III. Conclusion

The determinant of $\boldsymbol{H}$ is the product of the diagonal elements of the upper triangular matrix $\boldsymbol{U}$ [14]. Upon replacing the diagonal elements h_{ii} of $\boldsymbol{H}$ by $(h_{ii}-\lambda)$, the characteristic polynomial of $\boldsymbol{H}$ is the product of the diagonal elements of the companion upper triangular matrix $\boldsymbol{U}(\lambda)$; i.e.,

$$\det[\boldsymbol{H}-\lambda \boldsymbol{I}] = u_{nn}(\lambda)\cdots u_{22}(\lambda)u_{11}(\lambda).$$

References

[1] A. J. Laub, "Efficient multivariable frequency response computations," *IEEE Trans. Automat. Cont.*, vol. AC-26, no. 4, pp. 407–408, Apr. 1981.

[2] C. Van Loan, "Using the Hessenberg decomposition in control theory," in *Algorithms and Theory in Filtering and Control*, Mathematical Programming Study No. 18, D. C. Sorenson and R. J. Wets, Eds. Amsterdam: North Holland, 1982, pp. 102–111.

[3] A. J. Laub, "Algorithm 640: Efficient calculation of frequency response matrices from state space models," *ACM Trans. Mathematical Software*, vol. 12, no. 1, pp. 26–33, Mar. 1986.

[4] A. J. Laub and A. Linnemann, "Hessenberg and Hessenberg/triangular forms in linear system theory," *Int. J. Contr.*, vol. 44, no. 6, pp. 1523–1547 Dec. 1986.

[5] B. N. Datta, "An algorithm to assign eigenvalues in a Hessenberg matrix: Single input case," *IEEE Trans. Automat. Contr.*, vol. AC-32, no. 5, pp. 414–417 May 1987.

[6] G. S. Miminis and C. C. Paige, "A direct algorithm for pole assignment of time-invariant, multi-input linear systems using state feedback," *Automatica*, vol. 24, no. 3, pp. 343–356, May 1988.

[7] C. P. Neuman and J. J. Murray, "Customized computational robot dynamics," *J. Robotic Syst.*, vol. 4, no. 4, pp. 503–526 Aug. 1987.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations.* Baltimore: The Johns Hopkins Univ. Press, 1983, chap. 7.

[9] L. Kronsjo, *Algorithms: Their Complexity and Efficient*, second ed. Chichester, UK: Wiley, 1987, chap. 4.

[10] J. H. Wilkinson, *The Algebraic Eigenvalue Problem.* Oxford, UK: Oxford Univ. Press, 1965.

[11] S. Y. Kung, *VLSI Array Processors.* Englewood Cliffs, NJ: Prentice-Hall, 1988.

[12] J. D. Gardiner and A. J. Laub, "Implementation of two control system design algorithms on a message-passing hypercube," in *Hypercube Multiprocessors 1987*, M. T. Heath, Ed. Philadelphia: SIAM, 1987, pp. 512–519.

[13] P. R. Cappello and A. J. Laub, "Systolic computation of multivariable frequency response," *IEEE Trans. Automat. Contr.*, vol. AC-33, no. 6, pp. 550–558, June 1988.

[14] P. Misra and R. V. Patel, "A determinant identity and its application in evaluating frequency response matrices," *SIAM J. Matrix Anal. Appl.*, vol. 9, no. 2, pp. 248–255, April 1988.

Constraints Identification in Time-Varying Obstacle Avoidance for Mechanical Manipulators

B. H. LEE

***Abstract*—The time-varying obstacle avoidance problem is considered mathematically. The manipulator motion is described in terms of constrained motions that are governed by the environment and the manipulator itself. Various constraints are identified and derived, which are classified into two categories: the environment constraints and the manipulator constraints. These constraints are converted into the reachable path segment at each servo time instant to verify the existence of a collision-free trajectory. Discussions with regard to time-varying obstacle avoidance are also presented.**

Manuscript received September 6, 1987; revised April 23, 1988 and September 11, 1988.

The author is with the Department of Control and Instrumentation Engineering, Seoul National University, San 56-1 Shinrim-Dong, Kwanak-ku Seoul 151, Korea.

IEEE Log Number 8825076.

