RSD-TR-21-86

The Design of an End of Arm Tool Management System for Flexible Assembly Systems (FAS) Utilizing Industrial Robots

by

Paul G. Ranky

Visiting Associate Professor at the University of Michigan

Department of Mechanical Engineering
and Applied Mechanics

Center for Research on Integrated Manufacturing
The University of Michigan
Ann Arbor, MI 48109

September 1986

CENTER FOR RESEARCH ON INTEGRATED MANUFACTURING

Robot Systems Division

COLLEGE OF ENGINEERING

THE UNIVERSITY OF MICHIGAN

ANN ARBOR, MICHIGAN 48109-1109

TABLE OF CONTENTS

ABSTRACT

Flexible Assembly Systems should be able to accommodate a variety of different parts in random order, should provide robot tool changing capability at cell level and robot hand transportation, hand management, robot tool data collection and maintenance at system level. Automated tool changing at the robot cell level is important in order to provide the needed flexibility and the short changeover time for the system as a whole. Because of this, the number of robot tools, as well as their complexity, their sensing capability, etc. is growing in robotized assembly systems.

In order to keep track of the tools used by different robots as well as in order to provide real-time data regarding their location, status, wear, sensing and other capabilities, etc. complex assembly systems need a dynamic operation control system, that besides scheduling, balancing, capacity planning, etc. programs incorporates an "End-of-Arm-Tool-Management System" too.

The paper provides an overview of the most important design considerations of robot tool management systems, as well as describes a robot database and a robot tool database data structure currently implemented by the author and his students at the University of Michigan. (The outlined methodology is "generic" in a sense that with minor modifications, the presented concepts and data structure can be utilized for robotized systems dealing with processes other than assembly).

KEYWORDS End of Arm Tool Management, Robotics, Database Mangement Systems, Flexible Assembly Systems, FAS, FMS, CIM.

1. INTRODUCTION

There is an increasing number of robots capable of changing grippers, or in general hands. There are robot cells currently designed or being used that access a dozen or more tools in a robot hand magazine, thus "robot hand management" in flexible assembly and other robotized systems is becoming an important part of the assembly system's operation control software. [Figure 1].

The design of a robot tool management system incorporates a wast amount of analysis and system development work. It must be done by a team of engineers and data processing staff, headed by a experienced team leader who understands not only the data management problems, but also the production engineering aspects of robotized manufacturing systems, and the application possibilities of the large variety of robot tools and tool changing systems available today. (Figures 2 to 14 illustrate different robot tool changers and some of their application possibilities).

When designing the robot tool management system, one should consider the following steps:

1. Collect all current and possible future user and system requirements
2. Analyze the system (i.e. the data processing system and the FAS hardware and software constraints)
3. Design an appropriate data structure and data base for describing robot hands and tools and then

4. Specify, design and purchase (if possible) other subsystems accessing this data base as well as communicating with the real-time production planning and control system

Let us discuss the above list in more detail.

2. USER AND SYSTEM REQUIREMENTS

The most important question to be answered before starting to design a robot tool management system and a robot tool data base, to be accessed by the robot hand management system, is that *"Who?"* is going to use the data, *"when?"* and *"For what?"* purposes in the particular assembly system?

Robot tooling data in FAS (Flexible Assembly System), is typically going to be used by several subsystems, as well as human beings. These users can be summarized as follows:

- The production planning subsystem
- Process control
- Robot programming
- Robot hand maintenance
- Robot tool assembly (manual or robotized)
- Stock control and material storage
- Manufacturing cell/system design and simulation

Keeping in mind that a truly flexible system can accept parts in random order, for example the *production planning system* has to be informed in real-time about the availability of robot tools on stock, as well as in the tool magazines of the robots otherwise it will not be able to generate a proper scheduling program for the assembly line.

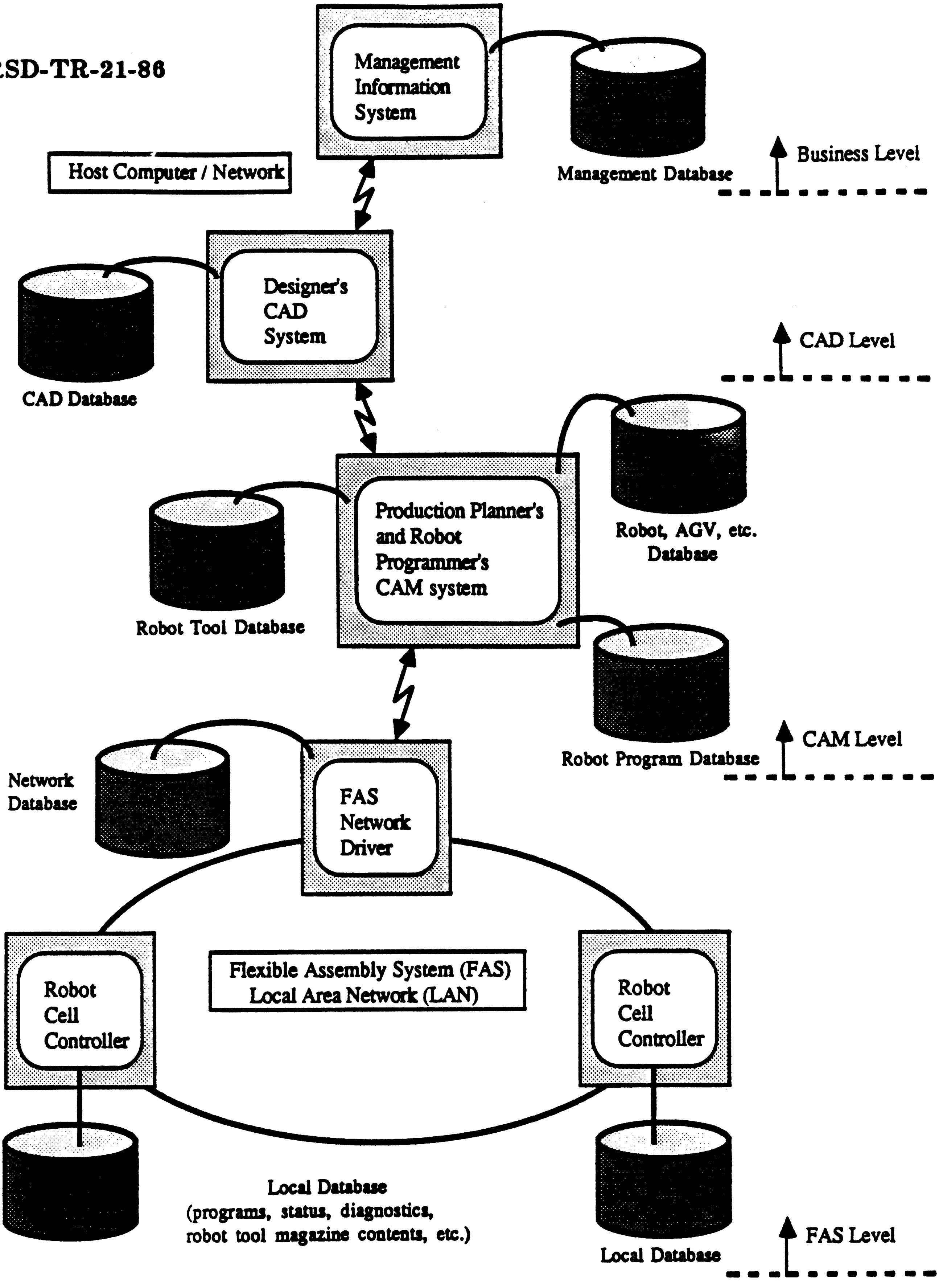


Figure 1 The overall architecture of the Robot Tool Management System for FAS applications.

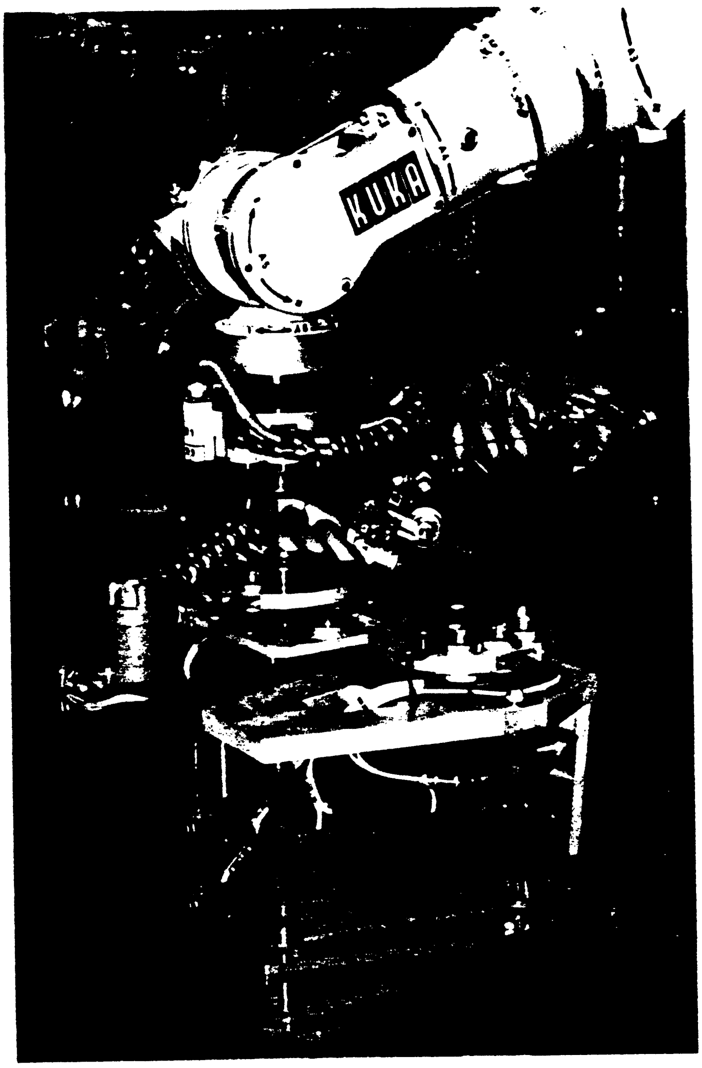


Figure 2 **Automated Robot Hand Changing System of Kuka, West Germany.**

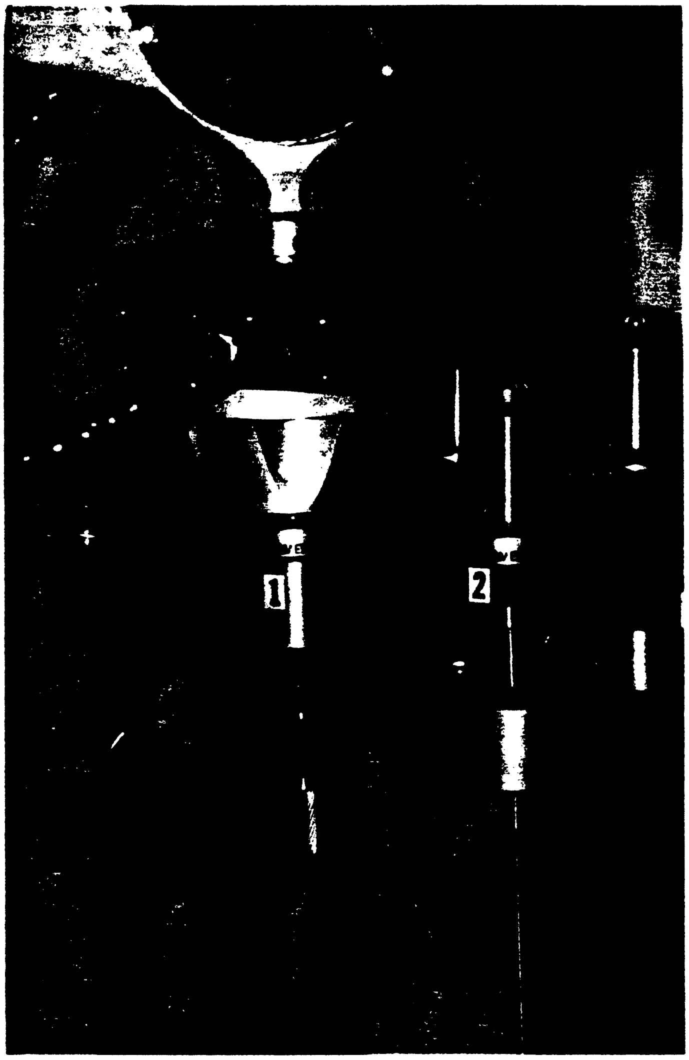


Figure 3 Automated Robot Hand Changing (ARHC) System, designed by Kennametal, USA.

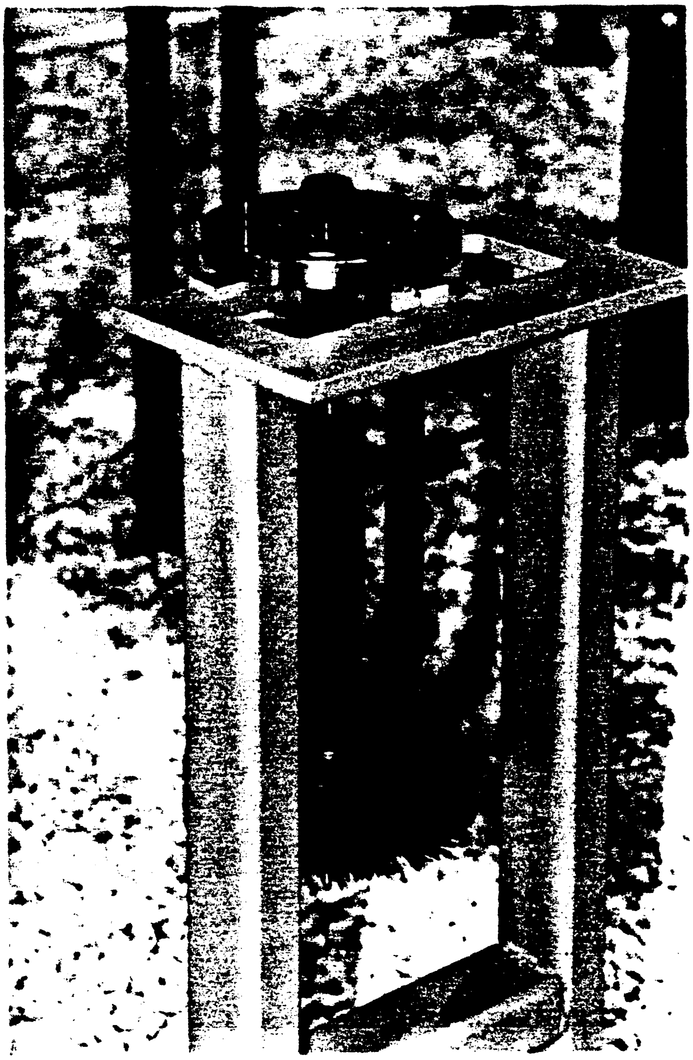


Figure 4 Automatically interchangable deburring tool by Cincinnati Milacron, USA

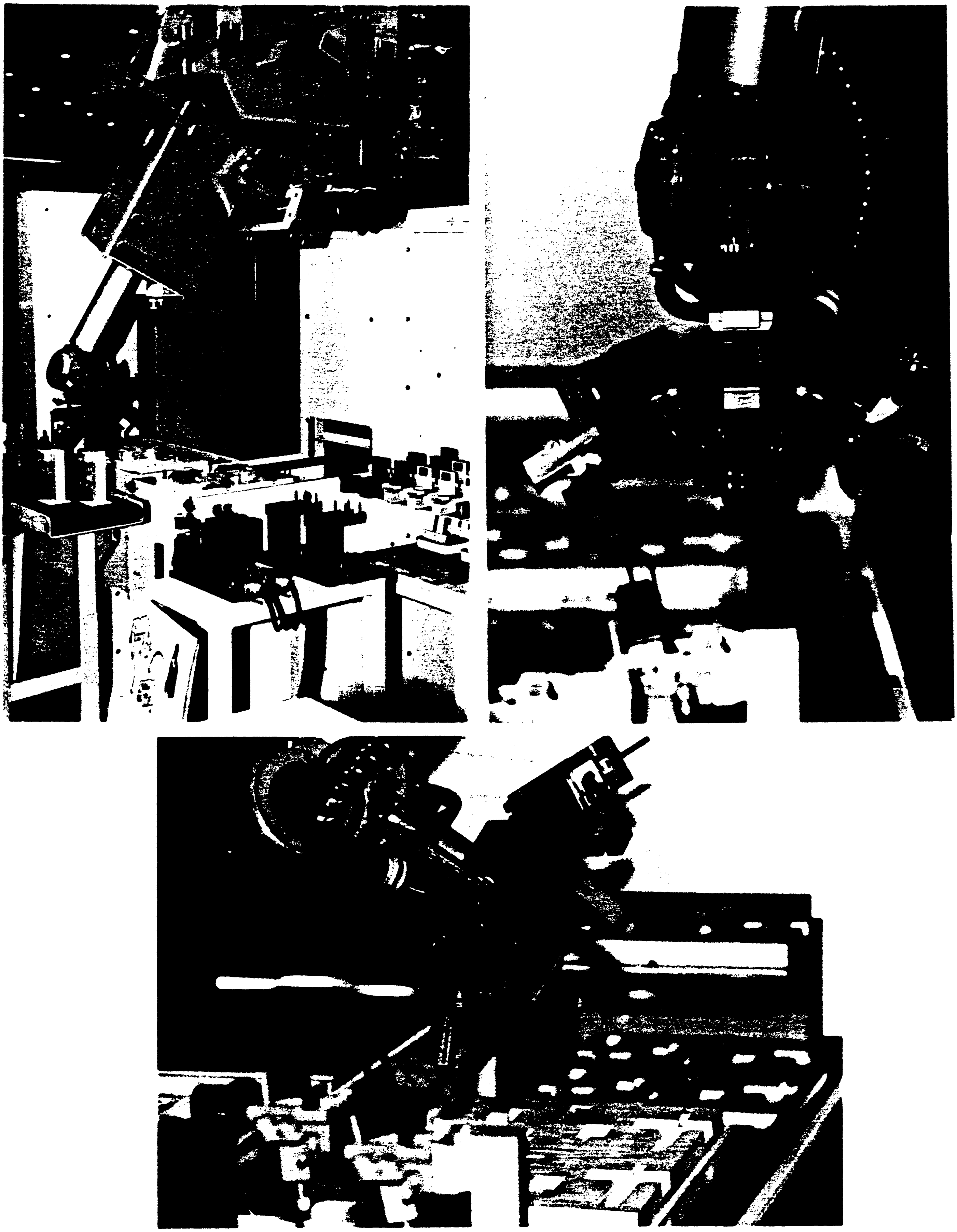


Figure 5 ASEA's extremely fast, rotary indexing tool changer driven by a DC motor at the last axis of the robot

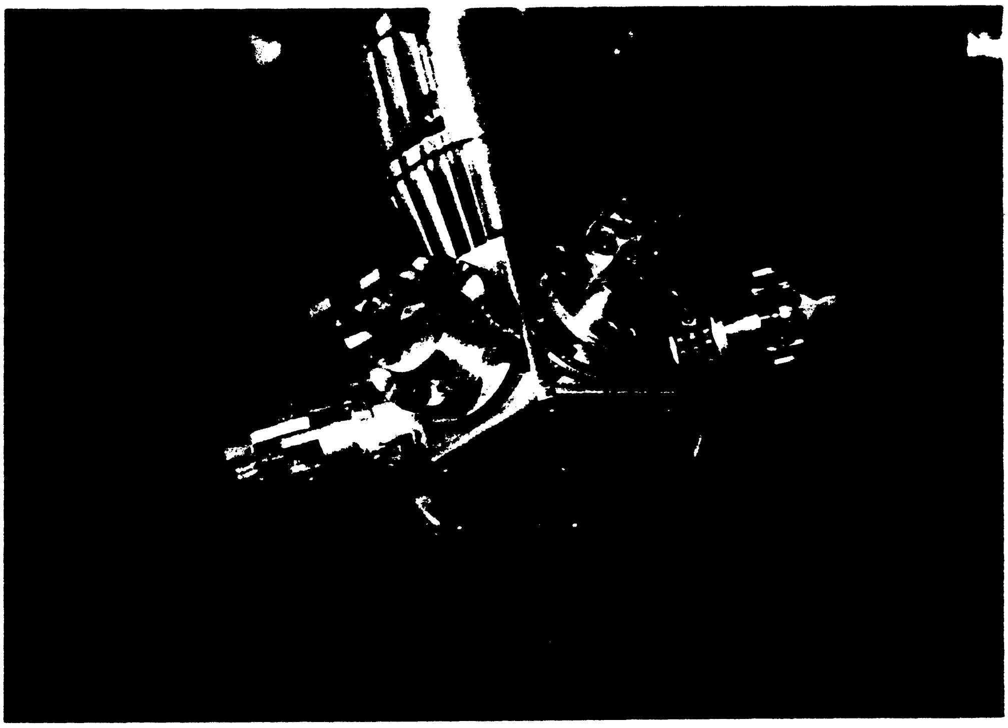


Figure 6 Flexible tool changer from BASE Robotics, for pneumatically operated, small size grippers.

