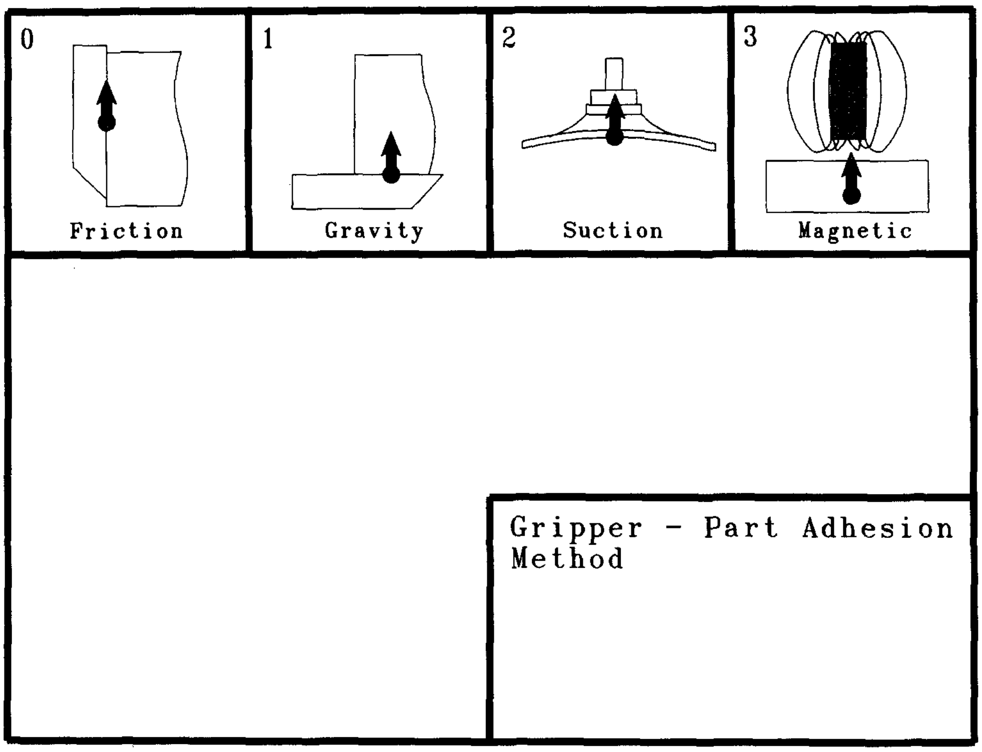
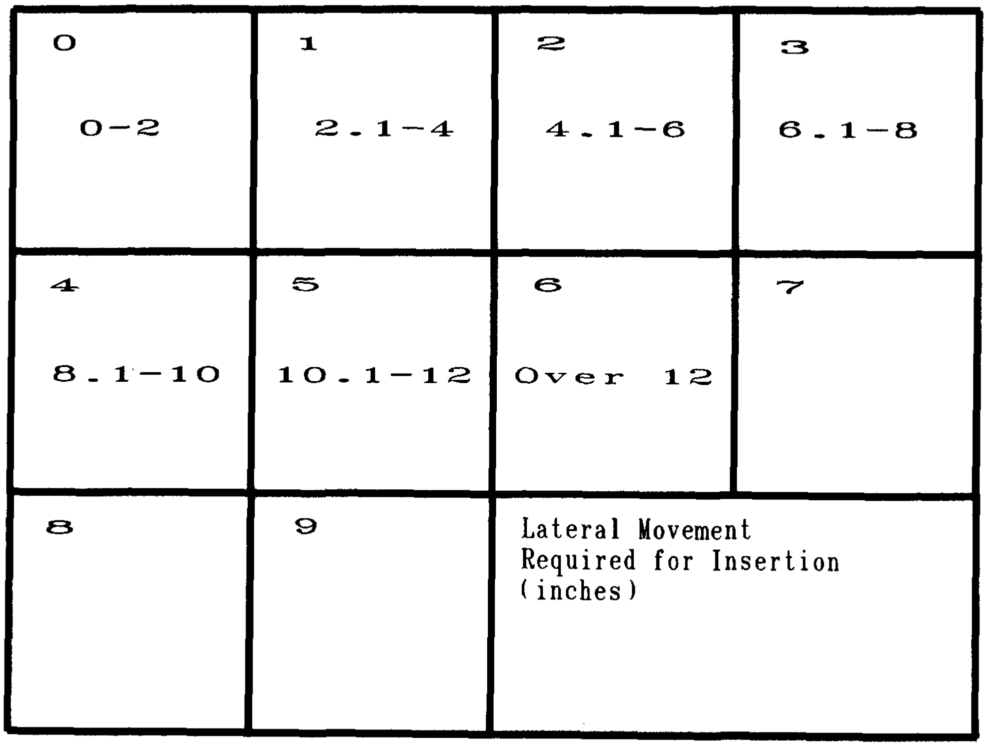