I. Introduction

Most of the existing off-line path planning schemes that concern obstacle avoidance concentrate on the problem of avoiding fixed and stationary obstacles in a workspace. Since the locations of obstacles are fixed and stationary, the obstacle avoidance can be achieved through collision-free path planning schemes. The path planning schemes usually convert the obstacle avoidance problems to geometric analysis for obtaining collision-free paths.

A time-varying obstacle is generally defined as an obstacle of which the position and orientation depends on time. There exist only few path planning schemes concerning the problem of avoiding the time-varying obstacles. E. Freund [1] analyzed the situation of two robots operating in a common workspace. Since one robot has the priority to move, the other may be considered as a time-varying obstacle. Motion commands for robots are stored in a database so that collision avoidance of two robots can be achieved. Fortune *et al.* [2] developed a useful algorithm for independent but synchronized motion of two Stanford arms. Erdmann *et al.* [3] used a configuration space-time technique to represent the constraints imposed on the moving object. The planner represented the space-time by using two dimensional slices that were then searched for a collision-free path. Tournassoud [4] presented a local method for collision avoidance based on the existence of separating hyperplanes between two manipulators. Its application was then extended to the coordinated motion of two manipulators. Recently, an analytic view to moving obstacle avoidance has been discussed by Lee [5], [6].

In this paper, we try to identify the various constraints in time varying obstacle avoidance for mechanical manipulators. In general, the prior knowledge required for solving the time-varying obstacle avoidance problem includes the description of the obstacle movement, the initial and final location of the manipulator, the physical manipulator constraints and the various environment constraints. The manipulator constraints (MC) include the velocity, acceleration, jerk and torque limitations. The environment constraints (EC) include a motion priority, a traveling time constraint for the manipulator, and the collision constraint between the manipulator and the obstacles. They are described in the following sections.

II. Environment Constraints

A. Time Constraint

The time constraint is a specified time period in which a manipulator must complete the desired motion from one location to another. If the time constraint does not exist and there exists a potential collision, a collision-free trajectory can always be found, as long as the obstacles do not collide with the manipulator at the initial location throughout their movement on a segment. In order to increase the productivity, the minimum-time collision-free trajectory is generally desired for the manipulator movement.

B. Priority Constraint

The priority constraint is defined as the moving priority between the manipulator and the obstacle. If the priority constraint is given to the obstacle, the manipulator involved needs to change its motion strategy for avoiding the potential collision with the time-varying obstacle. Since it is generally difficult and impractical to modify the trajectory of the time-varying obstacle, the priority constraint should be given to the time-varying obstacle. Three possibilities may exist for avoiding a potential collision: changing the path and the trajectory of the manipulator, rearranging the velocity profile of the time-varying obstacle, or modifying the path and trajectory for both the manipulator and the obstacle.

0018-9472/89/0100-0140$01.00

C. Path Constraint

The path constraint is a constraint that restricts the maximum deviation from the predetermined path. In order to simplify computation, we generally assume that the manipulator path and trajectory are composed of straight line segments. Thus, the initial location $[p(k_0), \Phi(k_0)]$ and the final location $[p(k_f), \Phi(k_f)]$ of a manipulator are given for each segment, where $p(k)$ and $\Phi(k)$ specify the Cartesian position and the Euler angle of the manipulator hand at time kT, respectively. Hereafter, we use the time index k to denote the time $t = kT$, where T is the sampling period for servo control of the robot. The manipulator hand is required to move in the Cartesian space along the straight line. The straight line equation that passes through these two locations is described by

$$p(k) = \lambda(k)\big(p(k_f) - p(k_0)\big) + p(k_0) \tag{1}$$

$$\Phi(k) = \lambda(k)\big(\Phi(k_f) - \Phi(k_0)\big) + \Phi(k_0) \tag{2}$$

where $0 \leqslant \lambda(k) \leqslant 1$, and k_0 and k_f are the initial and final discrete times, respectively, and $\lambda(k_0) = 0$ and $\lambda(k_f) = 1$.

D. Collision Constraint

It is assumed that the wrist of a robot is modeled as a sphere. Essentially, the sphere modeling eliminates the orientation problem of the wrist in the computation of potential collision detection at the cost of a very rough approximation. The radius of the sphere is determined from the wrist geometry and the size of the object grasped.

