5th International Symposium on Industrial Robots

IIT Research Institute
Chicago, Illinois
Sept. 22-24, 1975

Sponsored by:
- Society of Manufacturing Engineers
- IIT Research Institute
- Robot Institute of America

With Participating Sponsors:
- The Charles Stark Draper Laboratory, Inc.
- Stanford Research Institute, Artificial Intelligence Center
- National Bureau of Standards, Office of Developmental Automation and Control Technology

TABLE OF CONTENTS

TABLE OF CONTENTS - CONT'D

TABLE OF CONTENTS - CONT'D

TABLE OF CONTENTS - CONT'D

AUTHOR AND CO-AUTHOR INDEX

EMERGENCY RECOVERY IN INTELLIGENT ROBOTS

Giuseppina Gini, Maria Gini and Marco Somalvico

Milan Polytechnic Institute
(Italy)

ABSTRACT

The modern results of artificial intelligence research have provided new techniques useful for the design of more sophisticated and advanced industrial robots. The use of industrial robots in the solution of the automatization of the assembly process for mechanical systems represents a very important application of industrial robotics. The purpose of this paper is to illustrate how with the use of an automatic problem solver operating within the computer which monitors the industrial robot, it is possible to achieve the automatic emergency recovery from a failure occourring during the normal activity devoted to the assembly process. The results presented, have been implemented on the UNIVAC 1108 computer by utilizing the MICROPLANNER programming in the description to the computer of the industrial robot's activity.

I. INTRODUCTION

The recent development of new capabilities and performances which are allowed to the computer activity, and in particular the research results achieved within artificial intelligence [Feigenbaum and Feldman (1963)], [Minsky (1969)] have already projected their great utility in the solution of hard problems which man has to face in different aspects of his technological progress.

Industrial robotics represents an important fact within this modern trend. An industrial robot is considered as an artificial system, capable of interacting with the external world, which is given an intelligent behaviour by means of the controlling activity of an interconnected computer which has the task of continuosly monitoring its functions and operations [Barrow and Crawford (1972)], [Michie, et al. (1973)].

A very developed field of industrial robotics is related with the intelligent automatization of the mechanical assembly process, i.e., the task of building up a mechanical system which is composed of several component parts.

The industrial robots of this class, are devoted to the automatic execution of a fixed sequence of elementary assembly operations, which makes up the completely assembled system. Although quite sophisticated as artificial systems, mainly on the mechanical aspect, such industrial robots have no ability in overcoming the sudden difficulties which arise when an emergency situation occours.

Suchevenience can happen when, with the continuos repetition of the same assembly process, a defective component part is encountered and, by consequence, the execution of one elementary assembly operation fails. The solution of such occourrences is available only to the man who has to find out how to recover from the emergency situation, in order to start again, afterwards, the deterministic and automatic assembly process.

The purpose of this paper is the investigation of the techniques which allow the computer to provide, automatically and independently, the solution of the emergency recovery problem. It is shown that this new capability requires that the computer is provided with an automatic problem solver, i.e., of a program which is able to automatically construct the solution of a problem which has been represented, i.e., described by the man [Amarel (1968)], [Nilsson (1971)], [Slagle (1971)], [Mandrioli, et al. (1973)].

In this way it is shown how it exists an important interaction between the theoretical efforts pursued within artificial intelligence research, and the practical goals which are at the basis of industrial robotics. The results here presented, illustrate how such an intelligent robot interacts with a computer in which the standard assembly process is monitored within the execution of an human invented mechanical solution algorithm (deterministic programming aspect of the computer), while the automatic construction of the solution of an emergency recovery problem is provided by the automatic problem solver, by means of a search process operating on the information which is contained into the representation of such a problem (non deterministic programming aspect of the computer).

The research results have been experimented on our UNIVAC 1108 computer by utilizing the MICROPLANNER interpreter [Sussman, et al. (1970)] which acts like an automatic problem solver on the representation of the emergency recovery problem describbed within the MICROPLANNER goal-oriented language.

Such a problem arises when the execution of the standard assembly algorithm, which is carried on by the LISP interpreter (contained within the MICROPLANNER interpreter) has to be temporarily suspended, whenever a defective component part is encountred.

The MICROPLANNER interpreter is able to find automatically the solution of the emergency recovery problem, to monitor its execution, and, afterwards, to give back the control to the execution of the standard assembly program.

This work represents a part of the research activity which is being developed at the Milan Polytechnic Artificial Intelligence Project (MP-AI Project), and which is focused in the direction of the theory of problem solving, in the experimentation and design of representation languages (goal-oriented languages), and industrial robotics.

