A PLANNING SYSTEM FOR OBJECT RECONFIGURATION WITH ROBOTIC HAND AND THE REAL TIME REALIZATION OF THE MOTION

Gábor Vass * **Shahram Payandeh** ** **Béla Lantos** *

* *Budapest University of Technology and Economics*
Department of Control Engineering and Information Technology
H-1117 Budapest, Pázmány Péter sétány 1/D. Hungary
vass@scoger.iit.bme.hu, lantos@scoger.iit.bme.hu

** *Simon Fraser University*
Experimental Robotics Laboratory, School of Engineering Science
Burnaby, British Columbia, Canada V5A 1S6
shahram@cs.sfu.ca

Abstract: An object manipulation algorithm for a three-fingered dextrous robotic hand mounted on a 6 DOF robotic arm is proposed. The paper is divided into two parts. The first part describes a (currently) off-line planning system for object reconfiguration design based on simulated annealing, while the second part deals with the real time realization of the determined motion path. The presented model-based motion planner algorithm for object re-configuration uses simulated annealing (SA) algorithm for generating the relative motion between the object and the end-points of the fingers.

Keywords: Robotic manipulators, planning, object manipulation

1. INTRODUCTION

The problem of object manipulation with multiple mobile agents is called object re-configuration problem. The manipulation task is stated as the following: given an initial grasp of the object find the motion's trajectories of the manipulating agents to move the object to the desired configuration. During the grasping, contact forces should be exerted on the object by the manipulating agents. Only quasi-static motion is considered in the algorithm.

Manipulation algorithms are frequently applied in robotic research for multiple agents, such as dexterous robotic hands or multiple manipulators. In many cases the cooperating manipulators can be considered as abstract fingers. A survey of robot grasp synthesis algorithms is provided in (Shimoga, 1996). Several aspects of the basic mechanics of manipulating objects are explored in (Salisbury and Mason, 1985).

In the proposed algorithm, the physical parameters of the object have to be known a priori. A learning control method for a geometrically constrained object, of which dynamical parameters (such as the position of the mass center) are unknown is proposed in (Naniwa *et al.*, 1999).

Object manipulation algorithms can be classified by the mode of the relative motion of the fingers and the object. In this paper only rolling and sliding relative motions between the fingertips and the object are assumed. The planner algorithm consists of two main parts: the global and the local planner as follows. The main task of the global planner is to search for the nominal path in the configuration space among static obstacles between the initial and the desired position and orientation of the object. A graph search algorithm is used to generate points (subgoals) in the object's quantized configuration space (Cherif and Gupta, 1997) for the local planner, which attempts to find admissible motion of the manipulating agents in contact with the object between the subgoals.

The detailed motion trajectories of the agents are generated by the local planner between two subgoals. The local planner includes three main parts: a) the kinematics solution (computes the trajectories), b) the force computing algorithm (solves for the contact forces) and c) the generation of the relative velocities between the agents and the object. The motion of agents is dependent on the relative velocities between the agents and the object chosen by the local planner algorithm. The motion sequence of the manipulation system is represented by a relative velocity matrix.

In this paper simulated annealing algorithm (SA) is used at the local planner level to choose relative velocities for the mobile agents (Vass *et al.*, 1999). A good survey to SA is presented by (Ansari and Hou, 1997). The purpose of using SA is to avoid local minima of the motion planning objective function. Usually the objective function has got numerous local minima in case there are various constraints required for defining feasible motions and forces. SA is a general-purpose optimization tool which can be easily used with any constraints in the searching space. Motion planning can be treated as an optimization problem, therefore SA is useful if the energy function of the searching space has many minima. The drawback of SA is low speed, so usually it cannot be used real time. A possible algorithm for accelerating SA is proposed in (Vougioukas *et al.*, 1996). It combines stochastic gradient descent optimization together with SA random motion. The SA algorithm is accelerated by remembering local minima and by a special temperature schedule.