We assume that point $p_O(k)$ is the position of the obstacle at time k, where the subscript O denotes the obstacle. Point $p_r(k)$ is the position of the robot at time k. A collision may occur if the sphere of the robot wrist intersects the sphere of the obstacle during its motion. The distance between two set points on the robot and the obstacle path must be greater than $r_O + r_r$ for collision avoidance, where r_O is the radius of the sphere model of the obstacle and r_r is the radius of the sphere model of the robot wrist [5]. The portion that must be avoided by the robot at time k is the sphere of radius of r_O, centered at the point on the path of the obstacle at time k. That is, the collision avoidance can be achieved when the following condition is met:

$$X_{p_r(k)} \cap X_{p_O(k)} = \phi \tag{3}$$

where

$$X_{p_r(k)} = \{ x \in E^3 | \|x(k) - P_r(k)\| \leqslant r_r \} \tag{4}$$

and

$$X_{p_O(k)} = \{ x \in E^3 | \|x(k) - P_O(k)\| \leqslant r_O \}. \tag{5}$$

The equation of the straight line path of the time-varying obstacle is denoted as

$$p_O(k) = p_O(k_0) + \lambda(k)\cdot\big(p_O(k_f) - p_O(k_0)\big) \tag{6}$$

where $0 \leqslant \lambda(k) \leqslant 1$. Then, the existence of a potential collision is found by solving the equation

$$(r_O + r_r)^2 = \|p_r(k) - p_O(k)\|^2. \tag{7}$$

Replacing $p_O(k)$ by using (6), we have

$$\begin{aligned}(r_O + r_r)^2 = {} & \|p_r(k) - p_O(k_0)\|^2 - 2\lambda(k) \\ & \cdot\big(p_r(k) - p_O(k_0)\big)\cdot\big(p_r(k_f) - p_O(k_0)\big)^T \\ & + \lambda^2(k)\cdot\|p_r(k_f) - p_O(k_0)\|^2. \end{aligned} \tag{8}$$

Equation (8) has three possible solutions: 1) real roots do not exist; 2) two real roots, $\lambda_1^c(k)$ and $\lambda_2^c(k)$, exist ($\lambda_1^c(k) > \lambda_2^c(k)$.); and 3) only one real double root $\lambda_1^c(k)$ exists. When no real root exists, there is no collision between the robots at time k. It is notable that the coefficient of $\lambda^2(k)$ is always positive, and thus the feasible region lies outside the two roots. Thus, the constraint from the collision situation at time $t = kT$ can be written as

$$\lambda_1^c(k) \leqslant \lambda(k) \text{ or } \lambda_2^c(k) \geqslant \lambda(k). \tag{9}$$

If we use $\Delta\lambda(k)$ to denote the increment of $\lambda(k)$ from $\lambda(k-1)$, i.e., $\Delta\lambda(k) = \lambda(k) - \lambda(k-1)$, then the collision constraint $\Delta\lambda(k)$ can be expressed as

$$\Delta\lambda(k) \geqslant \lambda_1^c(k) - \lambda(k-1) \text{ or } \Delta\lambda(k) \leqslant \lambda_2^c(k) - \lambda(k-1). \tag{10}$$

III. Manipulator Constraints

We first derive some useful definitions. The position $p(k)$ and the Euler angle $\Phi(k)$ of a 6-degrees-of-freedom manipulator can be represented by a 6×1 vector and described by

$$\begin{pmatrix} p(k) \\ \Phi(k) \end{pmatrix} = N(q(k)) = \big(N_1(q(k)), \cdots, N_6(q(k))\big)^T \tag{11}$$

where $N(\cdot)$ is a 6×1 nonlinear vector function depending on the manipulator configuration. To initiate the discretized trajectory analysis, let us denote the sampling period for servo control of the robot as T (usually 3 ms $\leqslant T \leqslant$ 20 ms) and $q(k)$ represent the angular displacement $q(kT)$.

