S14-63
12 P.
188183

Using Automatic Robot Programming for Space Telerobotics

E. Mazer, J. Jones, A. Lanusse, T. Lozano-Perez, P. O'Donnell, and P. Tournassoud
Massachusetts Institute of Technology
Cambridge, MA 02139

MJ 700 [illegible]
AC [illegible]

1 Abstract

is described

~~This paper describes~~ the interpreter of a task level robot programming system called Handey. Handey is a system that can recognize, manipulate and assemble polyhedral parts from given only a specification of the goal. To perform an assembly, Handey makes use of a recognition module, a gross motion planner, a grasp planner, a local approach planner and is capable of planning part re-orientation. The possibility of including these modules ~~in~~ a telerobotics work-station is discussed.

2 Introduction

The projected increase in the use of robots in space will make their increased autonomy essential. Direct teleoperation of robots in complicated, repetitive tasks, such as those found in space, can be very tedious. Robot autonomy would relieve the operators from unnecessary fatigue as well as improving reliability and cost [1].

One step towards improving the autonomy of robots consists of having a system capable of planning simple grasp and assembly operations. This goal seems to be a fairly simple and short term objective and yet, it has only been achieved for very well structured environments. The early research on automatic planning of robot operations [2,3,4] focused exclusively on simple situations involving completely modeled environments.

More recent work has now made it possible to design systems working in much more general environments, including environments with significant uncertainty. These environments resemble the type of environment one can expect to find in the vicinity of a space station. This type of environment can include complex parts and six degree-of-freedom revolute arms, all of it modeled with a CAD system. In this paper we describe Handey, a new system that embodies many of the fruits of this more recent research. While Handey still makes strong assumptions about its environments and tasks, we believe that these assumptions are realistic enough so that Handey can contribute towards improving the tele-operation of manipulators.

3 Handey overview

Handey is a task-level [5] robot programming interpreter, that is, the commands given to the system are not robot motions or gripper operations as in VAL or LM [6,7] but simply by describing a certain desired state of an assembly. For example, a full sequence of robot motions, gripper operations and sensor calls can be replaced in a task-level robot programming system by a single statement: *PLACE PART A on PART B*. The interpreter is responsible for planning and carrying out the detailed motions and other actions which lead to this assembly. In its current stage of development, however, Handey makes use of the "perfect world" hypothesis and so, does not take in account problems related to uncertainty or unexpected events. For example Handey does not

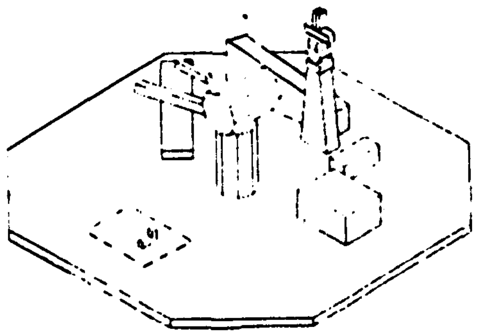


Figure 1: Experimental setup

include a compliant motion planner which would plan assembly strategies in the presence of uncertainty [8,9,10], and does not provide program verification techniques based on uncertainty propagation to patch a predefined plan [11,12]. This remains as future work.

Figure 1 represents a scene used during the development of the system, the doted line on the table shows the limits of the field of view of the laser range finder (this area is called the V-area in the rest of the paper).

Part A is assumed to be located in the V-area. Nothing is assumed concerning this area: part A can be in any location and it can be partially obscured from the range finder by other objects. Except for part A it is not necessary for parts entirely located in the V-area to be modeled in the CAD system. The location of part B is assumed to be known, as are the locations of obstacles in the workspace outside of the V-area, such as the laser-camera device, which we plan to use in the future to find the exact relative position between the gripper and the part.

The user describes the final assembly with a set of relationships between geometric features of parts A and B, then he activates the interpreter. The following is a typical sequence performed by the interpreter to plan the detailed motions and operations necessary to achieve the assembly.