Fig. 4. GUI screen for selecting the gripper–part adhesion method.



0 0–2	1 2.1–4	2 4.1–6	3 6.1–8
4 8.1–10	5 10.1–12	6 Over 12	7
8	9	Lateral Movement Required for Insertion (inches)	

Fig. 5. GUI screen for selecting the lateral movement required for insertion.

required actuator motivation power source (see Fig. 10).

Digit 11: number of contact points. This digit categorizes the number of contact points, i.e. implicitly defining the number of gripper "fingers" (see Fig. 11).

An illustration of this code is as follows (using the 0–9 scale for indicating level):

1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	5	1	1	0	0	0	1

This particular number describes a robotic insertion task (and associated EOAT configuration) with the following characterization:

(1) it is EOAT;

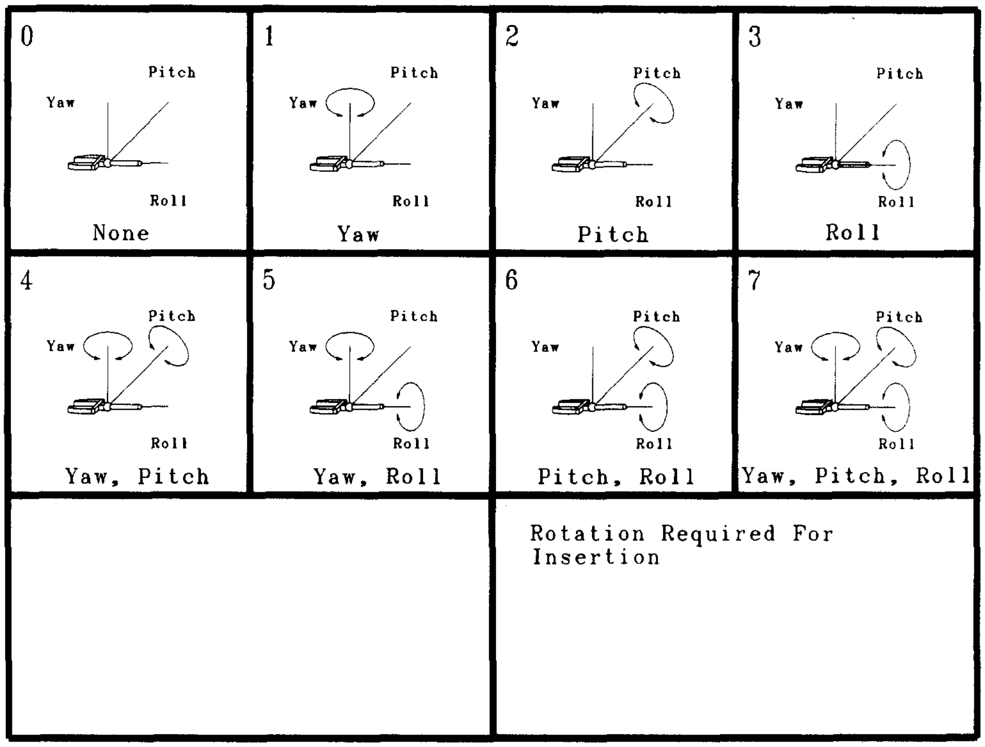


Fig. 6. GUI screen for selecting the rotation required for insertion.

