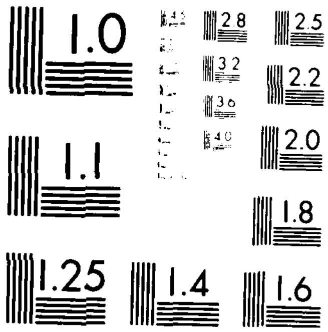
AD-A151 945 A MULTI-DIMENSIONAL GRAMMAR FOR MODELING TEMPORAL-SPATIAL IMAGES(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH B G LITHERLAND 1984 1/1

UNCLASSIFIED AFIT/CI/NR-85-16T F/G 5/7 NL

END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFIT/CI/NR 85-16T
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): A Multi-Dimensional Grammar for Modeling Temporal-Spatial Images
5. TYPE OF REPORT & PERIOD COVERED: THESIS/~~DISSERTATION~~
6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Barry G. Litherland
8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: AFIT STUDENT AT: Univ of Florida
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: AFIT/NR WPAFB OH 45433
12. REPORT DATE: 1984
13. NUMBER OF PAGES: 67
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): UNCLASS
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-1

LYNN E. WOLAVER
Dean for Research and Professional Development
AFIT, Wright-Patterson AFB OH

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

DTIC ELECTE MAR 27 1985 D E

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

11 056

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

A MULTI-DIMENSIONAL GRAMMAR FOR MODELING TEMPORAL-SPATIAL IMAGES

By

Barry G. Litherland

December 1984

Chairman: Leslie H. Oliver, Ph. D.
Major Department: Computer and Information Sciences

Image processing deals generally with well-defined multi-dimensional data with the intention of transforming them into a more useable form. This thesis proposes a grammar which is useful in describing, analyzing, and designing image processing in multi-dimensional terms. The concept of an m-dimensional to n-dimensional dimensioned modeling process (m:ndMP) employed by the proposed grammar allows a systematic approach to modeling image processes.

This type of modeling process is designed to be useful in manipulating and relating transformations of multi-dimensional data using networks of directed graphs. Inclusion of a confidence element in the grammar's notation allows inclusion of heuristics in describing image processing algorithms. The structured composition of

m:ndMPs parallels that of formal computer programming languages to ease translation of image processing models into software implementations.

BIBLIOGRAPHY

Elson, Mark, *Concepts of Programming Languages*. Science Research Associates, Inc., Chicago, Illinois, 1973.

Iliffe, J. K., *Advanced Computer Design*. Prentice-Hall International, Inc., Englewood Cliffs, New Jersey, 1982.

Knuth, Donald E., *The Art of Computer Programming*. Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1968.

Nilsson, Nils J., *Problem-solving Methods in Artificial Intelligence*. McGraw-Hill, Inc., New York, New York, 1971.

Rich, Elaine, *Artificial Intelligence*. McGraw-Hill, Inc., New York, New York, 1983.

Shooman, Martin L., *Software Engineering*. McGraw-Hill, Inc., New York, New York, 1983.

Tou, Julius T., and Gonzalez, Rafael C., *Pattern Recognition Principles*. Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1974.

A MULTI-DIMENSIONAL GRAMMAR
FOR
MODELING TEMPORAL-SPATIAL IMAGES

By

BARRY G. LITHERLAND

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

1984

This thesis is dedicated to

my wife,

Murl Lynn Worley Litherland.

ACKNOWLEDGEMENTS

I would like to express my undying love and deepest gratitude to my wife, Murl Lynn Worley Litherland, and my children for their support and understanding without which this thesis would never have been attempted. This thesis is truly the result of a family effort.

I also wish to express deep gratitude and admiration to Dr. Leslie H. Oliver for his patience and thoughtful guidance throughout the research and writing of this thesis. The cooperation and contributions of Dr. Douglas D. Dankel II and Dr. Donald G. Childers are also greatly appreciated.

I would like to thank the Air Force Institute of Technology for its financial and administrative support during my matriculation at the University of Florida.

Finally, I wish to thank my parents, Allyn C. Litherland and Regina K. Litherland, for their faith in me. They have given me an appreciation for the value of education which will stay with me for a lifetime.

TABLE OF CONTENTS

LIST OF FIGURES

processing algorithms. The structured composition of m:ndMPs parallels that of formal computer programming languages to ease translation of image processing models into software implementations.

Chairman

CHAPTER I
BACKGROUND AND INTRODUCTION

This thesis proposes a new grammar for modeling image processing algorithms in multi-dimensional terms. The new grammar is based upon relationships between dimensioned modeling processes describing data transformations.

A formal grammar can be defined as a list of rules describing a scheme for representing concepts or ideas. More than just a recipe for form, though, it also implies relationships between the expressed concepts dictated by their positional relationships within the structure imposed by the grammar.

Grammars generally are formally defined as one-dimensional strings of characters or symbols. Every written language has some form (at least one form) of grammar as a standard. Computer languages, specifically, have well-defined grammars, such as Backus normal form [IL82].

Some grammars have been defined as two-dimensional arrangements of symbols. The field of computer pattern recognition boasts a few of these grammars used to express multiple simultaneous relationships between various pattern elements.

The fundamental weakness of most grammars, in the author's opinion, is their tendency to be limited to the two dimensions easily represented on a printed page. To express a greater dimensionality, the grammar must be abused by giving some or all of the symbols multi-dimensional meanings. This requires the reader to mentally expand the meaning of the presented grammar beyond its designed capabilities.

A grammar is needed to express multi-dimensional relationships by design. Definition of such a grammar is the intention of this thesis. Specifically, the field of image processing, which deals generally with well-defined multi-dimensional data, could benefit from the definition of such a grammar.

Tou and Gonzalez [TO74] define a grammar as the 4-tuple $\{V_N, V_T, P, S\}$, where V_N is a set of nonterminals (variables), V_T is a set of terminals (constants), P is a set of productions or rewriting rules, and S is the start or root symbol, as shown in Figure 1.1. By defining the symbols representing variables and constants as multi-dimensional in nature, a multi-dimensional grammar is created which is easily represented even in two-dimensional printed form. And, since these symbols always imply a dimensional nature, the reader is tasked with one less mental interpolation when confronted with a multi-dimensional relationship.

$$G = (V_N, V_T, P, S)$$

Figure 1.1, Definition of a grammar.

$$\{S^m, T, R^n, C^k\}$$

Figure 1.2, Form of grammar for V_N, V_T, and S.

a.	$\{S^1, T_a, R^2\}$	one-dimensional source coordinate system; two-dimensional result.
b.	$\{A^6, B, C^3\}$	six-dimensional source coordinate system; three-dimensional result.
c.	$\{I_a{}^1, J_c, K_o{}^1\}$	one-dimensional source coordinate system; one-dimensional result.

Figure 4.1, Examples of notation used to describe transformations. The ordered 3-tuple contains a source coordinate system (S^1, A^6, and $I_a{}^1$), the transformation description name (T_a, B, and J_c), and the result coordinate system (R^2, C^3, and $K_o{}^1$).

CHAPTER IV
EXPANSION OF TRANSFORMATIONS

To discuss expansion of transformations it is convenient to introduce a simple notation representing the form and identity of transformations. Figure 4.1 shows several transformation examples. Associated with each element of these notational representations are definitions of the attributes and role of each dimension in the element's coordinate system, as shown in Figure 4.2.

Expansion of a transformation involves more than merely listing the subordinate transformations (hereafter called sub-transforms). The combination scheme associating the sub-transforms must be explicitly described, also. To accomplish this, a directed graph [KN68] can be used to depict the logical flow of transformations from the original (source) coordinate system to the final (result) coordinate system. Figure 4.3 shows a directed graph depicting the expansion of a transformation. Again, definitions of the dimensions' roles and attributes of the transformation and each sub-transform would be necessary.