A. Smoothness Constraint

All discretized control set points in the joint space must be within certain limits to maintain the smoothness of the trajectory. The smoothness constraint on the joint trajectory set points can be stipulated by a velocity bound (VB), an acceleration bound (AB), and a jerk bound (JB). These three bounds are given [7] as

$$|\dot{q}_i(k)| \leqslant \epsilon_i^v;\ \epsilon_i^v \quad 0 \text{ and } i = 1, \cdots, 6 \tag{12}$$

$$|\ddot{q}_i(k)| \leqslant \epsilon_i^a;\ \epsilon_i^a \quad 0 \text{ and } i = 1, \cdots, 6 \tag{13}$$

$$|\omega_i(k)| \leqslant \epsilon_i^J,\ \epsilon_i^J \quad 0 \text{ and } i = 1, \cdots, 6 \tag{14}$$

where ϵ_i^v, ϵ_i^a, and ϵ_i^J are the ith element of six-dimensional bound vectors for the manipulator (suppose the robot manipulator has six joints). The velocity bound (VB) and acceleration bound (AB) constrain the joint actuators from exceeding the maximum limits of the velocity and acceleration. The jerk bound (JB) reduces wear of joint actuators, and reduces excitation of vibrations. Hence, we impose VB, AB, and JB on the entire trajectory from one trajectory set point to another. Combining (13) and (14), we have

$$\ddot{q}_{i,\min}(k) \leqslant \ddot{q}_i(k) \leqslant \ddot{q}_{i,\max}(k); \quad i = 1, 2, \cdots, 6 \tag{15}$$

where

$$\ddot{q}_{i,\min}(k) = \max\{-\epsilon_i^a, \ddot{q}_i(k-1) - \epsilon_i^J \cdot T\} \tag{16}$$

$$\ddot{q}_{i,\max}(k) = \min\{\epsilon_i^a, \ddot{q}_i(k-1) + \epsilon_i^J \cdot T\}. \tag{17}$$

Similarly, we obtain (18) from (12) and (15) for the joint velocity constraint.

$$\dot{q}_{i,\min}(k) \leqslant \dot{q}_i(k) \leqslant \dot{q}_{i,\max}(k);\ i = 1, 2, \cdots, 6 \tag{18}$$

where

$$\dot{q}_{i,\min}(k) = \max\{-\epsilon_i^v, T\ddot{q}_{i,\min}(k) + \dot{q}_i(k-1)\} \tag{19}$$

$$\dot{q}_{i,\max}(k) = \min\{\epsilon_i^v, T\ddot{q}_{i,\max}(k) + \dot{q}_i(k-1)\}. \tag{20}$$

Then, we can obtain the joint position constraint as

$$q_{i,\min}(k) \leqslant q_i(k) \leqslant q_{i,\max}(k); \quad i = 1, 2, \cdots, 6 \tag{21}$$

where

$$q_{i,\min}(k) = T\cdot\dot{q}_{i,\min}(k) + q_i(k-1) \tag{22}$$

$$q_{i,\max}(k) = T\cdot\dot{q}_{i,\max}(k) + q_i(k-1). \tag{23}$$

We recall

$$p(k) \triangleq p(kT); \quad \Phi(k) \triangleq \Phi(kT). \tag{24}$$

Let us assume that at time $t = (k-1)T$, $p(k-1)$ and $\Phi(k-1)$ are given and that they are within the physical bounds from the straight line requirements, the smoothness and torque constraints. Then, we would like to find the next set point, $[p(k), \Phi(k)]$, such that it is again within the smoothness and torque constraints and must lie on the specified straight line path exactly. If we describe $\Delta p(k)$ and $\Delta\Phi(k)$ from linearization about $q(k)$ in (11)

$$\begin{bmatrix} \Delta p(k) \\ \Delta\Phi(k) \end{bmatrix} \cong [\nabla N(q(k))]\Delta q(k) \tag{25}$$

where $\Delta p(k) = p(k) - p(k-1)$, $\Delta\Phi(k) = \Phi(k) - \Phi(k-1)$, $\Delta q(k) = q(k) - q(k-1)$, and the elements of $[\nabla N(q(k))]$ are found to be

$$[\nabla N(q(k))]_{ij} = \frac{\partial N_i(q(k))}{\partial q_j(k)}; \quad i,j = 1,2,\cdots,6. \tag{26}$$