Determining the Final Location. Based on the specified symbolic geometric relationships between parts A and B, a geometric transform representing the relative location between A and B after the assembly is computed.

Recognizing and Localizing Part A. A model-based vision algorithm is executed to determine the location of part A.

Planning a Grasping operation. The grasp planner first tries to find a grasp which permits placing part A directly on part B. If this is not possible, a regrasping operation will be planned later.

Planning the first gross motion. A collision-free path is planned from the initial position to a point on the boundary of the V-area.

Planning a collision free approach. Since the scene in the V-area is not modeled in the CAD system, it is not possible to find a collision free path by the method used for gross-motion planning. Another planner, using the data provided by the range finder, plans a path for the gripper among the obstacles which are located in the V-area.

Planning re-orientation. In many occasions it is not possible to grasp and to assemble the part keeping the same relative position between the part and the gripper. In this case a regrasp operation is planed in a obstacle free portion of the work space.

Planning the remaining gross motions. The regrasp planner produces a number of intermediary locations to be reached by the robot during the re-orientation phase, this phase computes all the paths necessary the perform the regrasping and the path to the final destination.

4 Functional description of Handey

Handey is composed of several modules, most of these modules correspond to substantial pieces of code.

4.1 Experimental Environment

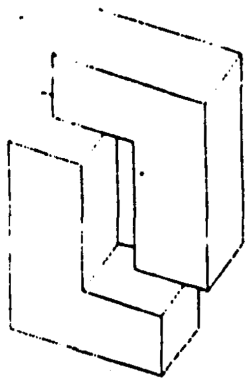
The hardware-dependent software primitives provide a way for the planner to ignore the details needed to operate real-world equipment. It is crucial that these modules be quite good, since they determine the overal system's precision in localizing and moving parts and, as a consequence, will determine the success of the experimentats. Handey makes use of a very limited number of such hardware dependent primitives.

- Range finder calibration: this primitive eliminates non-linearity due to the technology of the laser sensor and scanner hardware. It also determines the scale factors between the sensor and the model.
- Robot Vision calibration: this primitive determines accurately the relative location between the reference frames associated with the robot and the range finder.
- Depth map acquisition: this primitive activates the range finder and returns a depth map in standard units.
- Joint motion: in its current version Handey makes use of one robot motion primitive: "MOVE-JOINT TO (q1,q2,.....q6)" to command a coordinated motion of the robot. The gripper is operated in a binary mode.

4.2 The world modeling system

The world modeling system is used to construct polyhedral models of the parts involved in the assembly including the obstacles (table, ceiling, etc.) and the robot. It is also used to maintain a model of the world during the planning. Once geometric models have been created it is possible to use the following primitives to create, modify or interpret a scene:

- assign a location to a part,
- express the location of a part in a different reference frame,
- affix and unfix parts from the gripper of the robot



- **Face A of Part A is Parallel to Face A of Part B**
- **Face C of Part A is Against Face C of Part B**
- **Face B of part A is Against Face B of Part B**

Figure 2: Describing the final assembly

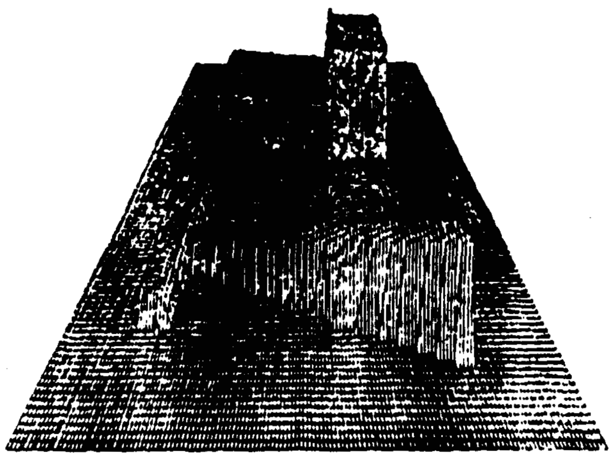


Figure 3: Depth-map produced by the laser range finder

4.3 Computing the final location of part A

The user can describe symbolically the next assembly step by specifying a set of geometric relationships which should hold between geometric features of part A and geometric features of the sub-assembly. Figure 2 represents the set of relationships used to describe the final location of part A.