Each sub-transform could, in turn, be expanded into its own sub-transforms. The recursive expansion of subordinate levels of transformations is theoretically limitless. However, in any design or implementation there

However, this transformation is not accomplished in one step. The exact method of transformation varies with the manfacturer's circuit design, but the result is essentially the same. Each method of transformation involves a combination of subordinate transformations which decode, amplify, massage, and interpret the data according to specific rules of combination prescribed by the circuitry involved. The description of a transformation in terms of a combination of subordinate transformations is called an expansion of the original transformation.

contained in the video signal transmitted to a television receiver. Figure 3.1 shows the format of a typical video signal. In effect, the information transmitted to the television receiver to represent a single full-screen picture or frame, is two-dimensional.

The temporal dimension (i.e., the dimension of time) is a linearly ordered set of time values. In Figure 3.1 this appears as a continuous horizontal axis increasing in value from left to right. It represents the encoded horizontal placement of picture elements on the television screen.

The signal dimension (i.e., the dimension of modulation) is a linearly ordered set of amplitude values. In Figure 3.1 this appears as a continuous vertical axis increasing in value from bottom to top. It represents the encoded vertical placement and brightness of picture elements on the television screen.

The signal illustrated in Figure 3.1 is two-dimensional, but the picture generated by the television receiver is three-dimensional. Each picture element of the video display has a vertical position, a horizontal position, and a brightness. Picture elements are generated by reference to their vertical and horizontal position in the picture, according to a transformation scheme known as raster scanning. So, transforming the two-dimensional signal data into three-dimensional picture data is the goal of a television receiver.

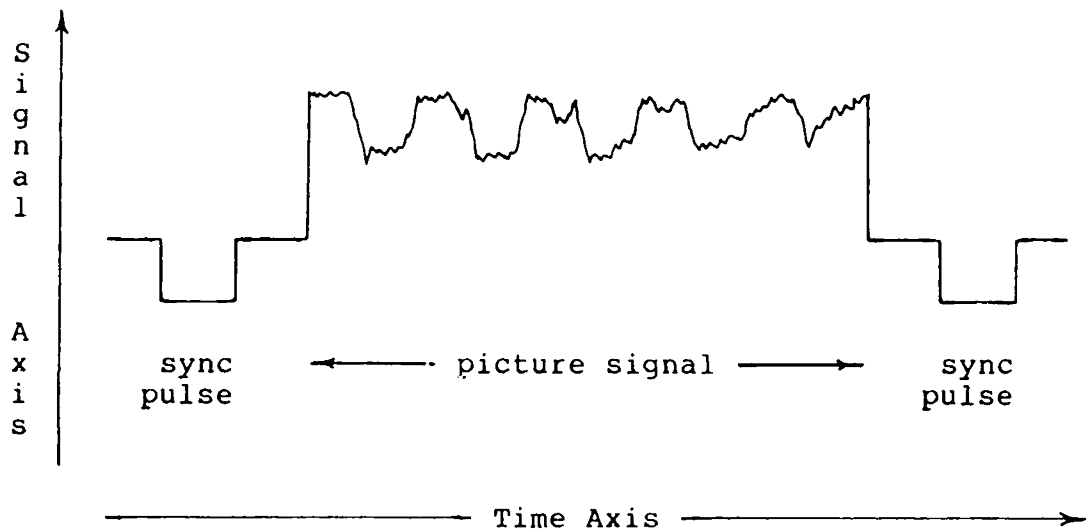


Figure 3.1, Example of a portion of the two-dimensional form of a television video signal. The dimensions describe values of intensity or encoded vertical positioning (amplitude) and horizontal positioning within the picture being transmitted (temporal value). The example is not to scale.

CHAPTER III
DIMENSIONED TRANSFORMATIONS

The utility of dimensioned image data is obvious. The attributes of dimensionality directly support the requirements of image processing when the acquired data are accurately described in terms of a completely defined dimensioned universe. Coordinate systems allow even more descriptive expressions of data without loss of the unambiguity of one-dimensional values.

This expressiveness is useful in another way. Image processing systems are designed to transform acquired image data into a more meaningful or more expressive form. The concept can be described simply as translating acquired image data from one coordinate system to another coordinate system. This translation is a change in form or content or both. The resulting data may be a subset of the source data, and the form used to express the data may change.

Furthermore, every transformation is equivalent to (at least one instance of) a combination or structure of transformations. The usefulness or utility of expanding a transformation into a combination of transformations depends on the applicability of either form to a specific task (e.g., a computer numerical processing algorithm).

To illustrate the concept of transformation, consider the two-dimensional description of the data

Dimensions, therefore, possess four important attributes: minimality, domain, range, and ordering scheme of member values. Coordinate systems are composed of one or more orthogonal dimensions. By imposing these attributes on a set or collection of data, dimensionality of the data is achieved.

The unambiguous nature of dimensionality makes it useful as a tool. Mathematicians rely heavily on these attributes to create a structure of axioms, lemmas, and theorems to model and measure the world and to propose alternative realities. Dimensions are used to describe the physical universe in terms of standard measurements and points of reference.

Image processing in particular relies on the unambiguous measurability of image data to support the accuracy, reliability, and enhancement of an image. It is not remarkable that the greatest effort in construction of image processing systems has been in the area of data acquisition hardware. Accuracy and reliability have been of paramount importance in most applications.

schemes, the first and last example sets have the same range.

Coordinate systems use dimensions to describe instances of data by one or more of their attributes. Each instance of a datum in a coordinate system has a single value within the range and domain of each dimension of the coordinate system. This method of expressing values is as unambiguous as expressing values using only a single dimension. Furthermore, the information contained in coordinate form is literally more explicit. Coordinates of a datum point within an n-dimensional space defined by a coordinate system are ordered n-tuples of values corresponding to the ordered dimensions of the coordinate system.

The property of orthogonality of a coordinate system is defined as the coexistance of one or more dimensions in the coordinate system universe while maintaining the separate identities of each dimension. Two or more dimensions within a coordinate system may in fact have identical domains, range, minimality, and ordering schemes. But, when they are used as coordinates describing a datum point in the coordinate system space, they specify values of different attributes. Thus, in the context of coordinate systems, each dimension is associated with a unique attribute of points within the coordinate system's space or universe. This unique attribute is called the role of the dimension in the coordinate system.

a.	[RED - VIOLET]
b.	[BLUE - YELLOW]
c.	[RED - YELLOW]
d.	[RED - VIOLET]

Figure 2.2, [illegible] of the ordered sets in Figure 2.1.

a. (RED,ORANGE,YELLOW,GREEN,BLUE,INDIGO,VIOLET)

b. (BLUE,GREEN,INDIGO,ORANGE,RED,VIOLET,YELLOW)

c. (RED,BLUE,GREEN,INDIGO,ORANGE,VIOLET,YELLOW)

d. (RED,ORANGE,INDIGO,GREEN,BLUE,YELLOW,VIOLET)

Figure 2.1, Ordered sets of seven colors of the visible light spectrum. The schemes used to order the sets are:

a. increasing frequency of the corresponding light,
b. ascending alphabetical order of the color names,
c. ascending order of the number of letters in the color names,
d. reverse ordering of the second scheme listed and rotated to make RED the first value.

CHAPTER II
DIMENSIONALITY EXPLAINED

A dimension can be described, in its most general sense, as a totally ordered set of values each of which is unique within that set. The universe consisting of every possible value within a dimension is called the domain of the dimension. The extreme values of the ordered set of values in a dimension define the range of the dimension.

Minimality is the characteristic of a dimension asserting the uniqueness of each of its member values within the dimension. This property guarantees unambiguity of relationships based on the ordering scheme within the dimension.

It is often possible to describe a set of data in terms of dimensions differing from each other only by the ordering scheme used. In these cases, the domains of the dimensions are identical, but the ranges of the dimensions may vary. Figure 2.1 shows several examples of dimensions described as ordered sets of seven colors in the visible light spectrum. Figure 2.2 gives the ranges of the corresponding ordered sets in Figure 2.1. Each of the example sets contains the same number of values, and the domains are identical. The ranges of the first three sets are different. Although they have different ordering computer programs into the proposed grammar can aid in the analysis of errors and weaknesses of the underlying algorithms with full consideration of the multi-dimensional nature of the image data.