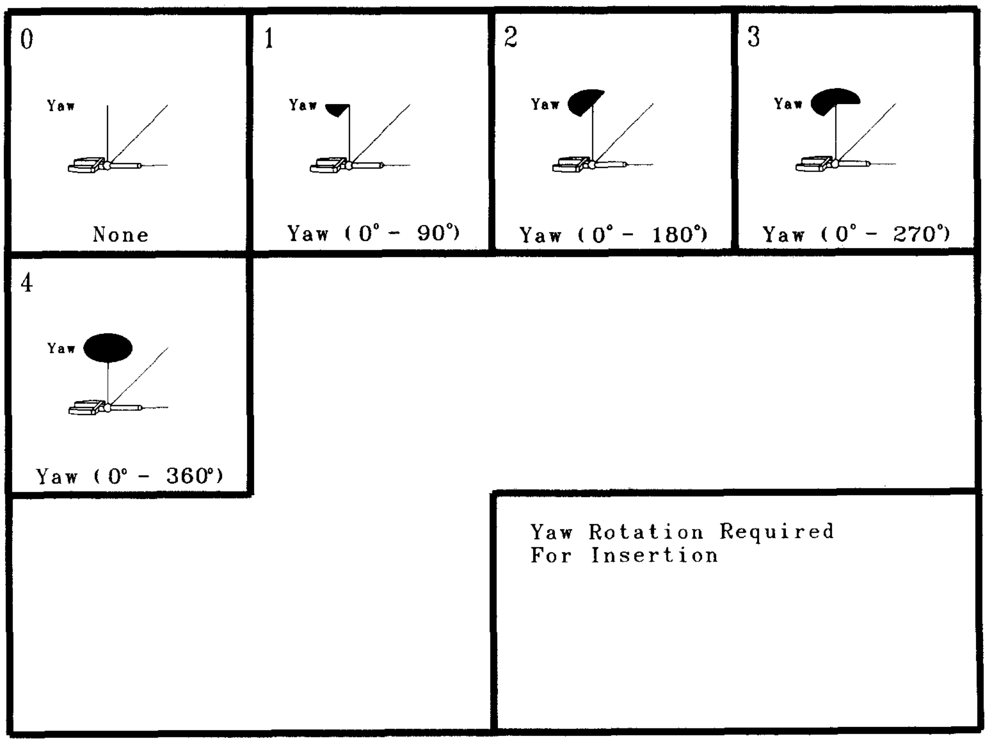


Fig. 7. GUI screen for selecting the yaw rotation required for insertion.

(2) a rectangular (nearly cubic) part requires insertion;
(3) the gripper will grasp the part externally, in a rather standard fashion;
(4) the maintenance of the grasp will be primarily by friction;
(5) the part must be moved from 10.1 to 12 in. from its feeder to the required assembly point;
(6) the orientation of the part must be rotated in only one direct, i.e. in terms of gripper orientation, a change in yaw is required;
(7) the change in yaw required is between 0 and 90°;
(8) no change in pitch is required;
(9) no change in roll is required;
(10) the actuator is pneumatically powered;
(11) there are two contact surfaces, i.e. two gripper fingers are required for grasping and holding the part.