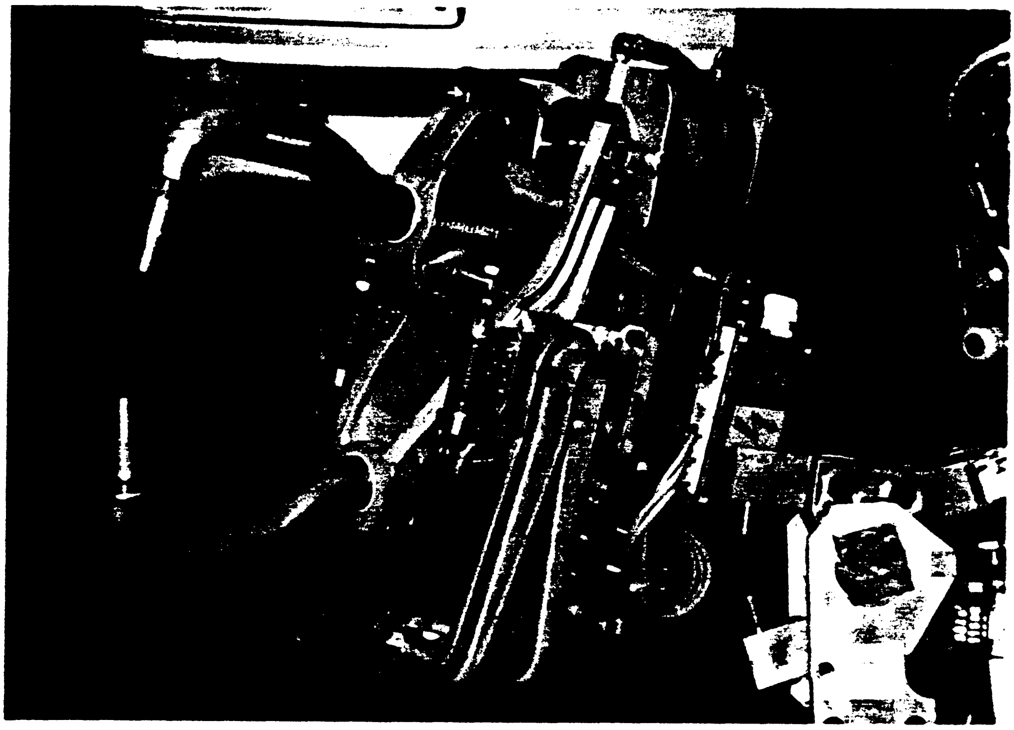
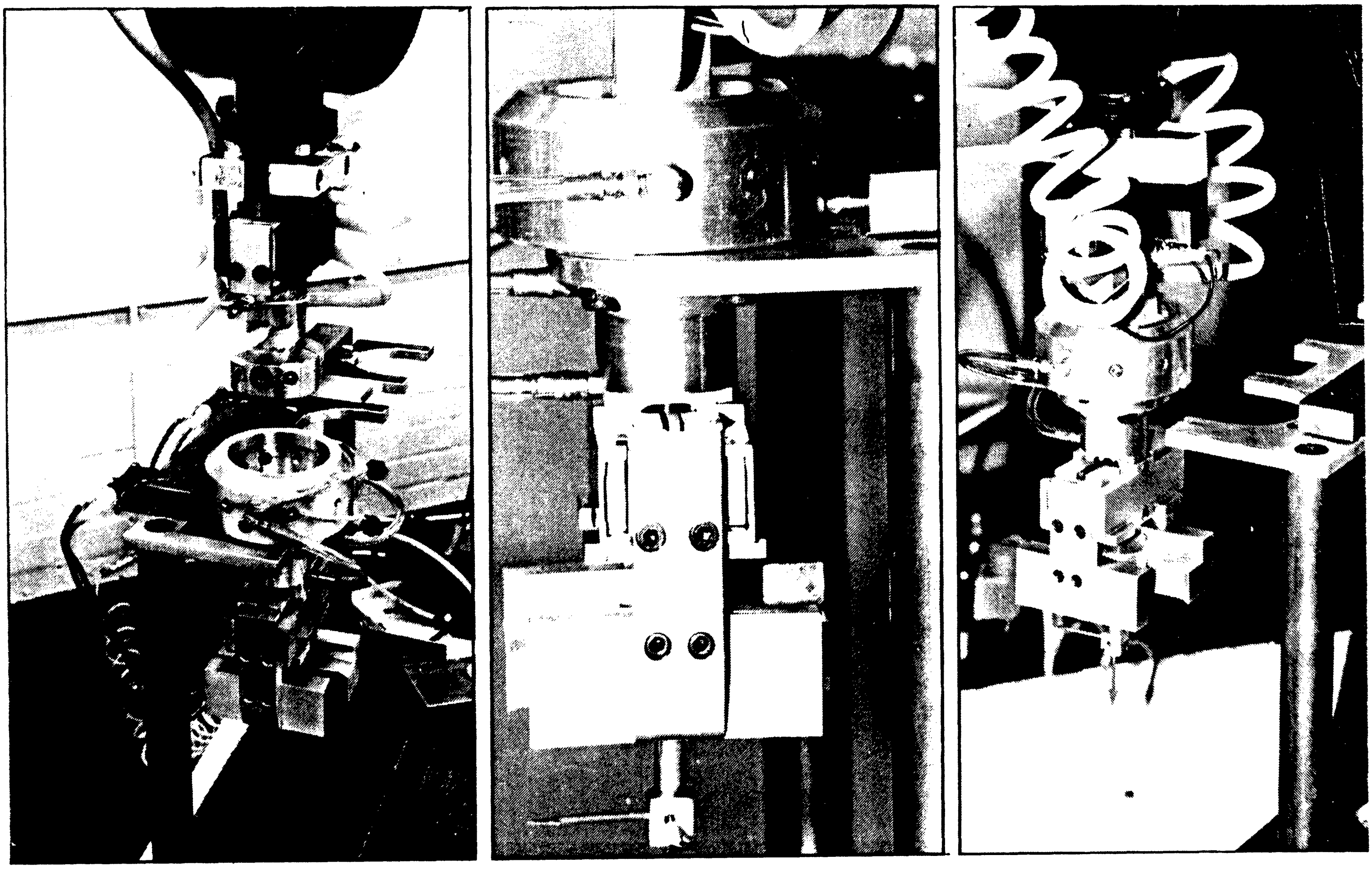


Figure 7 Automatically interchangeable welding gun makes automobile body assembly lines more flexible.



a) b) c)

Figure 8 Photos a) b) and c) demonstrate the "approach", "locate," and "pickup" operations, using the Automated Robot hand Changing System (ARHC) designed by the author in 1982. (The photograph marked a) illustrates the pueumatic and electronic connections of the Mark I design). (Photographs adapted from Ref. 3).

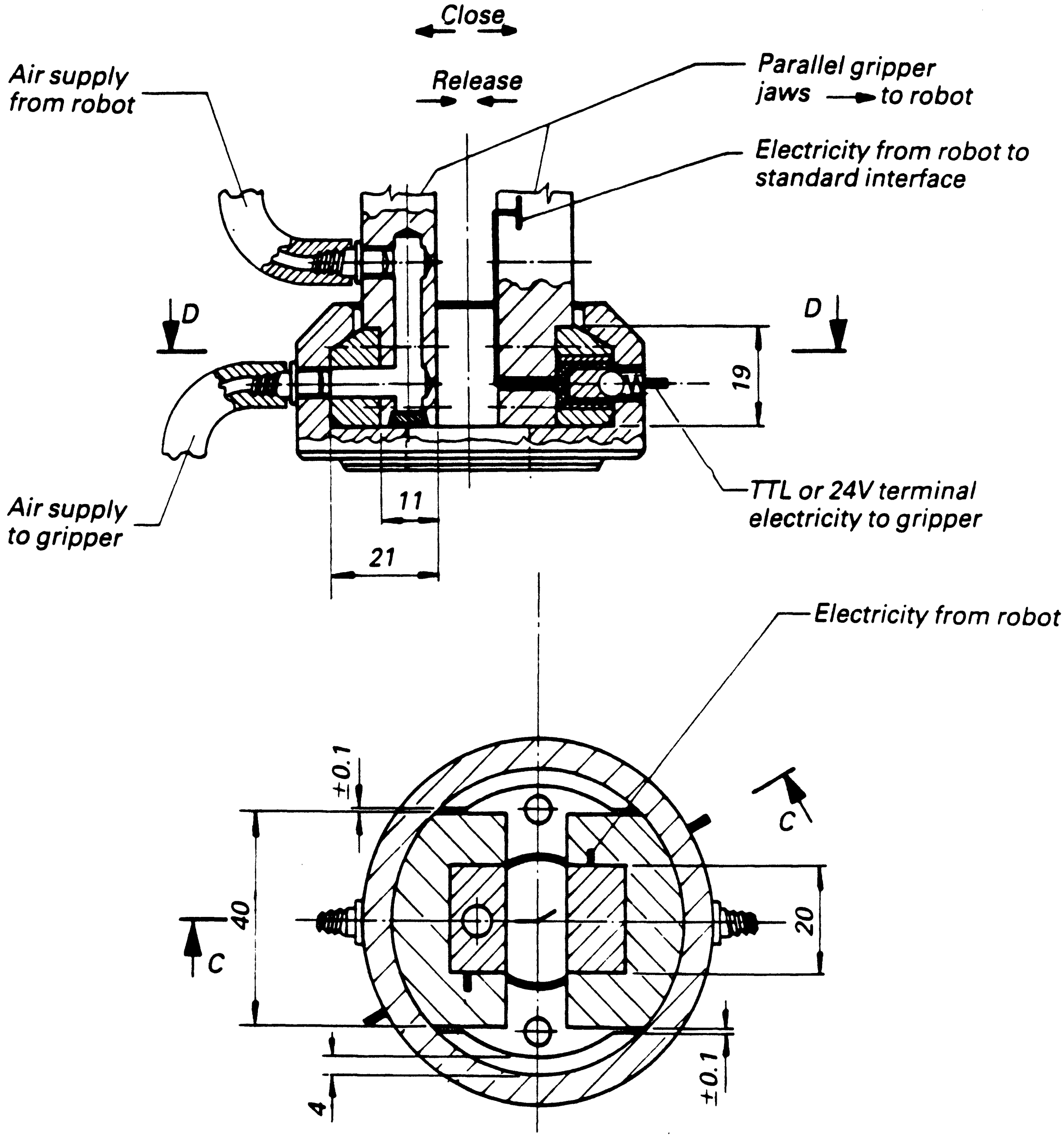


Figure 9 **The mechanical design and the recoupling action of the robot hand changing system, designed by the author, (Mark I version, adapted from Ref. 3) (Note the way the system can automatically recouple pueumatic and electronic power supplies).**

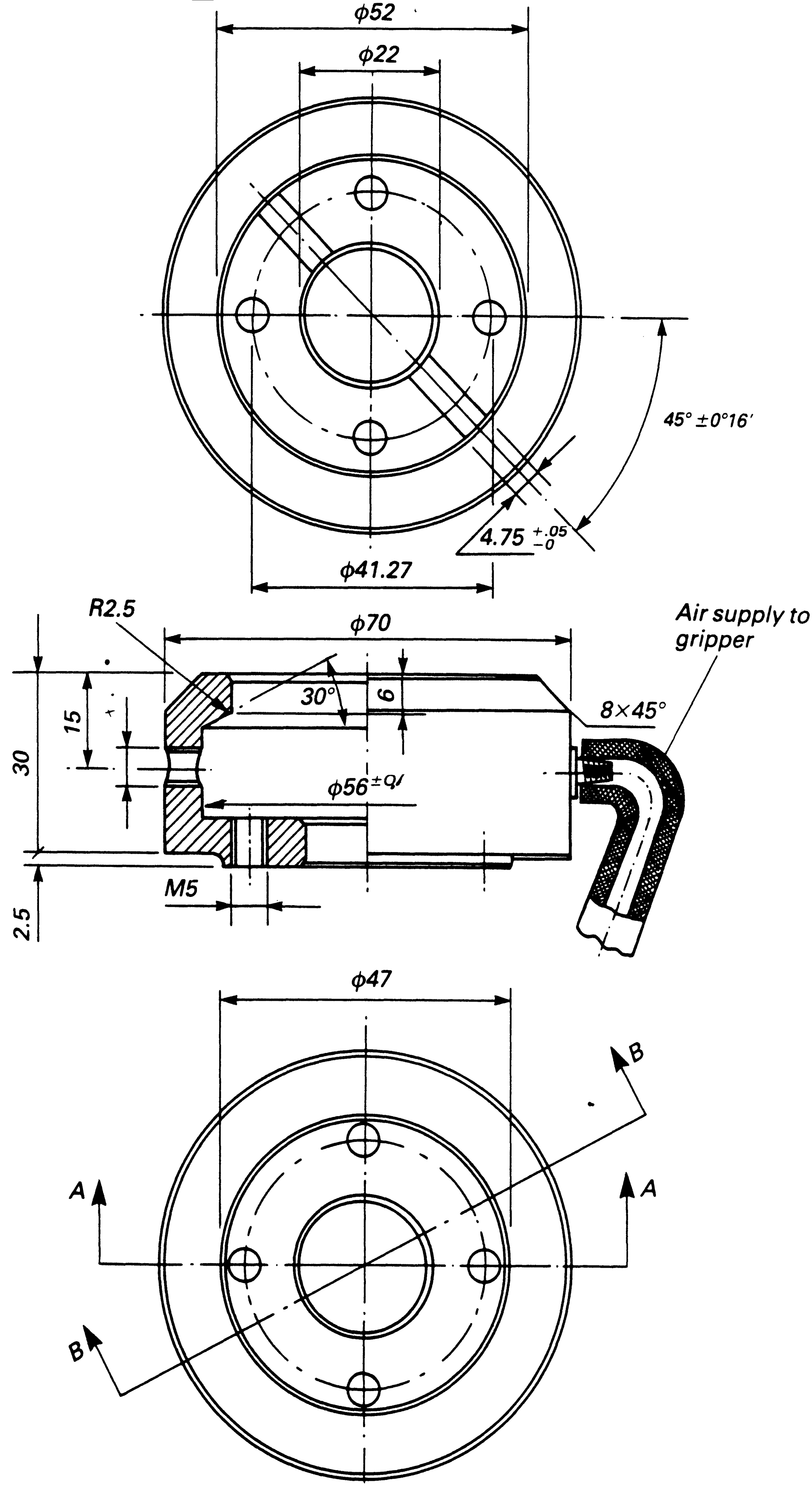


Figure 10 **The standard interface of the ARHC system, designed by the author (Adapted from Ref. 3) (Note, dimensions are in metric !).**

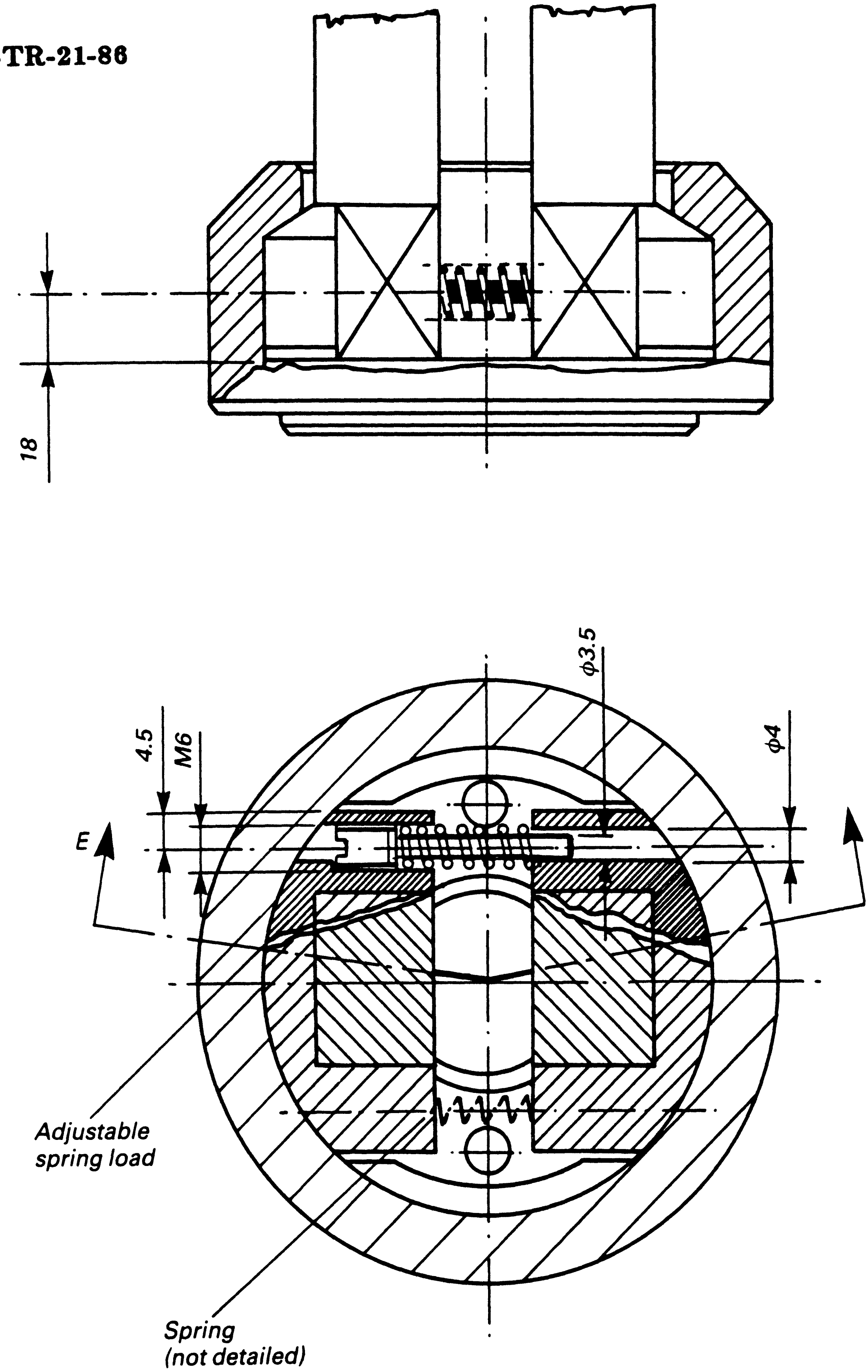


Figure 11 **The collecting fingers (the male part) of the Ranky-type ARHC (Adopted from Ref. 3). (Note that the dimensions are in metric !).**