For the implementation of the algorithm, QNX, a real time operation system is used. The robotic hand on which the algorithm will be tested is three fingered, called TUB-PC robotic hand. Every finger has three degrees of freedom, so relatively complicated tasks can be solved. The dextrous hand is mounted on a PUMA 560 arm (see fig. 1).

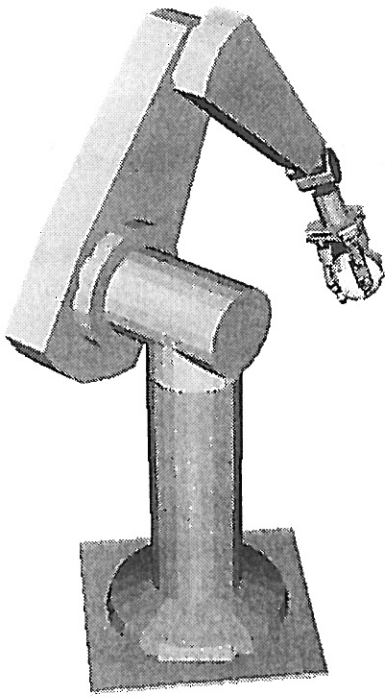

Fig. 1. *The manipulation system: configuration of the arm and the hand.*

The objective of this paper is to present a SA based object reconfiguration algorithm for a hand and arm system. The algorithm can be used to move a known shaped object to a different position and orientation in a known environment. The algorithm is off-line, and it is used in quasi-static cases. The organization of the paper is as follows: in section 2 the background of the problem is given. In section 3 the overview of the planning system is presented, section 4 describes the use of SA for motion planning. The hardware and software architecture is presented in section 5.

2. BACKGROUND OF THE KINEMATIC MODELING AND CONSTRAINTS

The relative motion of two contacting rigid and piecewise smooth bodies is solved by (Montana, 1988) by describing the motion of the contact point on the bodies' surface. The relative velocities between the contacting bodies, and the description of the contacting surfaces are required to solve the equations of the contact motion. The motion of the object and the agents is constrained during the manipulation, the motion constraints which should be satisfied are defined in the following:

Maintaining contact between the object and the fingertips - it is given by the solution of the differential equations of contact motion.

Equilibrium condition - in case of quasi-static motion the resultant force applied on the object has to be zero. $f_{O,ext}$ denotes the resultant vector of the external forces applied on the object.

Rolling constraint - in case the contact motion is rolling or stationary between the fingers and the object, the relative linear (v_i) and angular velocities (ω_i) are: $v_{ix} = v_{iy} = v_{iz} = \omega_{iz} = 0$ where z is the direction of the surface normal. The contact model is point contact with friction. The contact force has to be inside of the cone of friction. In the proposed algorithm this constraint is linearized: instead of cone of friction, pyramid of friction is used.

Sliding constraint - In case the contact motion is pure sliding ($v_{iz}, \omega_{ix}, \omega_{iy}, \omega_{iz} = 0$) the contact force has to be outside of the pyramid of friction. The tangential component of the grasping force at a given contact point should lie in the plane given by v_{ix} and v_{iy} and has to have the same direction and sign.

No collision is allowed among the object, the additional contact surface and the segments of the fingers (collision detection algorithm has to be implemented).

Workspace condition - The contact points have to be in reachable space of the fingers. Inverse kinematics for all components of the composite system (robot and fingers) should be performed.

3. STRUCTURE OF THE PLANNING SYSTEM

The planner algorithm consists of two main parts: the global and the local planners (fig. 2). The task of the *global planner* is to search for the nominal path in the configuration space among static obstacles between the initial and the desired position and orientation of the object. A graph search algorithm is used to generate points (subgoals) in the object's quantized configuration space, the local planner tries to find admissible motion of the fingertips in contact with the object between the subgoals. Collision avoidance between static obstacles and the object is also a task of the global planner.

The detailed trajectories of the motion (path) of the fingertips between two subgoals are generated by the *local planner*. The local planner includes three main parts: a) the kinematics solution (computes the trajectories), b) the force computing algorithm (solves for the contact forces) and c) the generation of the relative velocities between the fingers and the object. Collision detection between the composite system (robot and fingers) and the environment is also performed by the local planner.