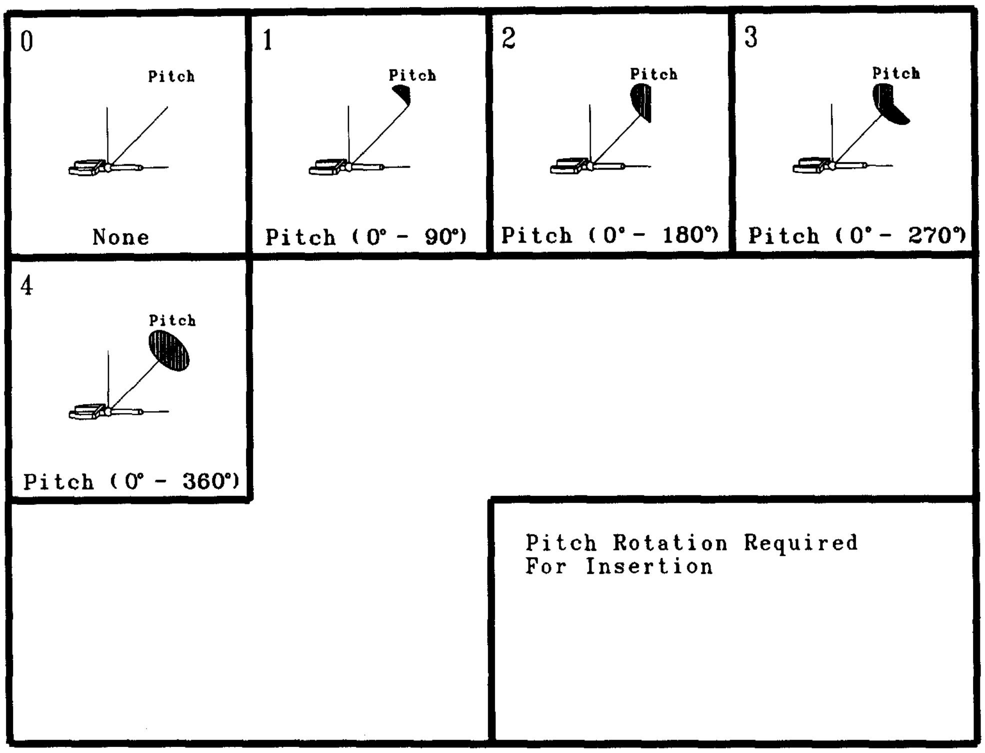


Fig. 8. GUI screen for selecting the pitch rotation required for insertion.

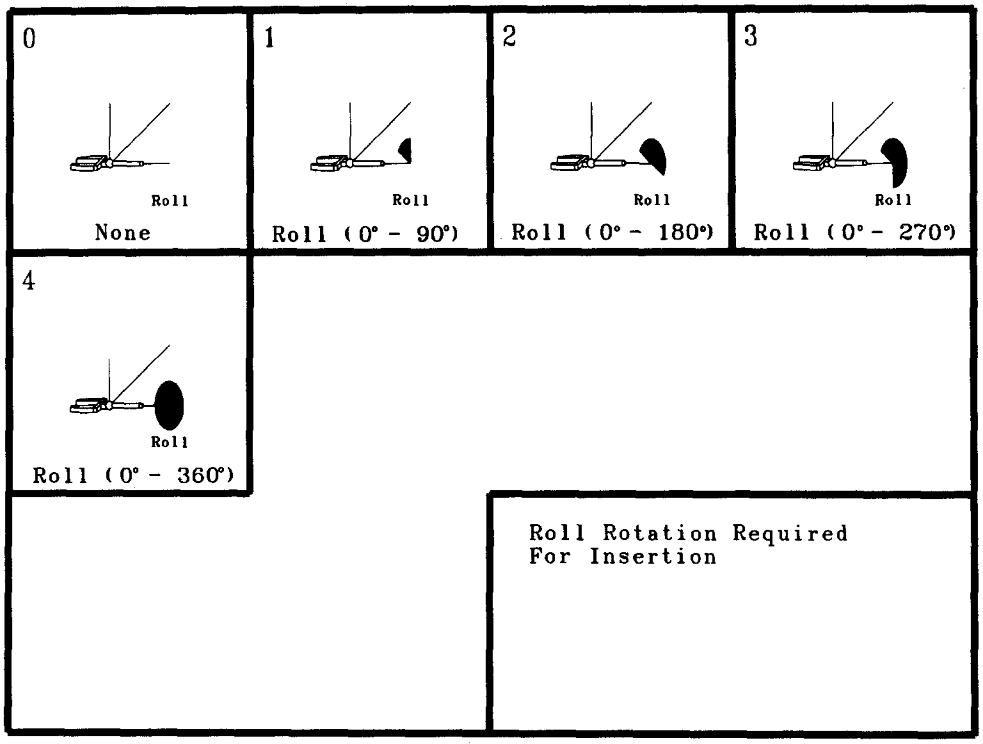


Fig. 9. GUI screen for selecting the roll rotation required for insertion.

With this system, a product design engineer may develop a relatively simple and portable description of the assembly task(s) associated with the design of a particular part to be assembled. Conversely, if the EOAT design engineer wishes to retrieve a gripper design with this characterization, this code specification could be used as the basis for a database search. As well, to broaden the search scope a user may eliminate one or more dimension from consideration by the retrieval mechanism, e.g.