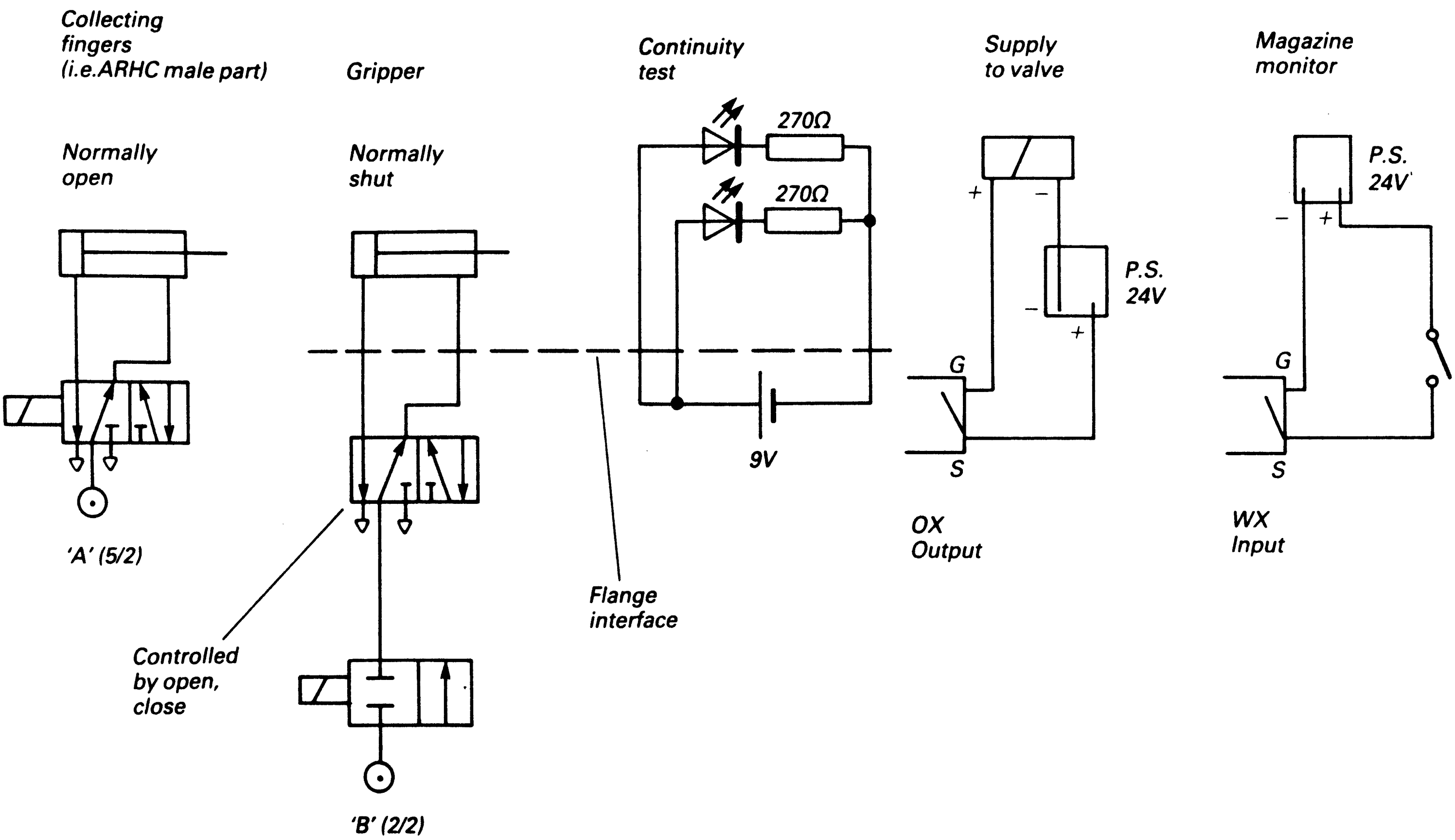


Figure 12 **The control circuits for the "Ranky-type" robot hand changer (Adopted from Ref. 3).**

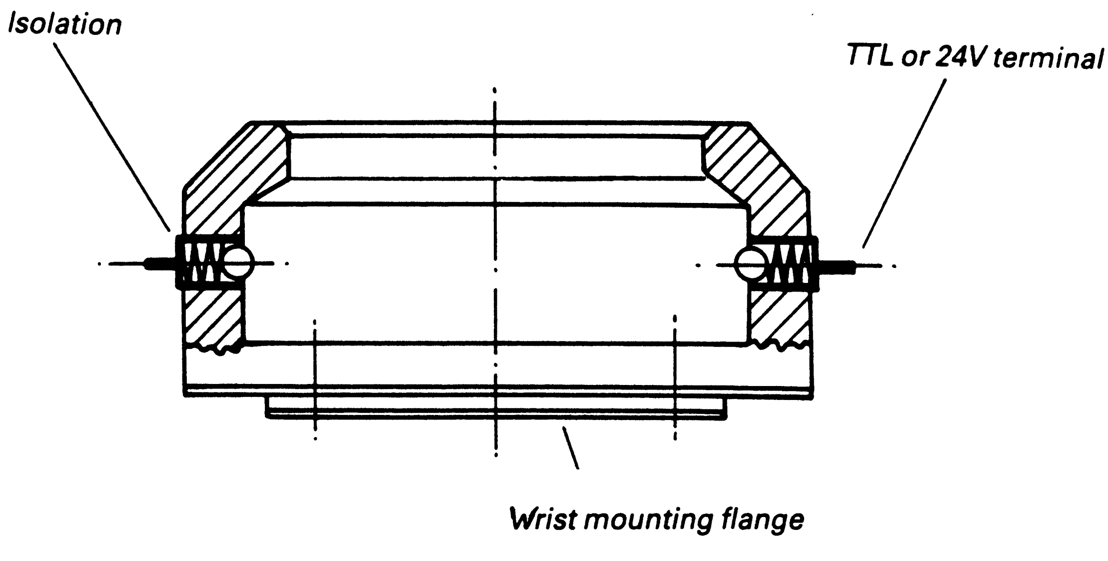


Figure 13 **The principle of the electronic connections (currently 8 in the Mark II version) in the Ranky-type robot hand changer.**

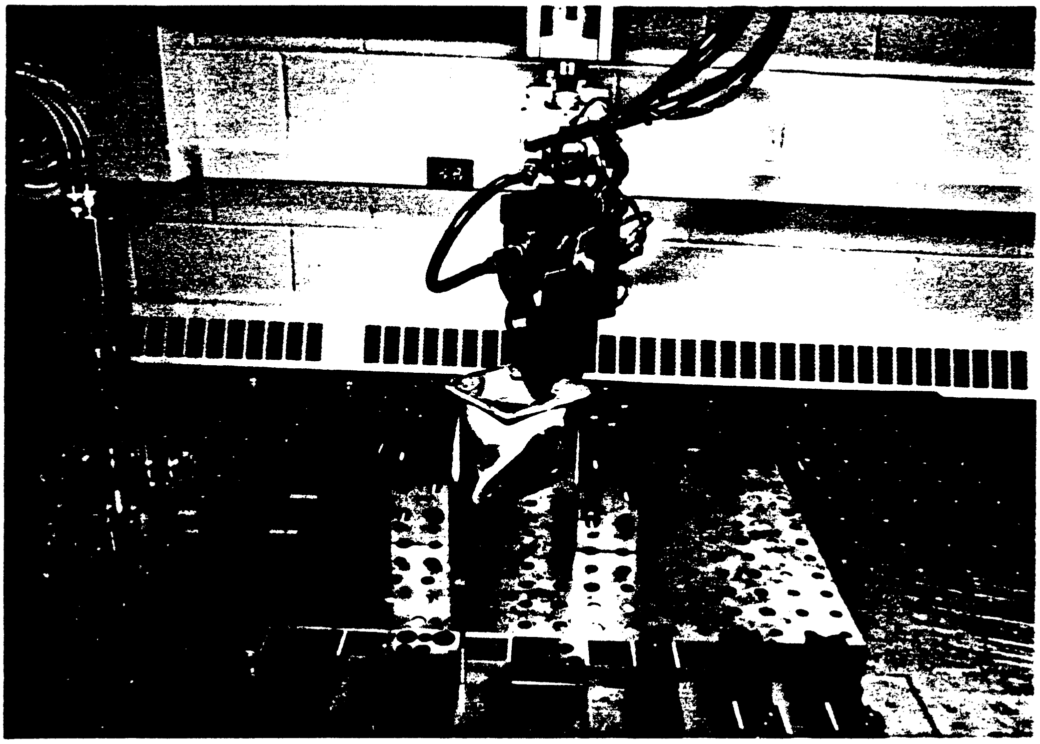


Figure 14a Note on the left hand side of the photograph the self powered, automatically changeable pneumatically and electronically operated nutrunning tool, designed for the IBM 7565 robot by the author and his FMS course students at the Mechanical Engineering Department, University of Michigan

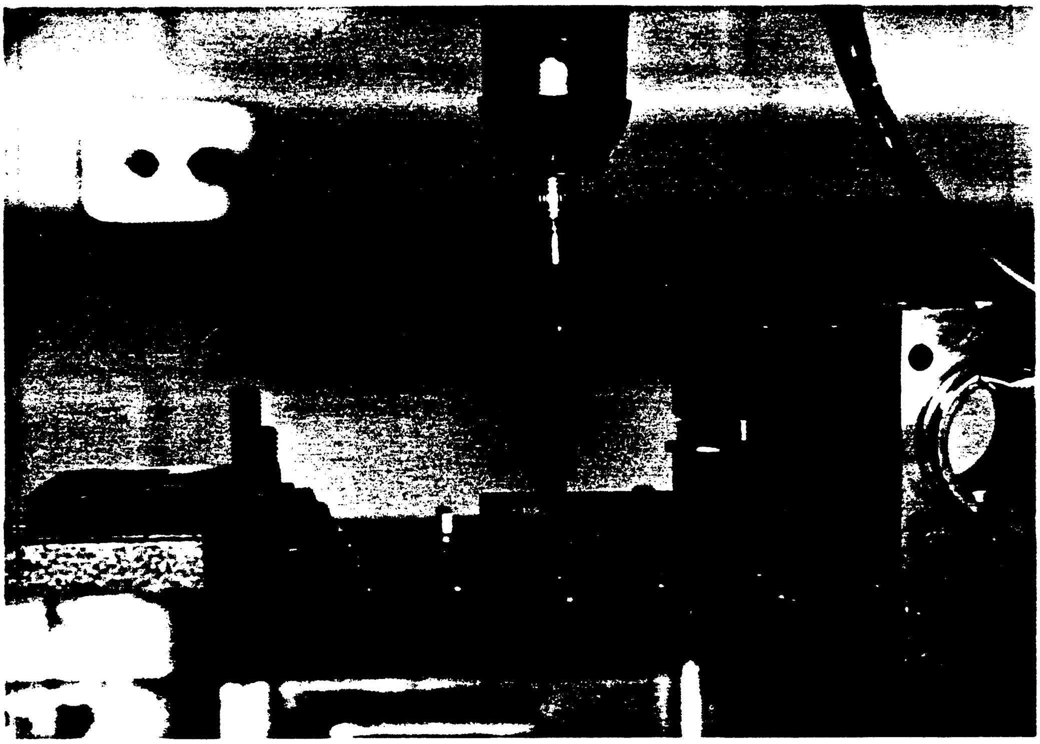


Figure 14b The nutrunning tool in operation

Both the *process control* and the *production* planning systems have to update any changes in real-time or the operation of the system can be disrupted.

The *robot tool preset station,* if required, must be able to inform the process control system about robot tool data, preferably via a digital tool preset unit linked directly to the data processing network of the FAS.

When writing *robot programs,* one has to know the actual sizes of tools, their sensors and their behavior in different operating conditions. The robot tool geometry is also used when checking for collision by graphics simulation. (Figure 15 to 25 illustrate different solid model designs and graphics simulation using the ROBCAD System).

3. SYSTEM CONSTRAINTS

If data types are kept separately and accessed by independent programs in independent files, one program will "not a know" in time when another program updates a file and eventually *panic situation will occur* in the real-time system.

Data Base Management Systems are considered to be the essential core of the FAS robot tool management system since they:

- Provide logical as well as physical data independence. *(Logical data independence* meaning that that new fields of records may be added to the system without rewriting the application program. *Physical data independence* means that changes can be made to some element of data on disk, or on any type of data storage media, leaving the application programs untouched.)

- Ensure a standard software interface for its users and fast information retrieval
- Ensure that data is compatible for all subsystems, reducing data access time and application program development costs
- Minimize data redundancy

To summarize, Data Base Management Systems enable the data as well as its data description to be interfaced with several different application programs written in different languages, running in different processors and operating systems.

To provide the required flexibility and high level of local intelligence for the tool management system, as well as for the other subsystems of the FAS, it is necessary to apply distributed processing theory both for communications, as well as for Data Base Management purposes.

The most important aspects of distributed processing and data management from this point of view are:

- The possibility of real-time communication and data update in the robot tool store room, at the tool assembly station, at the robots using the tools and in general between all subsystems accessing this facility within the FAS data processing network.
- Flexible and user friendly operator interface at all terminals where the robot tool management system's users must access the distributed system
- "Well designed hardware and software architecture", preferably based on intelligent nodes linked together by Local Area Networks (LANs)

When following these principles, the man-machine and the machine-machine communication systems will be more flexible and compared to "non-distributed systems" the system to be created will be more reliable.

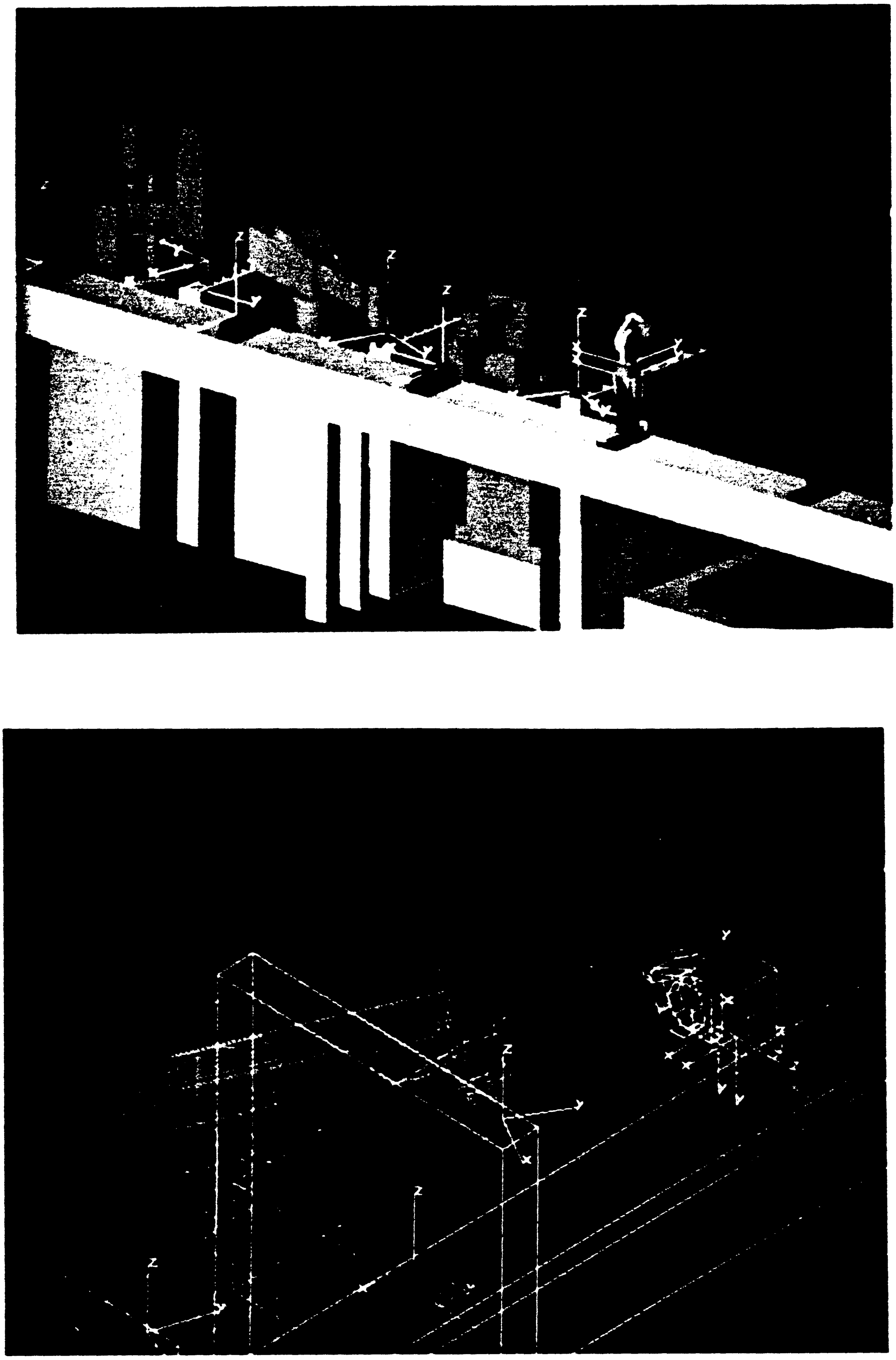


Figure 15 The solid model and a section of the wire frame model of a dedicated assembly line incorporating industrial robots, as well as part feeding and tool changing devices along a conveyor line.