In Section II we will present the characteristics of the mechanical assembly problem and the importance of the occourrence of emergency situations. In Section III we will discuss problem solving at the light of the interaction between deterministic and non deterministic programming as related with industrial robotics. In Section IV we illustrate the case study of assembly process by an industrial robot, which has been proposed by Olivetti , and we propose its solution. In Section V we describe the characteristics and the use of MICROPLANNER programming in the construction of the solution of the presented case study. In the Conclusions we summarize the results which have been obtained, and we outline the directions for future research work.

II. THE MECHANICAL ASSEMBLY PROBLEM

The advent of the modern technologies of electronics, computer science, and automatic control, together with the advanced and recent results in dygital systems and in mechanical trasducers and manipulators, has provided, in general, a great impact and potentiality on the automatization of industrial processes [Michie, et al. (1973)]. In particular, an improved and sophisticated use of the electronic computer has enabled the design and construction of much more powerful artificial systems than the traditional automatically controlled systems.

Thus, the advent of the notion and realizations of robots and industrial robots, has dramatically changed the environment and the technology in which to insert artificial systems considered as powerful tools for reducing labour and difficulty in human activities [Barrow and Crawford (1972)].

Since the beginning of the first studies and designs of intelligent robots, the problem of mechanical assembly has been considered as one of the most important and natural, in which the potentiality of such artificial systems could be matched with the exigencies, sometimes very hard, deriving from real industrial problems [Winston (1972)].

The mechanical assembly problem arises from the exigence of assisting and possibly substituting the man in the task of carrying on a sequence of elementary assembly operations which are necessary for making up a mechanical system (or subsystem) which is composed by a given number (usually some decades) of component parts.

The solution of this exigence has been achieved firstly by means of mechanical trasducers, controlled by the man, which were able to perform just one or few elementary assembly operations. Thus, the whole assembly process was organized with the use of assembly lines capable of producing at their outputs the assemblage of a certain number of mechanical systems. Different types of very simple mechanical trasducers, controlled by men constrained in continuosly repeating the same kind of elementary assembly operation, and distributed along the length of the assembly line, were necessary to carry on the assembly process.

The most modern solution to the same exigence has been obtained within the use of assembly stations where a concentrated and sequential activity, based on different kinds of elementary assembly operations, and performed by the same man assisted by complex and sophisticated multipurpose mechanical trasducers, was intended to completely provide the goal of assemblying a mechanical system.

The increasing automatization of such assembly stations has made necessary the use of more and more sophisticated multipurpose mechanical trasducers, namely, the programmable industrial robots. The main feature of these industrial robots, is characterized by their interaction with a controlling computer, generally a minicomputer or even a microcomputer (possibly, a special purpose wired program computer), which has the task of monitoring the execution, by the mechanical trasducers of the industrial robot, of the appropriate sequence of elementary assembly operations which is necessary to be executed in order to achieve the assembly of a mechanical system.

This activity,demanded to the industrial robot, requires that the man has once for all determined which is the sequence of operations which are sufficient in order to obtain, generally in an efficient way, the automatic assembly of the system.

However, the limits,which are connected with completely determined activity of the industrial robot, are very important and significant ones. Every time that, during one of the many operations of the assembly sequence, something goes wrong, i.e., an emergency arises due to some manufacturing defect of one of the component parts, then the robot's automatic activity has to be stopped, and the man has to take over the responsability of finding an appropriate solution to the need of recovering from the emergency which has been encountered.

Because of the too many different reasons and situations in which an emergency case might occour, it is not convenient to determine completely all such eveniences, and it is not practicable to provide the industrial robot with all the corresponding sequences of operations necessary for automatically recovering from such situations.

Therefore the need arises of increasing the intelligence of the robot, in the sense that it shall be given the responsability of finding automatically, within its own processing capability, an appropriate solution whenever it might be confronted by an unpredictable emergency recovery problem.

This new ability implies that the computer, interacting with the industrial robot, will be given the capability of automatically finding the solution to the emergency recovery problems. Therefore the computer will be programmed with an automatic problem solver, i.e., a program which is able to construct the solution of an appropriately represented problem.

In this way we are showing how it might exist an useful interaction between industrial robotics and the modern research results achieved within artificial intelligence, in general, and problem solving, in particular. In the next Section we are going to deal with such useful interaction.

III. PROBLEM SOLVING AND INDUSTRIAL ROBOTICS

The invention by the man of the sequence of operations is communicated to the computer by means of a deterministic program in which has been embedded the assembly algorithm providing the solution of the assembly problem. We will call such activity as deterministic programming.