1	2	3	4	5	6	7	8	9	10	11
0	0	*	0	0	5	1	1	0	0	*

This example differs from the previous by the inclusion of two wildcards, "*," in place of two of the

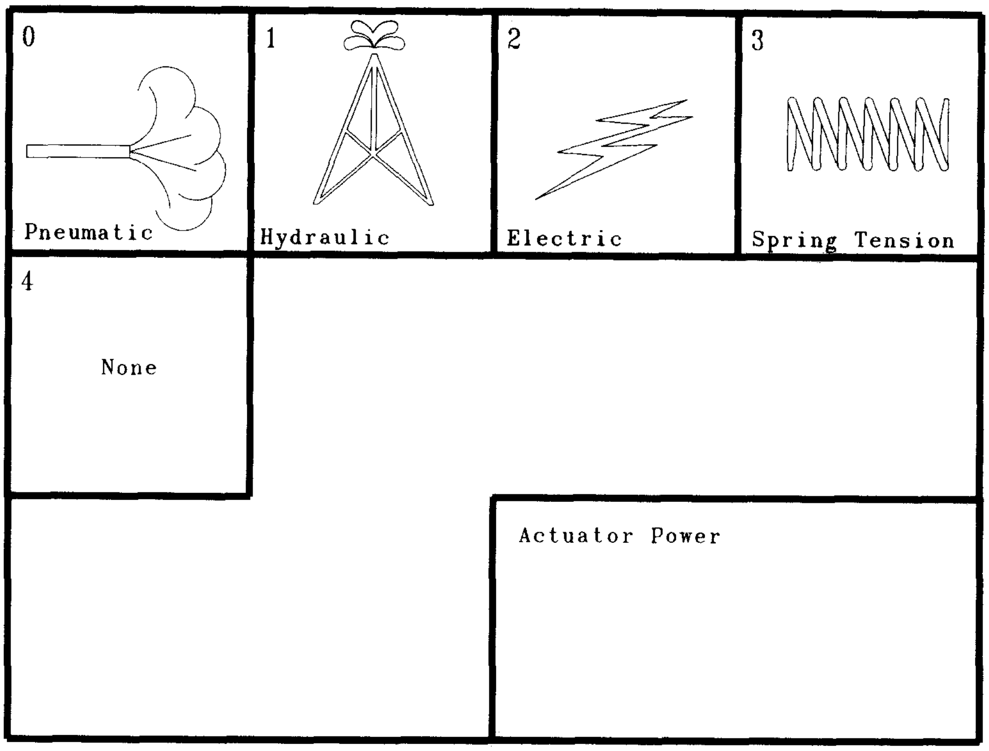


Fig. 10. GUI screen for selecting the EOAT actuator power method.

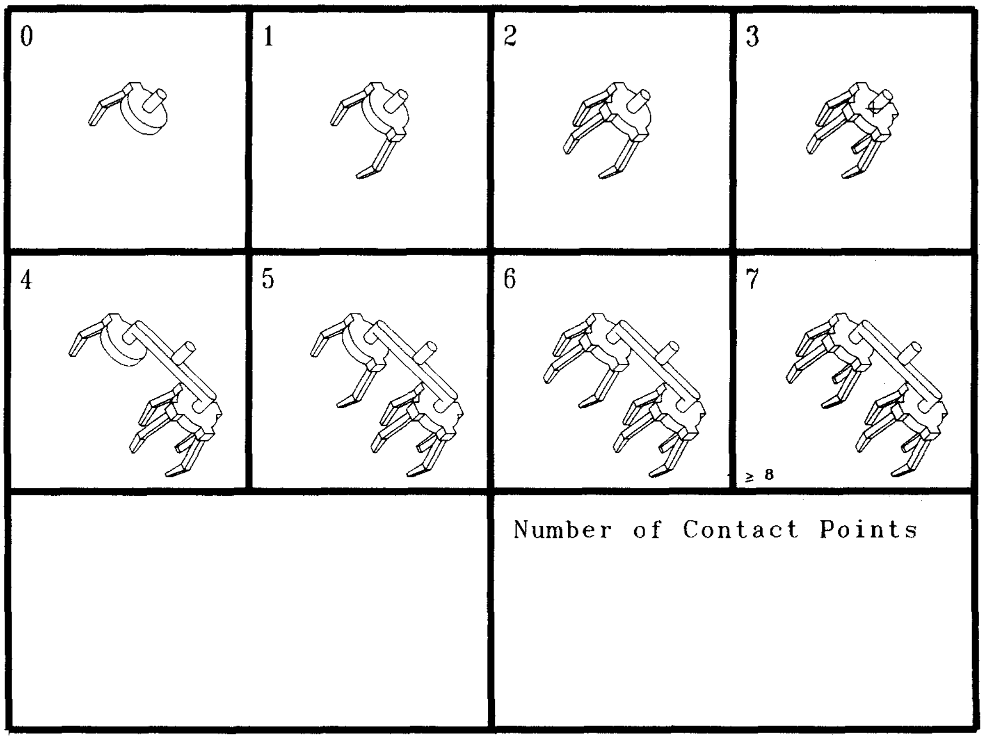


Fig. 11. GUI screen for selecting the number of EOAT contact points.

