Sea Grant College Program

Massachusetts Institute
of Technology
Cambridge, Massachusetts
02139

Equipment Arrangement Using Interactive Computer Graphics

C. Chryssostomidis
C. Graham, M. Meyers
P. V. Prakash

MITSG 86-3TN

Research sponsored by
Naval Sea Systems Command
United States Navy
in the
MIT Department of Ocean Engineering

MIT Sea Grant NA84AA-D-00046
R/T-28

Published by
Sea Grant College Program
Massachusetts Institute of Technology
Cambridge, MA 02139

April 1986

Table of Contents

ABSTRACT

This report is on the study of the arrangement of equipment in a ship compartment utilizing interactive computer generated graphics from an equipment library. Arrangement locations are sent via file to a ship's data base.

This study produced a methodology for managing computer arrangements and a program for transferring equipment arrangement locations via ASCII file to the ship's Integrated Data Base (IDB).

The program is written in VARPRO2 computer language for a ComputerVision (CV) 200X CGP minicomputer with the ComputerVision CADDS 4X CAD/CAM application program installed.

1 INTRODUCTION

1.1 General Background

The U.S. Navy has been developing computer supported design capabilities in all major ship design functional areas for many years. The principal shortfall of these design capabilities has been their lack of integration among functional areas. The Navy's present effort is to produce an integrated computer supported design process utilizing a common database system (Navy Integrated DataBase or IDB). Part of this effort will be to develop the capability to produce equipment arrangement drawings and to integrate the internal compartment arrangement data into the Navy IDB.

1.2 Problem Definition/History Timeline

OCTOBER 1984 - The proposed tasking of this study was to "develop the capability to produce compartment internal equipment arrangement drawings using the Computervision CAD/CAM system and to transfer this data to and from the ship's integrated database" [3]. The product of this study was to be a drawing and a description of the compartment in the IDB with the major emphasis on the method being user friendly. The ComputerVision system [1] was installed and made operational.

OCTOBER - DECEMBER 1984 - Presentations on system components (RIM,IDB,IGES,CV equipment, etc.) were conducted.

NOVEMBER 1984 - Navy RIM/IDB (Relational Information Management/Integrated DataBase) [2] containing some hull form (deck) files (but no real compartment data) were loaded into M.I.T.'s Joint Computer Facility's VAX mainframe computer. Utilizing SOFTECH Co.'s IGES (Initial Graphics Exchange Specification) processor an IGES file of DDG 51 Hold Deck is transferred to the ComputerVision System. This process proves to be very time consuming.

DECEMBER 1984 - NAVSEA divides the tasking: (1) SOFTECH is to produce a compartment extractor that would pull a single compartment's geometry out of the Navy's IDB, and to work on the IDB end of the data transfer; (2) M.I.T. to develop CV autoload routine based on IGES input from SOFTECH, and a methodology for compartment arranging management utilizing CV. NAVSEA suggests use of VARPRO2 as preferred programming language.

JANUARY - APRIL 1985 - CV system utilized to send data from library files to the compartment and then rearrange it. Methodology of arrangements and updates discussed. ASCII file format for data transfer discussed.

MAY - SEPTEMBER 1985 - VARPRO2 program development and refinement.

OCTOBER 1985 - Study report out, program presentation and demonstration.

2 METHODOLOGY

2.1 Basic Steps

Our approach for this study was to:

- broadly define the problem
- learn the component parts of the problem
- use the equipment to develop a feel for the process and the equipment constraints
- discuss the manner in which data is manipulated and transferred
- decide what information needs to be transferred between locations
- discuss who should control the data transfer
- redefine the problem objectives
- decide on the data file transfer type

- write and test a program

2.2 Constraints of the Equipment

Because of the limitations of the existing equipment only one workstation, the main processor, and the disc drive were put into operation. Files were transferred between VAX and CV via tape because a CV modem and interface were not available. These files were then processed by the IGES program in the CV computer. This proved extremely time consuming and aided in our decision to seek out the absolute minimum of data that would need to be transferred and the fastest way to transfer it.

2.3 Trade-Offs

Several basic trade-offs became apparent as this study proceeded. Some were presented by equipment constraints (as above), others by refining the problem definition, and all of the trade-offs are related in one manner or another.

- Universal application vs Machine specific: The study initially set out with the intention of designing an application program that would be universally transportable among all graphics machines and mainframe computers. This is possible by use of IGES (Initial Graphic Exchange Specification). The process of making information transfer machine independent requires far greater detail than making it machine specific. Additionally the time required to convert to and from IGES format becomes prohibitive as the level of detail and amount of information increases. It was decided (with NAVSEA concurrence) to make the process machine specific for the CV computer.

- IGES vs ASCII format: The time required to transfer detailed complex geometries through IGES format forced a hard look at the type and amount of data really required to be transferred. It was decided that complex equipment geometries were not required to be stored in the IDB since they would only be used in the CV process. Since geometries were not to be transferred and the process was to be machine specific, then the IGES format was no longer the only format acceptable. The ASCII format was then adopted because of its ease of use, speed in transfer, and its acceptability by all machines involved.

- Level of Detail: In order to reduce the amount of time involved in a transfer and to ascertain the accuracy of the information transferred, a minimum of detail would suffice to be sent. The absolute minimum for a given piece of equipment would be a bounding box, its location and orientation within the compartment, its generic geometric name (NFIG part name), and its unique component name. Any mass property information or calculations concerning a unique component would be maintained as part of the generic component file in the IDB. It was later decided to reduce the bounding box to three axes which will include access requirements for that piece of equipment (see Appendix I).

- User friendliness: In the problem definition it was stated that a high level of user friendliness was desired. This was translated into creating a program that requires the least number of inputs from the individual user and only a minimal understanding of the internal mechanism of this program.

2.4 Information Transfer

In order to send the least amount of data back and forth it was decided not to send the geometries (neither hull nor equipment part) back to the IDB once it was resident on the CV minicomputer. The data transfer file will contain only X Y Z Axis length and orientation with respect to the ship's geometry origin, equipment geometry (NFIG) name, unique equipment 'label', and compartment name. To speed up the transfer the ASCII file type was chosen over the IGES format. The equipment geometry (NFIG) name refers to the CV library part (NFIG) name. This is the generic CV part Nodal-Figure (NFIG) by which arrangers can use predefined geometries (NFIGs) from equipment libraries resident on the CV and not have to transfer bulk computer graphical geometries to the IDB and back. The equipment 'label' will identify each piece of equipment on the ship specifically and uniquely.