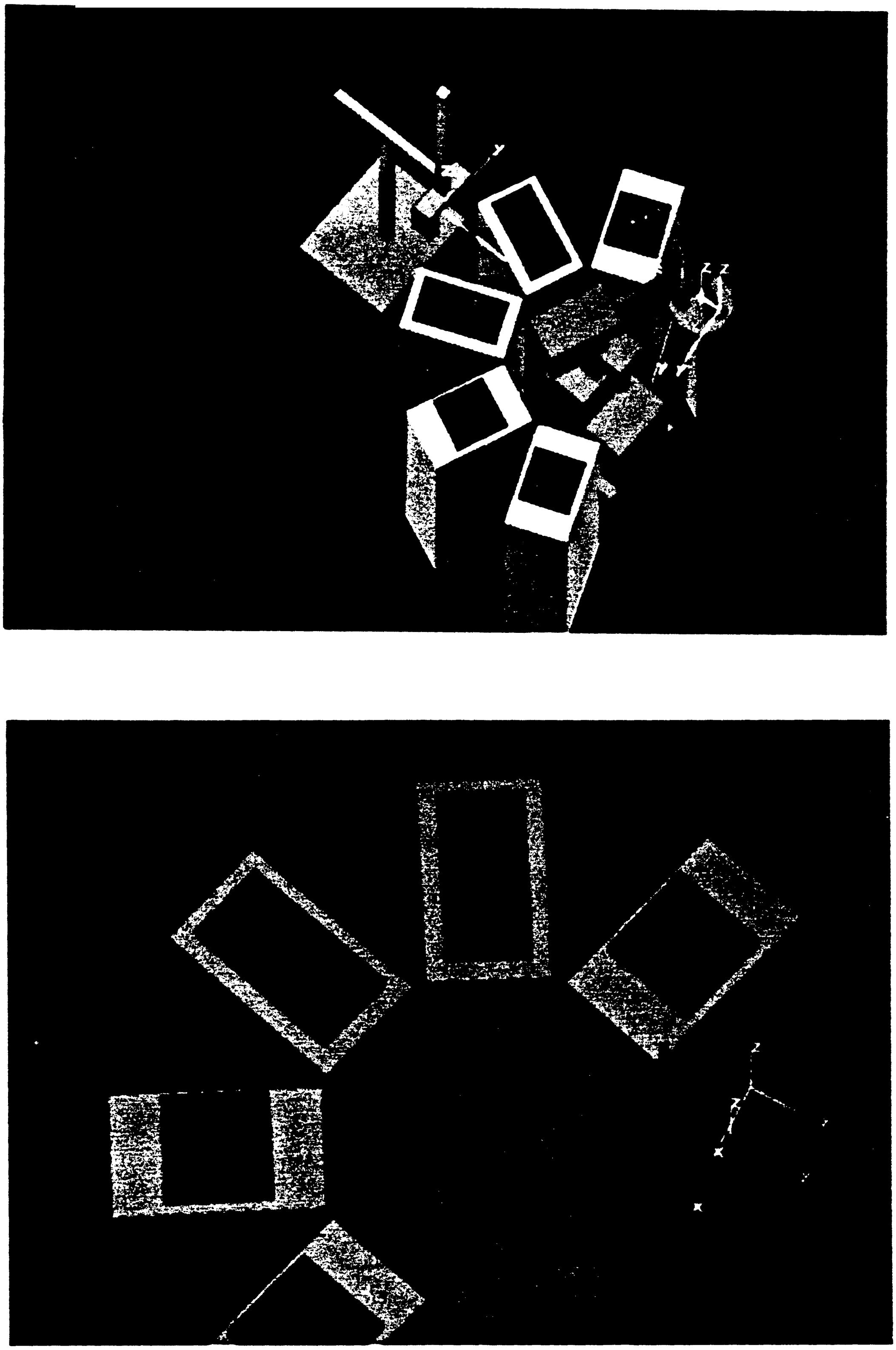


Figure 16 The solid model of a truly flexible assembly cell shows the robot in tool changing position. The tool changer in this example holds three tools in a rotary indexing magazine. (Larger robot tool magazines can also be applied if necessary).

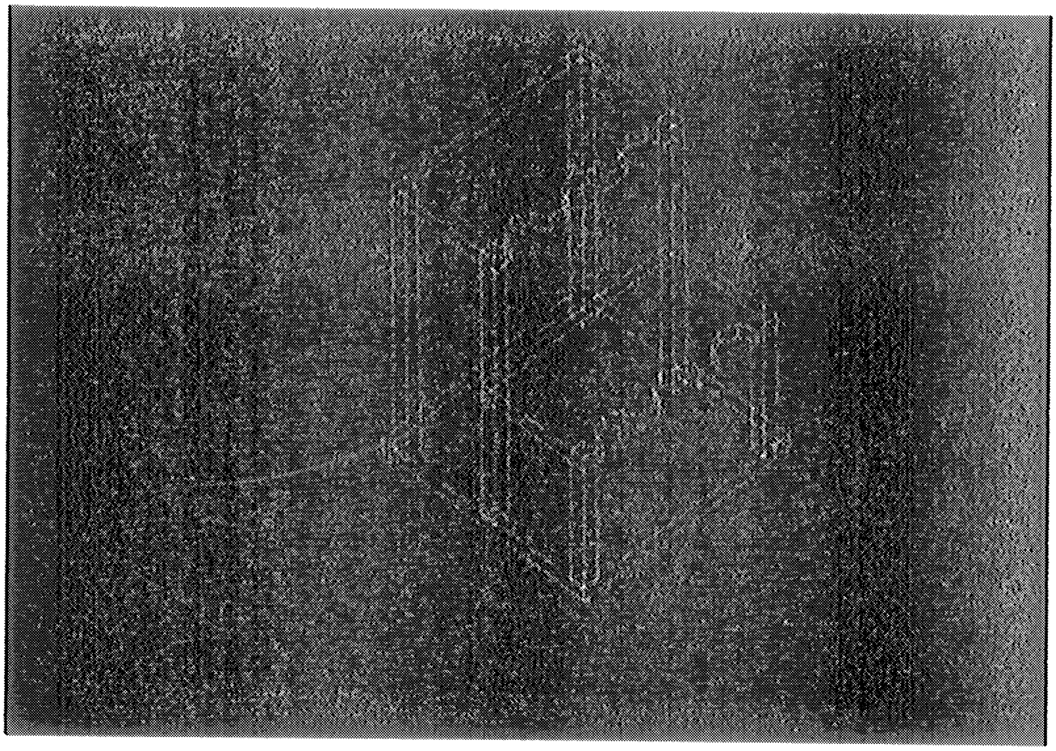


Figure 17 The "Mark II" ("double decker" version of the tool magazine designed by the author for his tool changing system.

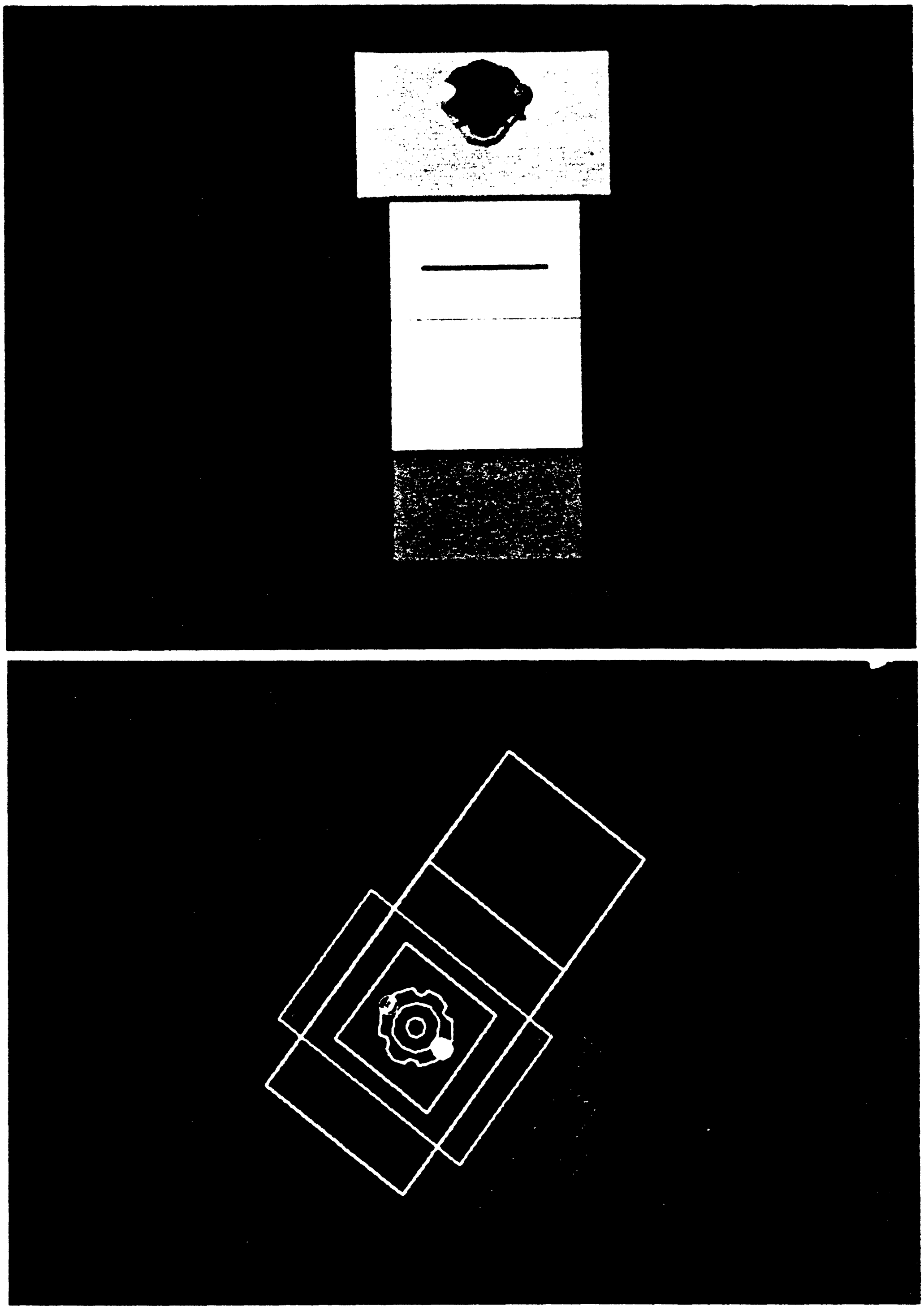


Figure 18 Closeups of and AGV docking station. The Automatically Guided Vehicle is loading a pallet to this station on the top of which there is a rotary indexing robot tool magazine capable of holding up-to six tools. (Note that currently there are only two tools in the magazine) Note that in this design the pallet is mechanically, electronically as well as pneumatically recoupled, as necessary, at these standard AVG docking stations.

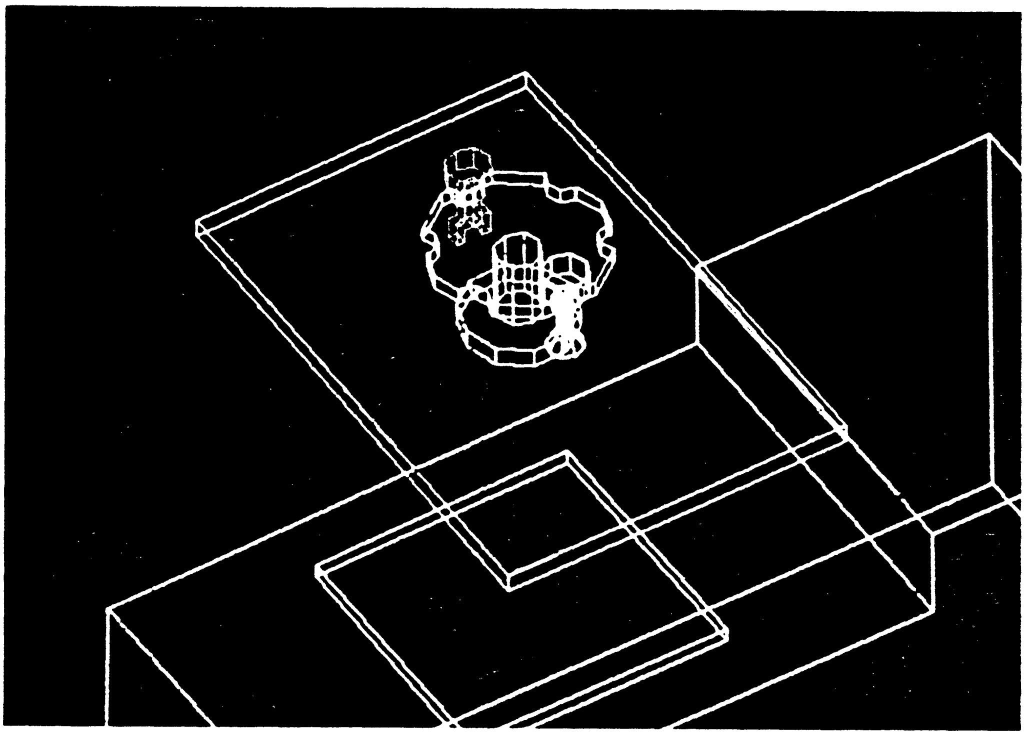
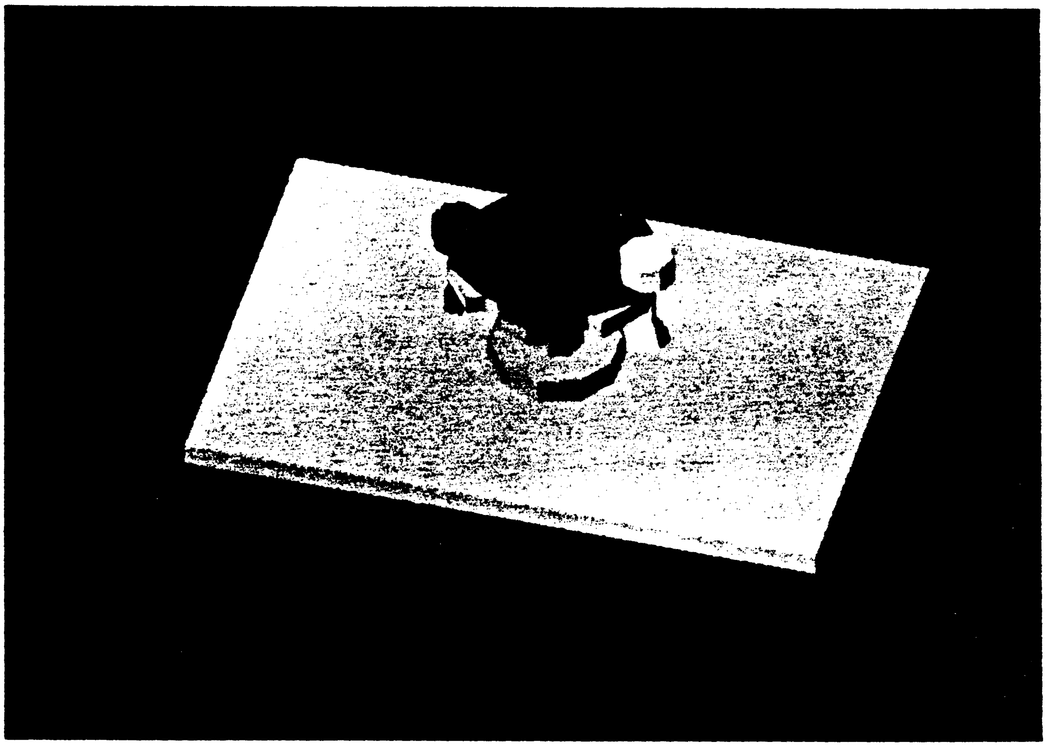


Figure 19 This closeup, shown as a wire frame model, illustrates the way the pallet is located by the AGV at the AGV docking station. (Note the four locating pins, providing accurate pallet positioning as well as power-supply/sensor recoupling facility at the docking station).



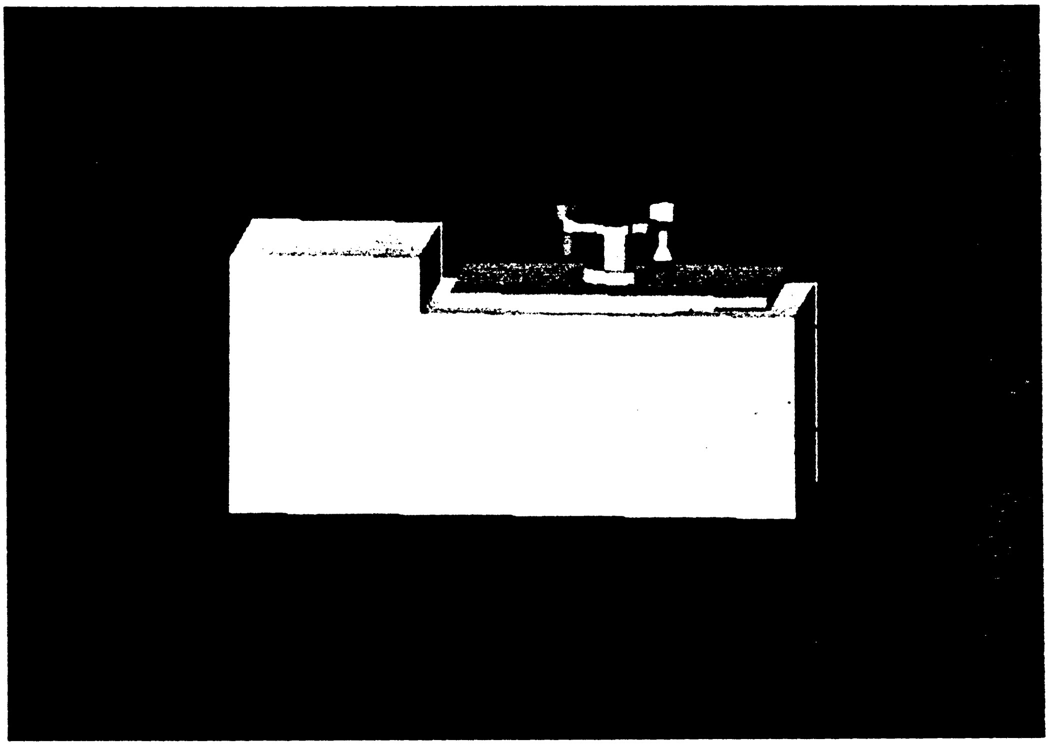


Figure 20 The pallets, the robot tools, the AVG, etc. are separate entities (i.e. programmable building blocks) of this solid model library, designed by the author and implemented in ROBCAD.

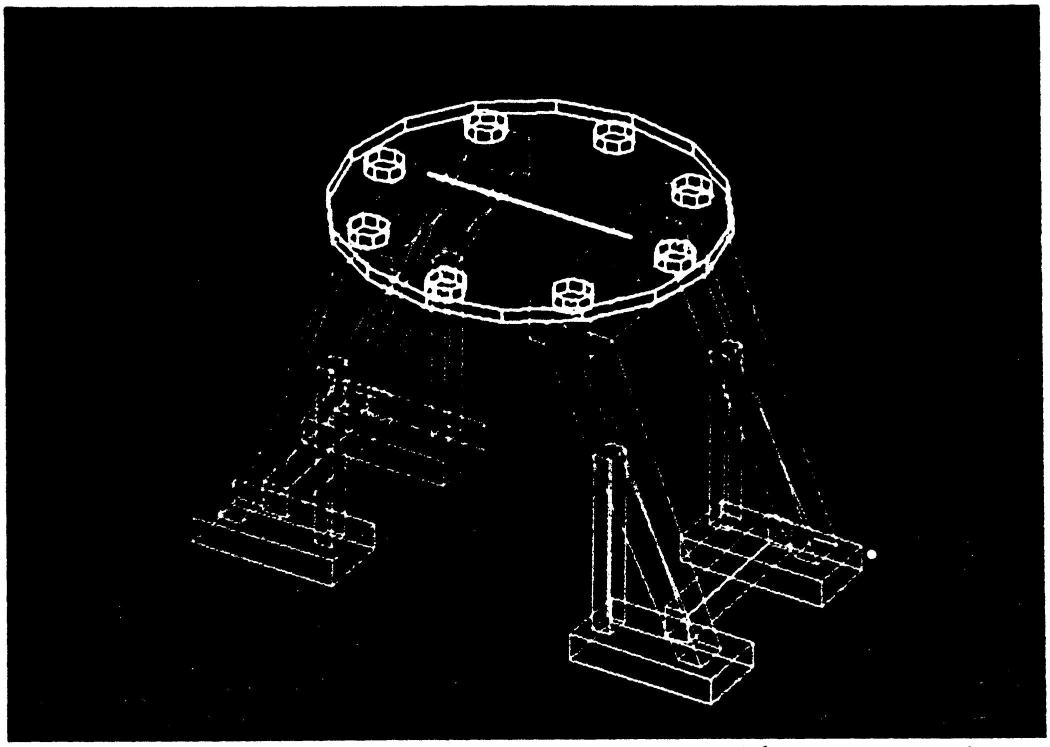


Figure 21 The photograph illustrates the rotary tool/part magazine in an AGV docking station.