The new requirements imposed on the computer for automatically solving the emergency recovery problems , demands a new kind of interactionbetween man and computer, which is called non deterministic programming, i.e., problem representation, in the sense that the man does not determine which will be the solution algorithm, which is being to be automatically constructed by the automatic problem solver, operating within the computer, and processing the informations contained only in the representation of the emergency recovery problem.

The design of automatic problem solvers involves mainly two essential questions, namely how to describe the problem to the computer (representation), and how to obtain a solution, possibly optimal, in an efficient way (search).

An useful approach to problem solving, which is indicated to discuss its relevance with the control of intelligent robots, is thestate-space approach to problem solving, called SSPS.

The elements which make up the SSPS representation are: (i) the set of all the possible situations, called states,of the problem, which is called state-space;(ii) a precisely defined initial situation, called initial state; (iii) a precisely goal situation, called goal state; (iv) a set of operations, called operators, which change one situation into another one.

The elements which make up the SSPS search are the most efficient algorithms which are able to process the SS in order to construct a solution, possibly optimal, to the problem,which is intended as a sequence of operators that change the initial state into a goal one.

In the case of an industrial robot's activity, the states correspond with any situation occourring during its operation, while the operators correspond with each elementary assembly operation available to any of its mechanical trasducers. The solution, i.e., the sequence of operators, corresponds to the assembly process intended as a sequence of elementary assembly operations.

IV. A CASE STUDY OF ASSEMBLY PROCESS BY AN INDUSTRIAL ROBOT

In the previous Sections we have illustrated the relevance in industrial robotics of the mechanical assembly problem, and the usefulness of the nondeterministic behaviour of an industrial robot in the evenience of emergency situations.

In this Section we are going to illustrate a real example which has oriented and motivated our research activity.

The case study here illustrated has been proposed by Olivetti Company, which is beginning to insert multifunction industrial robots in its mechanical assembly lines.

In Fig. 4.1 we show the schema of a real mechanical subsystem of a teletype which has to be assembled. The mechanical system is the driver of a teletype drum. The driver is composed by nine parts: four blocks and one bar on which the blocks must be fixed by four screws, in correspondence with four holes in the bar.

Fig. 4.1

Scheme of the Driver

In Fig. 4.2 we show the three different component parts of the driver.

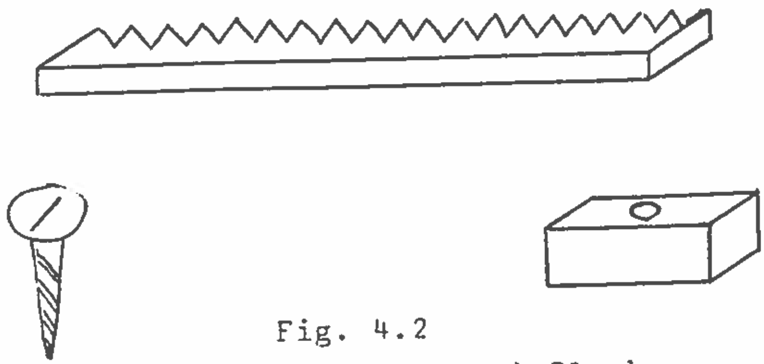

Fig. 4.2

Component parts : a) Bar, b) Screw, c) Block

The assembly task is performed by a computer-controlled robot, which is made up by a mechanical arm whose hand is used to pick up a component part from its loader, and to fetch it in the appropriate position of the assembly platform. Moreover the hand is used to screw together one block with the bar, by means of one screw in the corresponding hole. In the sequel we will simply mention the hand, without making any distinction on which of the two indicated actions we are referring to.

Our investigation on the emergency situations arising in the manipulation behaviour of such a robot, have been carried on a simulation of its activity on our UNIVAC 1108 computer.

Thus the problems connected with a real time control of the mechanical arm have been considered only with respect to the characterization of the real environment from which arise the emergency problems to be solved automatically.

Each one of the three component parts of the driver, namely the bar, the block and the screw, is positioned on a loader containing many instances of the same component part.

The assembly process is performed on a platform from where the assembled driver is taken away.

The elementary assembly operators which belong to the robot activity are the following ones :

1. GRASP : appropriate for picking up a component part or a (partially or totally) assembled driver.
2. UNGRASP: appropriate for leaving a component part or a (partially or totally) assembled driver.
3. TRANSLATE: appropriate to the necessary translation movements among the loaders, the platform, the wastebasket, and the driver's basket.
4. SCREW : appropriate to the mechanical connection between a block and a bar by means of a screw.
5. UNSCREW: appropriate to the mechanical disconnection between a block and a bar previously connected by means of a screw.