Finally, the inclusion of the fourth element, C, called the "confidence", of variables and constants, allows description of procedures including their limitations and applicability under various conditions. Thus, realistic assessment and evaluation of the image processing algorithms can be made, even in the face of heuristic or questionable computational methods.

Chapters 2 through 5 of this thesis progressively reveal the concepts forming the basis of dimensioned modeling processes. The remaining chapters discuss the various features of the modeling process grammar.

Image processing generally entails the transformation and manipulation of a finite set of precisely defined image data into a different and, hopefully, more usable form. Even though this data is precisely defined, it often is best described in multi-dimensional form. The multi-dimensional grammar described in this thesis endeavors to simplify the process of describing image processing. Design and analysis of multi-dimensional image processing systems is thereby simplified.

The concept of performing image processing by comparing image data to a set of ideal image data models is a useful and proven technique. Production systems using a set of rules, one or more databases, and a control or ordering strategy [RI83], perform transformations and manipulations of input data by comparison to the models they describe. The structure of the grammar proposed in this thesis lends itself very well to this modeling concept.

Thus, the proposed grammar is intended to be useful in many areas of image processing. As a basis for computer programming, it describes image processing algorithms as a network of transformations of the image data. As a tool for designing an image processing system, it characterizes the processing in terms which appreciate the multi-dimensional nature of the image data. Also, translation of existing image processing algorithms are

As the following chapters show, the form of this multi-dimensional grammar is based on four primary concepts. First, each instance of root, variable, and constant is represented by an ordered 4-tuple, $\{S^m, T, R^n, C^k\}$, of symbolic names, S, T, R, and C, with appropriate dimensionality superscripts, m, n, and k, as shown in Figure 1.2. Second, each instance of the first, third, and fourth element of every variable or constant is defined explicitly by a list of attributes. These attributes include the symbol's name, coordinate system, and a definition of each dimension within the coordinate system. Third, each instance of the second element of every variable is defined explicitly by a directed graph of variables and/or constants. Finally, each instance of the second element of every constant is defined textually in sufficient detail to describe the transformation or function.

Addition of rules, governing the relationships of the elements of variables and constants within a directed graph and the elements of the variable or root defined (in part) by the graph, is all that is needed to complete the grammar. Specification of these rules can be tailored to the desires of each user and should be based upon the utility of the conventions adopted within the applications anticipated. The flexibility and utility of these rules are discussed in the latter chapters of this thesis.

a. $\{S^1, T_a, R^2\}$

S^1: dimension #1:
role: numerical value
domain: integer numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order

T_a: associate even/odd status with numerical value

R^2: dimension #1:
role: numerical value
domain: integer numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #2:
role: odd/even status
domain: ['odd','even']
range: 'odd' to 'even'
ordering scheme: ['odd','even']

Figure 4.2, Descriptions of the transformations shown in Figure 4.1.

b. $\{A^6, B, C^3\}$

A^6: dimension #1:
role: x-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #2:
role: y-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #3:
role: z-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #4:
role: intensity value
domain: integer numbers
range: zero to positive infinity
ordering scheme: ascending numerical order
dimension #5:
role: color
domain: visible spectrum
range: RED to VIOLET
ordering scheme: ascending frequency
dimension #6:
role: temporal axis
domain: integer numbers
range: 0 to 100
ordering scheme: ascending numerical order

B: show only the spatial position of the source data

C^3: dimension #1:
role: x-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #2:
role: y-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order
dimension #3:
role: z-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order

Figure 4.2--continued.

c. $\{I_a^1, J_c, K_o^1\}$

I_a^1: dimension #1:
role: x-axis
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical order

J_c: show the trigonometric sine of each value

K_o^1: dimension #1:
role: x-axis
domain: real numbers
range: negative 1.0 to positive 1.0
ordering scheme: ascending numerical order

Figure 4.2--continued.

a. $\{A^2, T_o, Z^3\}$

b.

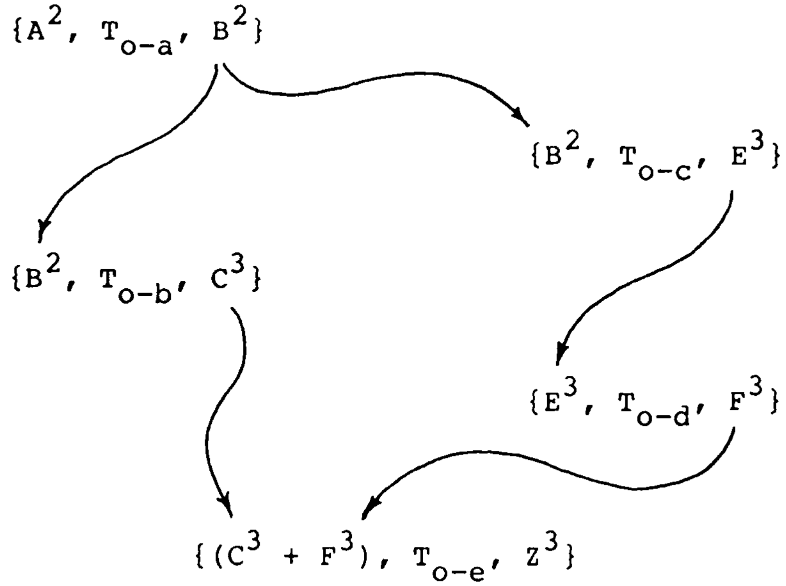


Figure 4.3, Use of a directed graph in expanding a transformation. The original transformation expressed in part a. is composed of the five sub-transforms shown in b. The union of the sets resulting from the sub-transforms T_{o-b} and T_{o-d} is used as the source for the sub-transform T_{o-e}. To perform a union, the coordinate systems of the participating result sets must be identical.

is invariably a practical and desirable limit to the depth of expansion of any transformation.

The expansion shown in Figure 4.3 contained a sufficient number and arrangement of sub-transforms to accomplish the transformation. This is an example of an exactly-defined transformation. An exactly-defined transformation is adequate for translation directly into computer code, but leaves little or no room for modification of the transformation without it becoming a hypo-defined transformation (discussed below).

If the directed graph included more than a sufficient number and arrangement of sub-transforms, then it describes a hyper-defined transformation. That is, the data resulting from all of the sub-transforms exceeds the minimum required to express the desired result. Hyper-defined transformations contain extraneous information which need not be included to perform the transformation desired. If, however, the transformation needs to be modified at some future time, and the desired modifications coincide with the extraneous sub-transforms, then no modification of the expansion would need to be done.

Alternatively, if the directed graph describing a transformation does not show all of the sub-transforms required for a complete definition of the desired result, then the transformation being described is called a hypo-defined transformation. In practice, this sort of definition is not adequate for immediate translation into computer code. However, it can be used as a tool to illustrate a partially complete design or to discuss only a specific portion of a very complex directed graph expansion representation.

CHAPTER V
PROCESSES AS EQUIVOCATED TRANSFORMATIONS

It is evident throughout the realm of human experience that there is much which remains unknown or uncertain to us. Nevertheless, judgements and decisions must be made even in the absence of complete information. These judgements rely on statistics and probabilities. Historical data used to make decisions are assumed to be representative samples of the same data source in the future.

The key word here is "assumed". Every assumption is based on unknown data or, perhaps, even unknowable data. For example, when a television image is used as a source of data in an image processing system, it is assumed to be a direct transformation of the light reflected, passed, or originating from the target surfaces and focused at a particular viewpoint plane in the television camera. In fact, there are various possibilities for this assumption to be false.

The atmosphere between the reflecting surfaces and the focal point may attenuate, reflect, filter, or otherwise block or modify the transmission of light. The camera image receiving equipment, transforming the light into an analogous electronic signal, may exhibit anomolous varying properties. This variance may be based on the room temperature, humidity, power fluctuations, mechanical vibrations, and more. Radiation in other spectra may influence the transformation of light to electronic signal (e.g., cosmic rays).