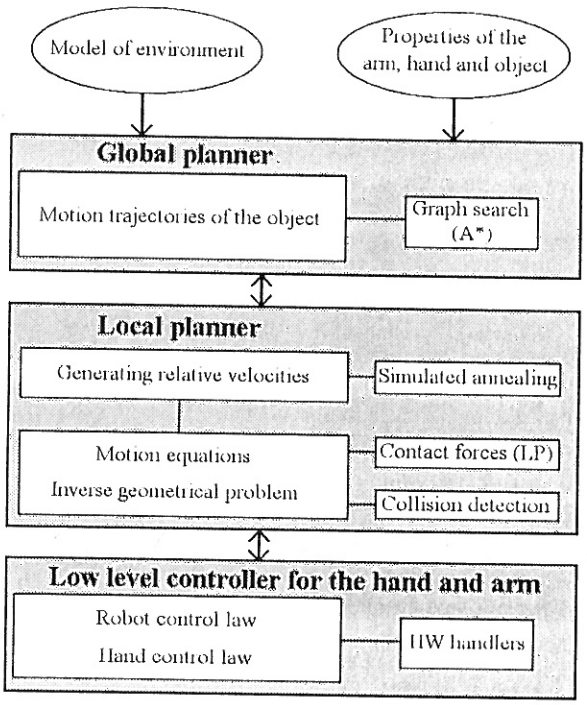

Fig. 2. *The structure of the planning system.*

The motion of the fingers depends on the relative velocities between the fingertips and the object chosen by the local planner algorithm. In the proposed method SA algorithm is used to select the relative velocities. The trajectories of relative motion of the fingertips in the local planner is given by the solution of the differential equations of contact motion (described in section 2) given the relative velocities.

The forces at the contact points are resolved by the solution of a linear program (LP). The aim of this task is to comply with the constraints and minimize the contact forces. The constraints of the LP task are the motion constraints (described in the previous section): the equilibrium constraint and the rolling and sliding constraints (depends on the relative motion at the given contact point).

In the LP task the sum of the normal component of the contact forces is minimized subject to the motion constraints (f_{i_z} denotes the normal component of the contact force at the contact point i):

$$\min_f \sum_i f_{i_z} \qquad (1)$$

Any solution of forces that satisfies the constraints is appropriate because the object's movement is already defined by the constraints and the geometry. The advantage of minimizing the normal component of the contact forces is to save in total energy and to spare the object if it is delicate.

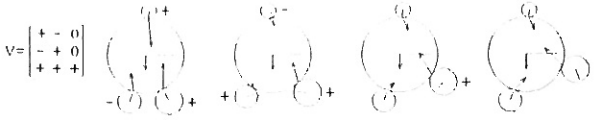

Fig. 3. *A motion sequence of the object and the fingertips.*

4. SA BASED LOCAL PLANNER

Consider the complete motion sequence of the object and the fingers between two subgoals. The motion of the fingertips is parameterized by relative linear and angular velocities. In fig. 3 the first configuration and the object's position and orientation at the last configuration are given by the global planner, the in-between frames are filled up by the local planner. The object is held by three fingers: the orientation of the object and of the fingertips are marked by a normal-direction radial line. The motion of the fingertips is described by the relative velocities, rolling clockwise and counterclockwise in this example, denoted by - and + respectively. The case of no relative motion between the object and the fingertip is denoted by 0. Arrows represent forces. Matrix **V** (introduced below) contains the velocity parameters: the rows and columns are related to the contact points and intervals respectively.

SA algorithm is applied to search for appropriate relative velocity parameters. The initial pose (position and orientation) of the fingers and the object, and the desired pose of the object are given by the global planner as subgoals. The relative velocities are needed to be computed to define the motion of the fingers between two subgoals. The SA method chooses from a large number of sequences using a relative velocity matrix (fig. 4), and accepts the best of those (which is near optimal). SA is preferred to other non-convex nonlinear optimization methods due to its neighboring search feature.