In the sequel we will utilize special names for describing the activity of each operator.

We will indicate the following position names :

1. P1, P2, P3, P4 indicate the central position of each block on the assembly platform;

2. P5 = g (P1, P2, P3, P4) indicates the central position of the bar on the assembly platform;

3. L1, L2, L3 indicate the picking up position from the loaders of the blocks, bars, and screws.

After having presented an intuitive description for each one of the operators of the robot, we shall now introduce a more formal and rigorous characterization for each one of them.

Each operator is characterized by two principal components: the preconditions, which must hold before that the operator may be applied, and the effects produced by the operator. [Fikes and Nilsson (1971)] .

We shall now illustrate the formal description of each one of the previously introduced operators.

If the first operator is described in such a theoretical formulation, we have the following description :

1. name : GRASP (hand grasps an object a in the position p)
   parameters : a,p
   Preconditions : IN (Hand,p), CLEAR (Hand), OBJECT (a),
   IN (a,p)
   effects :
   add list : CARRYING (a)
   delete list:CLEAR (Hand), IN (a,p)

The operator may be applied if the preconditions are satifyied, i.e. if an object a can be found in the position p, and if the robot's hand is clear and it is on the same position p.

When the operator is applied, the relationships CLEAR (Hand) and IN (a,p) are removed, and CARRYING (a) is added to the description of the actual state.

In the same way we will describe the remaining operators.

2. name : UNGRASP (object a is left in p)
   parameters: a,p
   preconditions : OBJECT (a), IN (Hand, p), CARRYING (a)
   effects :
   add list : CLEAR (Hand), IN (a,p)
   delete list:CARRYING (a)

Please note that this operator is just the reverse of the operator GRASP.

3. Name : TRANSLATE (there is a translation from p to q)
   parameters : p,q
   preconditions : IN (Hand, p)
   effects :
   add list : IN (Hand,q)
   delete list: IN (Hand,p)

4. name : SCREW (a screw a is screwed in the position p)
   parameters : a,b,c,d,e,f,c,p,q,r,s,
   preconditions : IN (Hand, p), CARRYING(a), SCREW(a), IN (b,p)›IN(c,q), IN(d,r), IN(e,s), BLOCK(b), BLOCK(c),BLOCK (d), BLOCK(e), IN(f,g(p,q,r,s)),BAR (f)

   effects :
   add list : CLEAR (Hand), Screwed (a,b,f,p)
   delete list:CARRYING(a), IN(b,p)

5. name: UNSCREW (a screw a is unscrewed in position p)
   parameters: a,b,f,p
   preconditions : IN(Hand,p), CLEAR(Hand), SCREWED (a,b,f,p),

   effects:
   add list : CARRYING (a), IN(b,p)
   delete list: SCREWED (a,b,f,p), CLEAR (Hand)

A state description for this world is a set of predicates which expresses the relations between objects, the state description represents the knowledge about the world at a certain instant.

The initial state is described by the following assertions :

A1. IN (Hand, Posin)
A2. CLEAR (Hand)
A3. IN (Block 1, L1)
A4. IN (Bar 1, L2)
A5. IN (Screw 1, L3)
$A6_1$.OBJECT (Block 1) ... $A6_m$. OBJECT (Block m)
$A7_1$.OBJECT (Bar 1) ... $A7_n$. OBJECT (Bar n)
$A8_1$.OBJECT (Screw 1) ... $A8_p$. OBJECT (Screw p)
$A9_1$.SCREW (Screw 1) ... $A9_p$. SCREW (Screw p)
$A10_1$. BLOCK (Block 1) ... $A10_m$.BLOCK (Block m)
$A11_1$. BAR (Bar_1) ... $A11_n$.BAR (Bar_n)

The assertions A1.- A5. describe the initial position and situation of the hand and of the component parts. The state can change by means of a modification of its defining assertions during the execution of the program.

The assertions $A6_1$.- $A11_n$, are always true; they define the characteristics of the objects.

Of course we have described only that part of the world which is strictly related to the robot activity and to its possible source of emergency situations. We have in fact described an open world which interacts with other parts of the world, such as the loaders, the wastebasket, and the driver's basket.

Of this external world we are not considering in detail its operators whose activity is parallel to the activity of the operators which are internal to the specific world of the robot.

In fact, the assertions A3, A4. and A5. are modified also by the operators external to the world of the robot. When one of these assertions is deleted by the operators of the world, another operator of the loader, here not defined, adds a new suitable assertion.