2.5 Compartment Arrangement Methodology

A generic compartment arrangement methodology can be developed to provide the general direction to designers and arrangers for the three basic types of compartment arrangements (machinery, combat systems, human support). Specific direction will additionally be required for the arranging of each of the three types of compartment. The methodology will be largely independent of the manual or computer aided design tools used.

The compartment arrangement methodology should include:

- A listing of the types of components/equipment required in the compartment. Standard lists for various compartments would serve as reminders to those developing a specific equipment listing for a particular compartment.

- A catalog of standard Navy component/equipment geometries (NFIG parts library of equipment already in the Navy) can be prepared. This catalog (along with physical descriptions) will serve as a reference for arrangers.

- Adjacency requirements Certain components of subsystems to be arranged in a compartment have adjacency requirements to the other components. Adjacency can be required for the efficiency of crew operation and maintenance, equipment limitations (e.g., pressure drop) and other rationale. Necessary and desirable adjacency criteria for standard Navy components should be specified in the above mentioned catalog.

- Clearance requirements: Clearance between components can be dictated by removal, maintenance access, or other operational reasons. Minimum and ideal clearance criteria should be specified. These requirements (adjacency and clearance) can be incorporated into the axis definitions of the equipment NFIGs in the parts library.

- Arrangement Efficiency Guidelines: Certain common-sense guidelines exist which can lead to minimum ship impact arrangement solutions. For example, in a combat system compartment, noncritical components such as file cabinets should be located along weather bulkheads to add to the fragment protection for critical components.

- Assessment: Design indices can be defined to assess the goodness of a compartment arrangement. For example, equipment footprint to compartment area ratios can prove an indication of overall arrangement tightness. These parameters can be determined for previous designs to serve as guidelines.

A check on the methodology can be obtained by providing it to a novice to use for arranging sample compartments and then compare them with existing 'successful' drawings.

As the primary objective of this project was to develop the methodology and programs for data transfer between the CV system and the IDB, the compartment arrangement methodology was not pursued. This is a topic for a future effort.

2.6 Management of the IDB

It is foreseen that there will be several versions of the IDB in existence at any one time (i.e., Approved, Proposed, Archived). These versions will be supervised by the IDB supervisor who is in turn controlled by the Ship Design Manager. The Ship Design Manager will state on a macro level which version of the IDB is approved and which is proposed. Each functional code will take the Approved IDB version data that they need to work with and install it in their code as the approved baseline. The codes will then manipulate and insert data as necessary producing a new 'proposed' file. The code supervisor will present the 'proposed' file to the Ship Design Manager. If the Ship Design Manager accepts these changes, then they will be allowed to be placed in the IDB's Proposed version. When the Ship Design Manager is satisfied with the IDB Proposed version, he will authorize the IDB supervisor to archive the old Approved version, make the Proposed the new Approved version, and release this new Approved version to all the functional codes so as to continue the iterative design process.

3 CONCLUSION

- Integrated Database - it is difficult to develop an interface mechanism for a database that is not fully defined.

- Data Transfer - it is best to transfer data in the simplest fastest manner possible, and to transfer the absolute minimum amount of data required.

- Geometry Transfer - only transfer geometries when absolutely unavoidable.

- Management of Data - there must be only one central controller/decision maker from whom versions of data are approved, proposed, or archived.

4 RECOMMENDATIONS

On the basis of our study, we recommend the following:

- Do not transfer the hull geometry back to the IDB where it is already resident.

- Use predefined equipment geometries (NFIGs) for the equipment library and store them at the CV end.

- Consider requiring manufacturers to submit IGES compatible NFIGs on magtape when providing the equipment.

- Build into the NFIGs the access requirements (maintenance and operational) (see axis definition in APPENDIX I section I-2).

- Use the lower-left-front corner as the local origin for each piece of equipment.

- Use ASCII file type for data transfer.

I. TRANSFER PROGRAMS MANUAL

I.1 General description of the program

The program to be described in this Appendix performs the tasks of entering and extracting information about specified objects from the CV database. These programs are written in VARPRO2 (a command language of the CV designer system). While this section does not assume knowledge of the CV CADDS system the other sections in this Appendix require concepts used in the CADDS environment.

In the program, the unit of information transfer is a compartment in a ship. The unit of information access and update is a piece of equipment whose geometry is defined in the parts library. The main physical unit of transfer is a file containing data about equipment in one compartment. The data consists of a box description of the equipment in a particular location in the compartment. A box description means the location and orientation of a box which surrounds the equipment completely. The placement of the box in a particular location instead of the equipment guarantees the reservation of the correct amount of space.

The extraction of the information from the database depends on a specialised but simple definition of an axis system in each library part. A library part is a component in the compartment whose geometry is defined separately as a part. The axis definitions are given in detail in Sec.I.2.1. The box description of the part is obtained from this specialised axis system attached to each part. Since this axis information is available only for parts whose geometry is defined in the library, only information about library parts may be sent to the IDB. In other words, in a compartment containing library defined parts and non-library entities only library parts may be sent back and forth. The non-library entities have to be those whose existence is not important at the IDB. (eg. Drawing aid notes etc.)

The main items of information that we need to know about the equipment ie. the box description are available by accessing the axis definitions of the equipment. These are

1. Its generic equipment name.
2. Its unique equipment id.
3. Its bounding box. (size, location and orientation)

The generic equipment name defines the unique <u>type</u> of object and is the name of the file describing its geometry in the parts library. The equipment ID uniquely identifies a particular piece of equipment in the whole database and provides the key to search for the part in the IDB. Another important piece of information not mentioned above is the status of the part in the compartment. The part could have been deleted or added/moved in the compartment. This information is conveyed through a flag indicating how to use the information sent over. This information is not included in the above list as it is not obtained from the axis definition but is explicitly provided by the user.

Parts are created in the CV database as library parts and then inserted into a compartment. During the arrangement of these parts in the compartment, they may be moved, deleted or left unchanged at the inserted position. Those parts that are newly inserted, moved or deleted must be sent to the IDB for update. Similarly, parts which have been changed at the IDB are sent over to CV for update. The only possible changes at the IDB are deletion and movement of a part to a different location. Both the forward and backward transfers ie. to and from the IDB use the same core information. The forward transfer (to the IDB) distinguishes only between parts that have or have not been deleted at the CV. The backward transfer (from the IDB) distinguishes only between items that have been deleted and those that have been moved at the IDB. The lack of need for unchanged information is because the updates at the CV end are made on the present compartment description.

On a user level, the main interaction is in the form of a series of digitizes (pointing mechanism) pointing at the part to be sent. The program prompts the user for the part identifying digitizes and to start sending the information to a file. At the end of each transaction, a file containing the information about the compartment and its parts is created. It is this file which is sent to update the IDB.