In general, if these undesirable influences can be minimized, then their effects to the image transformation will be minimized accordingly. However, these influences can not be eliminated entirely in many cases, and therefore pose a degree of uncertainty about the validity of the digitized television image. Judgements and decisions based on the digitized television image are at risk of being erroneous in direct proportion to the probability of errors in the digitized image.

In any chain or sequence of transformations, each depending on the results of the previous ones, the errors due to the uncertainties of each transformation could accumulate. The cummulative error is often compounded rather than merely summed.

Therefore, a fourth element is added to the transformation notation to define uncertanties of the transformation. The resulting 4-tuple is called a dimensioned modeling process. The convention chosen in this thesis is to include an element, called "confidence" and represented by the symbol "C", which defines the applicability of the transformation tuples within their stated coordinate systems and universes.

The resulting notation, shown in Figure 5.1, is called an m-dimensional to n-dimensional dimensioned modeling process. This is abbreviated as m:ndMP, where the values of m and n may be replaced by the appropriate positive non-zero integers. This is based on the concept that the C-element accurately describes the behavior of the transformation in Figure 5.2.

It is convenient to impose a specific universe and coordinate system on every instance of C in a group of m:ndMPs. This allows the manipulation and comparison of m:ndMPs confidence values without the need to translate them into a common coordinate system every time. Rules for such comparisons must form a valid and complete algebra for the universe and coordinate system used for the C elements, and must adequately describe the confidence of every transformation in the group. When such a convention is adopted for any group of m:ndMPs, the dimensionality superscript of every C element may be omitted from the notation without loss of information, if the convention is completely defined and described within the group.

To avoid confusion, throughout the remainder of this thesis the term "discussion" will be used to mean a group or collection of m:ndMPs with a completely defined and described C-element convention. A discussion may be a list of m:ndMPs, a complete or partial expansion of an m:ndMP, or even a single m:ndMP.

$$\{S^m, T, R^n, C^k\}$$

Figure 5.1, Notation for the m:ndMP.

$$\{S^m, T, R^n\}$$

Figure 5.2, Notation for the transformation represented in Figure 5.1.

One possible C-element convention adopts the use of a one-dimensional coordinate system in the domain of real numbers ranging from 0.0 to 1.0. This provides an infinite membership with a succinct and logically complete notation which is widely recognized. Each value of C could be mapped onto a universe of descriptive textual phrases, if desired. The rules of C-element comparisons and combination would have to be defined accordingly.

If C-element values are defined as describing the conditions for which the m:ndMP is valid, then it is possible to define rules describing the relationships between C-elements of subordinate m:ndMPs within an expanded m:ndMP to the C-element of the expanded m:ndMP.

For example, one such rule might be expressed as, "The C-element of a directed graph of m:ndMPs is equivalent to the union of the graph's final m:ndMP's C-element with the intersection of the C-elements of each m:ndMP directed to the S-element of the graph's final m:ndMP." Used recursively throughout a directed graph representing the expansion of an m:ndMP, this rule results in a formula of unions and intersections of every C-element in the graph. Figure 5.3 shows an example of an expanded 5:2dMP and its C-element formula.

It is possible to define the C-elements in many other ways and design different combinatorial rules. While there may not be a best C-element convention, it is important to choose a convention for each discussion of m:ndMPs which accurately and completely describes the confidence of every m:ndMP within the discussion. Furthermore, common sense suggests that such a convention should be flexible enough to accomodate any m:ndMP which may be included in future discussions which modify or refer to the current discussion.

Inclusion of the C-element enables analysis and description of transformations in a complete and real sense. This results from the concurrent description and acknowledgement of the uncertainties inherent in complex transformations (i.e., transformations which describe their counterparts in the real world).

However, it is readily apparent that the C-element represents a considerable obstacle in the way of developing or analyzing expanded m:ndMPs due to its potential complexity. Although the volume of information in a C-element is reduced by limiting it to assertions about the conditions under which its m:ndMP is applicable, there is a strong argument for even this to be a potentially infinite list or unknowable quantity. Based upon the concept that there may be information about the m:ndMP which is unknown and the fact that the information is unknown may also be unknown, it is theoretically impossible to assert with complete confidence and accuracy that a C-element is completely definable.

This is a potential disadvantage of the m:ndMP, but, as illustrated in the subsequent chapters, does not

a. $\{A^5, T, V^2, C\}$

b.

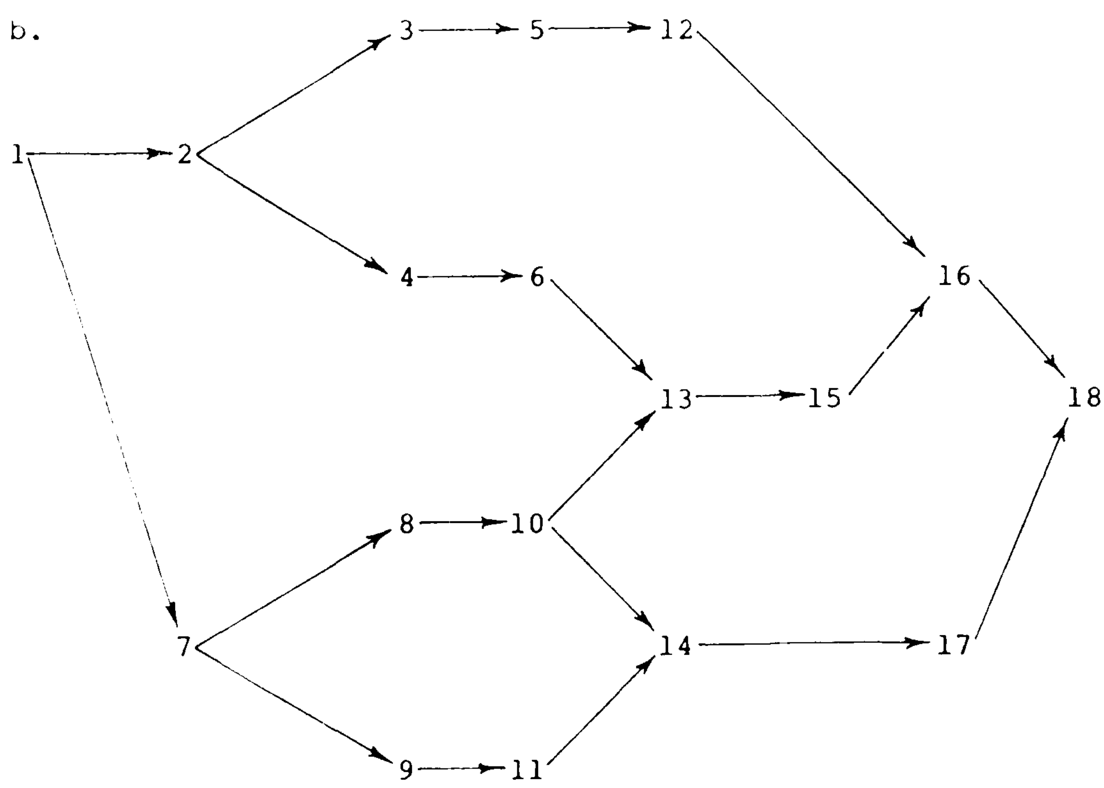


Figure 5.3, Example of ar expanded 5:2dMP and its C-element derivation.

a. The original 5:2dMP,
b. Directed graph of the expanded 5:2dMP,
c. Node m:ndMPs,
d. Formula for derivation of the original 5:2dMP's C-element from the C-elements of the constituant m:ndMPs. (+ is the union operation, * is the intersection operation)

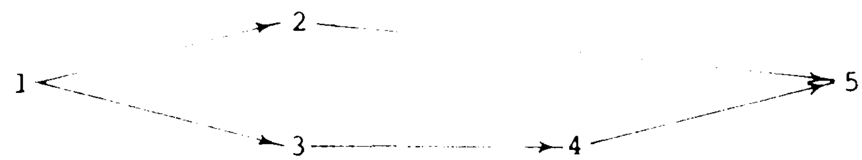
c.