For example :
if is deleted the assertion : IN (Block 1, L1)
the loader's operator adds: IN ($Block_{i+1}$, L1)

Our representation of the world is focused on the actions which the robot can perform in its world, and, not on the external world where there are other independent systems. The problem of the correlation between such systems is not considered in our exposition, because it is not relevant to our main concern, i.e. the arising of emergency problems related with the strict activity of the robot.

The solution of the previous assembly task is constituted by a sequence of operators which is illustrated in fig. 4.3
This sequence of operator is obtained automatically by a problem-solving, given a description of the final state :
SCREWED ($Screw_1$, $Block_1$, Bar_1, P1), SCREWED ($Screw_2$, $Block_2$, Bar_1, P2), SCREWED ($Screw_3$, $Block_3$, Bar_1, P3), SCREWED ($Screw_4$, $Block_4$, Bar_1, P4).

TRANSLATE (Posin, L1)
GRASP ($Block_1$,L1)
TRANSLATE (L1, P1)
UNGRASP ($Block_1$,P1)
TRANSLATE (P1,L1)
GRASP($Block_2$,L1) — Fig. 4.3 (a)
TRANSLATE (L1, P2)
UNGRASP ($Block_2$, P2) — Solution of the assembly problem
TRANSLATE (P2, L1)
GRASP ($Block_3$, L1)
TRANSLATE (L1, P3)
UNGRASP ($Block_3$, P3)
TRANSLATE (P3, L1)
GRASP ($Block_4$, L1)
TRANSLATE (L1, P4)
UNGRASP ($Block_4$, P4) — The four blocks are posed
TRANSLATE (P4, L2)
GRASP (Bar_1, L2)
TRANSLATE (L2 ,g(P1,P2,P3,P4))
UNGRASP (Bar_1, P5) — the bar is posed
TRANSLATE (P5, L3)
GRASP ($Screw_1$, L3)
TRANSLATE (L3,P1)
SCREW ($Screw_1$, $Block_1$, $Block_2$, $Block_3$, $Block_4$, Bar_1, P1,P2,P3,P4)
TRANSLATE (P1, L3) — GRASP ($Screw_2$, L3) — TRANSLATE (L3,P2)
SCREW ($Screw_2$, $Block_2$, $Block_1$, $Block_3$, $Block_4$, Bar_1, P2,P1,P3,P4)
TRANSLATE (P2, L3) — GRASP ($Screw_3$, L3) — TRANSLATE (L3,P3)

SCREW (Screw$_3$, Block$_3$, Block$_1$, Block$_2$, Block$_4$,
Bar$_1$, P3, P1, P2, P4)
TRANSLATE (P3,L3)
GRASP (Screw$_4$, L3)
TRANSLATE (L$_3$, P4)
SCREW (Screw$_4$, Block$_4$, Block$_1$, Block$_2$, Block$_3$)
Bar$_1$, P4, P1, P2, P3)   The Screws are screwed
TRANSLATE (P4,Posin)

Fig. 4.3 (b)

Solution of the assembly problem

The operators which are used to make up the solution of the normal assembly problem don't make use of the UNSCREW operator, which is used only when an emergency arises.

Since the problem is deterministic we may use this sequence of operators which is apt to solve the problem, and which can be directly programmed once for all.

The assembly process previously considered is based on the assuption that all the component parts are correct, i.e.,they have no defect. On the other hand, in a real situation,there might possibly be some component parts which are defective;in this case the assembly sequence of operators previously illustrated cannot be carried on.

The emergency problem is therefore that problem which arises whenever such defective component parts are encountered during an assembly process.

In particular, we consider the emergency problems determined by the following possible reasons, for which the robot cannot be explicitely aware:

1. The holes are defective (in width, filleting, or position) in the bar or in some blocks.

2. The screws are defective (in filleting, lenght or width).

The arising of such an emergency problem forces the robot to interrupt the assembly in the application of the operator SCREW.

As our first approximation, we assume that we have no knowledge about the reason of the emergency. In such situations, we can approach the emergency problem according to three different recovery strategies:

1. Changing of the screw;
2. Changing of the block and of the screw on which there is interruption;

3. Unscrewing of the previous blocks and changing of the bar and of the only block and only screw on which there is interruption.

Each one of these strategies can be viewed as a new problem, i.e. the emergency recovery problem, which has to be solved automatically by the computer, by searching a new sequence of operators and by using, if it is necessary, the operator UNSCREW.

After that a solution for the emergency problem has been reached, the normal assembly must be started again from an appropriate reentry point.

The standard assembly problem is in nature a deterministic program, i.e., it is a program for which the man knows all the informations about the standard assembly problem, and has designed a selfsufficient and consistent solution algorithm. The emergency problem can arise in any point of the standard assembly problem.