The set of geometric relationships is then translated into a set of algebraic equations [13]. The system then solvesthe the set of equations to compute the relative position between the part and the sub-assembly. At the present time this feature is not integrated into the on-line version of Handey but works separately.

4.4 Locating part A

The range finder is activated and produces a depth map (figure 3). The map is then processed as if it was an image except that the brightness corresponds directly to the elevation above the work table.

A standard edge operator [15] is run over the image and extended linear segments are identified in the resulting array. Note that this process identifies 3D edge segments,

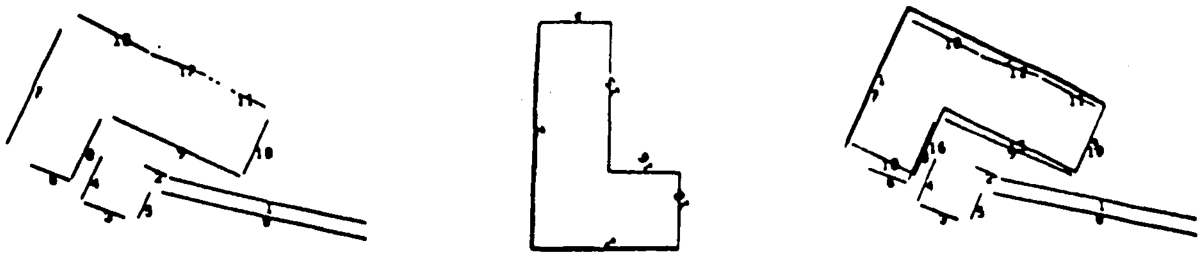


Figure 4. Matching model edges with scene edges

not just their projection in an image. The method used for object localization is a simple hypothesize-verify algorithm based on matching linear segments in the depth map to edges in the polyhedral model of the part. This method is a variation of the method described in Lozano-Pérez and Grimson [16] using edge data instead of face data. Figure 4 represents one matching between edges of the scene and edges of the model.

4.5 Planning collision-free motions

At a number of points in the operation of the system, a collision-free path is required from one specified location to another. Handey uses a simplified version of the path planner described in Lozano-Pérez [17]. This path planner uses the robot's joint space as the configuration space. The version of the path planner used by Handey never computes configuration spaces of dimension greater than three, but it allows motions requiring six degrees of freedom. Essentially, we assume that a path from the start to the goal exists such that the last three joints of the arm retain their starting values until some intermediate point where they change their values at the goal and never change after that. It is easy to construct cases where this assumption will fail, but it works in a large percentage of actual cases.

The actual planning proceeds as follows: An approximate arm model is built in which the last three joints are replaced by a box. This box must be large enough to enclose the last three links, the hand and any object in the hand, not only at their start and goal position but also at the intermediate positions between the two. The three-dimensional configuration space for this model can then be built. We find the closest free point in this configuration space to both start and goal positions, a path is then found between these two free points. Note that the complete robot is guaranteed to be safe along this path, for the whole range of values of the last three joints between the start and the goal. Therefore we can simply interpolate the values of the last three joints between the start and the goal values. Then, we plan a path using the original model of the robot between the free point and the start point itself. We also plan a path from the free point closest to the goal itself. In these two paths the value of the last three joints are fixed. The concatenation of these three paths form the desired path. Figure 5 represents the path found the final motion.