Node	m:ndMP
1	$\{A^5_8,\ T_1,\ B^8_5,\ C_1\}$
2	$\{B_5,\ T_2,\ D_4,\ C_2\}$
3	$\{D_5,\ T_3,\ E_5,\ C_3\}$
4	$\{D_4,\ T_4,\ F_3,\ C_4\}$
5	$\{E_5,\ T_5,\ G_4,\ C_5\}$
6	$\{F_8,\ T_6,\ H_7,\ C_6\}$
7	$\{B_7,\ T_7,\ J_6,\ C_7\}$
8	$\{J_7,\ T_8,\ K_3,\ C_8\}$
9	$\{J_6,\ T_9,\ L_4,\ C_9\}$
10	$\{K_3,\ T_{10},\ M_4,\ C_{10}\}$
11	$\{L_3,\ T_{11},\ N_2,\ C_{11}\}$
12	$\{G_3,\ T_{12},\ P_2,\ C_{12}\}$
13	$\{\ (H^4_4 + M^4_4),\ T_{13},\ Q^2_3,\ C_{13}\}$
14	$\{\ (M^4 + N^4)_4,\ T_{14},\ R^3,\ C_{14}\}$
15	$\{Q^2_4,\ T_{15},\ S^4_1,\ C_{15}\}$
16	$\{S_3,\ T_{16},\ T_1,\ C_{16}\}$
17	$\{R_3,\ T_{17},\ U_1,\ C_{17}\}$
18	$\{\ (T^1 + U^1),\ T_{18},\ V^2,\ C_{18}\}$

d.

$$
\begin{aligned}
C = C_{18} & * \\
((C_{17} & * C_{14} * \\
& (C_{11} * C_9 * C_7 * C_1) + \\
& (C_{10} * C_8 * C_7 * C_1)) + \\
(C_{16} & * \\
& (C_{12} * C_5 * C_3 * C_2 * C_1) + \\
& (C_{15} * C_{13} * \\
& ((C_6 * C_4 * C_2 * C_1) + \\
& (C_{10} * C_8 * C_7 * C_1)))))
\end{aligned}
$$

Figure 5.3--continued.



NODE	m:ndMP
1	{DATA-SOURCE^3, ISOL-DISH, DATA-DISH^3, true}
2	{DATA-DISH^3, ISOL-WALLS, DATA-TEMP^3, true}
3	{DATA-DISH^3, ISOL-CULTURE, DATA-UNFILT^3, true}
4	{DATA-UNFILE^3, FILTER, DATA-TEMP^3, true}
5	{DATA-TEMP^3, JOIN-DATA, DATA-RESULT^3, true}

DATA-SOURCE, DATA-DISH, DATA-TEMP, DATA-UNFILT, DATA-TEMP, and DATA-RESULT:

- dimension #1:
  - role: x-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum x-axis value
  - ordering scheme: ascending numerical order
- dimension #2:
  - role: y-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum y-axis value
  - ordering scheme: ascending numerical order
- dimension #3:
  - role: brightness value of picture element
  - domain: positive integer number
  - range: zero to maximum brightness value
  - ordering scheme: ascending numerical order

ISOL-DISH: Mask array data to pass data within the outer radius of the dish unmodified, and pass all other data as zero (background data).

ISOL-WALLS: Mask array data to pass only data outside of the inner radius of the dish.

ISOL-CULTURE: Mask array data to pass only data within the inner radius of the dish.

FILTER: Apply 60Hz filter to data (to remove noise).

JOIN-DATA: Pass all data unmodified.

Figure 7.2, Expansion of 3:3dMP in Figure 7.1.

7.1, is the isolation and enhancement of the dish image against a contrasting background.

This level of definition is no more than a formalized statement of the algorithm. Also designated at this point (but not shown) would be the definitions of the C-element conventions and any other shorthand notations being used.

Figure 7.2 shows an attempt to expand the 3:3dMP shown in Figure 7.1 into its constituents. Although the quantity of written information seems unwieldy, the frequent user of dimensioned modeling processes can reduce the volume through use of shorthand notations and perhaps computer-aided design software. Even an average quality word processing program can accelerate the process of expansion and alleviate the bulkiness of the notation.

Further expansion of the m:ndMPs of Figure 7.2 would describe operations on an element by element basis such as multiplication by mask values. Expansion continues until every variable is fully defined in terms of other variables and constants, and every constant is defined in terms of an existing operation or procedure on the implementation computer system.

The second example demonstrates the analysis of an existing image processing algorithm. For the sake of illustration, only the relevant portions of the image process's m:ndMPs are shown. The point being stressed here

{DATA-SOURCE^3, IMAGE-PROCESS, DATA-RESULT^3, CONFIDENCE}

DATA-SOURCE^3:

- dimension #1:
  - role: x-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum x-axis value
  - ordering scheme: ascending numerical order
- dimension #2:
  - role: y-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum y-axis value
  - ordering scheme: ascending numerical order
- dimension #3:
  - role: brightness value of picture element
  - domain: positive integer number
  - range: zero to maximum brightness value
  - ordering scheme: ascending numerical order

IMAGE-PROCESS:

Locate, isolate and enhance the boundary of the circular area representing the petri dish. Then, replace all background data external to the dish area with a contrasting value, and filter the data internal to the dish area to reduce the electrical interference effects.

DATA-RESULT^3:

- dimension #1:
  - role: x-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum x-axis value
  - ordering scheme: ascending numerical order
- dimension #2:
  - role: y-axis position of picture element
  - domain: positive integer number
  - range: zero to maximum y-axis value
  - ordering scheme: ascending numerical order
- dimension #3:
  - role: brightness value of picture element
  - domain: positive integer number
  - range: zero to maximum brightness value
  - ordering scheme: ascending numerical order

CONFIDENCE:

true

Figure 7.1, Example of the initial definition in top-down design of an image processing algorithm.

CHAPTER VII
DESIGN AND ANALYSIS OF DIMENSIONED MODELING PROCESSES

Design and analysis of image processing algorithms are two sides of the same coin. The process of designing an algorithm expresses concepts of data manipulation in terms of a structured network of transformations. Analysis, on the other hand, forms concepts of data manipulation through interpretation of an existing network of transformations.

An example of each of these is illustrated in this chapter. Only superficial representations are given in order to express the underlying concepts without becoming bogged down with the details of the actual image processes involved.

The first example deals with the top-down design of an image processing algorithm which manipulates data representing a video image of a petri dish containing an organic culture. The circular dish is precisely in the center of a square image. Brightness values in the entire image are distorted by electrical interference during acquisition. The video snapshot is a three-dimensional array of picture element data conforming to the definitions used for the DATA-SOURCE element in Figure 7.1.

The desired output of this algorithm, expressed in the definitions used for the DATA-RESULT element in Figure some implementations would define the division operation as a basic instruction available to the programmer.

Additionally, the representation of the information contained in the example could be markedly reduced in volume without loss of information. This can be accomplished by using a dictionary of symbols and shorthand notations for the most commonly used ranges, domains, and ordering schemes. Many of these could be contracted into single symbols or used by default whenever they are not explicitly defined. Caution must be exercised to ensure that the definitions of the notations being used do not vary within the discussion. No attempt is made herein to prescribe specific notations for this purpose.

