MSc in Information Technology: Knowledge Based Systems

Intelligent Assembly Systems

The Off-line Planning and Programmimg of the Assembly Task

Recommended Reading: Chapters 1 & [illegible] of *Robot Motion: Planning and Control,* eds Brady et al, MIT Press, 1982 part 1 of Chapter 6

Quotations from Chapter 1, *Introduction.*

> A basic problem in robotics is *planning* motions to solve some previously specified task, and then *controlling* the robot as it executes the commands necessary to achieve those motions. Robots are typically linked structures with motors at the joints between adjacent links that can set the joint to any value in its range. The motors can also set the first and second derivatives of the joint value to any value in the associated range.

A robot.

The assumption is made that planning how to achieve some assembly task is necessarily a question of planning *motions*. This is the assumption that distinguishes the *classical approach* from the *behavioural approach.* — approach taken at Edinburgh.

> The most common type of commercially available programmable robot essentially consists of a feedback controller that moves the manipulator joints independently from the current values to some set of values specified by the user. .... the effect of the motion of one axis upon another is viewed as a disturbance the feedback control system must reject.

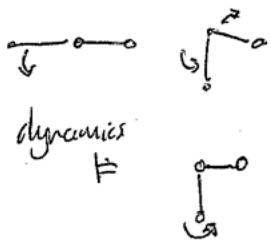
Solution of the inverse dynamics within the computational scope of a robot controller would enable much faster motion and more efficient robots. So would avoiding the inverse dynamics computation by the use of smart adaptive joint controllers that rapidly settled down into a good approximation to the optimum. Current joint-independent controllers are a compromise - the best we can do at the moment.

> The process of computing a sequence of positions, velocities, and accelerations is called *trajectory planning.* (straight-line motion).

Note that this is a kind of planning which has been successfully accomplished in the on-line system.

> Pure position control is, however, inadequate for tasks that involve tight tolerances, or motions that are subject to geometric constraints that are not known exactly.

This not quite true. Useful uncertainty reduction can be accomplished by the use of sliding motions, either pushing the part into place, or permitting the grip to slip. This is often referred as *constrained motion.*



> By *kinematics* is meant the position, velocity, and acceleration relationships among the links of the manipulator. *Statics* is the relationship between the force and torque that the manipulator is exerting on the environment and the forces and torques at the links. *Dynamics* is the relationship between the kinematics and the statics.
>
> ....
>
> There are three types of dynamic torques. The *inertial* torques are torques proportional to the joint accelerations. The *centripetal* torques are torques proportional to the square of joint velocity. Finally, the *Coriolis* torques are proportional to the product of joint velocities from two different links. .... When expanded in this manner, the dynamic equations increase in complexity dramatically with the number of joints.

Finding effective on-line solutions to this problem is still a research topic.

> The role of feedback control is to ensure that a planned sequence of motions and forces will execute correctly in the face of unpredictable errors arising from inaccuracies of kinematic and dynamic models of the robot, limitations of computational precision, and mechanical effects such as static friction and vibration.

Feedback control permits the construction of new types of operators. In the case of these joint position and force feedback controls, they enable goal-seeking operators to be constructed in the [redacted]-line system which will bring the robot to a desired combination of position and force exertion. These operators are decomposed by the joint servos into a sequence of varying torque commands to the motors, i.e., currents. The classical position assumes that higher level task specifications should ultimately be decomposed off-line in a top-down decomposition into these position/force operators. The behavioural position permits the construction of higher levels of goal-seeking operator in the on-line system, which extends the scope of the kinds fo uncertainty coped with on-line from inaccuracies in the model of the robot to inaccuracies in the positions of features of the parts.

> Since it is by no means easy to achieve Cartesian space straight line motion, we might be content to *approximate* Cartesian space straight lines by splines composed of primitive trajectories that are easier, or more efficient, to achieve. .... [This approach] was originated by Taylor, who calls such trajectories *bounded deviation paths*. .... Paul has suggested a method for smoothing the transition at knot points, and Taylor has adapted it to bounded deviation algorithm.
>
> ....
>
> Given a trajectory planner and a position controller, it is relatively simple to construct a system which will move the manipulator to any position in the workspace. .... Unfortunately there is an abundance of manipulator tasks which cannot be adequately expressed as a sequence of positions. Whenever the manipulator is constrained by external positional constraints some alternative mode of control must be used. There are two common classes of motions involving external position constraints: *guarded motion*, used when the manipulator is about to contact a surface; and *compliant motion*, used when the manipulator is in continuing contact with a surface.

move along z with ray flexible.