The information from the IDB to update the CV end will be in a format similar to the file that was created when the parts were sent over to the IDB. This information is used to update the particular compartment. It is expected that only parts whose positions have been changed or deleted at the IDB will be transmitted back. This process is carried out by pulling in the compartment in question and examining the first part in that file. If this part is to be deleted, it is deleted in the compartment. If it is to be moved, it is deleted at its present position and a new instance of the part is brought in at the new position. The unique part ID information from the IDB is then inserted into the axis system to uniquely identify this new instance as an old existing part. A text label of the same ID appears at a standardised position relative to the part. Any equipment in the compartment whose status is unchanged at the IDB will remain unaffected. The program then files away the new updated compartment separately.

Section I.3 describes the input and output of the program and its interpretation. Section I.4 steps through a sample run of the program to clarify the steps involved. The next section describes the preparatory steps required to create the axes in the library files and the environment that is required for the successful execution of the programs.

I.2 Preparation steps

I.2.1 How to create a library part

The creation of a library part in model space involves the definition of coordinate frame and a description of its geometry in relation to that coordinate system. The specification here only deals with the position of a coordinate frame in relation to the geometry and addition of a few entities at or near the origin.

The following steps specify the additional steps to be taken to create the required axes for a part whose geometry is defined

1. Place the origin at a point exterior to the geometry of the part and which is most useful in an insertion of the part into a compartment. eg. A chair should have its origin on one corner of its base while a ceiling attachment should have its origin at one corner of its top face. The exterior requirement is included so as to be able to include access space requirements of the part in an arrangement context.

2. Create line entities along the three axes of length such that a box having these three lines for the sides will completely enclose the equipment and any access space associated with it.

3. Insert the properties into the x-axis line in the following way

Property name	Description
PARTCODE	Name of the library part whose axis is taking this property.
NOTE	The equipment ID slot for insertion upon use in a compartment.

4. Group the line X, line Y, line Z using the CONSTRUCT GROUP command of the CV CADDS system.

I.2.2 Creating a particular instance of the part

The insertion of the NOTE property is superflous inside the library as the part is not uniquely defined in the library. On creating a particular instance of the part in the compartment a text string that uniquely identifies this part must be inserted into the NOTE property. So two parts may have a common library part describing its geometry but would have a different NOTE property definition and hence a different identity.

I.3 Input/Output

I.3.1 User interaction

The input to the program that does the CV to IDB transfer is minimal and involves entry of the following pieces of information.

1. The name of the file into which the information is to be written.
2. An answer to the question 'Any more parts to send ?' with an Yes or No.
3. A response to a query about the status of the part to be sent ie. a delete or change status. If the status is 'delete', the part will be deleted and a delete signal will be sent to IDB. The change status is the default setting for this query.
4. A digitise to identify the part to be sent. Since the program is looking for the grouped axis definition, a digitize close to the axis is required.

The output of the program is in the form of a file. This file is unique for this run and contains only information about one compartment. The information is organised into a header block containing compartment information, date and time and a block for every part digitised to be sent over. This block contains size, location and orientation information about the part and two textual pieces of information.

The exact formats are given below along with the description of each field. Sec.I.3.2 defines the terms used in the input and output of the program.

```
Compartment block

COMPARTMENT_NAME  DRAWING_NAME DATE TIME   (4A10)
2 blank lines

Part block

Identifier text   PARTNAME                 (2A10)
Identifier text   EQUIP_ID                 (2A10)
```

```
Identifier text   STATUS                        (A10,I3)
Identifier text   X Y Z                         (A10,3F10.4)
Identifier text   DX1 DY1 DZ1                   (A10,3F10.4)
Identifier text   DX2 DY2 DZ2                   (A10,3F10.4)
Identifier text   DX3 DY3 DZ3                   (A10,3F10.4)
blank line
```

The field identifier text is just an aid for visual inspection of the file and is not required information for entry into IDB. The part blocks are repeated for every part that was sent over.

The input to the program that does the IDB to CV transfer only requires the file to be used for the update. The structure of this input file is identical to the above format except for parts whose position/orientation has been changed. In such cases the old origin is required along with the new origin and the axis vectors. The block for such a part in the input will look like this

```
Compartment block

COMPARTMENT_NAME  DRAWING_NAME DATE TIME   (4A10)
2 blank lines

Part block

Identifier text   PARTNAME                      (2A10)
Identifier text   EQUIP_ID                      (2A10)
Identifier text   STATUS                        (A10,I3)
Identifier text   Xo Yo Zo                      (A10,3F10.4)
Identifier text   Xn Yn Zn                      (A10,3F10.4)
Identifier text   DX1 DY1 DZ1                   (A10,3F10.4)
Identifier text   DX2 DY2 DZ2                   (A10,3F10.4)
Identifier text   DX3 DY3 DZ3                   (A10,3F10.4)
blank line
```

I.3.2 Definitions

COMPARTMENT_NAME : Unique name given to the compartment and is used as the part file name for activation of the compartment.

DRAWING_NAME : Any drawing associated with that compartment from which the parts were sent over. It has no use at the IDB end but is used for activation of the compartment while coming back.

DATE AND TIME : Date and time at start of sending out of the information.

PART_NAME : The name of the library part(CV) that contains the NFIG of the generic equipment. It is also the text of the PARTCODE inserted into X-axis line of the library part definition. (See section on Preparation steps)

EQUIP_ID : is the unique name/ID given to a particular instance of the generic equipment. It is incorporated into the axis system of the generic part after inserting the part at the appropriate location in the compartment. It is used as a key in locating the part in the IDB.

STATUS : A flag indicating the part has been deleted or changed at the CV end and whether it has been deleted or moved to a different location at the IDB. (0 - change; 2 - delete)

ORIGIN x y z : The position of the origin of the equipment coordinate system in the compartment coordinate system.

X AXIS dx dy dz : A free vector along the x axis of the parts coordinate system described in the compartment's coordinate system.

Y AXIS dx dy dz and **Z AXIS dx dy dz** are similar to the X AXIS definition.

A box drawn in the compartment with one vertex at the position vector and described by X, Y, and Z AXES free vectors at the origin would bound the instance of the part specified by EQUIP_ID.

I.4 Example

In this example we will go through the sequence of creating a new library part and its axes definitions. It is assumed that the reader has knowledge of the CADDS command language and with the general use of CADDS modeling concepts. Assume that a library part named LIB.TABLE exists with the geometry of the table completely defined ie. previously created. We will insert the axes required to hold the 'box' information of the part that will be useful in a transfer to the IDB. The equipment ID is then inserted into the axis system after inserting the part into the compartment. This equipment ID inserted after entry in the compartment will be different for different instances of the same type of object. Later, on execution of the transfer program the various steps in the execution are shown and a sample output file is given. The phase of reverse update is also explained by the execution of the program that updates the CV database based on changes in the IDB.