Result[2]	dimension #1: role: numerical value of result domain: real numbers range: negative infinity to positive infinity ordering scheme: ascending numerical value dimension #2: role: error code of transformation domain: [LEGAL, ILLEGAL] range: LEGAL to ILLEGAL ordering scheme: ascending value
T_{check}:	If B is zero then error code is ILLEGAL.
T_{sense}:	If only one of A and B is negative, then the sense is NEGATIVE, otherwise the sense is POSITIVE. Make A and B both positive (absolute value).
T_{divide}:	Divide A by B to get a result.
T_{sign}:	If the sign is NEGATIVE, then make the result negative, otherwise make the result positive.
C-a:	true
C-b:	true
C-c:	true
C-d:	true

Figure 6.3--continued.

```
temp-b^4:      dimension #1:
                    role: numerical value of A
                    domain: real numbers
                    range: zero to positive infinity
                    ordering scheme: ascending
                    numerical order
               dimension #2:
                    role: numerical value of B
                    domain: real numbers
                    range: zero to positive infinity
                    ordering scheme: ascending
                    numerical value
               dimension #3:
                    role: error code of transformation
                    domain: [LEGAL, ILLEGAL]
                    range: LEGAL to ILLEGAL
                    ordering scheme: ascending value
               dimension #4:
                    role: sign of result
                    domain: [NEGATIVE, POSITIVE]
                    range: NEGATIVE to POSITIVE
                    ordering scheme: ascending value

temp-c^3:      dimension #1:
                    role: numerical value of result
                    domain: real numbers
                    range: zero to positive infinity
                    ordering scheme: ascending
                    numerical order
               dimension #2:
                    role: error code of transformation
                    domain: [LEGAL, ILLEGAL]
                    range: LEGAL to ILLEGAL
                    ordering scheme: ascending value
               dimension #4:
                    role: sign of result
                    domain: [NEGATIVE, POSITIVE]
                    range: NEGATIVE to POSITIVE
                    ordering scheme: ascending value
```

Figure 6.3--continued.

Check ⟶ Sense ⟶ Divide ⟶ Sign

Node	m:ndMP
Check	{Parameters^2, T_{check}, temp-a^3, C-a}
Sense	{temp-a^3, T_{sense}, temp-b^4, C-b}
Divide	{temp-b^4, T_{divide}, temp-c^3, C-c}
Sign	{temp-c^3, T_{sign}, Result^2, C-d}

Parameters^2:

dimension #1:
- role: numerical value of A
- domain: real numbers
- range: negative infinity to positive infinity
- ordering scheme: ascending numerical value

dimension #2:
- role: numerical value of B
- domain: real numbers
- range: negative infinity to positive infinity
- ordering scheme: ascending numerical value

temp-a^3:

dimension #1:
- role: numerical value of A
- domain: real numbers
- range: negative infinity to positive infinity
- ordering scheme: ascending numerical order

dimension #2:
- role: numerical value of B
- domain: real numbers
- range: negative infinity to positive infinity
- ordering scheme: ascending numerical value

dimension #3:
- role: error code of transformation
- domain: [LEGAL, ILLEGAL]
- range: LEGAL to ILLEGAL
- ordering scheme: ascending value

Figure 6.3, Expansion of the 2:2dMP in Figure 6.2.

{Parameters^2, Function, Result^2, Confidence}

Parameters^2: dimension #1:
role: numerical value of A
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical value
dimension #2:
role: numerical value of B
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical value

Function: If legal to do so, divide A by B and return the result and an error code of LEGAL, otherwise return an error code of ILLEGAL

Result^2: dimension #1:
role: numerical value of result
domain: real numbers
range: negative infinity to positive infinity
ordering scheme: ascending numerical value
dimension #2:
role: error code of transformation
domain: [LEGAL, ILLEGAL]
range: LEGAL to ILLEGAL
ordering scheme: ascending value

Confidence: true

Figure 6.2, An example of an m:ndMP used by image processing algorithm. Note the abbreviated form used to describe the confidence element, implying that the value is absolutely true for every possible case of the m:ndMP.

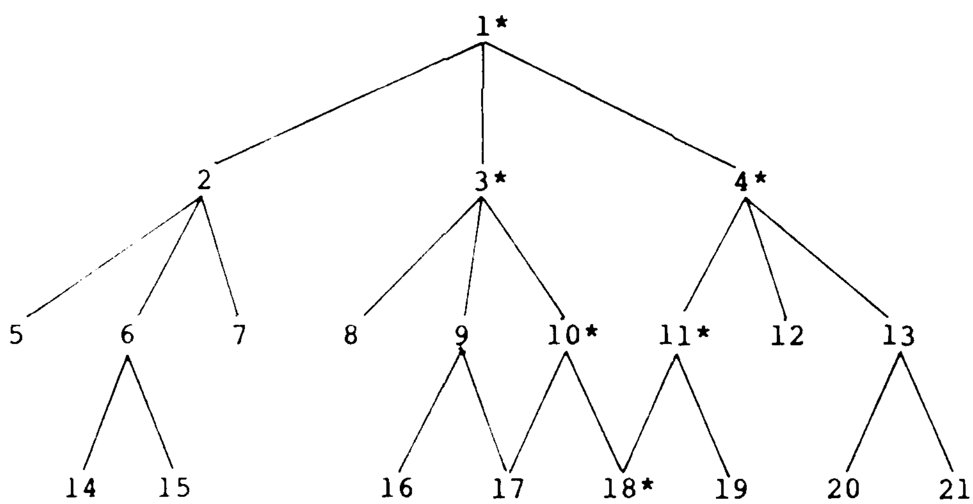


Figure 6.1, Illustration of an algorithm solution graph. Each node represents an m:ndMP. Arcs connect m:ndMPs with the m:ndMPs in their expansion. Asterisks mark nodes affected by the confidence element at node 18.

Heuristics may be included in the algorithm design using the m:ndMP confidence elements. A heuristic would be defined as an m:ndMP with a confidence wholly or partially unknown. Typically, heuristics are based upon empirical data presumed to have a high probability of being applicable to some domain. Once a heuristic is proven to be true for its domain, it is no longer a heuristic, but an algorithm [RI83].

In the solution graph of any algorithm the presence or a heuristic at any level is propagated upwards to the root of the graph via the expansion rules concerning confidence elements. Figure 6.1 illustrates this propagation.

The same propagation effect is true with any aspect of confidence elements. A similar propagation is inherent in the definition of the relationship between an m:ndMP and the constituant m:ndMPs in its expansion.

An example of an m:ndMP used by image processing algorithm, as shown in Figure 6.2, is the division of a real number A by another real number B. The expected result can be either a new real number or an error code signifying an overflow condition. A 2:2dMP representing this division operation is expanded into a graph of constituent m:ndMPs, as shown in Figure 6.3. Although the example describes each constituant m:ndMP as a constant, for most implementations thay are probably variables composed of even more basic operations. On the other hand, The top-down approach to algorithm design is a decomposition process focusing initially on the overall aspects of the task, and progressively dealing with greater and greater detail. The expansion of m:ndMPs represents precisely this concept.

Bottom-up design focuses initially on the most basic operations to be performed in the final algorithm, and then aggregates them into progressively more general operations. If the process is successful, then the final aggregation is equivalent to the algorithm desired. The most basic operations in m:ndMP terms are the constants defined by textual descriptions only, not by expansions. Aggregation of these constants via directed graphs into variables is analogous to bottom-up design.

Hybrid design uses a mixture of top-down and bottom-up design approaches, and probably more closely resembles the technique employed by most humans in general problem solving. The underlying concepts work the same ways.

In any approach to structured programming the final result can be viewed as an acyclic graph consisting of nodes and arcs. This is called the algorithm's solution graph. As illustrated by example in Figure 6.1, the nodes represent the variables and constants ed as building blocks, and the arcs represent the membership in a variable's expansion.

CHAPTER VI
MODELING WITH DIMENSIONED PROCESSES

How does one use m:ndMPs to design or describe an image processing algorithm? The answer to this question relies on the major attributes of the m:ndMPs and the grammar employing them. A clear and complete set of expansion rules, inclusion of confidence elements, multi-dimensionality, and a structured nature all combine to guide the designer systematically throughout the task of algorithm design.

The structured nature of m:ndMPs parallels that of most of the high level languages introduced along with the third generation of computers. Algol, PL/1, Pascal, C, Basic-plus, Fortran-77, and ADA support most structures. Even Fortran, Cobol, and Basic offer support for some structures [SH83].