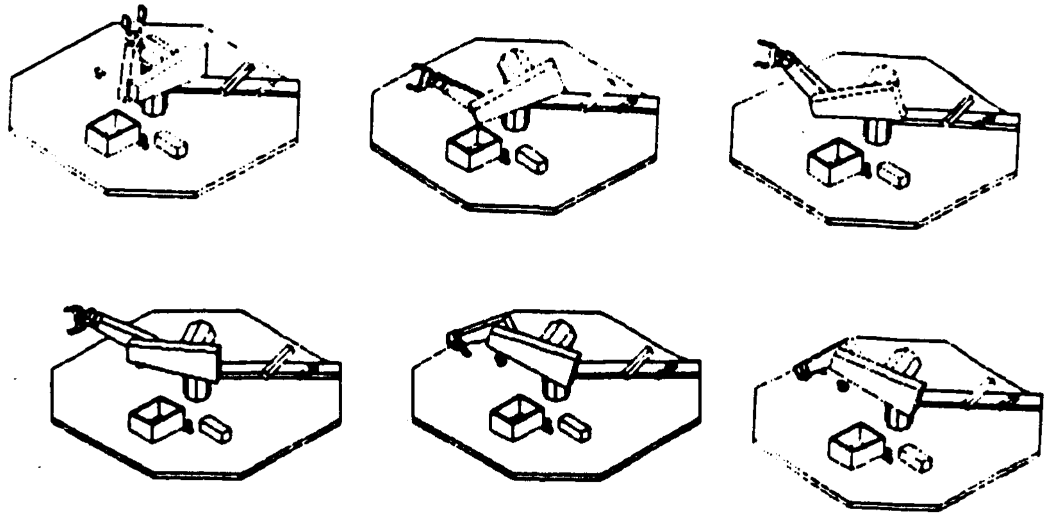


Figure 5 Example of a path generated by the path plann·r

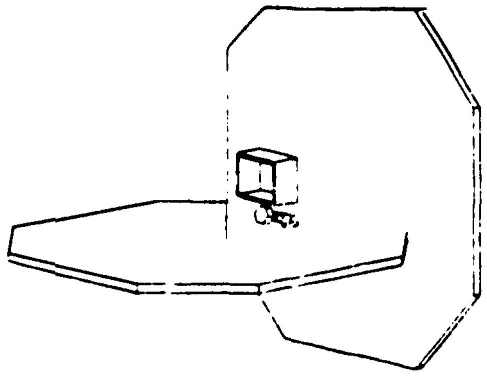


Figure 6 Back-projection of parts

4.6 Grasping

Once the part has been located Handey chooses a grasp. This operation has to take in account several constraints:

- the grasp should be stable,
- a path should exist inside the V-area to reach the grasp,
- the grasp should permit assembling the part once it is in the gripper.

The last constraint can be satisfied by "back projecting" all the obstacles in the V-area. After this operation virtual obstacles exist in the V-area, these obstacles have the same relative location with part A that the real obstacles will have in the final sub-assembly. If one can find a grasp in this environment then it is guaranteed than the grasp will permit assembling the part (figure 6).

4.6.1 Finding a stable grasp

In its current version, Handey uses a grasp planner designed for a gripper equipped with parallel jaws. A future version of Handey will include a more sophisticated planner designed for the three fingers JPL-Stanford-MIT hand [14]. Currently, the planner

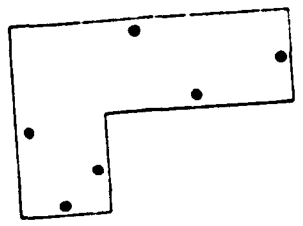


Figure 7 Grasp-points associated with one face

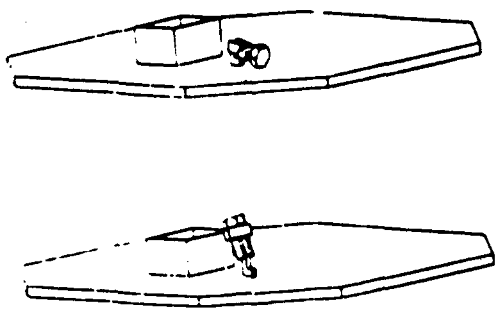


Figure 8 Angular range associated with a grasp

associates two grasps for each locally convex edge of the model. A grasp is defined by one of the face adjacent to the edge and a grasping point. The grasping point is located on the face at a prespecified distance from the edge. Figure 7 represents all the grasping points of one face of part A.