I.4.1 Create an axis definition in a library part

The old existing part in the library is activated using the CADDS commands. The name of the library part file is LIB.TABLE and the drawing name used is ISO for this example. The name of the drawing can be any valid drawing under the part.

```
# ACTIVATE PART LIB.TABLE
# ACTIVATE DRAWING ISO
```

Fig. 1 shows the description of the part at this stage. If the origin of the coordinate system is inside the library object's geometry then the origin must be moved to a point outside the object. Lines are then drawn along the x, y and z axes so that their lengths bound the part and any accompanying access space requirements.

```
# INS LIN: x0y0,xP
# INS LIN: x0y0,yQ
# INS LIN: x0y0,zR
```

where P,Q and R are the length, breadth and deptth of the bounding box required. Now insert properties that hold textual information about the library partname (LIB.TABLE) and a equipment ID. Let the test equipment ID be the TABLE X for this example. In CADDS we use the command

```
# INSERT PROPERTY PARTCODE "LIB.TABLE"  NOTE "TABLE X": d
```

where the digitize represents the X axis line entity. At this stage the entry of the NOTE property is not required but is just a place holder for its subsequent entry when inserted in a compartment. Fig. 2 shows the axes along with the geometry of the library part. The library part is then filed away as an extended NFIG.

I.4.2 Insertion in a compartment

The compartment is first activated and then the library part is pulled in at the right location. In this example, NP.COMPT is the name of the part file describing the compartment geometry.

```
# ACTIVATE PART NP.COMPT
# ACTIVATE DRAWING ISO
# INS NFIG LIB.TABLE: X1Y1
# CHANGE PROPERTY NOTE "TABLE A" : X axis of
                                   newly inserted table
```

The part CHAIR (LIB.CHAIR) exists when the compartment is activated and has been created earlier through a similar process and carries the equipment ID CHAIR A. A second insertion of the part LIB.TABLE at a different location would require insertion of a different NOTE property to identify it separately. Here we have used TABLE B. As a separate example we also insert a file cabinet (LIB.FILCAB) and change its note property to read CABINET A. Fig. 3 shows how the parts in the compartment look at this stage schematically. The compartment is now ready for any information to be sent over to the IDB.

I.4.3 Sending the information to IDB

The sending involves the execution of a program file which prompts the user for digitizes specifying which parts are to be sent. If program is on file NP.CVIDB then the command

EXECUTE FILE NP.CVIDB

runs the program. A listing of the program NP.CVIDB. The program first requests the name of the file into which the data is to be written.

LET &FILE = '<VAR>NP.IDBIN'

The underlined text is user supplied file name and should represent a valid CV file name. The next user input is a digitize of the part to be sent.

IDENTIFY 3,&AXES

GRO d

where d is a digitize on the axis system of the part to be sent. In case, the digitize is made wrongly and is to be repeated, type GRO for 'group' and then follow it with the digitize. Here we digitized the axis system corresponding to TABLE A.

The next two inputs are flags to indicate if more parts to be sent or not and if the part just digitised is to be deleted from the compartment and at the IDB (see section on INPUT/OUTPUT for the options). At this stage, we have more parts to be sent and do not want to delete the part.

LET &S,&STAT(&N) = <VAR> 1,0

The 'identify' and flag inputs are requested for each part to be sent over. In our case we do the following sequentially.

IDENTIFY	&S	&STAT(&N)
x axis of TABLE B	1	0
x axis of CHAIR A	1	2
x axis of FILCAB A	0	0

The input of the value for the flags may be skipped if the default values are to be used. The defaults are &S = 1 and &STAT(&N) = 0. ie. if

LET &S, &STAT(&N) = <VAR> 1,0 CTRL X

may be reduced to just

LET &S, &STAT(&N) = <VAR> CTRL X

. After the send flag of &S = 0 is given the program loops through each digitised part to extract the information and prints the information to a specified file. The file created for our present run when all parts were sent over will look like the sample input file in Sec.I.6.2. The file is now resident in CV and awaits transfer to the system where IDB is installed via tape or a data communication medium.

I.4.4 Re-entry of updated information

Assuming the file containing the information sent back from IDB exists this process is executed with no user interaction except to specify the file to be used.

EXECUTE FILE VAXCV

will prompt the user for a file name. A listing of the program IDBCV is given in Sec.I.6.4.

LET &FILE = "<VAR>NP.VAXOUT"

where the underlined text is the user-supplied file name. A sample input file for IDBCV program ie. an output file from IDB is shown in Sec.I.6.3. It will then update the compartment mentioned in the file and file away a new version along with the old version of that compartment. This process can be viewed as it goes through the entry sequence. This program also requires access to the library definitions ie. the use of the library NFIG's used in the original construction of the compartment. This new file is filed under the old compartment file name with the .NEW extension. Eg. Compartment 'NP.COMPT' would have the new version filed under 'NP.COMPT.NEW'. A final view of the compartment for our example is shown in Fig. 5. Note the deletion of the file cabinet as requested by IDB and shown in the input file as a status of 2.

I.5 Variable listings

&AXES(3) - an entity array which temporarily holds the three line entities corresponding to the three axes of the part digitized.

&ALIST(i,i+1,i+2) - holds the three line entities as in &AXES but for all the entities digitized. Set of three array indices corresponds to one part.

&END1(3),&END2(3) - location variables that recieve the end points of the three axis lines. END1 is one end. END2 is the other.

&ORIG - location of the origin of the part digitized.

&XAXIS, &YAXIS, &ZAXIS - location variables holding free vector data about the three axes relative to the &ORIG.

&PCNT - A number indicating how many properties have been retrieved.

&P, &Q - Integers bounding elements of the array &ALIST that contain entity variables corresponding to one part.

&S - flag indicating more parts to send or not. (1 - yes, continue ; 0 - no, start sending)

&STAT(i) - flag indicating status of the i^{th} part being sent.

&DRGNAME - name of the drawing from which the EXECUTE FILE command is given.

&CMPNAME - name of the compartment from which the EXECUTE FILE command is given.

&DATE, &TIME, &TASK - date, time and task (CV terminology) at which the program was executed.

&PRTNAME - file name of the library part corresponding to one digitize. Text held in the PARTCODE property of the part.

&PRTLABL - label of the library part corresponding to one digitize. Text held in the NOTE property of the part.

&NOTE - text variable used to hold the text 'NOTE'.

&PRTCODE - text variable used to hold the text 'PARTCODE'.

&FILE - text variable used to hold the user-supplied file name.