For example let the interval between two subgoals (t_s) be divided into n sub-intervals ($\Delta t = t_s/n$). For every Δt interval, relative velocities should be generated for all the contact points. Consider the complete motion, there are $n * k$ parameters, where k is the number of contact points, hence the complete motion can be parameterized by a $n * k$ size matrix of the relative velocities denoted by $\mathbf{V} \in \mathrm{R}^{n \times k}$. The relative velocities v_{ij} are chosen automatically by the SA algorithm which attempts to optimize the $n * k$ size matrix of the relative velocities considering given constraints (shown in section 2).

Let the variable v_{ij} be defined as the relative velocity at the i^{th} contact point in the j^{th} interval. The values of the variables can be quantized

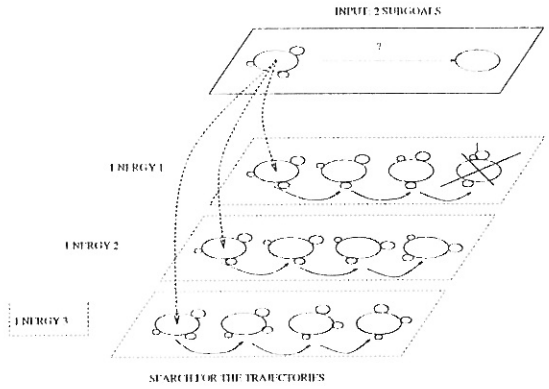

Fig. 4. *Example for generation of the velocity matrix* **V**. *The local planner chooses from different motion sequences which were generated from different velocity matrices. The decision is based on the energy function.*

to decrease the search space, for example $v_{ij} \in \{0, v_{x_{max}}, -v_{x_{max}}, v_{y_{max}}, -v_{y_{max}}, \omega_{x_{max}}, -\omega_{x_{max}}, \omega_{y_{max}}, -\omega_{y_{max}}\}$ (e.g. relative linear and angular velocities along the x and y axes of the contact frame on the object surface). Fig. 3 demonstrates the relative velocity matrix using a 2D example.

In the proposed approach the energy function of the SA algorithm can be a combination of analytical (e.g. equilibrium and no collision) and heuristic expressions (e.g. small contact forces). For example, if the energy function is defined as the number of collisions during the motion then minimizing the energy function will minimize the number of collisions. The energy function of the SA algorithm has several components. The nature of the energy function depends on the problem, it is defined as follows (e.g. equation 2). The components are defined to be:

a) *force equilibrium* (applied on the object),

b) *no collision* of fingers,

c) *the absolute sum of contact forces* for all the contact points in all Δt interval has to be small.

This is useful if the object is delicate or if the objective is to minimize the energy. Given the pose (configuration) of the fingers and the object, equation 1 gives the minimal contact forces. The SA algorithm chooses from different configurations in this case, preferring that configuration which has the minimal absolute sum of contact forces.

$$E(\mathbf{V}) = K_c \sum_{j=1}^{n} c_j + K_e \sum_{j=1}^{n} e_j + K_f \sum_{i=1}^{k} \sum_{j=1}^{n} |f_{ij}| \quad (2)$$

The first sum of equation 2 is the penalty due to collision (the number of collisions multiplied by a constant K_c), the second is the penalty due to no force equilibrium. The third part is the absolute sum of the contact forces. The definitions of the variables and constants are:

$E(\mathbf{V})$ is the energy function to be minimized, $\mathbf{V}$ contains the relative velocities, the variables of the system (parameters to be optimized), K_c, K_e, K_f are positive weight constants for collision, equilibrium and forces respectively; n and k denote the number of intervals between two subgoals and the number of contact points respectively.

$$c_j = \begin{cases} 0 \text{ if there is no collision in the j}^{\text{th}} \text{ interval} \\ 1 \text{ otherwise} \end{cases}$$

$$e_j = \begin{cases} 0 \text{ if there is equilibrium in the j}^{\text{th}} \text{ interval} \\ 1 \text{ otherwise} \end{cases}$$

f_{ij} is the contact force applied at the i^{th} contact point at the j^{th} interval (computed by equation 1). $K_c \approx K_e$, and $K_c, K_e \gg K_f$ because no collision and equilibrium are more important than small contact forces. The algorithm prefers those motion sequences which have the smaller sum of contact forces, less number of collisions and velocity changes and more configurations with equilibrium.