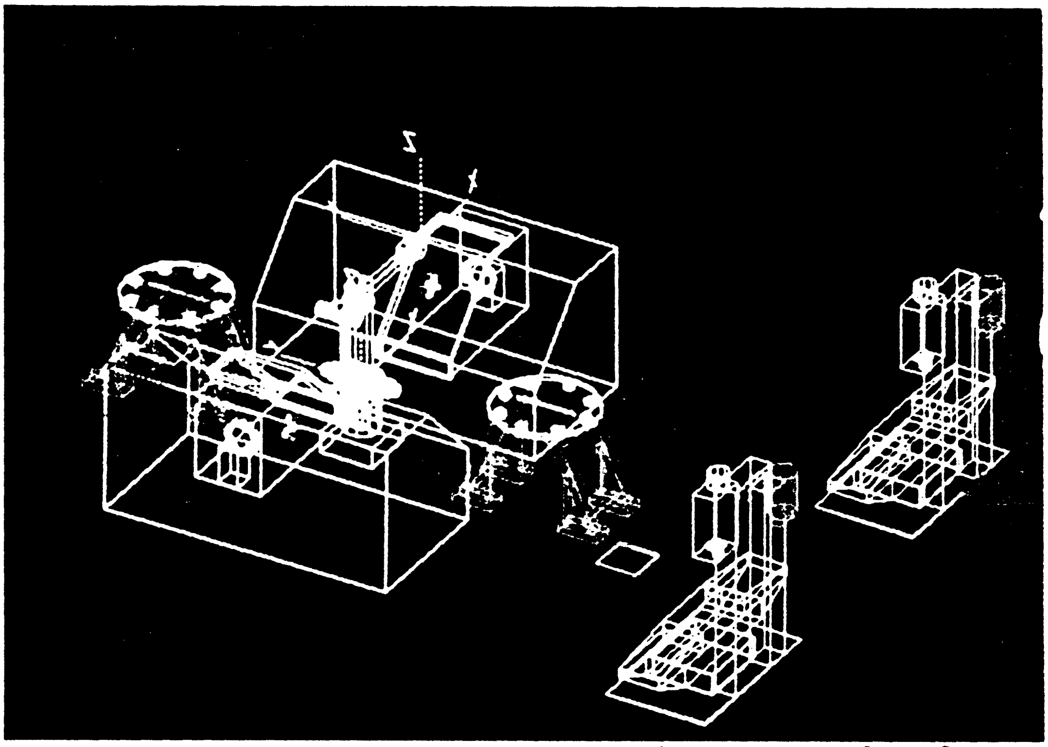
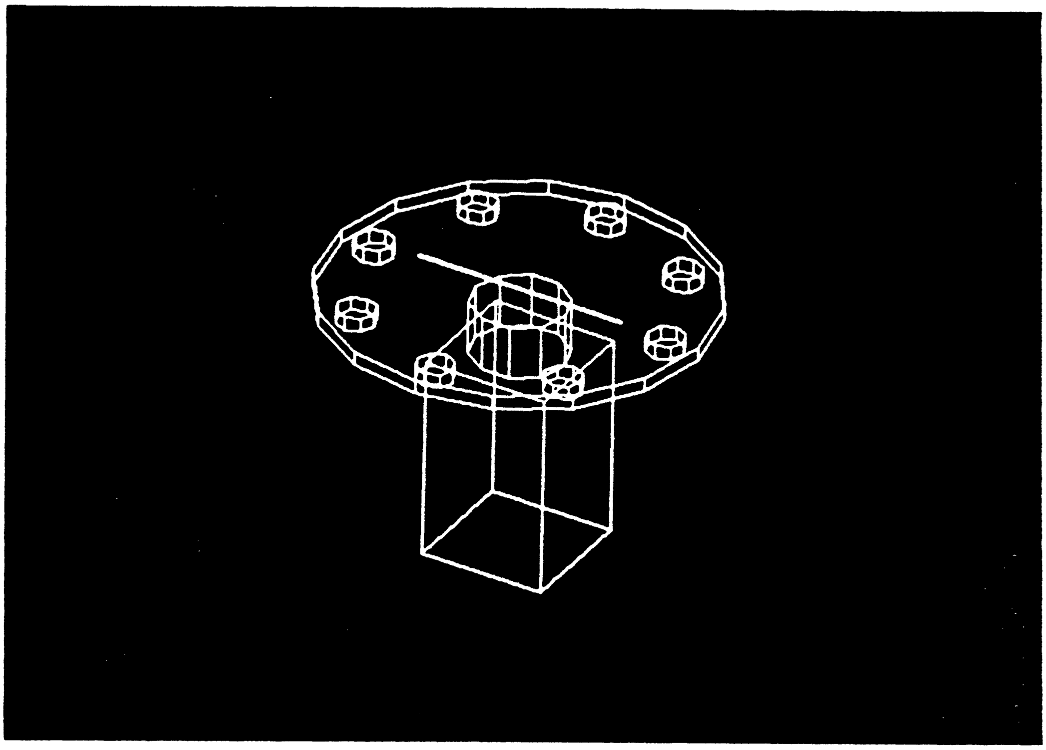


Figure 22 Machining cells (turning and milling) utilizing the above shown rotary indexing part magazine.



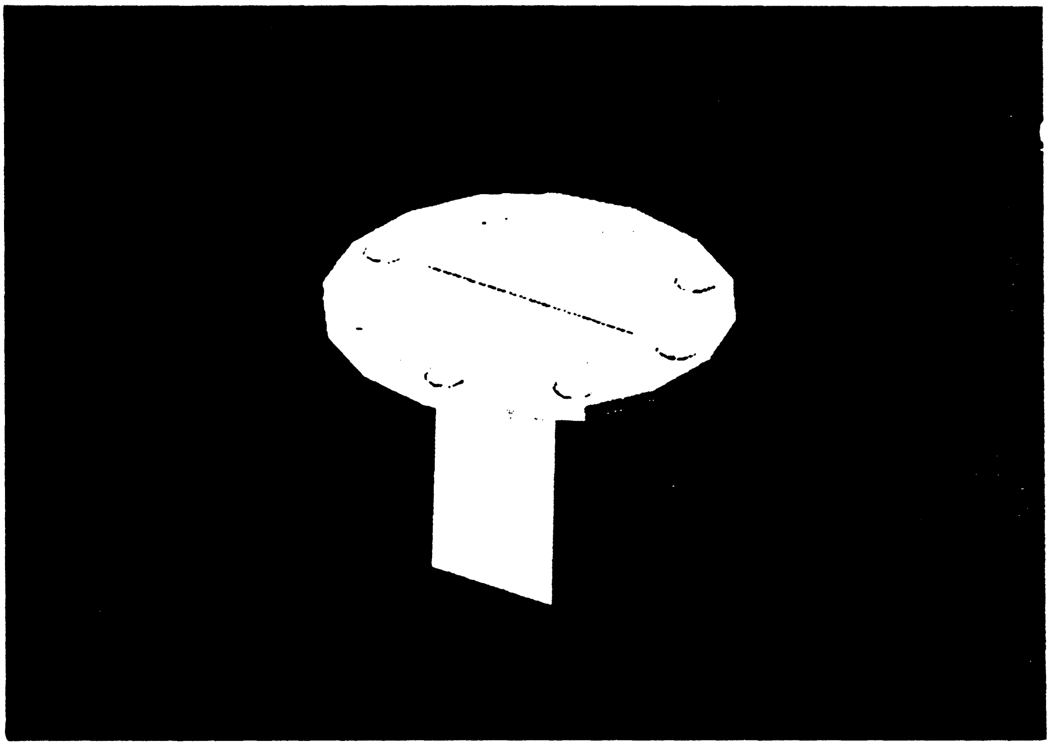


Figure 23 The wire frame model and the solid model of a rotary indexing table, that can be used as a part storage facility for rotary parts, as well as a robot hand magazine. This design allows the device to be loaded and unloaded automatically by an AGV .

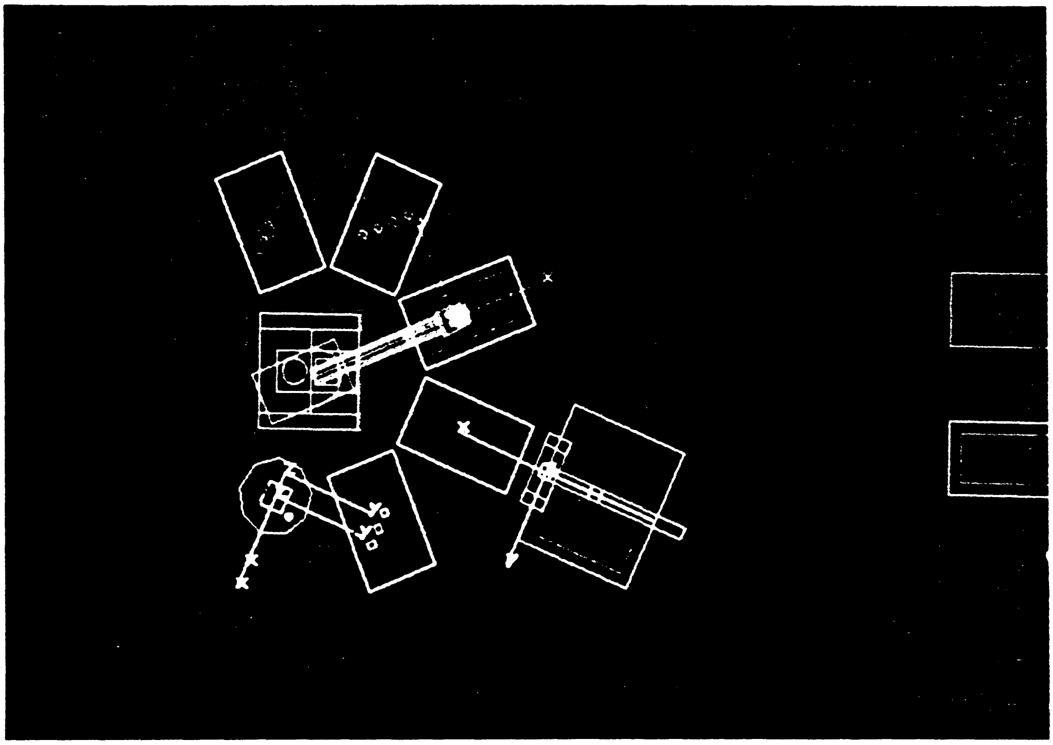


Figure 24 The layout of the truly flexible assembly cell, incorporating an industrial robot and AGV docking stations. The AGV can load/unload palletized parts as well as part feeding devices and a tool magazine in any programmed order, providing high level " material handling flexibility" both at cell, as well as at system level.

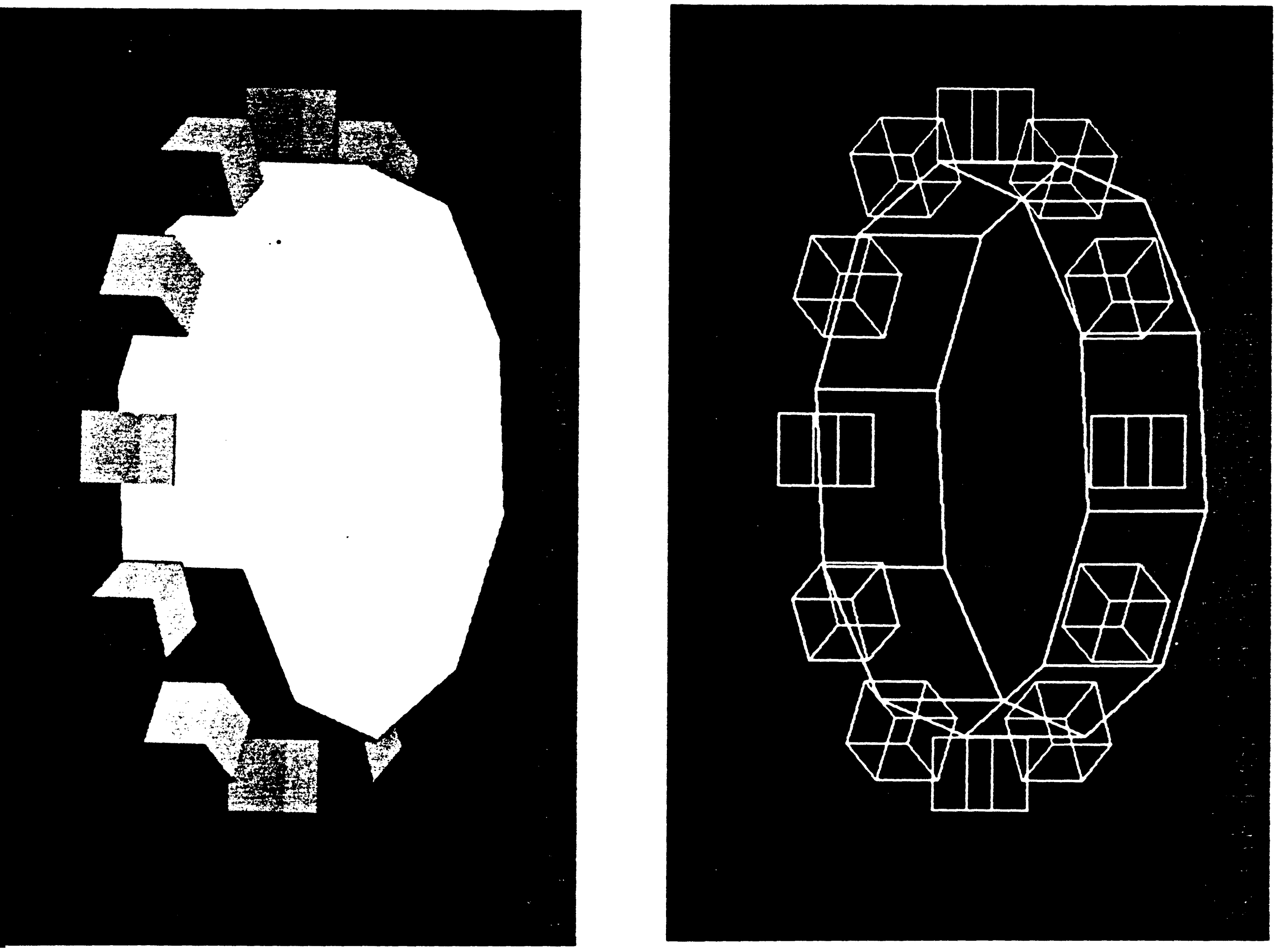


Figure 25 The conceptual design of a vertically indexing, rotary robot hand changer. (The rotary indexing station can be controlled pueumatically, or by means of the sixth or seventh axis of the robot, if available) (The photographs show both the solid model, as well as the wire frame model of the tool changer).

4. ROBOT TOOL MANAGEMENT SYSTEM DATA STRUCTURE DESIGN AND FLOWCHART

This section demonstrates the concept of a structured tool description method, developed by the author, using a Relational Data Base Management System. The method is "generic" thus can be used and/or adapted relatively easily for more or less complex applications, for small or large computers, networks and FMS/FAS systems.

The concept of handling robot and robot tool data sets is similar to a LEGO kit. It allows total flexibility for each implementation, while providing a "generic" data structure as a general guideline.

Following the structure and the robot tool description method new tools can be described and added to the system and necessary changes can be made as the system grows up-to the physical limits of the particular data processing hardware and software without any complications. (Figure 26 to 45 illustrates as well as outlines the design of the system). (Note that further publications are currently prepared by the author and his students documenting the real-time and off-line levels of this Robot Hand Management System.)

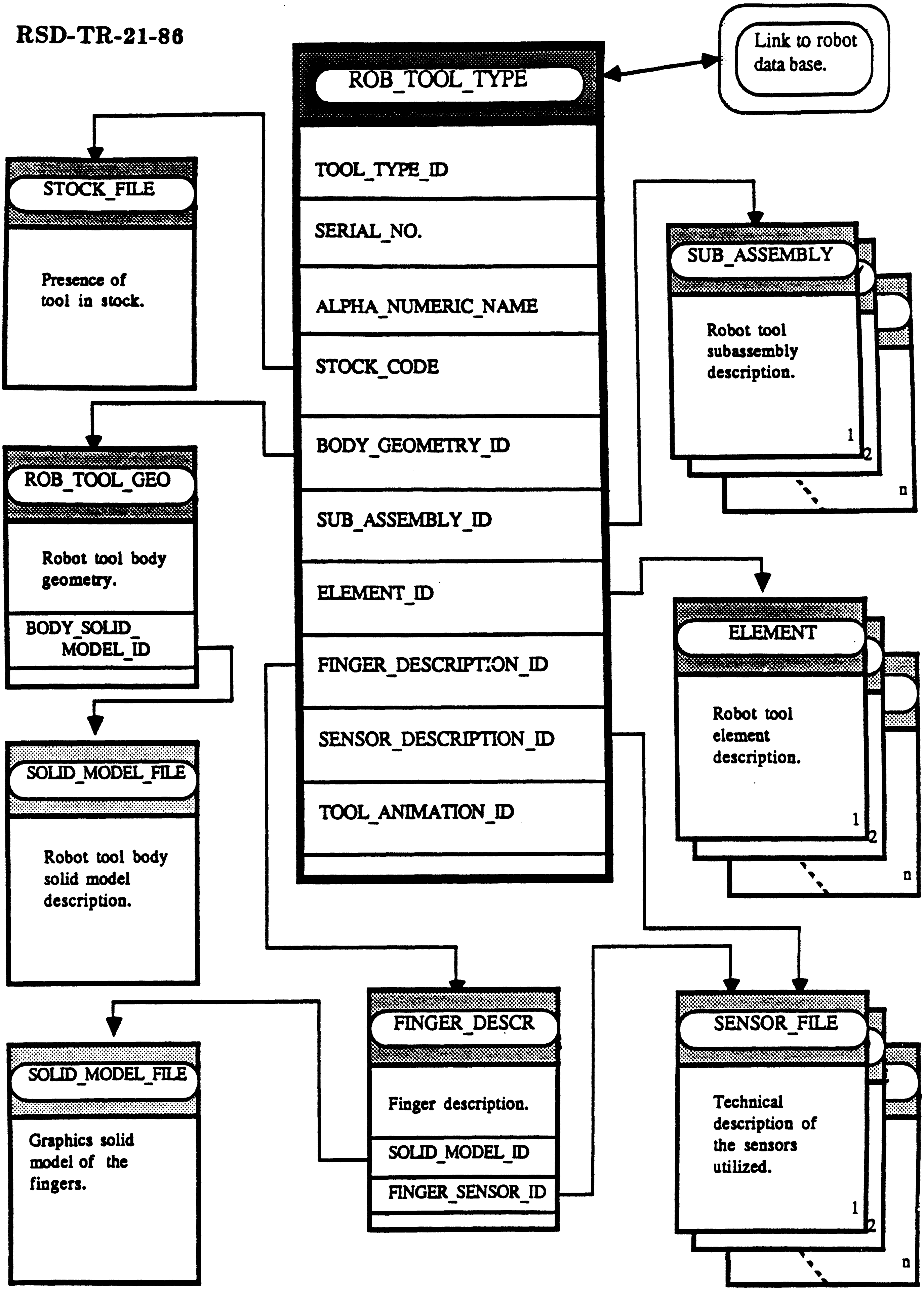


Figure 26 The data structure and some sample files of the robot tool data base.