&WRITE1 - file pointer corresponding to file &FILE.

&EMP - empty textstring

&NEW - file name of the new version of the updated compartment.

&DLOC - location variable used for deletion and other miscellaneous purposes.

&COUNT - a temporary counter used to count the records in the file to be skipped.

I.6 Program listings and sample I/O files

I.6.1 CV to IDB program

```
BEGIN VARPRO2
CLEAR
*;
*;   DECLARE ALL VARIABLES
*;
DECLARE ENT &AXES(3),&ALIST(90)
*;
DECLARE LOC &END1(3),&END2(3)
DECLARE LOC &ORIG,&XAXIS,&YAXIS,&ZAXIS
*;
DECLARE NUM &PCNT
DECLARE NUM &P,&Q,&STAT(100),&S,&N
*;
DECLARE TEXT &CMPNAME,&DRGNAME,&PRTNAME
DECLARE TEXT &DATE,&TASK,&TIME
DECLARE TEXT &PRTLABL,&NOTE,&PRTCODE,&FILE
*;
LET &NOTE = "NOTE"
LET &PRTCODE = "PARTCODE"
*;
*;    OBTAIN ALL STATUS DATA
*;
OBTAIN STATUS &DATE,&TIME,&TASK,&CMPNAME,&DRGNAME
*;
LET &FILE = "NP.VAXIN"
*;
*;
*; ENTER THE FILE NAME AT THE <VAR> PROMPT INTO WHICH THE DATA IS
*; TO BE WRITTEN. DEFAULT IS NP.VAXIN
*;
LET &FILE = "<VAR>"
*;
OPEN WRITE &WRITE1 &FILE
WRITE &WRITE1 (4(A15))  &CMPNAME,&DRGNAME,&DATE,&TIME
WRITE &WRITE1 (' ')
WRITE &WRITE1 ('  ')
```

```
LET &S = 1
WHILE &S .GT. 0
LET &N = &N+1
*;
*;  IDENTIFY THE PART BY DIGITIZING CLOSE TO ONE OF THE
*;  AXES OF THE PART.
*;
IDENTIFY 3,&AXES
GRO <VAR>
LET &P = 3*(&N-1) + 1
LET &Q = &P + 2
LET &ALIST(&P.TO.&Q) = &AXES
*;
*; ENTER THE 'CONTINUE OR SEND' FLAG VALUE (0 OR 1) FOLLOWED
*; BY THE STATUS FLAG. THE DEFAULT FLAGS ARE '0 = CONTINUE'
*; AND '0 = CHANGE'. USE CTRL X TO END VARIABLE INPUTS.
*;
LET &S,&STAT(&N) = <VAR>
*;
ENDWHILE
*;
WHILE &N.GT.0
LET &P = 3*(&N-1) + 1
LET &Q = &P + 2
LET &AXES = &ALIST(&P.TO.&Q)
*;
*;  OBTAIN THE COORDINATES OF LINES FORMING THE AXES
*;
OBTAIN LINE 1,&AXES(1),&END1(1),&END2(1)
OBTAIN LINE 1,&AXES(2),&END1(2),&END2(2)
OBTAIN LINE 1,&AXES(3),&END1(3),&END2(3)
*;
*;   GET THE NAMES DEFINED IN PROPERTIES
*;
OBTAIN PVAL &AXES(1),&NOTE,&PCNT,&PRTLABL
OBTAIN PVAL &AXES(1),&PRTCODE,&PCNT,&PRTNAME
*;
*;        CREATE THE ACTUAL VECTORS FOR THE AXES
*;
LET &ORIG = &END1(1)
LET &XAXIS = &END2(1) - &END1(1)
```

```
LET &YAXIS = &END2(2) - &END1(2)
LET &ZAXIS = &END2(3) - &END1(3)
*;
*;   WRITE OUT THE DATA
*;
WRITE &WRITE1 ('PARTNAME  ',A20) &PRTNAME
WRITE &WRITE1 ('EQUIP-ID  ',A20) &PRTLABL
WRITE &WRITE1 ('STATUS    ',I3) &STAT(&N)
IF &STAT(&N).NE.2
WRITE &WRITE1 ('ORIGIN    ',3F10.4) &ORIG
WRITE &WRITE1 ('X AXIS    ',3F10.4) &XAXIS
WRITE &WRITE1 ('Y AXIS    ',3F10.4) &YAXIS
WRITE &WRITE1 ('Z AXIS    ',3F10.4) &ZAXIS
ENDIF
WRITE &WRITE1 (' ')
IF &STAT(&N).EQ.2 , DELETE ENT: NFIG&ORIG
*;
*;   SEE IF OKAY
*;
LET &N = &N - 1
ENDWHILE
CLOSE &WRITE1
PRINT FILE &FILE
FILE PART <VAR>
END VARPRO2
```

I.6.2 Sample output file of Program CV-IDB

```
 NP.CMP.TEST    ISO              11-08-85              01:13:57


PARTNAME  LIB.FILCAB
LABEL     FILCAB A
STATUS      0
ORIGIN        5.0906    4.9020    0.0000
X AXIS        2.2500    0.0000    0.0000
Y AXIS        0.0000    3.7500    0.0000
Z AXIS        0.0000    0.0000    4.2500

PARTNAME  LIB.CHAIR
LABEL     CHAIR A
STATUS      2

PARTNAME  LIB.TABLE
LABEL     TABLE B
STATUS      0
ORIGIN        3.7324    1.2567     0.0000
X AXIS        2.5000    0.0000     0.0000
Y AXIS        0.0000    1.7500     0.0000
Z AXIS        0.0000    0.0000     1.7500
```

I.6.3 Sample input file for Program IDB-CV

```
 NP.CMP.TEST     ISO              11-08-85         01:13:57


PARTNAME  LIB.FILCAB
LABEL     FILCAB A
STATUS      2
ORIGIN        5.0906     4.9020      .0000
X AXIS        2.2500      .0000      .0000
Y AXIS         .0000     3.7500      .0000
Z AXIS         .0000      .0000     4.2500

PARTNAME  LIB.TABLE
LABEL     TABLE B
STATUS      1
OLD-ORIGIN    3.7324     1.2567      .0000
ORIGIN        3.7324     1.2567     2.0000
X AXIS        2.5000      .0000      .0000
Y AXIS         .0000     1.7500      .0000
Z AXIS         .0000      .0000     1.7500
```