Combining (1), (2), and (25) at time $t = (k-1)T$ and $t = kT$, we have

$$[\nabla N(q(k))]\Delta q(k) \cong \Delta\lambda(k)\begin{bmatrix} p(k_f) - p(k_0) \\ \Phi(k_f) - \Phi(k_0) \end{bmatrix} \tag{27}$$

where $\Delta\lambda(k) = \lambda(k) - \lambda(k-1)$. If $[\nabla N(q(k))]$ is nonsingular at time $t = kT$, then

$$\Delta q(k) \cong \Delta\lambda(k)Q(k) \tag{28}$$

where

$$Q(k) = [\nabla N(q(k))]^{-1}\begin{bmatrix} p(k_f) - p(k_0) \\ \Phi(k_f) - \Phi(k_0) \end{bmatrix} = [Q_1(k), \cdots, Q_6(k)]^T. \tag{29}$$

Physically, $Q(k)$ is the vector which relates the angular displacement of each joint with $\Delta\lambda(k)$ of a given straight line. Since the servo time interval T is very small, let us assume that, for the joint position at $t = kT$,

$$q(k) \cong q(k-1) + \Delta q(k). \tag{30}$$

Then using (28), we have

$$q(k) \cong q(k-1) + \Delta\lambda(k)Q(k). \tag{31}$$

Combining (21) and (31), we have

$$\Delta\lambda_i^-(k) \leqslant \Delta\lambda(k) \leqslant \Delta\lambda_i^+(k) \tag{32}$$

where

$$\Delta\lambda_i^-(k) = \frac{T\cdot\dot{q}_{i,\min}(k)}{Q_i(k)}; \Delta\lambda_i^+(k) = \frac{T\cdot\dot{q}_{i,\max}(k)}{Q_i(k)} \tag{33}$$

for $i = 1,\cdots,6$, if $Q_i(k) \geqslant 0$. In addition, when $Q_i(k) = 0$, there will be no constraint from joint i, and when $Q_i(k) < 0$, the derivation should be slightly modified to reflect the negative sign of $Q_i(k)$.

B. Torque Constraint

Due to the physical limitations of the joint motors, the robot is limited on how far it can move from current set point to the next on the straight line path. The required joint torques to move the robot are described by its dynamic equations of motion and depend on the instantaneous joint position, velocity, acceleration, and the load that it is carrying. In general, the dynamic behavior of a robot can be computed by the Lagrange–Euler equations of motion as

$$\tau(kT) \triangleq \tau(k) \cong [D(q(k))]\ddot{q}(k) + h(q(k), \dot{q}(k)) + c(q(k)) \tag{34}$$

where $\tau(k)$ is a 6×1 applied torque vector for joint motors, $c(q(k))$ is a 6×1 gravitational force vector, $h(q(k), \dot{q}(k))$ is a 6×1 Coriolis and centrifugal force vector, and $[D(q(k))]$ is a 6×6 acceleration-related matrix. The approximated equality results from the discrete-time approximation of q, $\dot{q}$, and $\ddot{q}$.

If $q(k)$, $\dot{q}(k)$, and $\ddot{q}(k)$ are given, the required piecewise joint torques can be computed by treating the equations of motion as an inverse dynamics problem. In a simplified notation

$$\tau(k) = [D_k]\ddot{q}(k) + h_k + c_k \tag{35}$$

where $D_k = D(q(k))$, $h_k = h(q(k), \dot{q}(k))$ and $c_k = c(q(k))$, and we assume

$$\dot{q}(0) = 0; \ \ddot{q}(0) = 0; \ w(0) = 0 \tag{36}$$

at $k_0 = 0$. Let us further assume that the torques generated from (35) are constrained by limits that are dependent on the joint position (due to the manipulator actuator geometry) and on the joint velocity (due to the back electromotive force terms or other actuator effects) as

$$\tau_{i,\min}(q(k), \dot{q}(k)) \leqslant \tau_i(k) \leqslant \tau_{i,\max}(q(k), \dot{q}(k)) \tag{37}$$

or in a simplified notation as

$$\tau_{i,\min}(k) \leqslant \tau_i(k) \leqslant \tau_{i,\max}(k). \tag{38}$$

Since the joint torque is represented by (35), we have

$$\tau_{i,a}^-(k) \leqslant D_{i,k}\ddot{q}(k) \leqslant \tau_{i,a}^+(k) \tag{39}$$

where $D_{i,k}$ represents the ith row of the matrix D_k, and $\tau_{i,a}^-(k)$ and $\tau_{i,a}^+(k)$ are written as

$$\tau_{i,a}^-(k) = \tau_{i,\min}(k) - h_{i,k} - c_{i,k} \tag{40}$$

$$\tau_{i,a}^+(k) = \tau_{i,\max}(k) - h_{i,k} - c_{i,k} \tag{41}$$

where $h_{i,k}$ and $c_{i,k}$ are the ith element of the vectors h_k and c_k, respectively.