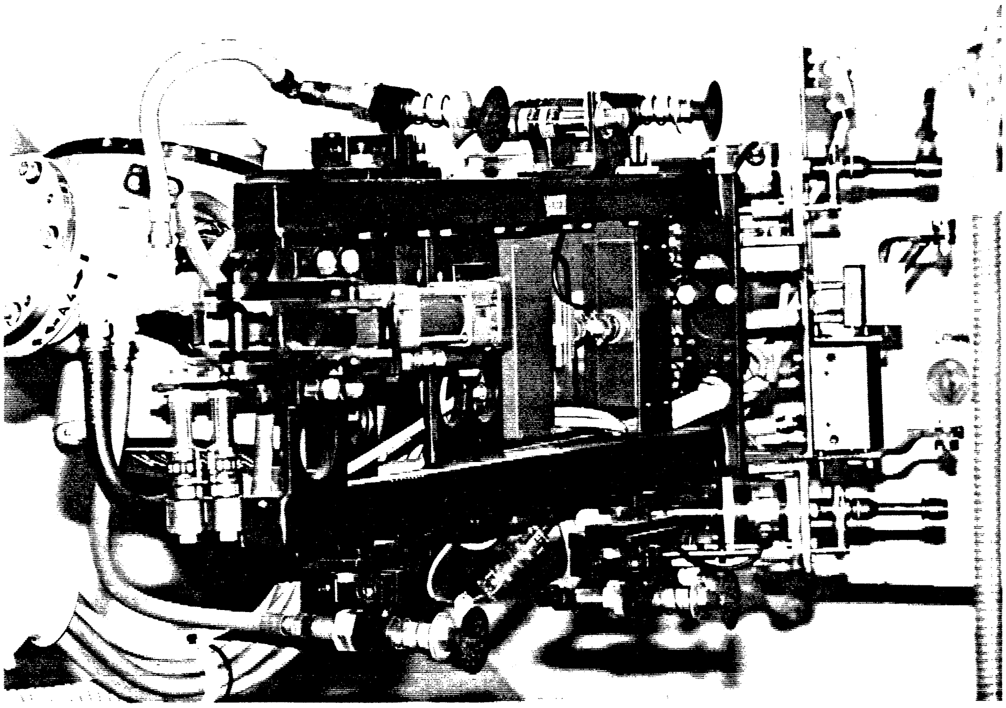


Figure 27 A robot tool management system and data base system must often co-op with such complete sensory tools as the one shown above for inserting welded door panels into automobiles in an assembly line, developed by KuKa, West Germany

ORDERCODE	ALPHA, MAX 20	CHARACTERS	PRIMARY KEY
GRAPHCODE	ALPHA, MAX 20	CHARACTERS	SECONDARY KEY
NAME	ALPHA, MAX 20	CHARACTERS	SECONDARY KEY
SUPPLIER	ALPHA, MAX 20	CHARACTERS	SECONDARY KEY
DEGFREE	LONGMATH		SECONDARY KEY
MAXLOAD	LONGMATH		SECONDARY KEY
POSERR	LONGMATH		SECONDARY KEY
COORDSYS	ALPHA, MAX 10	CHARACTERS	SECONDARY KEY
PROGRMODES	ALPHA, MAX 20	CHARACTERS	SECONDARY KEY
PRICE	LONGMATH		SECONDARY KEY
ASSEMBLY	BOOLEAN		SECONDARY KEY
PICKPLACE	BOOLEAN		SECONDARY KEY
PAINTER	BOOLEAN		SECONDARY KEY
WELDER	BOOLEAN		SECONDARY KEY
MACHINING	BOOLEAN		SECONDARY KEY
INSPECTION	BOOLEAN		SECONDARY KEY
OTHER	ALPHA, MAX 20	CHARACTERS	DATA FIELD

Figure 28 General description file of the robot.

This file contains useful information for quick robot searches. Robot application possibilities (ASSEMBLY-yes/no, WELDING-yes/no, etc.) are simply stored as boolean variables, in order to give a broad level and quick orientation to the searching program. Accurate search / evaluation programs should further access the TEST_RESULTS file before selecting the proper robot.

1) CONTRIDENT	ALPHA, MAX 20	CHARACTERS	PRIMARY KEY
2) CONTR_TYPE	ALPHA, MAX 40	CHARACTERS	SECONDARY KEY
3) POWER_EXT	ALPHA, MAX 40	CHARACTERS	SECONDARY KEY
4) POWER_INT	ALPHA, MAX 40	CHARACTERS	SECONDARY KEY
5) MEM_SIZE	LONGMATH		SECONDARY KEY
6) MEM_TYPE	LONGMATH		DATA FIELD
7) LANGUAGE	ALPHA, MAX 40	CHARACTERS	DATA FIELD
8) INPUT_PORT	LONGMATH		DATA FIELD
9) INPUT_TYPE	ALPHA, MAX 20	CHARACTERS	DATA FIELD
10) OUT_PORT	LONGMATH		DATA FIELD
11) OUTP_TYPE	ALPHA, MAX 20	CHARACTERS	DATA FIELD
12) ANALOG_INP	BOOLEAN		DATA FIELD
13) SENSING	ALPHA, MAX 40	CHARACTERS	DATA FIELD
14) PERIF1	ALPHA, MAX 20	CHARACTERS	DATA FIELD
15) PERIF2	ALPHA, MAX 20	CHARACTERS	DATA FIELD
16) PERIF3	ALPHA, MAX 20	CHARACTERS	DATA FIELD
17) SAFELOCK	ALPHA, MAX 20	CHARACTERS	DATA FIELD
18) WEIGHT	LONGMATH		DATA FIELD
19) AUXFUNC	ALPHA, MAX 20	CHARACTERS	DATA FIELD
20) GRAPHICS	ALPHA, MAX 20	CHARACTERS	DATA FIELD

Figure 29 Robot controller description file (CONTROLLER).

The explanation of the fields is as follows:

1) Identifier of the controller (e.g. the order code, or a serial number with some extension).
2) Type of controller, e.g. point-to-point, 6-axis continuous path, etc.
3) External power supply, e.g. 220V/AC
4) Internal power supply, e.g. 24V/DC
5) Memory size (preferably specified in Kbytes)
6) Type of memory, e.g. COMOS, ferrit, etc.
7) Description of the programming language (e.g. name, version, number, etc.)
8) Maximum number of input lines at the input port.
9) Type of input the controller can receive while controlled from the robot program.
10) Maximum number of output lines at the input port.
11) Type of output the controller can generate while controlled from the robot program.
12) This field is true if the controller can handle analog inputs (e.g. from a force sensor.)
13) Description of sensory feedback processing.
14) 15 / 16) PERIF1,2,3 describe the attachable periferals, e.g. disk, teach-box, etc.

17) Safety devices and signals the robot controller can handle.
18) Weight of the controller.
19) List of the auxiliary functions the controller can handle.
20) Solid model graphics file name for simulation purposes.

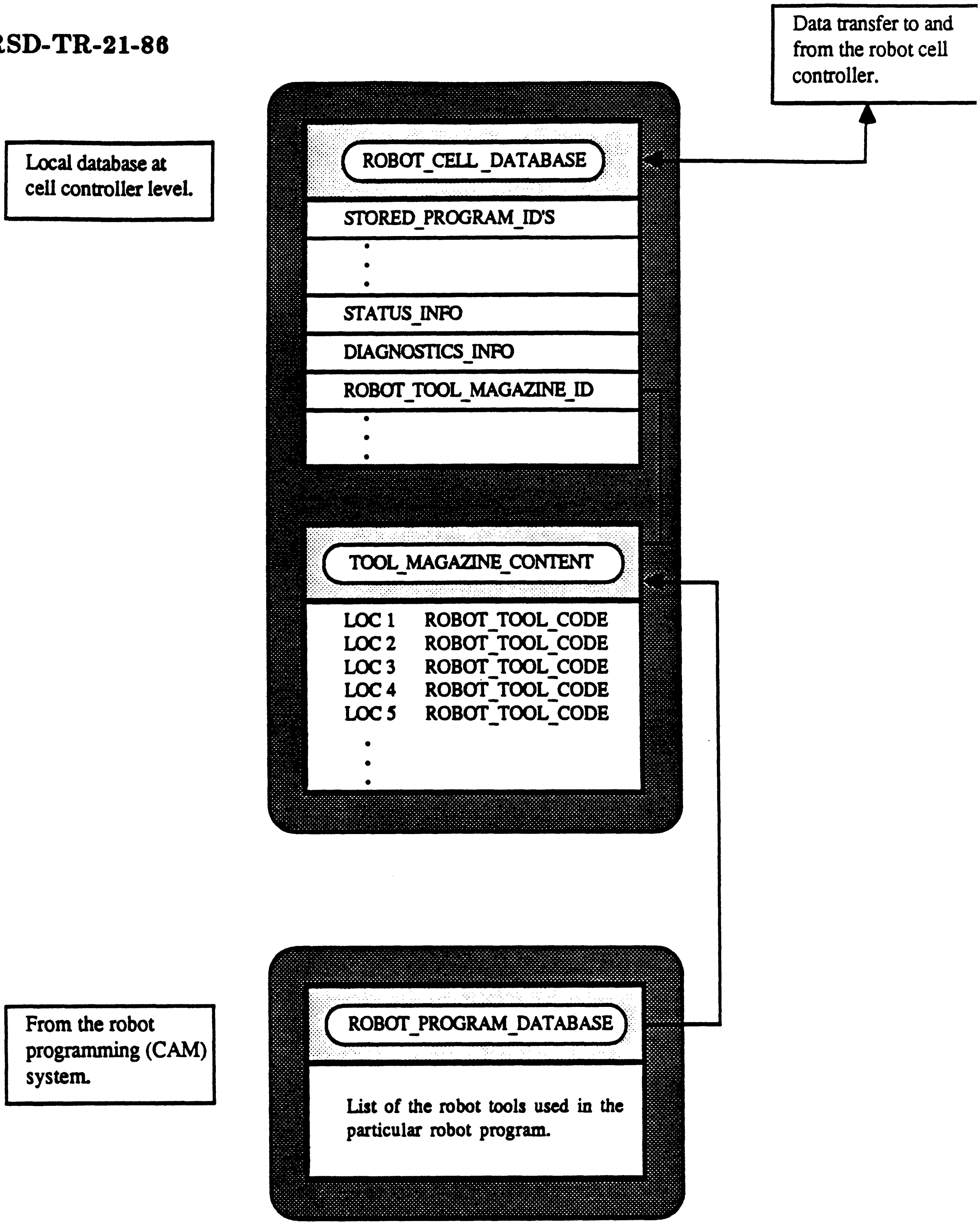


Figure 30 Typical real-time query when identifying the appropriate assembly robot, with the appropriate robot hand magazine contents, before finalizing the schedule of the FAS (Flexible Assembly System).

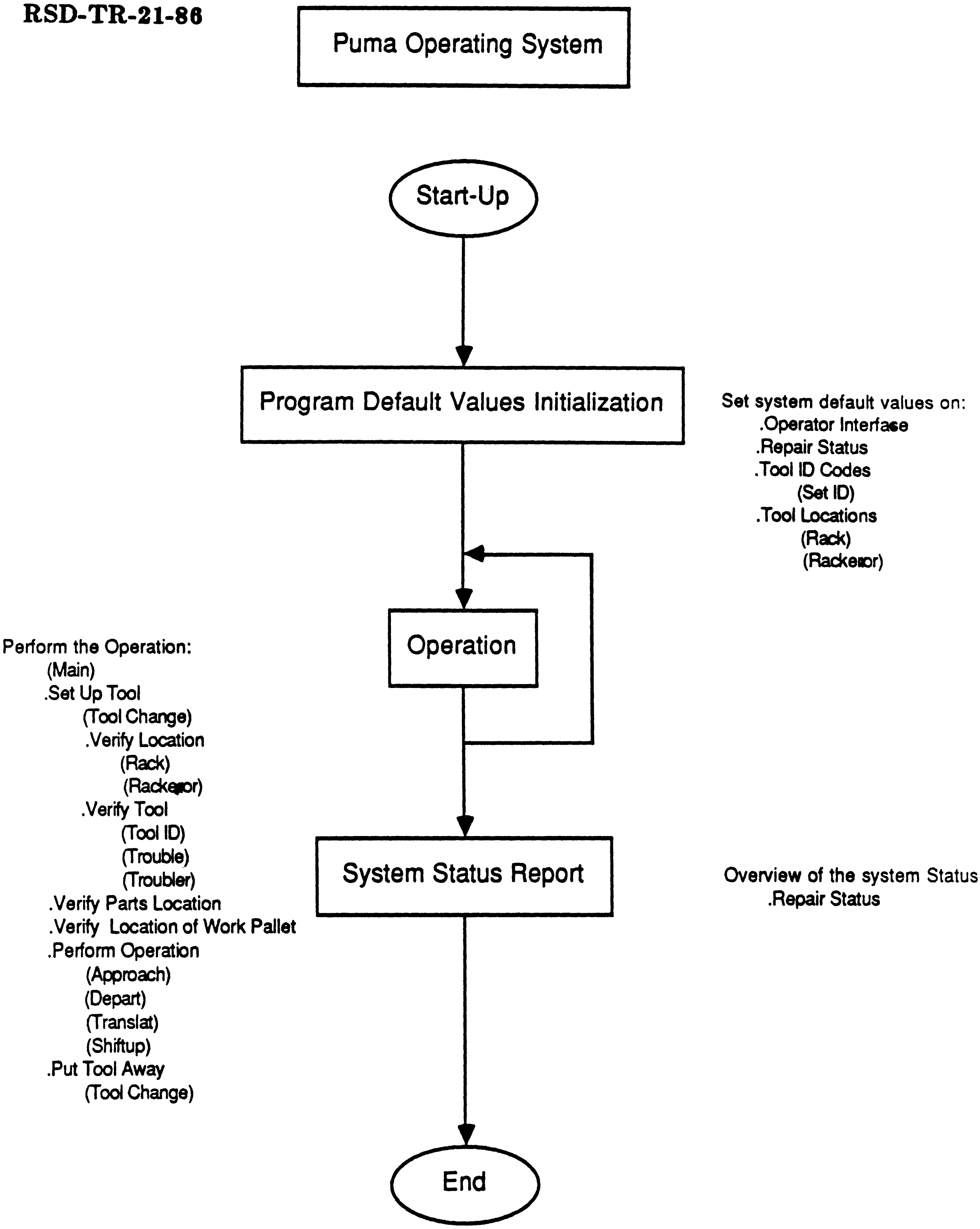


Figure 31 The interpetation of the real-time tool management routines into the Val II system (selected for convenience). [Ref.5]

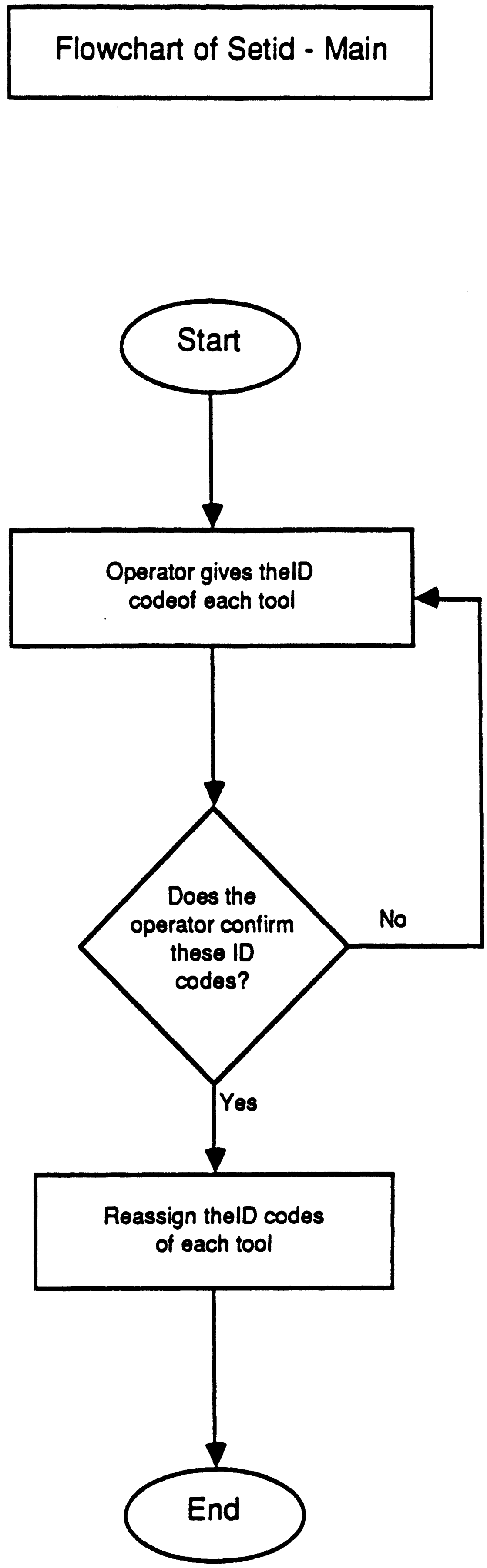

Flowchart of Setid - Main
Start
Operator gives theID codeof each tool
Does the operator confirm these ID codes?
No
Yes
Reassign theID codes of each tool
End

Figure 32

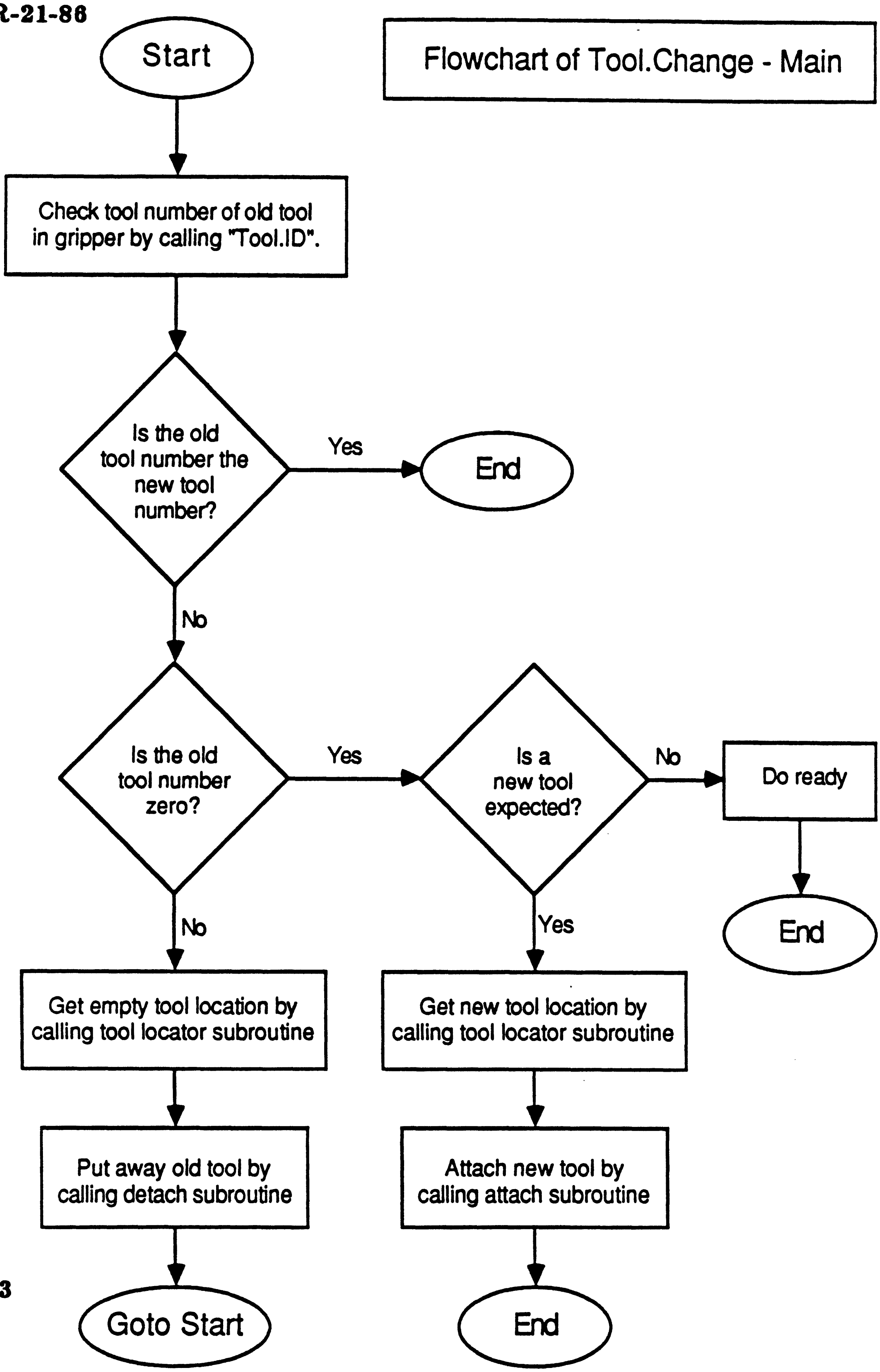

Flowchart of Tool.Change - Main
Start
Check tool number of old tool in gripper by calling "Tool.ID".
Is the old tool number the new tool number?
Yes
End
No
Is the old tool number zero?
Yes
Is a new tool expected?
No
Do ready
End
No
Yes
Get empty tool location by calling tool locator subroutine
Get new tool location by calling tool locator subroutine
Put away old tool by calling detach subroutine
Attach new tool by calling attach subroutine
Goto Start
End