Compliant motion is the special case of continuing contact with a surface when the robot is firmly holding the part or tool, and thus must *comply* with the motion the surface contact forces upon the part or tool. *Constrained motion* is the special case of continuing contact with a surface when the robot is *not* holding the part or tool firmly, i.e., the robot and the part are free to slide with respect to one another.

rotational compliance ≠ motion compliance

also: "opening doors"

> A controller which can simultaneously control force along certain co-ordinate axes and control position along the remaining co-ordinate axes will be referred to as a *hybrid controller*. .... By discarding the unwanted components of the force and position errors .... [is implemented] the division of the freedoms of the manipulator between a space of force-controlled variables and a space of position-controlled variables.

These are strictly separate. No interaction.

have compliance plane fixed through axis of door hinges.

Quotations from *Task Planning*, Chapter 6 of op cit.:

> Even when the tasks are relatively simple, as are today's industrial robot tasks, the cost of programming a single robot application may be comparable to the cost of the robot itself.
>
> ....
>
> In a task-level language robot actions are specified only by their effects on objects. For example, users would specify that a pin should be placed in a hole rather than specifying the sequence of manipulator motions needed to perform the insertion. A *task planner* would transform the task-level specification into manipulator-level specifications. To do this transformation, the task planner must have a description of the objects being manipulated, the task environment, the robot carrying out the task, the initial state of the environment, and the desired final state. The output of the task planner would be a robot program to achieve the desired final state when executed in the specified initial state. If the synthesized program is to achieve its goal reliably, the planner must take advantage of any capabilities for compliant motion, guarded motion, and error checking. We assume that the planner will not be available when the program is executed. Hence the task planner must synthesize a program which includes commands to access and use sensory information.

Of course, the task planner need not concern itself with commands to access and use sensory information where these are handled inside behavioural operators in the on-line system, just as in current systems there is no need for the planner to concern itself with the use of the joint sensors in the motor control feedback loops.

> We divide task planning into three phases: modelling, task specification, and manipulator program synthesis.

Lozano-Perez defines two distinct approaches to the task specification, the first state-based, the second operator-based. This is the first.

> A task specification is therefore, at first approximation, a model of the robot's world together with a sequence of changes in the configurations of the model components.

....

The world model for a task must contain the following information:

1. geometric descriptions of all objects and robots ....

2. physical description of all objects, ...

3. kinematic descriptions of all linkages;

4. descriptions of robot characteristics, ...

Some of these can be taken for granted.

Models of the task states also must include the configurations of all objects and linkages in the world model. Moreover, the model must specify the uncertainty associated with each of the configurations.

Uncertainty must be specified, because the use of sensors to cope with uncertainty has been defined to be part of the planning task.

Much of the complexity in the world model arises from modelling the robot, which is done only once. Geometric, kinematic, and physical models of other objects must be provided for each new task, however. The underlying assumption is that this information would readily be available as part of the design process of these objects. If this assumption does not hold, the modelling requirement for a task-level specification might dwarf the effort needed to generate a manipulator-level program to carry out the task.

Much current research work, including the Edinburgh Designer System, is aimed at precisely this point - the capture of CAD and design information for use by robot assembly planners, numerically controlled machine tool planners, etc..

A model state is given by the configurations of all the objects in the environment; tasks are actually defined by sequences of states of the world model. The sequence .... needed to fully specify a task depends on the capabilities of the task planner. The ultimate task planner might need only a description of the initial and final state of the task.

Lately some doubt has been thrown on the notion that the output of a task planner would be a fixed assembly sequence, e.g. [Fox and Kempf 85], [Malcolm and Fothergill 87]. There are usually choices in the ordering of the assembly sequence, and it can improve the flexibility and efficiency of the on-line execution of the assembly if these choices are preserved for the use of the on-line system.

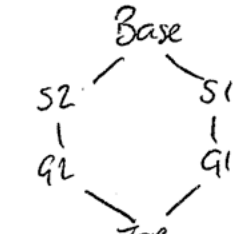
Three alternative methods for specifying configurations [i.e., the positions and orientations of the parts] have been developed:

1. using a CAD system ....

2. using the robot ....

3. using symbolic spatial relationships ....

.... in(shaft, hole1). etc.



Non-linear planner.

A more fundamental limitation [of specifying the assembly task as a sequence of geometric states of the world] is that geometric and kinematic models of an operation's final state are not always a complete specification of the desired operation. One example of this is the need to specify how hard to tighten a bolt during an assembly.

....

need compliant motion to move an object until contact is detected.