Combining (31) and (39), we have

$$\Delta\lambda_{i,\min}(k) \leqslant \Delta\lambda(k) \leqslant \Delta\lambda_{i,\max}(k); \ i = 1,\cdots,6 \tag{42}$$

where

$$\Delta\lambda_{i,\min}(k) = \frac{T^2\tau_{i,a}^-(k) + D_{i,k}q(k-1) - D_{i,k}q(k-2)}{D_{i,k}Q(k)} \tag{43}$$

$$\Delta\lambda_{i,\max}(k) = \frac{T^2\tau_{i,a}^+(k) + D_{i,k}q(k-1) - D_{i,k}q(k-2)}{D_{i,k}Q(k)} \tag{44}$$

where $\Delta\lambda_{i,\min}(k)$ and $\Delta\lambda_{i,\max}(k)$ are the minimum and maximum constraints for $\Delta\lambda(k)$ from the ith joint torque constraint. Similarly, the same arguments following (33) in the previous section must be applied to the denominator of (43) and (44) to acknowledge the inequality sign of (42).

IV. Conclusion

When there is only a time-invariant obstacle, the existence of a collision-free path only depends on the geometry of the obstacle and the manipulator. However, when a time-varying obstacle exists, the existence of collision-free path depends not only on the geometry of the obstacle and the manipulator, but also on the manipulator constraint and dynamic information of the time-varying obstacle. Hence, determining whether a trajectory satisfies all the constraints or not becomes the difficult part in finding a collision-free trajectory.

Since the maximum $\Delta\lambda(k)$ results in the maximum local speed at $t = kT$, it is desirable to formulate the problem as

$$\text{find} \max \Delta\lambda(k); \quad k = 1, 2, \cdots$$

subject to the smoothness constraint

$$\Delta\lambda_i^-(k) \leqslant \Delta\lambda(k) \leqslant \Delta\lambda_i^+(k) \tag{45}$$

and the torque constraint

$$\Delta\lambda_{i,\min}(k) \leqslant \Delta\lambda(k) \leqslant \Delta\lambda_{i,\max}(k) \tag{46}$$

where $i = 1, 2, \cdots, 6$, and the collision constraint

$$\Delta\lambda(k) \geqslant \lambda_1^c(k) - \lambda(k-1) \text{ or } \Delta\lambda(k) \leqslant \lambda_2^c(k) - \lambda(k-1). \tag{47}$$

It is notable that the time constraint is converted to find the maximum $\Delta\lambda(k)$ and the priority constraint is given to the time-varying obstacle. Also, the path constraint is embedded in the collision constraint.

As indicated in Section II-A, a collision-free trajectory always exists if there is no time constraint, which can be interpreted in terms of the aforementioned formulation as follows. Since there is no time constraint, the departure time of the robot can be sufficiently delayed enough to ignore the collision constraint in (47), and the minimum time trajectory will suffice as long as the constraints in (45) and (46) are not violated.

If there exists a time constraint, the existence of the common region for $\Delta\lambda(k)$ in (45), (46), and (47) is not completely guaranteed. In fact, even though we neglect the torque constraint by the assumption that the robot is not operated at the maximum velocity and acceleration of the joint motors, there is no direct relationship between the smoothness constraint in (45) and the collision constraint in (47). Thus, the existence of a collision-free trajectory is questionable.

As a matter of fact, if we have the whole motion information for the time-varying obstacle with respect to the manipulator path, and measure the necessary deviation for the manipulator path from the original path enough to avoid the obstacle, then there will be no collision between the obstacle and the manipulator. It is interesting to note that if the time constraint becomes weaker, then the path deviation will become smaller. Ultimately, the deviation will become zero when there is no time constraint.

V. References

[1] E. Freund, "Hierarchical control of guided collision avoidance for robots in assembly automation," in *Proc. 4th Int. Conf. on Assembly Automation*, Japan, Sept. 1983, pp. 91–103.