Figure 33

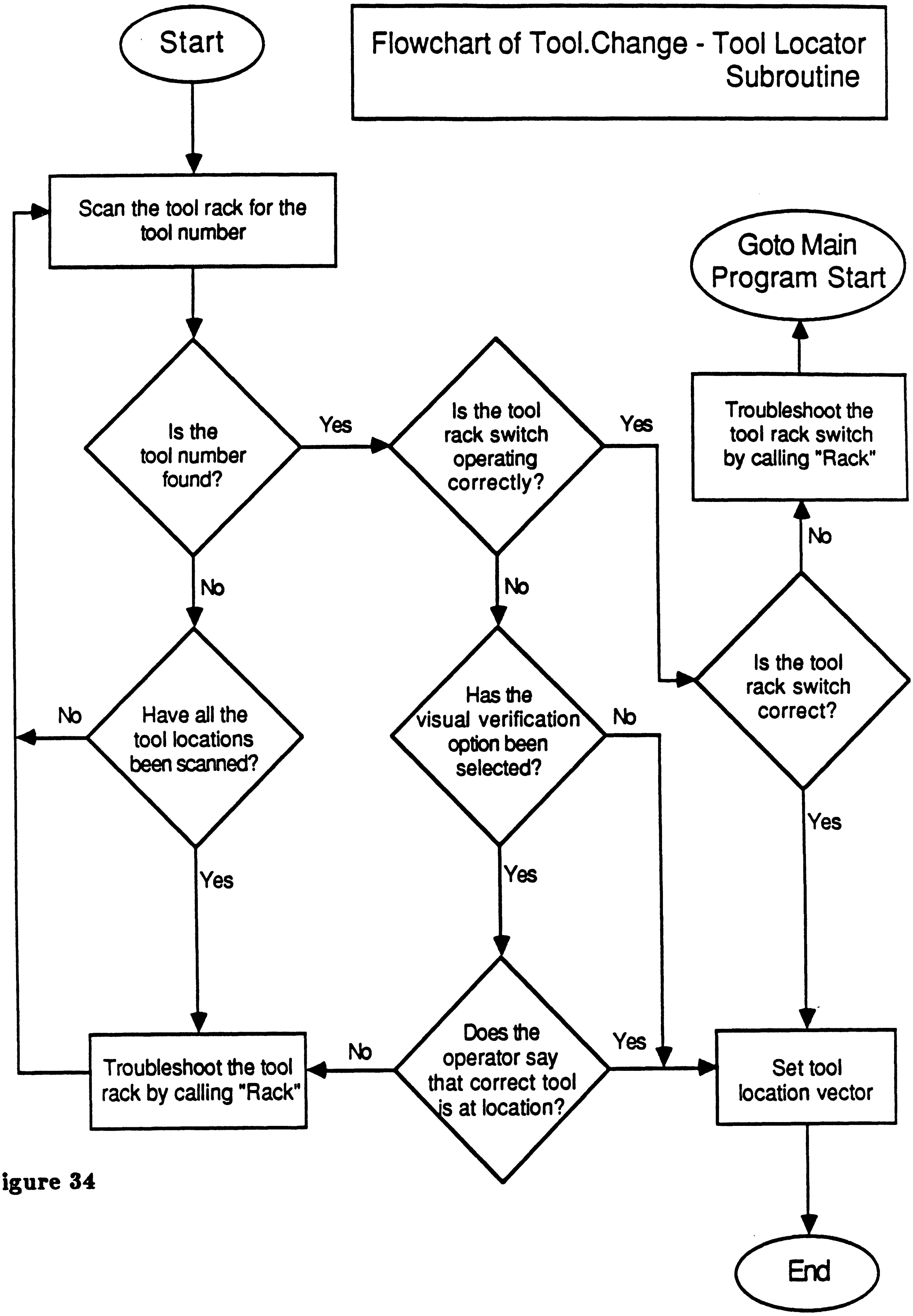

Start
Flowchart of Tool.Change - Tool Locator Subroutine
Scan the tool rack for the tool number
Goto Main Program Start
Is the tool number found?
Yes
Is the tool rack switch operating correctly?
Yes
Troubleshoot the tool rack switch by calling "Rack"
No
No
No
Have all the tool locations been scanned?
No
Has the visual verification option been selected?
No
Is the tool rack switch correct?
Yes
Yes
Yes
Does the operator say that correct tool is at location?
No
Yes
Troubleshoot the tool rack by calling "Rack"
Set tool location vector
End

Figure 34

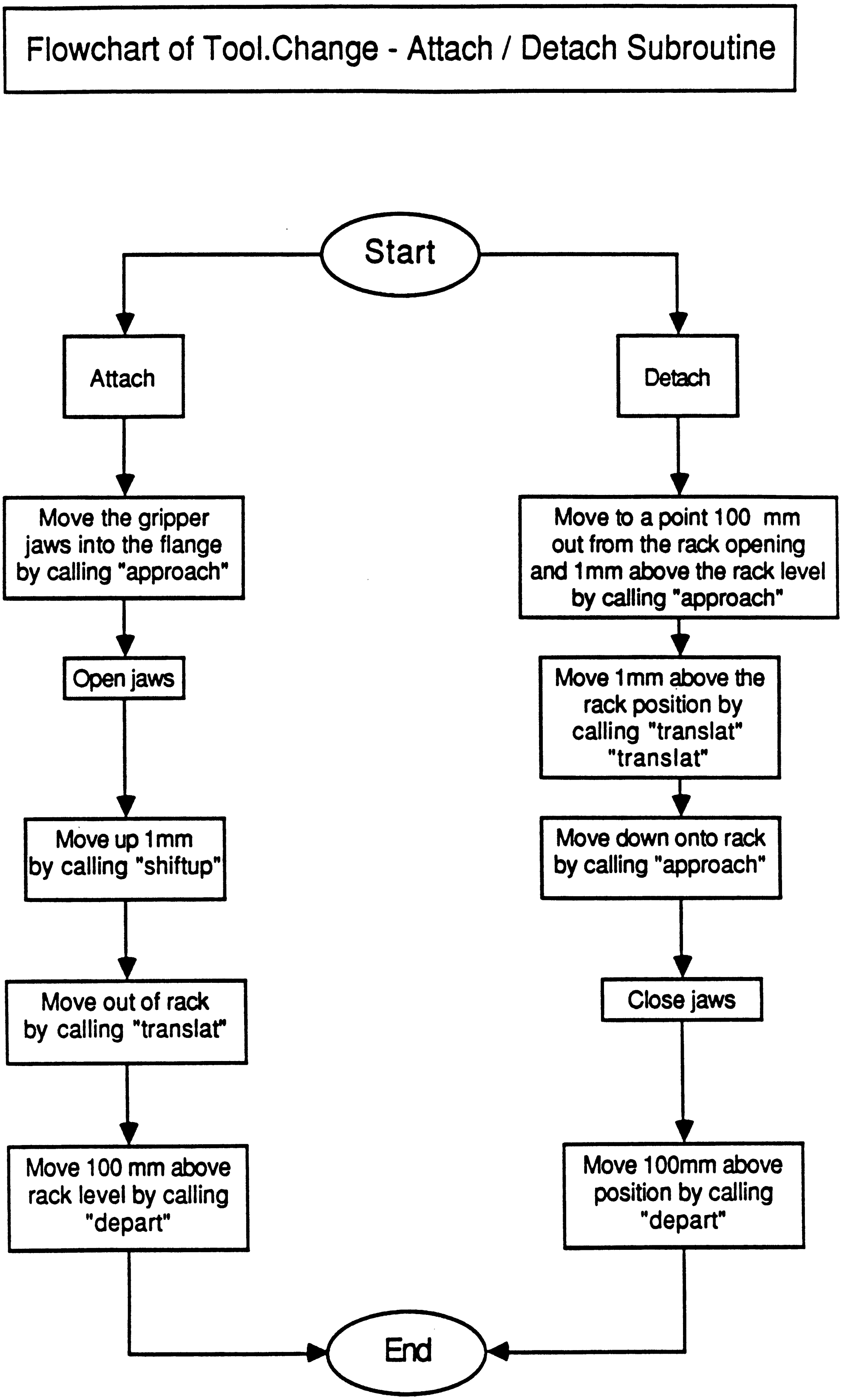
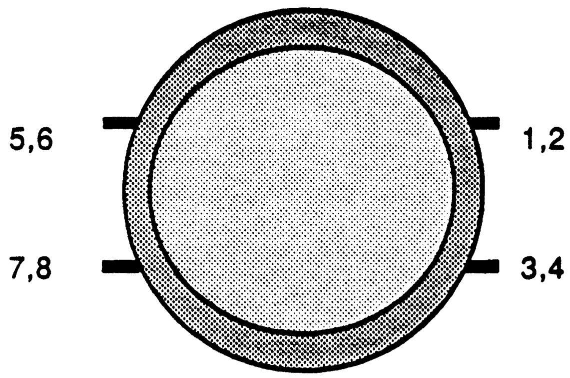

Flowchart of Tool.Change - Attach / Detach Subroutine
Start
Attach
Detach
Move the gripper jaws into the flange by calling "approach"
Open jaws
Move up 1mm by calling "shiftup"
Move out of rack by calling "translat"
Move 100 mm above rack level by calling "depart"
Move to a point 100 mm out from the rack opening and 1mm above the rack level by calling "approach"
Move 1mm above the rack position by calling "translat" "translat"
Move down onto rack by calling "approach"
Close jaws
Move 100mm above position by calling "depart"
End

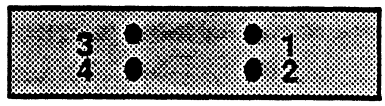
Figure 35

Ranky - Puma Standard Tool Connector Circuitry

Top View



Side view



Connector Numbering for the Connector

There are eight electrical connectors on the Ranky - style Puma tool connector. Of the circuits, two are used for the tool identification, and the remainder are used to operat the tool. The extra lines are used to run sensors, measure voltages, etc.

The tool electrical connections are further dedicated as follows:

.three of the connections are for 15V feed lines.

.five of the connections are for return lines.

One of the feed and return lines is for tool identification, and the remaining two fe and four return circuits are for use to operate the tool.

The following are the circuit pinouts of the Digitizer and Standard Gripper.

Digitizer Pinout		Std. Gripper Pinout
ID feed	Pin 1	ID feed
ID return	Pin 2	open return
Circuit 2 return	Pin 3	ID return
Circuit 1 return	Pin 4	open return
open return	Pin 5	open return
open return	Pin 6	open return
open feed	Pin 7	open feed
Circuit 1 feed	Pin 8	open feed

Figure 36

Digitizer Pinout Std. Gripper Pinout

Val Terminal I/O Port Identification

Signal 1	Pin 1	Pneumatic Feed 1
Signal 2	Pin 2	Pneumatic Feed 2
Signal 3	Pin 3	Tool Rack Switch Feed
Signal 4	Pin 4	Tool Connector ID Circuit Feed
Signal 5	Pin 5	Tool Connector Pin 7 Feed
Signal 6	Pin 6	Tool Connector Pin 8 Feed
Signal 7	Pin 7	Part Pallet Probe Feed
Signal 8	Pin 8	Working Pallet Probe Feed

Output Port Identification

Signal 1001	Pin 9	Tool Rack Switch 1 Return
Signal 1002	Pin 10	Tool Rack Switch 2 Return
Signal 1003	Pin 11	Tool Rack Switch 3 Return
Signal 1004	Pin 12	Tool Connector Pin 2 Return
Signal 1005	Pin 13	Tool Connector Pin 3 Return
Signal 1006	Pin 14	Tool Connector Pin 4 Return
Signal 1007	Pin 15	Tool Connector Pin 5 Return
Signal 1008	Pin 16	Tool Connector Pin 6 Return

Input Port Identification

Figure 37

```
*****************************************************************************************

                              Tool Change Program

*****************************************************************************************

;This program changes the tool on the robot arm.  It is set up
            .;to determine whether there is an old tool to put away
            ;before it attaches a new tool onto ther gripper,
            ;whether there is a new tool to attach at all.

;The program further verifies the identification of the to
            ;in the gripper to confirm that each tool it return
            ;the calling program is indeed correct.

;This program further troubleshoots malfunctions or
            ;inconsistencies in the operation of the tool rack
            ;switches and the gripper sensor.
;Finally, this program returns the robot to the calling pro
            ;in either a safe position above the tool rack when a
            ;tool was attached to the gripper or in the ready position
            ;of the gripper was left empty.

;To operate this program, simply call it after setting the
            ;variable old.tool to the number of the tool that is
            ;currently in the gripper and the variable new.tool to
            ;the number of the tool which should be put onto this
            ;gripper.  If either of these tools are "no tools",
            ;then set their respective variables to zero.

;Note that this program can also serve simply to verify the
          ;identification of any too on the gripper without
          ;changing that tool.  To have it do this,  simply call it
          ;after  setting the old tool number equal to the new tool
          ;number.

;    Programs called from this program are as follows:

;            .trouble
;            .rack
;            .toolid
;            .translat
;            .approach
```

Figure 38 **The tool change program in VAL II [Ref. 5] Note that the listed routines represent only a few of a larger robot program library developed for real-time tool management.**

```
;               .depart
;               .shiftup


;    Variables to be downloaded into this program are:

;               .old.tool                          (Integer)
;               .new.tool                          (Integer)
;               .vis.ver                           (T/F)

;    Variables internal to the program are as follows:

;               .tl                              (Integer)
;               .approach                        (0,1,2)
;               .tool.position                   (Integer)
;               .tool.number [ ]                 (Integer)
;               .tool.rack.switch.failure        (T/F)
;               .tool.rack.position.failure      (T/F)
;               .all.tool.position.check         (T/F)
;               .occupancy                       (Integer)
;               .tool.rack.ref                   (Location)
;               .tool.location                   (Location)
;               .point                           (Location)
;               .switch.signal                   (-1,0)

;    Signals activated in this program are

;    .signal 1: feed to tool rack switch 1
;    .signal 2: feed to tool rack switch 2
;    .signal 3: feed to tool rack switch 3
;    .signal 1001: feedback from signal 1
;    .signal 1002: feedback from signal 2
;    . signal 1003: feedback from signal 3

;     Tool Rack Reference Vectors:

;     .Position 1: shift (reference point by 0,20,0,0,0,0)
;     .Position 2: shift (reference point by 0,1,0,0,0,0)
;     .Position 3: reference point
;     .Position 4: gripper
;     .Tool rack reference point: trans (100,100,0,90,0
```

```
*************************************************************
                    Tool Change - Main

   This Val II program guides the flow of the tool changing program.
        First, the tool number of the tool on the gripper is
        confirmed.  Then, if necessary, a location vector is
        established for an empty tool location in the tool rack and then the
        old tool is put away.  Otherwise a location vector for a
        new tool is  established and that new tool is put on the wrist of the robot
        after this the program returns control to the calling program.
*************************************************************

; Establish the correct tool number of the tool on the
;         robot arm by running the "toolid" program.
;         The returned tool number should then be checked
;         in case it is the new tool number.

 call toolid
 if old.tool == new.tool then
    go to 50
end

  ;  If the old tool number is nonzero, then the location vector of
  ;         the empty spot on the tool rack is established to put
            the old tool away.  Otherwise, if the new tool number is
            non-zero, the location vector of the new tool is determined
            to get the tool.  If the new tool number is zero, then no
            tool location is determined and the program continues at label 5.
            The location vectors are determined in the "tool locator routine".

  if old.tool < > 0 the
     tl = old.tool
     go to 10
  end
  if new.tool == o then
     go to 5
  end

  tl = new.tool
  goto 10

   ;     Then attach / detach is performed.  If the old tool number is
   ;            non-zero, then the approach is set to 1.  Otherwise
   ;            it is set to 2.

5  if old.to < > 0 then
      approach = 1
   else
       if new.tool < > 0 then
          approach = 2
      else
          approach = 0
      end
   end
   go to 40

50  new.tool = 0
     return
    ;
```

Figure 39

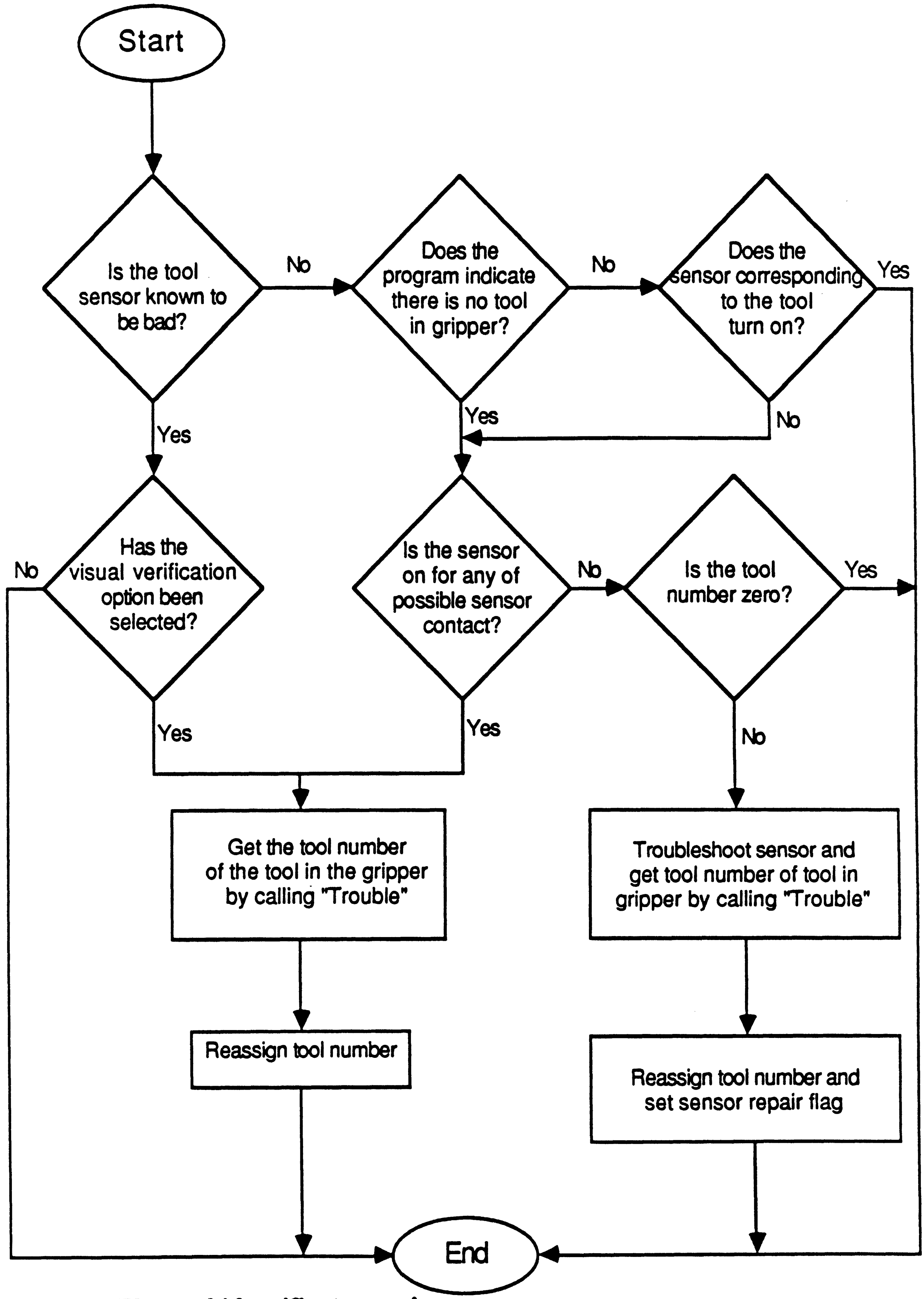


Figure 40 The tool identificator main program.