To be valid a grasp should satisfied three conditions:

1. a parallel face should exist and should permit a grasp (mutual visibility [18]) with an allowed distance between the two faces.
2. The gripper should be capable of some rotation around the grasping point (grasping range),
3. an inverse kinematic solution should exist at the grasping point.

The grasping range can be computed using a submodule of the path planner. The grasps are sorted with such grasp permitting the most vertical approach. Figure 8 represents the gripper into two end-points of an angular range.

4.6.2 Planning motions in the V-area

Since no information on obstacles exists in the world-modeling system for the V-area, we must take in account the presence of objects reflected in the depth map. For this purpose we use a planner specialized for planning the motion of the hand in the grasp plane. The grasp plane is a plane parallel and at equal distance from the two faces defining the grasp. When approaching a grasp the fingers remain parallel to the grasp

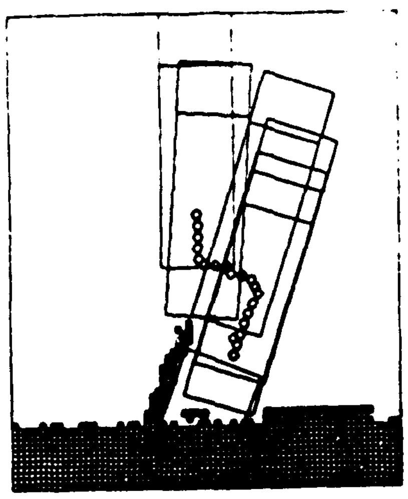


Figure 9: Planning a path in the grasp plane

plane and centered about it but, otherwise, are free to rotate and translate in the plane. Obstacles are projected into this plane to reflect the possibility that they collide either with one of the two fingers, the cross-piece of the hand, or the force sensor.

The planner uses a method loosely modeled on the potential field method for obstacles avoidance [19]. The goal of the grasp planner is to bring a gripping point located between the fingers as close as possible from the grasping point without colliding. The grasping point attracts the gripping point of the gripper while projected obstacles on the grasping plan repel the boundaries of the projected gripper parts (fingers, cross-piece and force sensor). These pseudo-forces are combined in such a way that the gripper is guided toward the goal in X,Y,Θ on the grasp plane. The initial position and orientation of the gripper is given by the grasping planner. Figure 9 represents the evolution from the initial position toward the goal.

4.7 Regrasping

Back-projected objects are artificially added to the depth map so that the final grasp will also permit the assembly. However this may constrain the problem so much that a feasible solution cannot be found. Handey will backtrack among sorted grasps a limited number of times before giving up and trying to find a solution with regrasping. The V-area path planner is then called without back-projected parts and a regrasping operation is planned.

For each part the grasp planner uses two data structures: placements and grasps.

1. A *placement* is a way of placing the part at a particular location on the work table. This location is chosen in an area known to be free of any obstacle, the regrasping will take place in this area. A parameter φ is associated with each placement P. Changing this parameter corresponds to rotating the part on the table around a vertical axis. All the placements P_i are computed automatically by computing the stable faces of the convex hull of the part.