The SA algorithm used is based on an accelerated SA method. It combines stochastic gradient descent optimization together with SA random motion and remembers for local minima. In the temperature schedule, freeze and heat cycles are altered. During a freeze cycle the temperature of SA is zero, thus the algorithm behaves as a downhill search, which converges to the nearest local minimum fast. When a local minimum is reached, the temperature is raised again, so the system moves away from the local minimum.

Remembering for all the explored local minima, the already examined areas in the search space can be avoided. Let $\mathbf{V_{min_h}}$ be the h^{th} explored local minimum of the energy function $E(\mathbf{V})$. In each interval the current velocity matrix $\mathbf{V}$ is compared with all the local minima represented matrices $\mathbf{V_{min_h}}$. If 90% of the corresponding elements of the current velocity matrix and a local minimum matrix are equal, then the current velocity matrix is considered to be close to a local minima. Therefore moving away is desirable, so the temperature is risen for a short period to escape from the local minimum.

5. THE REALIZATION OF THE MOTION

The Department of Control Engineering and Information Technology at the Budapest University of Technology and Economics launched a project on the construction and manufacturing of a new type of dextrous gripper in 1995. The mechanical interface of the TUB-PC hand is rather universal, therefore the hand can be fitted any PUMA or SCARA type robots. The TUB-PC hand is equipped with three fingers, arranged symmetrically on a planar palm. Fingers are identical, each finger has four degrees-of-freedom according to the TR$\|$R$\perp$R$\|$ joint formula. The rotational joints are tendon operated, driven by miniature DC motor-planetary gear units, 12 actuators in total. Drives are fitted to a proximal segment of the robot. Translational slides can be positioned only manually with turn knobs mounted to spindles.

The control system is divided into subsystems. The first subsystem is the *robot control system* consisting of a host PC with extension rack and the power electronics of the axes servo systems (analog PI current controllers with PWM). The extension rack contains three special ARC boards. Every board contains an i386EX processor, a DSP (TMS320C31) for the realization of the robot control algorithms, a tacho-processor receiving the encoder signals of four axes and DAC's for the reference signals of the low level current (torque) control loops of four axes. The boards are connected to the host PC by dual port RAMs. The communication amongst the boards can be performed by using dual port RAMs or CAN-bus. The calibration process of the robot is supported by a PCL analog input card of the host PC. The ARPS robot programming language runs on the host PC under QNX real-time operation system.

The robot control subsystem is designed to be capable to perform different type of robot control algorithms: decentralized cascade position and velocity control, self-tuning adaptive control, computed torque technique, hybrid position and force control, etc. The control system of the robot is described in (Tevesz, 1998).

The *hand control subsystem* of the tendon-driven TUB dextrous hand consists of a PC with extension rack containing an analog input card (PCL-818 with 16 inputs) for the analog position signals and an analog output card (PCL-727 with 12 outputs) for the servo amplifiers of the driving motors of the tendons. The servo amplifiers of the motors are removed to a separate rack. Each finger is driven by 4 tendons. The tendon controller PC is running under QNX and communicates with the host PC by Ethernet. At the actual level the hand control program performs the path design in Cartesian coordinates, the solution of the inverse kinematics, the conversion of the joint variables of the fingers into the position angles of the motors and the coordinated position control of the motors driving the tendons (4 motors are driving 3 finger joints in a coordinated way). Algorithms for the kinematics and dynamics of the hand were discussed in (Ludvig, 1997).

The low-level controller program runs on a PC. Inverse kinematic solutions for all the fingers and the arm are needed, the joint coordinates are com-

puted real time from the Cartesian coordinates of the fingertips. Only the joint coordinates are observed, other sensors are not used yet. In the future contact force sensors will be attached to the fingertips.