The alternative to task specification by a sequence of model states is specification by a sequence of operations. Thus, instead of building a model of an object in its desired configuration, we can describe the operation by which it can be achieved. .... Most operations also include a goal statement involving spatial relationships between objects. The spatial relationships given in the goal not only specify configurations, but also indicate the physical relationships between objects that should be achieved by the operation. Specifying that two surfaces are *Against* each other, for example, should produce a compliant motion that moves until the contact is actually detected, not a motion to the configuration where contact is supposed to occur. For these reasons, existing proposals for task-level programming languages have adopted an operation-centred approach to task specification .... .

RAPT.
align relationship
against "
parallel "

....

Robot programs must tolerate some degree of uncertainty if they are to be useful, but programs that guarantee success under worst case error conditions are difficult to write and slow to execute. Hence, the task planner must use expectations on the uncertainty to choose motion and sensing strategies that are efficient and robust .... . If the uncertainty is too large to guarantee success, then additional sensory capabilities or fixtures may be used to limit the uncertainty .... . For this reason, estimated uncertainties are a key part of task specification.

....

Decomposition into modules.

The synthesis of a manipulator program from a task specification is the crucial phase of task planning. The major steps involved in this phase are grasp planning, motion planning, and error detection.

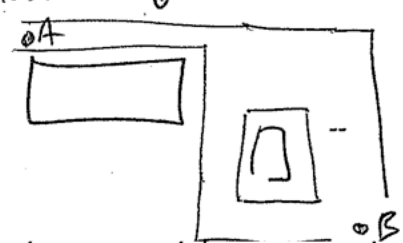
....

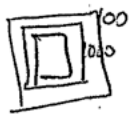
The typical motion required for an assembly operation has four parts: a guarded [?] departure from the current configuration, a free motion towards the destination configuration of the task step, a guarded approach to contact at the destination, and a compliant motion to achieve the goal configuration. During free motion, the principal goal is to reach the destination without collision; therefore planning free motion is a problem in obstacle avoidance, ... Guarded and compliant motions, both of which are sensor-based motion regimes, are used to achieve or maintain desired relative configurations among objects even in the presence of uncertainty in the absolute configurations.

....

The algorithms for robot obstacle avoidance can be grouped into the following classes: .... hypothesize and test .... penalty function .... explicit free space

Corridors:





etc.

CM/24/Jan/1988

closer to object the greater the penalty.

Movement between objects

space ....

....

Most of the proposed obstacle avoidance methods are fundamentally tied to the use of object approximations. .... This assumption does not hold .... where the goal is to bring objects into contact.

convex hulls.

....

There are three [four] principal considerations in choosing a a grasp configuration .... safety .... reachability .... stability .... certainty ....

place "rubber band" around object.

....

The approaches to different aspects of grasping remain to be synthesized into a comprehensive method for grasp planning. (has been done).

....

The discussion above suggests that task planning, for tasks requiring compliant motions, may be done by first finding a collision-free path, on the model *C-surface* for the task .... and then deriving a force control strategy that guarantees that the path of the robot stays on the actual *C-surface*, close to the desired path. This assumes that the robot is already on the desired *C-surface*, that is, in contact with some object in the task .... the problem of achieving this contact .... is the role played by guarded motions.

....

It is not enough to plan the nominal path; the task planner must generate a sensor-based motion strategy that will guarantee that the desired *C-surface* is reached. Four strategies .... can be analyzed using the configuration space diagram [a simple peg in hole]: Tilting .... Chamfers .... Search .... Biased Search ....

This is not simple. Consider the case of a cuboid being brought into contact with a plane. A fixed sequence contact strategy must plan first to achieve contact with a particular vertex, then contact with and edge, and finally contact of plane against plane. The complexities of more general cases are analysed in [Koutsou 85].

The basic idea .... is that motion strategies for particular tasks can be represented as parameterised robot programs, known as *procedure skeletons*. A skeleton has all the motions, error tests, and computations needed to carry out a task, but many of the parameters needed to specify motions and tests remain to be specified. The applicability of a particular skeleton to a task depends, as before, on the presence of certain features in the model and the values of parameters such as clearances and uncertainties.

REFERENCES

B R Fox and K G Kempf, "Opportunistic Scheduling for Robotic Assembly", Proc. IEEE Inter. Conf. on Robotics and Automation, 1985, 880-889.

A. Koutsou, "A Geometric Reasoning System for Moving an Object while Maintaining Contact with Others", Proceedings of the 1st ACM Symposium on Computational

Geometry, Baltimore, June, 1985.

C A Malcolm and A P Fothergill, "Some Architectural Implications of the Use of Sensors", invited paper at NATO Pisa conference Sep 1-5, 1986, in "Languages for Sensor Based Control in Robotics", ISBN 0-387-17665-9, eds Rembold and Hormann, Springer-Verlag, NATO ASI Series F, vol 29, pages 101-124, 1987 (DAI Research Paper 294).

----------O----------