The solution of the emergency problem brings up the execution of a non deterministic program, i.e., the computer activity in constructing automatically the solution of the emergency problem.

The reentry point in the control flow of the deterministic program after having executed the non deterministic program, is dependent on the following two factors:

i. The point of the deterministic program where the emergency has arisen;
ii. The way in which the non deterministic problem has been solved.

The way in which the emergency problem can be solved may be selected as well according a global strategy, which takes account of additionals informations which might be easily obtained from the working status of the robot.

These informations can be related to the emergency causes (for example, the point of depth in which the screw has been blocked).

Moreover additionals informations such as the cost of the component parts and the cost of the operations can influence the selection of appropriate strategies for solving the emergency.

Therefore the choice of the solution path for the emergency problem has been shown to be obtainable in a complete automatic way.

A global strategy, when the emergency problem arises can esamine all the informations and can choose a particular problem to be solved in order to reentry in the normal assembly.

The use of such additional informations constitutes a possible direction for further research work.

V. MICROPLANNER SOLUTION OF THE EMERGENCY PROBLEM

In the previous Section we have examined an example of mechanical assembly, and we have discussed the arising of emergency problems, and the automatization of their solution construction.

In this Section we examine the choice and characteristics of goal-oriented languages as the most suitable representation languages apted to describe to the computer both deterministic and non deterministic programs.

Moreover we will illustrate the experimental results which have been obtained on our UNIVAC 1108 computer, by utilising, as our selected goal-oriented language, MICROPLANNER [Sussman et al, (1970)].

We are now going to show how the implementation of the examined assembly problem can be realized, in a very natural way, by a goal-oriented language.

The use of such a language allows us to easily translate the descriptions given in the previous Section for each operator into suitable procedures. The activation and the execution of these procedures provides the construction of the solution of the problem.

We will make use, as our selected goal-oriented language of the MICROPLANNERlanguage, which runs on our UNIVAC 1108 computer and which has been yet used for control of intelligent robots [Gini and Gini (1975 a) and (1975 b)].

We are now going to illustrate, as an example, the description of the MICROPLANNER translation of the operator GRASP. Comments which illustrate such a translation are presented as well.

```
(PUT 'GRASP 'THEOREM '(THCONSE (OBJ POS)
        (CARRYING $&OBJ))
(THGOAL (OBJECT $&OBJ))
(THGOAL (IN $&OBJ  $&POS))
(THGOAL (CLEAR HAND))
(THGOAL (IN HAND $&POS) $T)
(THERASE(IN $&OBJ $&POS))
(THERASE(CLEAR HAND))
(THASSERT(CARRYING $&OBJ))         ))
```

This theorem, of consequent type, is characterized by a pattern, namely (CARRYING $&OBJ) which expresses the result of the application of the theorem, and by a sequence of instructions.

The theorem is written in a way such that "the body implies the pattern".

If we want to demonstrate the truth of the pattern we must demonstrate, i.e. execute successfully, the body of the theorem. Every step of the theorem is an instruction, whose evaluation gives "success" or "failure", as result.

The accomplishment of successive deductions is provided because it is possible with those instructions to call other theorems, and, thus, to set up a chain of deduction steps.

When the problem to be solved is expressed by the goal:

(THGOAL (CARRYING BLOCK) $T)
the system sets up a search among the assertions and successively, among the theorems, which extract, by pattern matching, the appropriate element of the problem base. If there are different assertions or theorems whose pattern matches the one of the goal, the system makes an arbitrary choice, backing up and trying another automatically, whenever the selected one leads to a failure.

However this non determinism reduces the program efficiency, thus it is convenient to limit the scope of the search as far as possible.

The solution proceeds normally in a <u>top-down</u> or goal oriented way. It reduces the problems to subproblems, with the objective of reducing the original problem to a set of solved subproblems. There is also the possibility of <u>bottom-up</u> behaviour. In this case new assertions are derived from the old ones always with the objective of deriving a solution of the original problem.

We can make the following observations which compares the theoretical description illustrated in Section IV with the MICRO-PLANNER program.

1. The parameters of the operators become the variables of the theorem; the theorem can have other variables as well. The variables are indicated by a prefix $&.

2. The pattern indicates that the operator GRASP should be used only in order to achieve a goal (CARRYING $&OBJ). The preconditions of the operators are satisfied by the goals :
   (OBJECT $&OBJ)   (IN $&OBJ $&POS)
   (CLEAR HAND)   (IN HAND $&POS)
   Every goal can be obtained by searching among the assertions or by calling another procedure (the form $T enables

us to call a procedure whose pattern matches the assigned one).