I.6.4 IDB to CV program

```
 BEGIN VARPRO2
CLEAR
DECLARE TEXTSTRING &CMPNAME,&PRTNAME,&PRTLABL,&DRGNAME,&EMP
DECLARE TEXTSTRING &DATE,&TIME,&NEW,&FILE
DECLARE NUMERAL &S
DEC NUM &STAT,&COUNT
DECLARE LOCATION &DLOC,&END1(3),&END2(3)
*;
*; THE ONLY INPUT REQUIRED FOR A RUN OF THIS PROGRAM IS
*; THE NAME OF THE FILE FROM WHICH THE UPDATE IS TO BE MADE.
*; TYPE IN THE NAME OF THE FILE AT THE <VAR> INPUT PROMPT.
*; ONLY THE FILE NAME MUST BE INPUT WITHOUT THE QUOTATION MARKS.
*; IF A MISTAKE IS MADE IN THE ENTRY, ABORT THE VARPRO2
*; PROGRAM AND START AFRESH. THE DEFAULT FILE NAME IS NP.VAXOUT.
*;
*;     READ IN THE DATA FOR THE COMPARTMENT
*;
LET &FILE = "NP.VAXOUT"
LET &FILE = "<VAR>"
OPEN READ &READ &FILE
READ &READ (2(A15)) &CMPNAME,&DRGNAME
READ &READ (A10) &EMP
READ &READ (A10) &EMP
ACT PAR &CMPNAME
ACT DRA &DRGNAME
*;
LET &S = 1
WHILE &S .GT. 0
*;
READ &READ (10X,A10) &PRTNAME
IF &V2EOF.GT.0 , EXITLOOP
READ &READ (10X,A10) &PRTLABL
READ &READ (10X,I3) &STAT
*;
IF &STAT.EQ.0
LET &COUNT = 1.0
WHILE &COUNT.LE.5
READ &READ (A10) &EMP
LET &COUNT = &COUNT + 1
```

```
ENDWHILE
ENDIF
*;
*;      DELETE THE PART REMOVED AT THE IDB
*;
IF &STAT.EQ.2
READ &READ (10X,3F10.4) &DLOC
LET &COUNT = 1
WHILE &COUNT.LE.4
READ &READ (A10) &EMP
LET &COUNT = &COUNT + 1
ENDWHILE
DELETE ENTITY : NFIG &DLOC
ENDIF
*;
IF &STAT.EQ.1
READ &READ (10X,3F10.4) &DLOC
DELETE ENT: NFIG &DLOC
*;
READ &READ (10X,3F10.4) &END1(1)
READ &READ (10X,3F10.4) &END2(1)
READ &READ (10X,3F10.4) &END2(2)
READ &READ (10X,3F10.4) &END2(3)
READ &READ (A10) &EMP
ENDIF
*;
IF &STAT.EQ.0.OR.&STAT.EQ.2 ,GOTO  MAINLOOP
*;
*;
*;         ACTIVATE THE MODEL AND INSERT THE SPECIFIC FIGURE
*;
LET &END2(1) = &END1(1) + &END2(1)
LET &END2(2) = &END1(1) + &END2(2)
DEF CPL XYPLANE :&END1(1),&END2(1),&END2(2)
SEL CPL XYPLANE
INS NFIG &PRTNAME :&END1(1)
SEL CPL TOP
LET &DLOC = &END1(1) + LOC(1.0,1.0,0.0)
INS TEXT &PRTLBL:&DLOC
LET &DLOC = 0.5*(&END1(1) + &END2(1))
CHANGE PROP NOTE &PRTLABL:  &DLOC
```

```
DEL CPL XYPLANE
LABEL MAINLOOP
ENDWHILE
*;
*;
LET &NEW = &CMPNAME+".NEW"
LET &NEW = XPAD( &NEW,0)
DEL PART &NEW
FILE PART &NEW
EXIT PART Q
END VARPRO2
ACT PART &NEW
ACT DRA &DRGNAME
```

References

[1] *CADDS 4 User Interface Reference*
ComputerVision Corporation, Feb 1984.

[2] *RIM Users manual*
Boeing Computer Services, .

[3] *Integration of Equipment Arrangements with Ships Data Base using Interactive Graphics*
Sea Grant Proposal (M. I. T. R/T - 28), 1984.

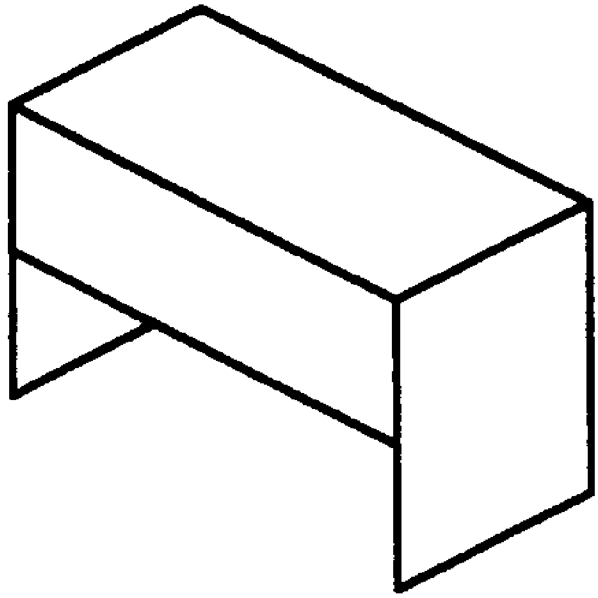


Fig. 1 Geometry of a library part, eg. a table before the axes are defined.