[2] S. Fortune *et al.*, "Coordinated motion of two robot arms," in *Proc. IEEE Conf. Robotics Automation*, San Francisco, CA, 1986, pp. 1216–1223.

[3] M. Erdmann *et al.*, "On multiple moving objects," *Proc. IEEE Conf. Robotics Automation*, San Francisco, CA, 1986, pp. 1419–1424.

[4] P. Tournassoud, "A strategy for obstacle avoidance and application to multi-robot systems," in *Proc. IEEE Conf. Robotics Automation*, San Francisco, 1986, pp. 1224–1229.

[5] B. H. Lee, "A view to moving obstacle avoidance for robot manipulators," *IEEE Conf. Syst. Man Cybern.*, Atlanta, GA, Oct. 14–17, 1986, pp. 476–481.

[6] B. H. Lee and C. S. G. Lee, "Collision-free motion planing of two robots," *IEEE Trans. Syst. Man Cybern.*, vol. 17, pp. 21–32, Jan. 1987.

[7] B. H. Lee, "Algorithmic approach to straight line trajectory planning for mechanical manipulators," in *Proc. Amer. Contr. Conf.*, Seattle, WA, June 18–20, 1986, pp. 121–126.

Group Decision Support with MOLP Applications

HSIAO FAN WANG and SHENG YUAN SHEN

Abstract —**The paper develops a group-decision support-system (GDSS) to aid a group of members to find a consensus solution in a systematic way. There are two groups of participants: One group consists of professional staff and the other of decisionmakers (DM's). Through the participants' reciprocation and complementation, a three-step decision process utilizing a general multiple objective linear problem is designed. The three steps are the generation of alternatives, the group-decision operation and the evaluation of the final result. The GDSS is evaluated on four criteria: complete information, full participation, equity principle and economic decision time. Based on these criteria, the proposed system has shown its potential to improve the quality of decisionmaking and to shorten the time required for the process. Further research and refineness are also suggested.**

I. Introduction

Group-decisionmaking involves a group of members, h, with different preference structures, p_l, $l = 1, \cdots, h$, who try to find a consistent collective decision, p_G, from a set of alternatives $\{x_j, j = 1, \cdots, n\}$. That is, $p_G = f(p_1, \cdots, p_l, \cdots, p_h)$. Here the function f represents the group decision rules that should be effective and efficient so that a "good" alternative can be selected within a reasonable period of time and the conflict existing among the members can be reduced to the minimum. Several researchers, e.g., [1], [5], [6], [11], [18], [19], [26], [31], have been involved in the relevant research.

While Arrow [1] and Keeny and Kirkwood [19] discussed the existence of group utility function and derived a social welfare function (SWF), Yu [31] then applied the concept of ideal point to propose a method of using distance norms to measure the degree of group regret. Both approaches articulate the preferences of the decisionmakers (DM's) prior to solving any multiobjective problems [14]. According to these approaches, once the DMs' worth or preference structures are built into the formulation of a mathematical model for a multiobjective problem, any changes of preferences cannot affect the result. However, this study indicates that the worth or preferences of DM's apparently are dynamic, particularly in group decisionmaking. Once more information from different DM's is communicated, the original opinions are often changed and so are the preferences. Besides, group utility function is derived from the individual utility functions, which are difficult to define, and the independence condition is difficult to satisfy [10].

Seo and Sakawa [26], and Blin and Whiston [2], by applying the fuzzy set theory, obtained a single attribute group utility function from the single-attribute individual utility functions. They also applied a multiattribute utility theory to obtain an implicit group utility. During the process, the group preference on the attributes is first ordered by the defined membership function, then the preference order is fed back to the DM's to determine the weights of the attributes that are further aggregated to form the group weights. Because this approach provides a feedback procedure, the DM's have an opportunity to change their preferences. In addition, there is no explicit individual utility function required in the process. However, in this approach, there is a problem with finding the proper definition of a membership function. If an improper function is adopted, it might distort the relative values of the attributes. Besides, the

Manuscript received January 30, 1988; revised September 11, 1988.

The authors are with the Department of Industrial Engineering, National Tsing Hua University, Hsinchu, Taiwan, Republic of China.

IEEE Log Number 8825074.

0018-9472/89/0100-0143$01.00 ©1989 IEEE