Advantages afforded by structured programming far outweigh their disadvantages [SH83]. Not the least important of these are clarity and modularity. Design, development, and maintenance are intrinsically easier to perform on structured programs than on unstructured programs.

Several techniques are available which complement the structured approach to programming. Among these are the top-down, bottom-up, and hybrid design techniques.

render the m:ndMP concept unusable. The C-element, in fact, describes the entire m:ndMP, not just the transformation portion, and is therefore descriptive of itself as well. In fact, the acknowledgement of the philosophical concepts of unknowns and unknowables is the very feature that makes m:ndMPs so well-suited to design and analysis of image processing.

Image processing, like any worthwhile endeavor, is often advanced through inspiration and experimentation. It is therefore the result of both scientific research and artistic expression. Exclusion of the creativity offered by the artist may therefore stifle the advancement of image processing.

The uncertainties of the applicability of transformations based on guesswork and inspiration used to perform image processing can be expressed conveniently in the C-element of the m:ndMP which models the algorithm. When this m:ndMP is expanded, its subordinate m:ndMPs will reflect these uncertainties. Some of the subordinate m:ndMPs may have C-elements asserting that their m:ndMP is completely known, while others will assert unknown values. Ensuing analysis of the algorithm versus its actual performance can focus on the unknown values in subordinate m:ndMPs. In effect, the C-elements asserting unknown values flag their m:ndMPs as probable sources of unexpected results in actual performance of the algorithm.

is that inherent weakness or flaws are even apparent under cursory inspection.

Figure 7.3 shows a section of an expanded m:ndMP. As in Figure 7.2, the literal value of each node's C-element is included in the illustration. Asterisks mark the unknown C-elements in the expansion graph shown. If the expanded process is a heuristic demonstrated under actual testing to fail to yield the expected data, then the questionable C-elements flag their m:ndMPs as likely sources of the erroneous definitions.

a. $\{A^5, T, V^2, C\}$

b.

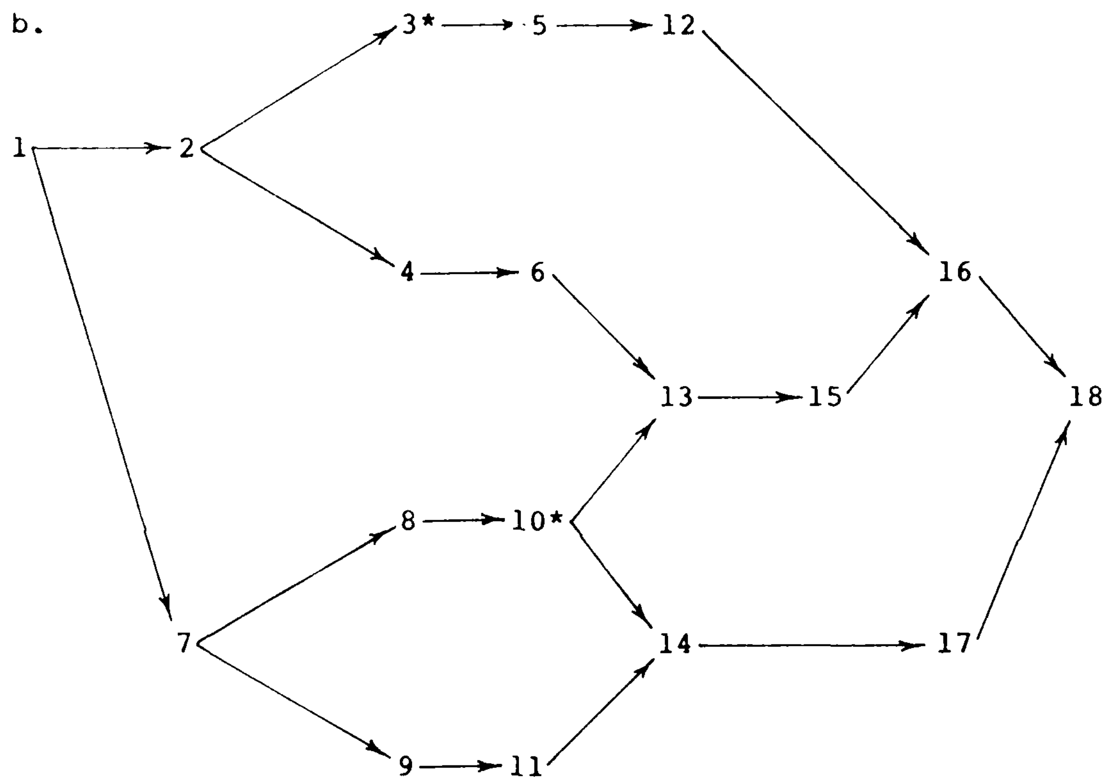


Figure 7.3, Example of an expanded 5:2dMP.

a. The original 5:2dMP,
b. Directed graph of the expanded 5:2dMP,
c. Node m:ndMPs.

c.

Node	m:ndMP
1	$\{A^5_8,\ T_1,\ B^8_5,\ true\}$
2	$\{B^8_5,\ T^1_2,\ D^5_4,\ true\}$
3*	$\{D^5_5,\ T^2_3,\ E^4_5,\ unknown\}$
4	$\{D^5_4,\ T^3_4,\ F^5_3,\ true\}$
5	$\{E^4_5,\ T^4_5,\ G^3_4,\ true\}$
6	$\{F^5_8,\ T^5_6,\ H^4_7,\ true\}$
7	$\{B^8_7,\ T^6_7,\ J^7_6,\ true\}$
8	$\{J^7_7,\ T^7_8,\ K^6_3,\ true\}$
9	$\{J^7_6,\ T^8_9,\ L^3,\ true\}$
10*	$\{K^6_3,\ T^9_{10},\ M^4_4,\ unknown\}$
11	$\{L^3_3,\ T^{10}_{11},\ N^4_2,\ true\}$
12	$\{G^3,\ T^{11}_{12},\ P^2,\ true\}$
13	$\{(H^4_4 + M^4_4),\ T^{12}_{13},\ Q^2_3,\ true\}$
14	$\{(M^4 + N^4),\ T^{13}_{14},\ R^3,\ true\}$
15	$\{Q^2_4,\ T^{14}_{15},\ S^4_1,\ true\}$
16	$\{S^4_3,\ T^{15}_{16},\ T^1_1,\ true\}$
17	$\{R^3,\ T^{16}_{17},\ U^1,\ true\}$
18	$\{(T^1 + U^1),\ T^{17}_{18},\ V^2,\ true\}$

Figure 7.3--continued.

CHAPTER VIII
MAINTAINABILITY AND NATURALNESS OF DIMENSIONED MODELING PROCESSES

Over the past thirty years, the ratio of hardware versus software costs of a new computer system has shifted markedly. Whereas hardware related costs dominated in 1955 by an average ratio of more than four to one, software costs now dominate by an average ratio of nearly ten to one [SH83]. Out of this large proportion of software costs, the largest share is software maintenance (i.e., for the improvement of the existing system and correction of its software errors).

Due to its modularity and clarity, a structured design is helpful in localizing errors and redesigning the offending sections of code [SH83]. Since m:ndMPs parallel the characteristics of structured programming languages, they benefit from the same modularity and clarity. Localization of errors tends to minimize the number of m:ndMPs suspected of contributing to a given error thus reducing the costs of error correction. Similarly, if a new feature needs to be added, the affected m:ndMPs are localized. Complete redesign of the entire image process is unnecessary if only a small section is affected.

In addition to the convenience of maintainability, m:ndMPs seem to provide a naturalness of expression for humans as well as computers. People and machines do not "think" the same way. No machine commercially available today even comes close to approximating the neurological structure of the average human being. On the other hand, computers can do some computations quickly, which human's would find difficult or even impossible to do within a normal life span.