digits. Retrieving from the engineering design database on the basis of this code will yield a list of EOAT designs which includes all exact matches to the dimension levels numerically specified, *and* all possible levels of digits 3 and 11 [and, in general, any dimensions represented by a wildcard (*) specification]. Thus, a user is able to control the search (within the context of the overall dimension specifications), reacting to the possible sparseness of specific searches by "loosening" the specificity of the "target" code.

Potential extensions of this facet of the code retrieval structure, i.e. the "loosening" of specific digit levels would be worthwhile. That is, if it is necessary to search on the basis of the immediate example above, but we wish to consider level 1 or level 2 of the first dimension we currently would be required to conduct two searches. The ability to modify the target code to allow for concurrent searches of this fashion (as discussed in Section 2.3) would be beneficial.

4. GRAPHICAL INTERFACE FORMAT

In Fig. 12, we present the general format of the screen which will be the focus of the user interface. Each digit of our GT code will have a different screen, and the user may select a level by pressing the mouse button with the cursor within the boundary of that region. In the first nine of the "blocks" on each screen, we have four distinct regions, i.e. A, B, C, and D, described in the following:

Region A: This region presents the level specification identifying the level referenced by this "box" of the screen. A click of the mouse in this "box" will (when encoding or retrieving) assign this level into this digit of the code.

Region B: This region presents an icon which will represent to the user the flavor (if not an exact description) of that particular level. Some of the icons used are rather roughly stylized, with their final form determined by that which would minimize coding error, rather than for artistic quality.

Region C: This region (not always included) presents textual material in support of the information contained in region B.

Region D: This region contain a computed value which is the number of designs available in the database associated with the code developed so far (if this level is specified) with a wildcard specification (see region E below) for all subsequent digits. This aids making "on-the-fly" judgements concerning whether or not to use a wildcard for a given digit.

Further, we have two more regions, located in the tenth "block" and the large rectangular block in the lower left corner (of Fig. 12), described in the following:

Region E: The wildcard region (not always included). This region is referenced in retrievals only. The encoding of a new EOAT design with a wildcard specification for any of its digits will result in an error notification required respecification.

Region F: information concerning the general description is addressed by this particular code digit.

Since the tenth block is most often used for "other" category, we use half of this block to define a region for the user to input the wildcard.