3. The add list and delete list are translated in the Thassert and Therase form.

4. The order of the preconditions is a way of controlling the language and it is very important.

5. The set theoretical description of the operators does not include any indication about the strategy to be used in order to apply an operator. In the MICROPLANNER language the program includes the following components: the order in which the elements of the precondition set must be satisfied, the task for which the theorem can be used (i.e., the pattern), a backtracking monitor which will consider the different choices available to the system.

The result of the activation of the theorem constitutes a plan for the robot. The plan produced consist of a list containing in their proper sequence all movements of the robot and blocks.

The existence and the position of the blocks,bars and screws in their loaders is simulated by the MICROPLANNER assertions :

(IN BLOCK L1)
(IN BAR L2)
(IN SCREW L3)

The assertions are never erased and they simulate the continuous arising of the component parts by the loaders.

It is possible to avoid to enumerate all the objects in the system. That is an advantage deriving from those of MICROPLANNER instead a formal language, like predicate calculus . The presence of apparently contradictory assertions, like

(IN BLOCK L1) and
(IN BLOCK P1)

does not create any difficulties because it is controlled by the program itself.

Some suitable counters are incremented every time a component part is taken or rejected.

When there are no component parts, i.e. the value of a counter is greater then a prefixed number, the system stopsthe assembly process.

In everymoment it is possible to check the number of the component parts employed in the assembly or rejected.

The assembly process is realized by the MICROPLANNER program. If there is no emergency the program is executed in a deterministic way and produces the deterministic assembly illustrated in the previous section.

In Fig. 5.1 is illustrated the example of this deterministic assembly. The sequence of the applied operators is indicated in the listing and commented by some drawings.

When an emergency problem arises, a particular emergency problem is activated on the basis of a prefixed strategy. In Fig. 5.2 there is an example of the emergency problem arisen in screwing the screw in P4: the solution is illustrated from this point.

```
*0:
(Ø6(DRIVER)  ØT)

TRANSLATE (POSIN, L1)
GRASP (BLOCK, L1)
TRANSLATE (L1, P1)
UNGRASP (BLOCK, P1)

                                          =


TRANSLATE (P1,L1)
GRASP (BLOCK, L1)
TRANSLATE (L1, P2)
UNGRASP (BLOCK, P2)

                                          =    =


TRANSLATE (P2, L1)
GRASP (BLOCK, L1)
TRANSLATE (L1, P3)
UNGRASP (BLOCK, P3)

                                          =    =    =


TRANSLATE (P3, L1)
GRASP (BLOCK, L1)
TRANSLATE (L1, P4)
UNGRASP (BLOCK, P4)

                                          =    =    =    =


TRANSLATE (P4, L2)
GRASP  (BAR, L2)
TRANSLATE (L2, P5)
UNGRASP (BAR, P5)

                                        ++±++++±+++++±+++++±+++
                                          =     =     =     =
```

Fig. 5.1 (a)
Deterministic assembly program

```
TRANSLATE (P5,L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P1)
SCREW-IN (SCREW, P1)

                                          ++±+++++±++++±+++++±++

TRANSLATE (P1,L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P2)
SCREW-IN (SCREW, P2)

                                          ++±+++++±++++±+++++±++

TRANSLATE (P2, L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P3)
SCREW-IN (SCREW, P3)

                                          ++±+++++±++++±+++++±++

TRANSLATE (P3, L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P4)
SCREW-IN (SCREW, P4)

                                          ++±+++++±++++±+++++±++

END OF ASEMBLY
TRANLATE (P4, POSIN)
*OVAL : ((DRIVER))
```

Fig. 5.1 (b)
Deterministic assembly program

```
                                          ++±+++++±++++±+++++±++

TRANSLATE (P3, L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P4)
SCREW-IN (SCREW, P4)
      EMERGENCY PROBLEM
TRANSLATE (P4, WASTEBASKET)
UNGRASP (SCREW, WASTEBASKET)
TRANSLATE (WASTEBASKET, L3)
GRASP (SCREW, L3)
TRANSLATE (SCREW, P4)
SCREW-IN (SCREW, P4)
      EMERGENCY PROBLEM
TRANSLATE (P4, WASTEBASKET)
```