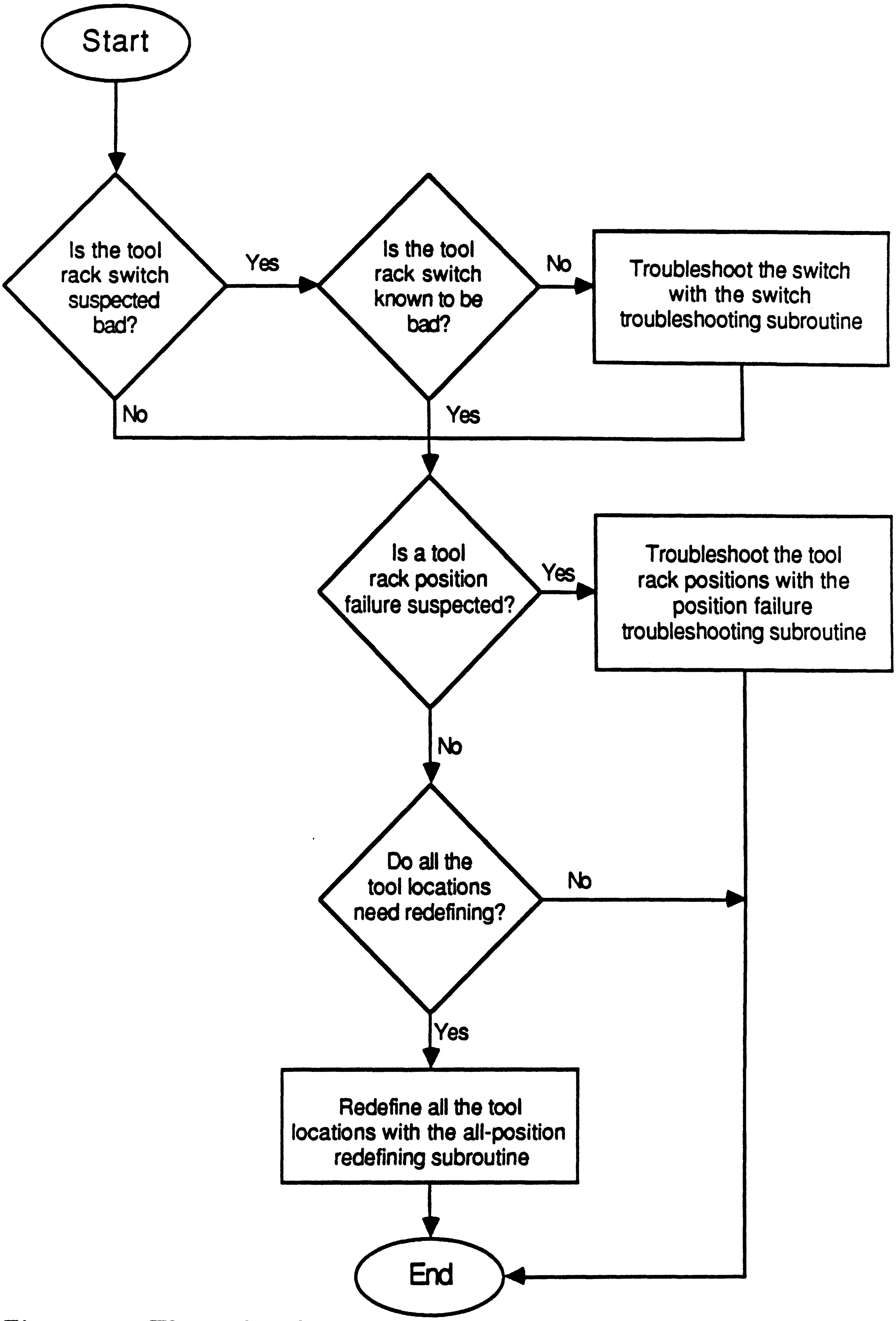


Figure 41 The tool rack management main program.

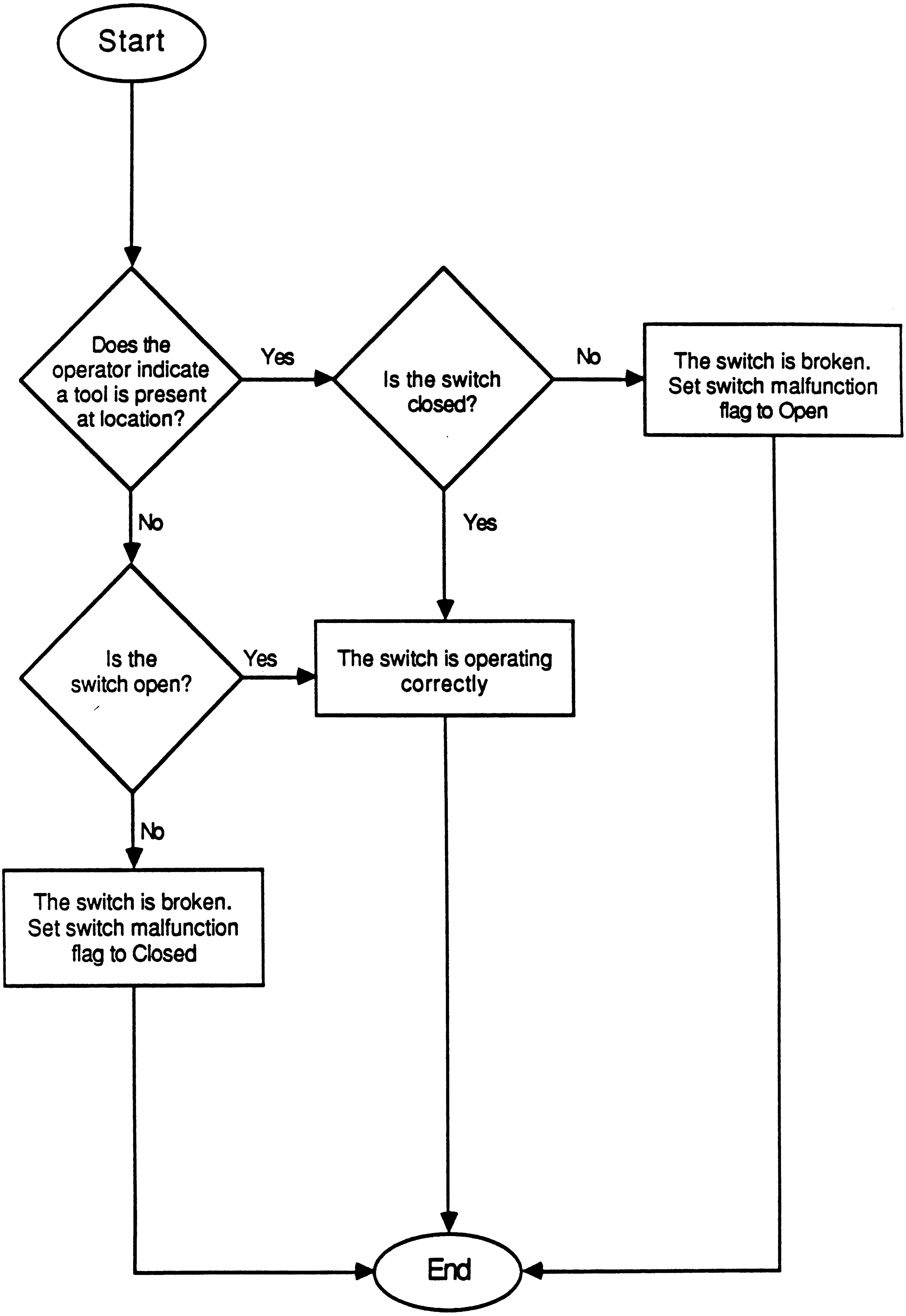


Figure 42 **The switch troubleshooting program of the tool-rack management main program. (Note that there is a micro switch sensing tool arrival/departure at each tool location inthe tool-rack)**

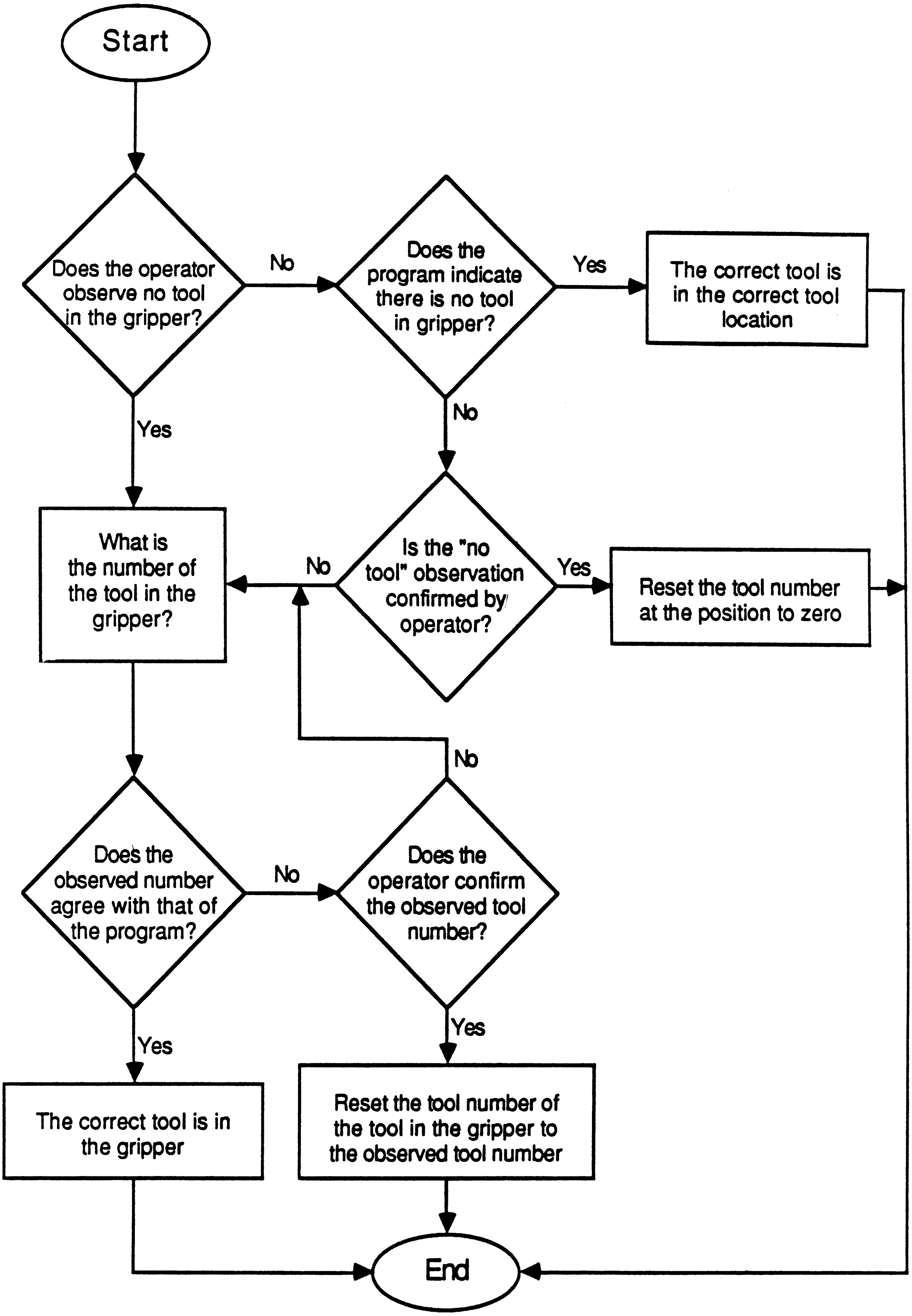


Figure 43 The tool rack tool load position failure troubleshooting subroutine.

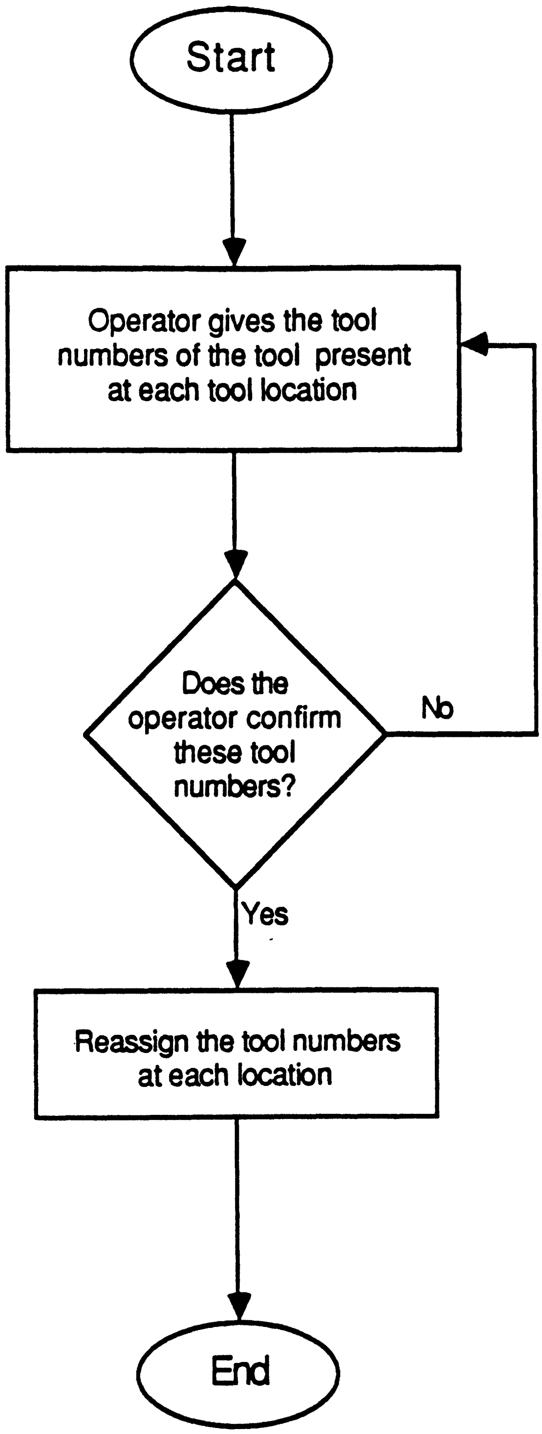


Figure 44 Tool rack setup routine

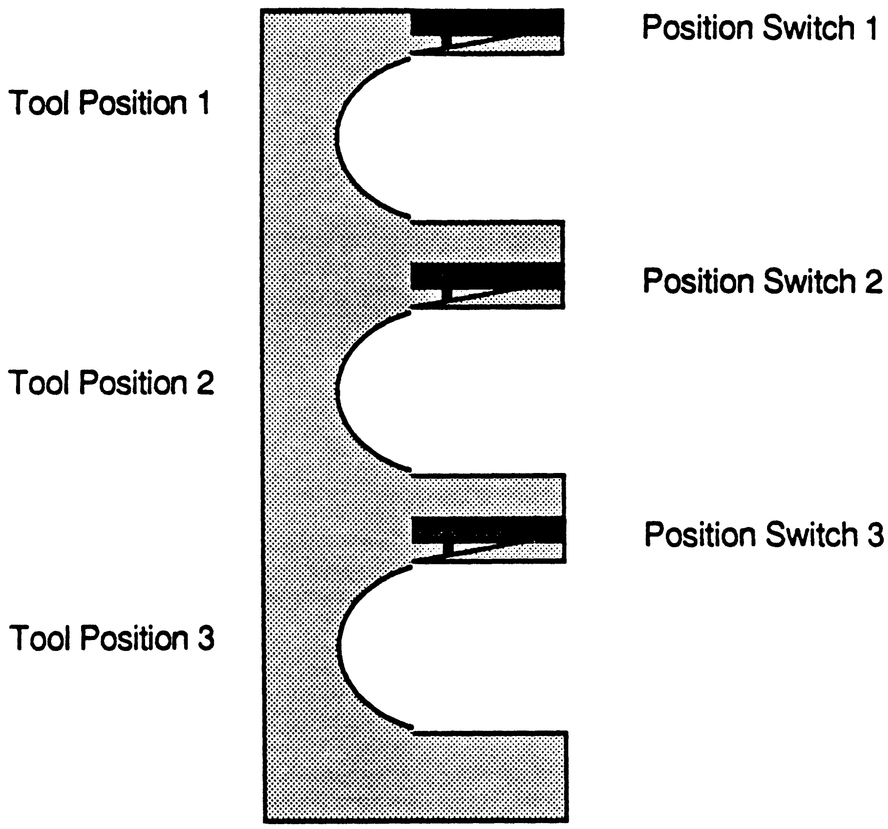


Figure 45 **The top view of the tool rack design indicating the switches, used in all tool rack management routines.**

5. CONCLUSION

There are no machine tools using a single tool. Very soon there will be no robots using a single hand, or gripper, but several hundred or more thus the robot hand management system design is of crucial importance to those who design, simulate, run an implement flexible assembly, welding, etc. systems. (Note that further details regarding the system under development can be found in the references and in papers currently prepared for publication.)

6. ACKNOWLEDGEMENTS The author would like to express his thanks to the University of Michigan, CRIM, and MEAM for supporting this research project.

REFERENCES AND FURTHER READING

[1] *Paul G. Ranky:* The Design and Operation of FMS, Flexible Manufacturing Systems), IFS Publications (Ltd) and North-Holland, 1983. 348 p.

[2] *Paul G. Ranky:* Computer Integrated Manufacturing, An Introduction with case studies, Prentice Hall International, 1985. 528 p.

[3] *Paul G. Ranky and Peter/C. Y./ Ho:* Robot Modeling, Control and Applications with software, IFS (Publications) LTD and Springer Verlag New York, 1985 348 p.

[4] *Paul G. Ranky:* Programming industrial Robots in FMS, Robotica (1984), Vol. 2. Cambridge University Press, Cambridge, UK, p. 87-92.

[5] *Paul G. Ranky-instructor and John Revelt-student:* ARHC Software Status Report 2, University of Michigan, Working Paper, 1986

[6] *Henry W. Stoll:* Design for Manufacturing: An Overview, Applied Mechanics Rev., Vol. 39, No. 9 Sept. 1986, ASME, USA

[7] *Paul G. Ranky:* Dynamic Simulation of Flexible Manufacturing Systems (FMS)., Applied Mechanics., Vol. 39, No. 9., Sept. 1986, ASME, USA

[8] *Paul G. Ranky:* End of Arm Tool Management System for Robotized Assembly Lines, SME Robots West Conference, Long Beach, CA, USA, Sept. 1986.

[9] ROBCAD System and use Manuals, ROBCAD/Detroit, 1985-86, USA.

[10] *Warnecke, H. J. and Schraft, R.D:* IndustrieRoboter, Mainz 1973, Krauskopf Verlag.