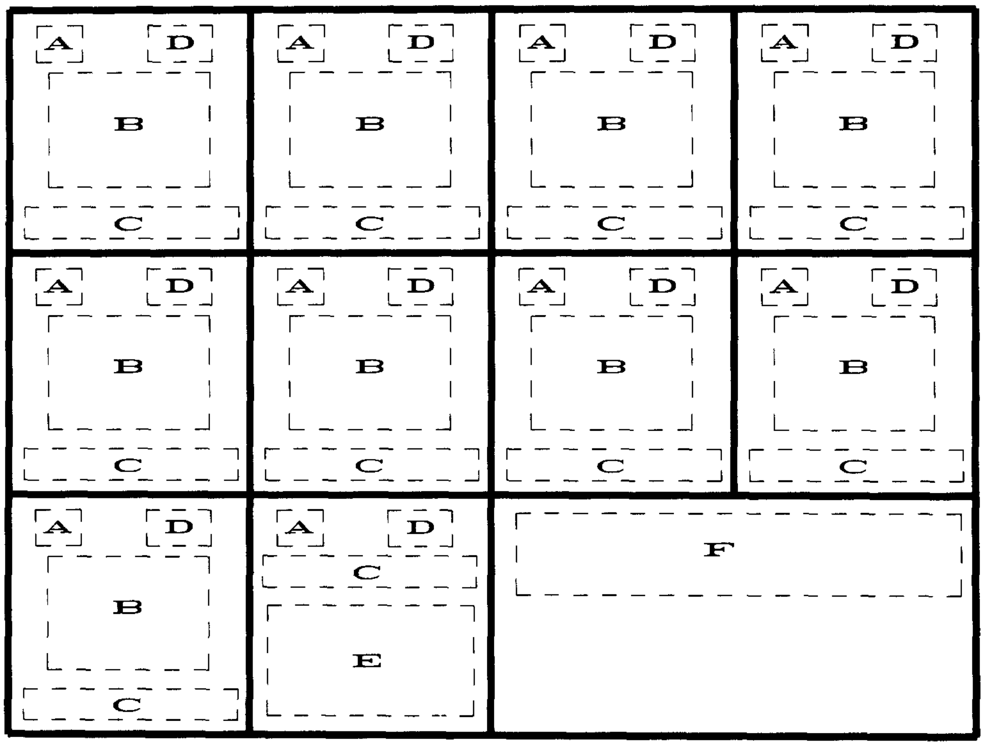


Fig. 12. GUI screen map.

5. DISCUSSION OF OPERATIONAL CONSIDERATIONS

The system described in the previous section reflected a series of design iterations, where classifications were proposed to the client firm and feedback was used to direct modifications. The primary purpose of the system was to aid the EOAT engineers to directly make the EOAT design and acquisition effort less expensive.

The secondary purpose was to infiltrate the new product design process in a relatively painless fashion. The client firm was and is focussing a great deal of effort attempting to closing the gap between the design and manufacturing functions. It was observed that EOAT represented a significant cost in automated manufacturing. The most basic principles of design-for-manufacturability^3 dictate that the design engineer possess information concerning existing manufacturing process options and constraints, in particular, associated with automated assembly systems, the EOAT. Candidly, it was also observed that many of the EOAT designs were relatively short-lived, with may grippers simply gathering dust in storage, often after a useful life span which was simply too short.

The notion was, that in design meetings, part of the decision making associated with designing a part and/or assembly could utilize information concerning the availability of EOAT necessary for accomplishing the required assembly tasks. This, of course, was to complement efforts to adapt product designs such that "standard" EOAT (which are much less expensive than "custom" designs) could be utilized.

The implementation of this system in an INTEGRAPH platform (the CAD system of preference within the firm) was to facilitate the connection between the product design and the process design functions.

6. CONCLUSIONS

In this paper we discuss a number of theoretical aspects of the GT classification and coding system design. What is conspicuously missing from this discussion is the notion of restricting such classification and coding systems to pieceparts or other "large" design populations. We use these principles (coupled with strong user input) to develop a "custom" small-scale (in terms of design population) system, directed at remedying a high-cost element of automated assembly.

We feel that the rationale used to justify the effort associated with the design of our system, *and* the paradigm used for the system design, is broadly applicable. A major extension to GT made by this paper is the realization (and operationalization) of the fact that GT-based classification and coding systems may be flexible in attribute specification (and entity population), to the extent that any quantitative or qualitative attributes which determine a *meaningful* characterization of an entity in question may be included. The code-design logic inherent in GT is that a code structure should be used as a *porthole* to view what a manufacturing firm is *really* doing. The GT-based code is the device allowing management to "see the forest, in spite of the trees." In this sense, the code structure aids in decision making via the use of a "crude" version of an expert system; and, in this context, like all expert system applications, it is problem-specific.

Finally, we develop a rough-cut evaluation of the classification and coding system presented in this paper with respect to the principles discussed in Section 2.

(1) **Completeness of the system:** As discussed in Section 2.1, the notion of completeness references a classification and coding system which enables complete encoding and consideration of the designated design population. For the EOAT project, all corporate designs were made available for inclusion in the system, and all were codable using our system.

(2) **Specification consistency:** The uniqueness of the code assignments were questionable only in two digits. In the "part contact configuration" digit (#3—Fig. 3), the ability of the EOAT to accomplish a secondary assembly action was not included. For example, in a number of cases, the EOAT was used to grasp, transport, and insert a given part, *and*, subsequently, the EOAT was reoriented and used to push or press the part into its "final" position. After discussion with the client firm, it was decided to focus upon the "primary" assembly action associated with a given part. Similarly, the "actuator power" digit (#10—Fig. 10) was difficult to code in multi-gripper configurations, for example where one gripper was pneumatic and the other was electric. Personnel from the client firm decided that this situation was such a rarity that it could be omitted. Further, since this was such an uneconomical EOAT configuration, and would violate the *consistency* principle, these particular designs were removed from the database and action was taken to discourage their design in the future.