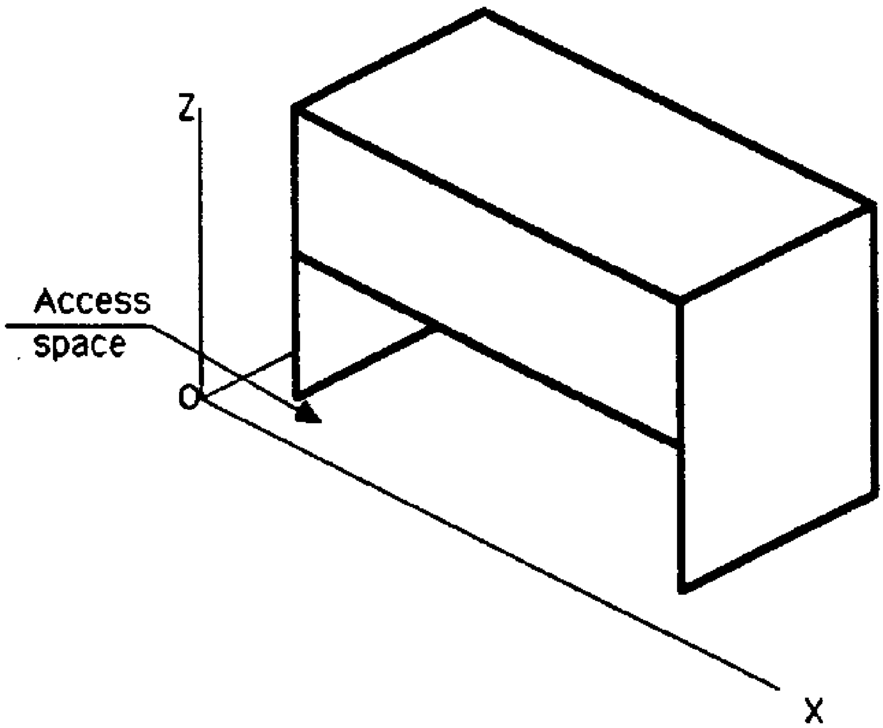
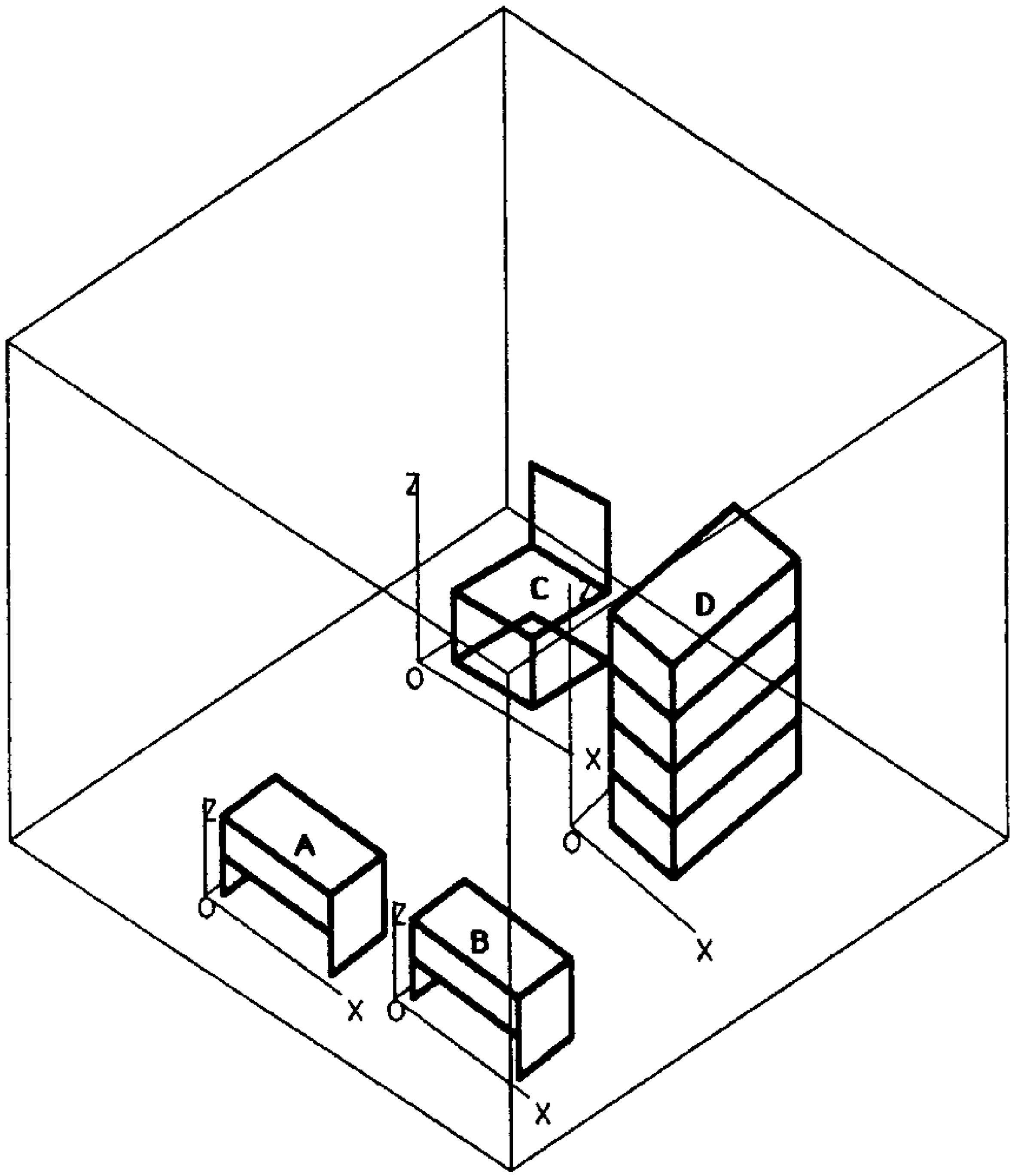
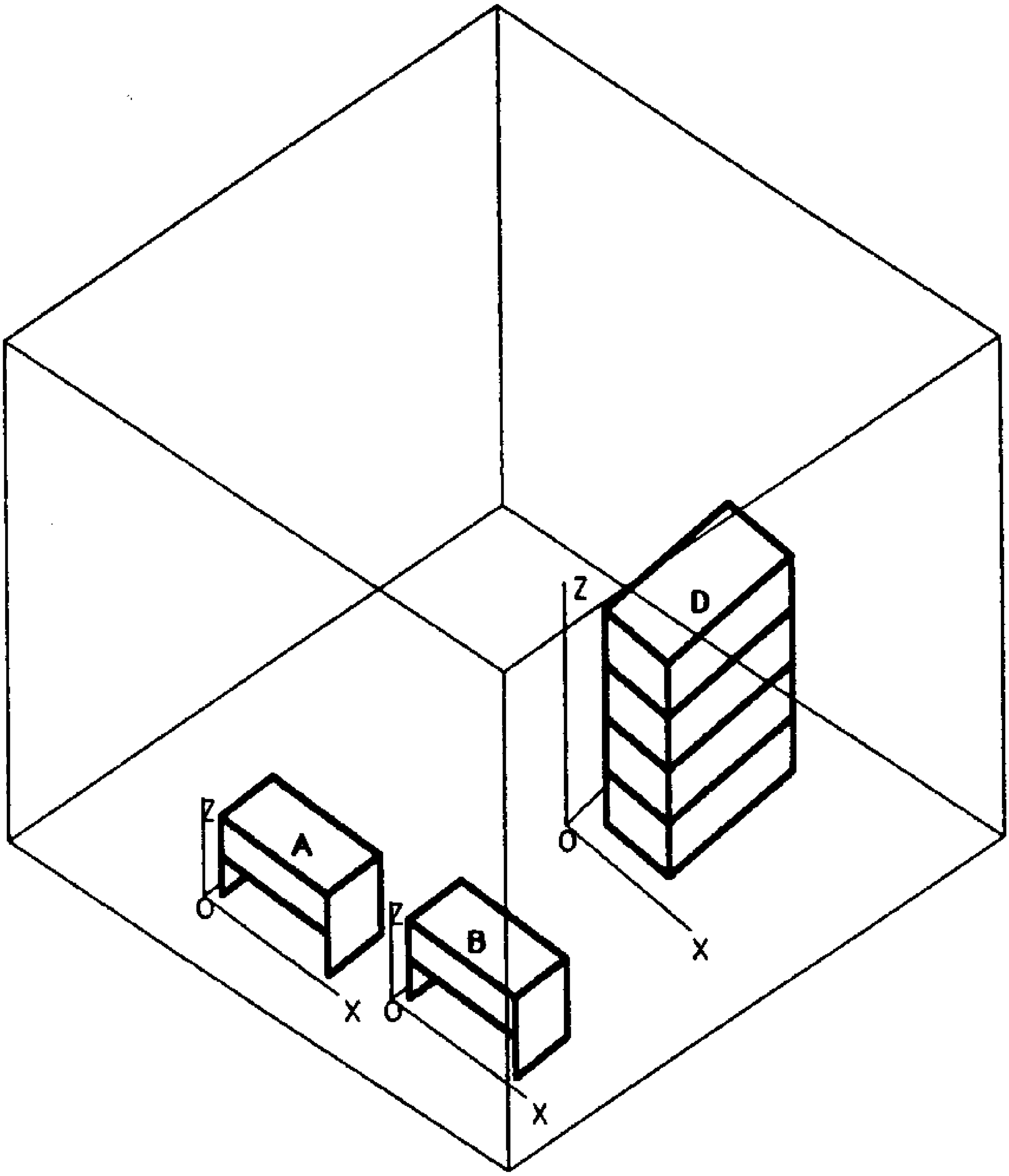