Fig. 5.2 (a)
Solution of an emergency recovery problem

```
UNGRASP (SCREW, WASTEBASKET)
TRANSLATE (WASTEBASKET, P5)
GRASP (BAR, P5)
TRANSLATE (P5, RESERVATION)
UNGRASP (BAR, RESERVATION)

TRANSLATE (RESERVATION, P4)
GRASP (BLOCK, P4)
TRANSLATE (P4, WASTEBASKET)
UNGRASP (BLOCK, WASTEBASKET)
TRANSLATE (WASTEBASKET, L1)
GRASP (BLOCK, LI)
TRANSLATE (L1, P4)
UNGRASP (BLOCK, P4)

TRANSLATE (P4, RESERVATION)
GRASP (BAR, RESERVATION)
TRANSLATE (REEERVATION, P5)
UNGRASP (BAR, P5)

                                        ++++++++++++++++++++++++

TRANSLATE (P5, L3)
GRASP (SCREW, L3)
TRANSLATE (L3, P4)
SCREW-IN (SCREW, P4)

                                        ++++++++++++++++++++++++

END OF ASSEMBLY
TRANSLATE (P4, POSIN)
*OVAL : ((DRIVER))
```

Fig. 5.2 (b)
Solution of an emergency recovery problem

The example of emergency recovery problem, which has been automatically solved by the MICROPLANNER interpreter, shows that the solution of such problems is reasonably quite a simple one. Therefore such solutions can be obtained automatically by the machine, while it would have been very cumbersome for the man to go through the examination of all the possible emergency situations, and the subsequent programming of each solution.

VI. CONCLUSIONS

In this paper we have illustrated the utility of the recent research results obtained within artificial intelligence investigation, mainly in problem solving and representation languages, towards the design of sophisticated intelligent robots which are capable of solving emergency recovery problems arising during their standard computer controlled activity.

We have examined a case study related with emergency situations and recovery problems originating during the automatization of the solution of the mechanical assembly problem (and, in particular, of a mechanical system made up by nine component parts).

The research results presented in the paper are part of the activity of the Milan Polytechnic Artificial Intelligence Project (MP-AI Project), and they have been implemented by MICROPLANNER programming on the UNIVAC 1108 of the Milan Polytechnic.

Our future research activity will be devoted to the study of the computational interaction between the deterministic and the non deterministic programming, both on the theoretical and on the practical levels. The investigation of more complex emergency recovery problems in industrial robotics will be examined as well.

We want to acknowledge the useful cooperation and interaction with Dr. D'Auria and Dr. Salmon of Olivetti, for having suggested this research problem, hand for having pointed out its relevance for industrial robotics.

REFERENCES

Amarel, S. On Representations of Problems of Reasoning About Actions. Michie, D. (ed). Machine Intelligence 3, pp. 131-171. American Elsevier Publishing Company, Inc., New York, 1968.

Barrow, H.G. and Crawford, C.F. The Mark 1.5 Edinburgh Robot Facility. Meltzer, B. and Michie, D. (eds). Machine Intelligence 7, pp. 465-480. Edinburgh University Press, Edinburgh, 1972.

Feigenbaum, E. and Feldman, J. (eds). Computers and Thought. Mc Graw-Hill Book Company, New York, 1963.

Fikes, R.E. and Nilsson, N.J. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Proceedings Second IJCAI, London, 1971.

Gini, G. and Gini, M. Control of Intelligent Robots and Goal-Oriented Languages. Proceedings Conference on Applied Robotics, Ceske Budejovice, May, 1975.

Gini, G. and Gini, M. Programming a Robot by Goal-Oriented Languages. Proceedings Third I.C.C.S.-WOGSC, Bucharest, August, 1975.

Mandrioli, D., Sangiovanni Vincentelli, A., and Somalvico, M. Towards a Theory of Problem Solving Marzollo, A. (ed). Topics on

Artificial Intelligence. Springer-Verlag New York, Inc., New York, 1973.

Michie, D., Ambler, A.P., Barrow, H.G., Burstall, R.M., Popplestone, R.J., and Turner, K. Vision and Manipulation as a Programming Problem. Proceedings First Conference on Industrial Robot Technology, Nottingham, 1973.

Minsky, M. (ed). Semantic Information Processing. The M.I.T. Press, Cambridge, Massachusetts, 1968.

Nilsson, N.J. Problem-Solving Methods in Artificial Intelligence. Mc Graw-Hill Book Company, New York, 1971.

Slagle, J. Artificial Intelligence. The Heuristic Programming Approach. Mc Graw-Hill Book Company, New York, 1971.

Sussman, G.J., Winograd, T., and Charniak, E. MICROPLANNER Reference Manual. AI TR 203, M.I.T., Cambridge, Massachusetts, 1970.

Winston, P.H. The MIT Robot. Meltzer, B. and Michie, D. (eds). Machine Intelligence 7, pp. 431-463. Edinburgh University Press, Edinburgh, 1972.