2. A *grasp* is defined by a parameter θ associated to each grasp G. Changing this parameter corresponds to rotating the gripper along an axis perpendicular to the

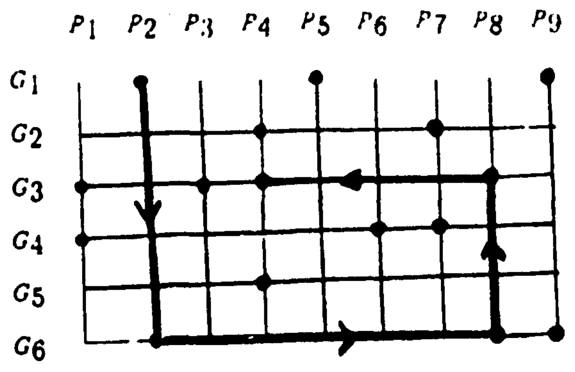


Figure 10: Finding successive placements and grasps

grasping face and containing the grasping point. The set of grasps G_j is also computed automatically.

In order to plan a regrasping operation it is necessary to compute all the vertices of the "regrasping graph". A vertex consists of a data structure defined by a pair $P_i G_j$ having a non-empty $\theta\varphi$ map. A map is built by sampling φ and θ over the interval $(0.0, 2\pi)$. Each φ, θ specifies a single position of the gripper. To be valid, a pair should correspond to a position of the gripper where a solution of the inverse kinematic exists. The map is the set of all the valid $\theta\varphi$ pairs.

There are two operations necessary to perform a re-orientation.

1. Moving from one placement P_i to another P_k. This is possible when a grasp G_j exists, such that the maps associated with $P_i G_j$ and $P_k G_j$ have at least one valid $\theta\varphi$ pair.
2. Changing from one grasp G_j to another grasp G_k. This is possible when a placement P_i exists such that the map associated with $P_i G_j$ and $P_i G_k$ has at least one valid $\theta\varphi$ pair.

The regrasp planner is given an initial position of the part inside the gripper and a final G_f, φ_f grasp which permits the assembly operation. The goal of the regrasp-planner is to find a way through various placements and grasps between the initial and the final grasps. This is represented in figure 10. Horizontal arcs in the figure represent motions of the part from one placement to another and vertical arcs represent motions of the gripper to change the grasp.

5 Applications to Telerobotics

Using telemanipulators in earth orbit has long been recognized as a difficult task. The trend has been to increase the level of commands available to the operator [20]. Prototype telerobotics workstations have been built integrating such high level teleoperation commands. For example, hybrid-control permits the computer to maintain a drill on a given axis while the operator can concentrate controlling the force necessary to perform the drilling operation.

Based on our experience we believe that it is not unrealistic to add some of the capabilities of Handey to such a work-station. As explained in 4.1 the number of primitives used by Handey is fairly limited and should be available in such a workstation anyway.

The most limitating factor being the possibility of including a range finder on the mobile remote servicer (RMS). Once integrated, one can imagine to use a system such Handey in three modes.

- **autonomous mode:** The operator describes the next assembly step. The system computes the sequence of operations and sensor calls to perform the assembly a graphic simulation is presented to the operator before actual execution.
- **partially automatic mode** In this mode the operator asks the system to plan certain portions of the assembly, for example, the system can plan the trajectory to align two axes so that a drilling operation can take place under the control of the operator.
- **monitoring** In this mode the operator first describes what result he expects to achieve.' The system monitors the task and sends an alarm when it detects that the present configuration of the system makes it difficult or impossible to reach the goal. For example grasping a part in a way that the final assembly or an intermediate path is difficult or impossible.

6 Conclusion

Watching Handey performing an assembly is always astonishing and fun, no operator would ever program a task the way our system does. Potentially, we believe that future versions of Handey could be more efficient in performing assembly tasks than typical operators. It could plan paths more effectively in term of time, energy, and safety. It would be less likely to make a mistake such as grasping a part and not being able to move it at a later stage of the assembly because of mechanical stops or collisions. Handey is based on well-establish geometric principles which can make it a robust system. For this reason, it is possible to think of the Handey interpreter as a target system for higher-level planners. The current Handey implementation on a Lisp Machine is still quite slow; it takes approximately 10 min to plan a single pick and place operation. But, we believe that is possible to reduce this time significantly simply by reimplementing it in a machine with fast floating point hardware.