Fig. 2 The library part after the definition of the axes. The PARTCODE and NOTE properties have been inserted into the x axis as LIB.TABLE and TABLE A respectively.



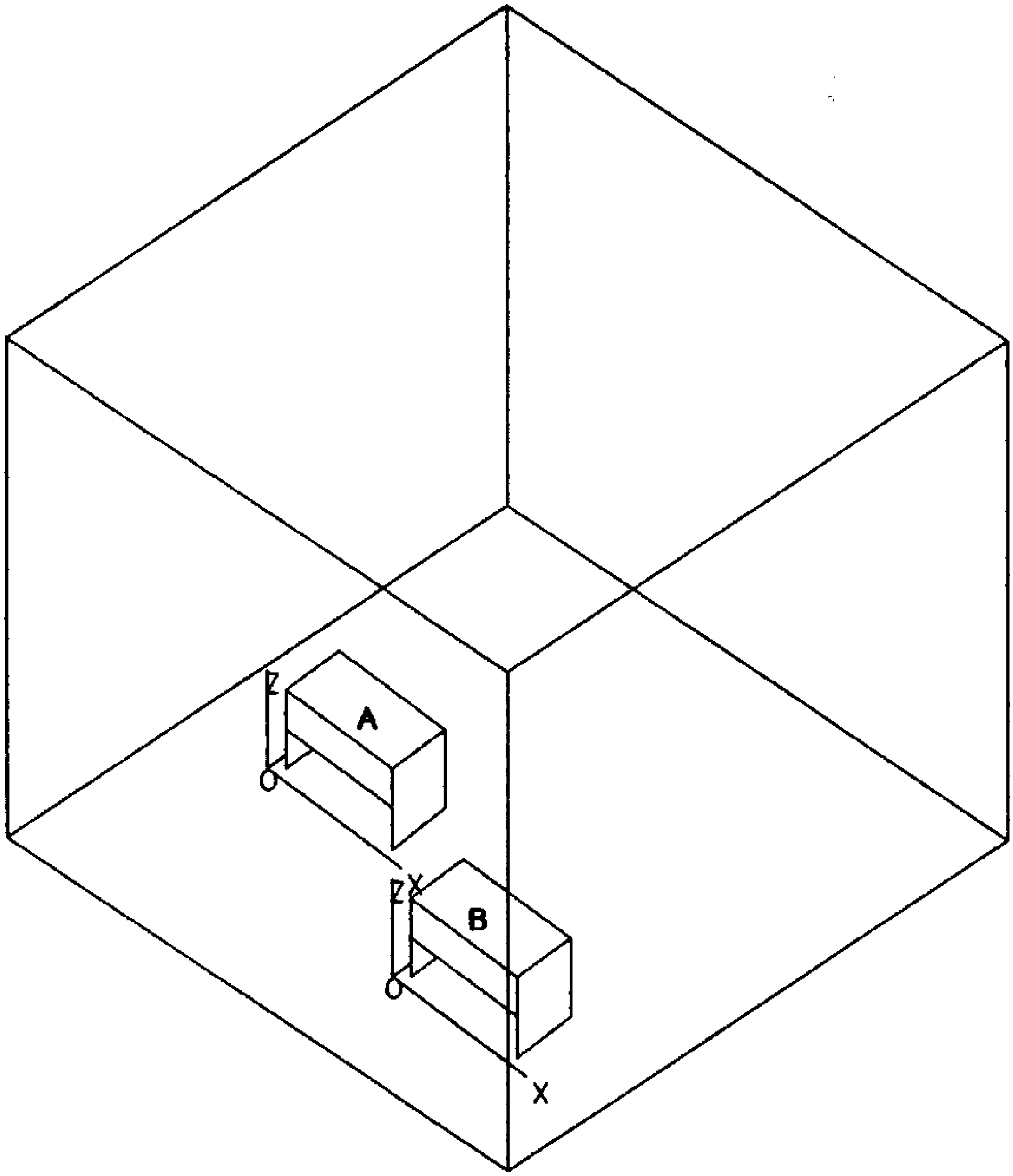
PART	PARTCODE	NOTE
A	LIB.TABLE	TABLE B
B	LIB.TABLE	TABLE A
C	LIB.CHAIR	CHAIR A
D	LIB.FILCAB	CABINET A

Fig. 3 Compartment before the execution of the CV to IDB program



PART	PARTCODE	NOTE
A	LIB.TABLE	TABLE B
B	LIB.TABLE	TABLE A
D	LIB.FILCAB	CABINET A

Fig. 4 Compartment after the execution of the CV to IDB program to show the requested delete of the chair.



PART	PARTCODE	NOTE
A	LIB.TABLE	TABLE B
B	LIB.TABLE	TABLE A

Fig. 5 Compartment after an update by the IDB to CV program using the sample input file of Sec. 6.3. Note the changed position of TABLE B.