The controlling system consists of seven processes, which run using time share, the scheduling of the processor time is handled by the main process. Five processes handle the kinematics of the fingers (one for each finger), the hand control law, and the information exchange between robot and hand controllers. The remaining process reads the off-line, previously planned path (the speed of the SA algorithm is slow, so the global and local planners don't run real time yet). Because of multiple processes, the communication among them have to be handled.

Solving the equations of the contact motion, the position and orientation is given for all the fingertips in the contact points. Thus, for three contact points 3*6 =18 parameters should be satisfied. Since the given manipulation system has only 15 degree of freedom (6 DOF of the arm and 3*3 DOF of the hand), the position and orientation of only two fingers can be assured according to the equations of the contact motion (costing 2*6=12 DOF). However, the third (remaining) fingertip can be set to its generated position (with the remaining 3 DOF). Setting only the position is not enough in case of rolling relative motion, though in case of sliding the orientation of the (spherical shaped) fingertip is irrelevant. Thus, a modification of the original algorithm should be made:

1. Assure that in each subinterval the relative motion of one of the fingers is sliding.

2. Ignore the orientation data of the sliding finger produced by the equations of the contact motion, and use only the position data. In other words, use only the information about the contact point on the surface object (and not on the fingertip).

3. In the LP during off-line motion design, an additional constraints should be used, which assures that only those contact forces may be generated in the contact point of the sliding finger, which can be satisfied by the three DOF of the finger.

6. CONCLUSION

The object re-configuration problem is to find appropriate trajectories for manipulating agents to move the object to a desired position and orientation. The contact motion of the agents relative to the object can be rolling or sliding. Our contribution is a SA based motion planner algorithm for a three fingered dexterous robotic hand mounted on a robotic arm. Since SA is slow, the trajectory was computed off-line. Applying the proposed planning algorithm, a real-time controlling architecture was described. For 6 DOF robot and 9 DOF hand a modified algorithm was introduced, in which the motion of one finger is sliding.

ACKNOWLEDGMENT

Support for the research is provided by the Natural Sciences and Engineering Research Council of CANADA (NSERC) grant No. 611205 and the Hungarian National Research Programs under grant No. FKFP 0417/1997 and OTKA T 029072.

7. REFERENCES

Ansari, N. and E. Hou (1997). *Computational intelligence for optimization.* Kluwer Academic Publishers.

Cherif, M. and K. K. Gupta (1997). Planning quasi-static motions for re-configuring objects with a multi-fingered robotic hand. In: *IEEE International Conference on Robotics and Automation.* pp. 986–991.

Ludvig, L. (1997). *Construction, modeling and itelligent control of robotic hands (in Hungarian).* PhD Thesis, Hungarian Academy of Sciences.

Montana, D. (1988). Geometric phases and robotic locomotion. *International Jornal of Robotics Research* **7**(3), 17–31.

Naniwa, T., S. Arimoto and K. Wada (1999). A learning control method for coordination of multiple manipulators holding a geometrically constrained object. *Advanced Robotics* **13**(2), 139–152.

Salisbury, J. K. and M. T. Mason (1985). *Robot Hands and the Mechanics of Manipulation.* MIT Press.

Shimoga, K. B. (1996). Robot grasp synthesis algorithms: A survey. *IJRR* **15**(3), 230–266.

Tevesz, G. (1998). Architectural problems of the hybrid position and force control system of robots. *Periodica Polytechnica Ser. El. Eng.* **42**(3), 251–262.

Vass, G., S. Payandeh and B. Lantos (1999). Application of simulated annealing for object manipulation with multiple agents. In: *Proceedings of the Second IASTED International Conference, Control and Applications.* pp. 566–571.

Vougioukas, S., M. Ippolito and U. Cugini (1996). Path planning based on accelerated simulated annealing. In: *Proceedings of the Research Workshop of ERNET - European Robotics Network.* pp. 259–268.