This presents a problem to the designer of an image processing system. How does one translate human ideas of image processing algorithms into a form implementable in software and hardware on a computer? Probably the best answer to this question today is to employ a means of communication which possesses a sufficient vocabulary and is understood both by humans and computers.

High level languages attempt to provide just such a communication medium. Some languages sacrifice a large vocabulary in order to cater to some specific system design technique or subject matter. These languages are typically referred to as expert systems. They perform remarkably well within their limited arena of application, but are incapable of supporting other applications.

The languages based on structured grammars seem to have been most successful for general programming tasks. Their modular approach is easily translated into machine code. Most humans seem to be capable of viewing problems in terms of interrelated subproblems. This is the basis of the structured approach to problem solving.

The structured approach echoed in programming languages and m:ndMPs is an attempt to make them more natural. That is, they attempt to provide the means of communication of human algorithmic ideas to computers.

In the case of m:ndMPs, this communication is enhanced by the generality of the grammar. By defining the coordinate systems used and including the confidence qualifiers appropriately, practically any intellectual concept can be expressed in terms of m:ndMPs. Once expressed, the manipulation of these concepts becomes feasible even for a computer.

Perhaps the strongest argument favoring the description of m:ndMPs as having the characteristic of naturalness, is the inclusion of the confidence elements. As discussed in previous chapters, this element allows the inclusion of questionable and unproven algorithms called heuristics. Humans use heurstics habitually in their daily life, even when absolute solutions are possible. Many times this is due to the speed of the heuristic solution versus that of the absolute solution's algorithm. As long as the heuristic answer's failure rate is relatively low or acceptable according to the user's standards, its use can save precious decision-making time.

Image processing time on computers can be precious, too. This is especially true in real-time image processing systems. In these cases, the same priciple applies. Heuristics are often used in these applications. However, the fact that they are heuristics is not expressed in the design language (except, perhaps as documentary comments). When an error occurs or an update to the program is desired, these heuristics are prime candidates for future troubles if the full implication of their uncertain nature is not taken into account.

It is primarily for this reason that the confidence element is included in the m:ndMP grammar definition. Even when modifications are made to the image processing system, the existing confidence elements remain and, due to the combination rules of extended m:ndMPs, their effects are not overlooked. Hence, the communication of the heuristic nature of an algorithm is preserved by the m:ndMP grammar, unlike other languages.

There is at least one more argument in support of the naturalness of m:ndMPs, based on the idea that humans involved in designing image processing systems may want to anticipate future problems or changes in their design. This anticipation usually results in computer code less efficient but adapting well to the anticipated changes.

Once again, this feature of a design is not expressed in the programming language used to implement the design (unless it is included as a documentary comment in a source listing). To take full advantage of the foresight incorporated in the original algorithm, designers modifying a design should be aware of the intentions of the original design in this respect.

The anticipation of future modifications included in m:ndMPs generally results in hyper-defined transformations and confidence elements exceeding the minimum necessary confidence to perform the original transformation. When a modification needs to be made, these excesses are readily apparent and suggest themselves to the redesigner as means of realizing the new design. Thus, the naturalness of the original plan is preserved by the m:ndMPs used in the design.

CHAPTER IX
SUMMARY AND FUTURE CONSIDERATIONS

The m-dimensional to n-dimensional dimensioned modeling process (m:ndMP) has been defined. A grammar has been developed using m:ndMPs to describe image processing systems. Features of the grammar and m:ndMPs have been elaborated including multi-dimensionality, structured nature, ease of system design, analysis and maintenance, and the naturalness of dimensioned modeling processes. Furthermore, m:ndMPs have been designed to be generally applicable to any image processing task, rather than being restricted to a specific implementation or method of description.

Several cursory examples have been given illustrating these concepts. However, there seemed to be a tendency for the bulk of m:ndMP descriptions to become unwieldy. Some suggestions have been given to alleviate this, including adoption of a notational shorthand and predefinition of default and common dimensional descriptions.

Initial users of this grammar may experience a tendency to define every m:ndMP element in excruciating detail. Frustration is unavoidable if confidence elements are literally and completely defined. Many philosophical questions may arise about the nature of number systems and numerical or logical operations, which are unanswerable or for which a complete answer would not be necessary for the application being considered.

This tendancy, once tempered with a sense of relevancy of the information being described, would be easily overcome without significant effect on the positive features of the grammar and m:ndMPs. The effort expended in this learning process should not be any greater than that expended in learning any high level programming language.

Many of the features of the grammar presented herein parallel those of most high level programming languages available today. Foremost among these is the use of structured problem description. The advantages of the structured approach far outweigh any of its potential disadvantages. The grammar is fully compatible with currently accepted software engineering practices and theories.

The grammar exceeds the expressiveness of most high level languages, however. Inclusion of the confidence element in m:ndMPs allows accurate description of heuristics used in the image processing algorithm. Every aspect of system design, analysis, and maintenance is favorably affected by the inclusion of this element and its accompanying rules of combination.

If this grammar is to be widely adopted as a standard design tool, it would be highly desirable to standardize the confidence element definitions and a set of commonly used dimension descriptions. This would ease the comparison of independently created designs and help to standardize the education of users of the grammar. It could also assist in reducing the bulkiness of notation.

The concept of dimensioned modeling processes could be easily extended into a higher form of modeling process. With the appropriate definition of combinatorial, relational, and transformational rules, a set-based modeling process grammar is feasible. Such a grammar might be useful in the fields of general problem solving, database management, and artificial intelligence.

BIBLIOGRAPHY

[EL73] Elson, Mark, Concepts of Programming Languages. Science Research Associates, Inc., Chicago, Illinois, 1973.

[IL82] Iliffe, J. K., Advanced Computer Design. Prentice-Hall International, Inc., Englewood Cliffs, New Jersey, 1982.

[KN68] Knuth, Donald E., The Art of Computer Programming. Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1968.

[NI71] Nilsson, Nils J., Problem-solving Methods in Artificial Intelligence. McGraw-Hill, Inc., New York, New York, 1971.

[RI83] Rich, Elaine, Artificial Intelligence. McGraw-Hill, Inc., New York, New York, 1983.

[SH83] Shooman, Martin L., Software Engineering. McGraw-Hill, Inc., New York, New York, 1983.

[TO74] Tou, Julius T., and Gonzalez, Rafael C., Pattern Recognition Principles. Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1974.

BIOGRAPHICAL SKETCH

On October 26, 1952, Barry G. Litherland was born to Allyn C. Litherland and Regina K. Litherland in Jacksonville, Florida. At the age of six he moved to South Patrick Shores, one of the many residential areas around Patrick Air Force Base, Florida. He attended Sea Park Elementary School, DeLaura Junior High School and Satellite High School.

In 1970, the United States Air Force granted him a four-year scholarship at the University of Florida. On August 24, 1974, he graduated with a Bachelor of Science degree from the College of Engineering and the Department of Computer and Information Sciences. At the same time, he was commissioned as a Second Lieutenant in the United States Air Force.

On September 14, 1974, Murl Lynn Worley and Barry G. Litherland were married. During the next nine years, they shared the experiences of pilot training and military duty in North Dakota and Guam. They also were blessed with five beautiful children during these years: Anthony Michael Litherland, Christopher Jason Litherland, Jonathan Patrick Litherland, Timothy Matthew Litherland, and Kathryn Jeanette Litherland.

In 1983, the United States Air Force granted Captain Barry G. Litherland another scholarship. On December 15, 1984, he graduated from the University of Florida again, with a Master of Science degree from the College of Engineering and the Department of Computer and Information Sciences.

Upon graduation, Captain Barry G. Litherland was assigned to the United States Air Force Space Command.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Leslie H. Oliver, Chairman
Associate Professor of
Computer and Information
Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Douglas D. Dankel II
Assistant Professor of
Computer and Information
Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Donald G. Childers
Professor of Electrical
Engineering

This thesis was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School, and was accepted as partial fulfillment of the requirements for the degree of Master of Science.

December 1984

Dean, College of Engineering

Dean for Graduate Studies and
Research

END

FILMED

5-85

DTIC