Telerobotics is often presented as feasible alternative to an infeasible autonomous robot. This is certainly true at the present time, but the contrary may be true in the future, that is, the technology necessary to achieve good tele-presence may be more sophisticated than the technology necessary to provide on-board intelligence and dexterity.

7 Acknowledgments

This work was founded primarily by the Office of Naval Research under contract N00014-85-K-0214. Additional support was provided by an NSF Presidential Young Investigator Award (Lozano-Pérez) and the French CNRS (Mazer).

References

[1] D.L. Akin, M.L. Minsky, E.D Thiel and C.R. Kurtzman. *Space Applications of Automation, Robotics and Machine Intelligence System (ARAMIS)- Phase II.* Na-

tional Aeronautics And Space Administration Contractor Report 3734, Volume 1, October 83

[2] L.I. Lieberman, and M.A Wesley. *AUTOPASS: an automatic programming system for computer controlled mechanical assembly*, IBM Journal of Research Development. July 1977.

[3] T.L. Lozano-Pérez. The Design of a Mechanical Assembly system, MIT Artificial Intelligence Laboratory, TR 397, December 1976.

[4] R.J. Popplestone. A.P. Ambler and I.M Bellos. *An Interpreter for a Language for Describing Assemblies*, Artificial Intelligence 14, 1980.

[5] J.C Latombe. *A Survey of Advanced Software for robot manipulators*", AICA congress, Padoue, October 1982.

[6] Unimation, Inc. *User's guide to VAL, a robot programming and control system*, Unimation Inc. February 1979

[7] J.C. Latombe and E. Mazer. *LM: a High-level programming language for controlling manipulators*, 11th International Symposium on Industrial Robots, Tokyo, October 1981.

[8] M.A. Erdmann. *On Motion Planning with Uncertainty*, MIT, Artificial Intelligence Laboratory, TR 810, 1984.

[9] B.R Donald. *A theory of Error Detection and Recovery: Robot Motion Planning with Uncertainty in the Geometric Models of the robot and Environment*, International Workshop on Geometric Reasoning, Oxford University, England July 1986.

[10] S.J Buckley. *Planning and Teaching Compliant Motion Strategies* PHD Thesis, MIT Artificial Intelligence Laboratory, January 1987.

[11] R.A. Brooks. *Symbolic Error Analysis and Robot Planning*, International Journal of Robotic Research, Vol 1, no. 4 December 1982

[12] J. Pertin and P. Puget. *Controle dans le Système de Programmation Automatique Sharp, Gestion des contraintes d'accessibilité et d'incertitudes* IMAG TR 615, June 1986.

[13] E. Mazer. *Geometric Programming of Assembly Robots LM-GEO*, International Meeting on Advanced Software in Robotics, Liège, May, 1983.

[14] N.G. Van-Duc. *The Synthesis of Stable Force-Closure Grasps*, MIT Artificial Intelligence Laboratory, TR 905, May 1986.

[15] J.F Canny *Finding Edges and Lines In Images*, MIT Artificial Intelligence Laboratory, TR 720 May, 1983.

[16] W.E.L. Grimson and T. Lozano-Pérez. *Recognition and Localization of Overlapping Parts from Sparse Data*, MIT Artificial Intelligence Laboratory, A.I Memo 841 June, 1985.

[17] T. Lozano-Pérez. *Motion Planning For Simple Robot Manipulators*, Third International Symposium on Robotics Research, Paris, October 1985.

[18] J. Pertin. *Modelisation du Raisonnement Géométrique pour la programmation des Robots.* These INPG, July 1986.

[19] O. Khatib *Real-Time Obstacle Avoidance for robot Manipulators and Mobile Robots* The International Journal of robotics Research. Vol5. No. 1 Spring 1986

[20] A.K Bejczy. *Robots as Man-extension Systems in Space*, IJAC 9th Triennial World Congress, Budapest, Hongary, 1984.