(3) **Accurate discrimination:** A number of digits do not accurately measure the notion of closeness as discussed in Section 2.3. The digit describing the "geometry of the part to be inserted" (#2—Fig. 2) is an example of this. The ordering selected here attempts this to some degree, assuming frequency dictates the number of the level specification, e.g. level specifications 0–3 are very common geometries. For the rest, the client firm decided that arranging similar geometries in columns, i.e. $0\rightarrow4\rightarrow8$ for "rectangular" parts, $1\rightarrow5\rightarrow9$ for "cylindrical" parts, and $2\rightarrow6$ for "oblique/rectangular" parts, would improve user friendliness sufficiently to warrant violation of this principle. When developing "part contact configuration" (#3—Fig. 3), and "gripper–part adhesion method (#4—Fig. 4), the order of level specification was in concurrence with estimates of expense of the

design and acquisition of the particular EOAT, i.e. the higher the level number, the more expensive was the EOAT. This implies a partial ordering and some level of discrimination capability.
The difficulties associated with the digit describing "rotation required for insertion" (#6—Fig. 6) are described in Section 2.6.

(4) **Corporate consistency:** Extensive communication with the client firm's project team ensured that corporate consistency was maintained within all digits.

(5) **User friendliness:** This aspect of the code was supported by a code design process where users had direct input, and by an operational environment for use of the code which was "inside" the in-house CAD environment. The ultimate plan was that the code will have two identical versions, one accessible using a drop-down menu "inside" the corporate CAD system, and one operated, independent of the CAD system, on a transportable microcomputer, thus facilitating the use of the retrieval capabilities of the system in product design meetings without consideration of the meeting location or the availability of CAD workstations.

(6) **Adaptable to numerical analysis:** Here, for many of the reasons mentioned in the above evaluation of adherence to the other code design principles, we fall short of the level desired. The original purpose of the project, from the perspective of the client firm was the development of an EOAT design retrieval system. This was accomplished. It is hoped that insights concerning the development of a system more adaptable to pattern seeking numerical techniques will come from extensive use of the system in its current form.

REFERENCES

1. Bao, H. P.: Group technology classification and coding for electronic components: development and applications. In *Productivity and Quality Improvements in Electronics Assembly*, Edosomwan, J. A., Ballakur, A. (Eds). New York, McGraw-Hill. 1989.
2. Billo, R. E., Rucker, R., Shunk, D. L.: Enhancing group technology. *J. mfg Systems* **7**(2): 95–106, 1987.
3. Boothroyd, G., Poli, C., Murch, L. E.: *Automatic Assembly*. New York, Marcel Dekker. 1981.
4. Houtzeel, A.: MI-CLASS, a classification system based on group technology. SME Working Paper No. MS75-721, 1975.
5. Hyde, W. F.: *Improving Productivity by Classification, Coding, and Database Standardization*. New York, Marcel Dekker. 1981.
6. Ingram, F. B.: Group technology. *Prod. Inventory Mgmt* **23**(4): 19–34, 1982.
7. Marion, D., Rubinovich, J., Ham, I.: Developing a group technology coding and classification scheme. *Ind. Engng* **18**(7): 90–97, 1986.
8. Mosier, C. T., Janaro, R. E.: Toward a universal classification and coding system for assemblies. *J. Ops Mgmt* **9**(1): 44–64, 1990.
9. Mosier, C. T., Taube, L.: The facets of group technology and their impacts on implementation—a state-of-the-art survey. *OMEGA* **13**(5): 381–391, 1985.
10. Opitz, H.: *A Classification System to Describe Workpieces*. Oxford, Pergamon Press. 1970.
11. Styslinger, T. P., Melkanoff, M. A.: Group technology for electronics assembly. In *Productivity and Quality Improvements in Electronics Assembly*, Edosomwan, J. A., Ballakur, A. (Eds). New York, McGraw-Hill. 1989.