INTELLIGENT TRAJECTORY GENERATOR FOR ROBOT ARMS

مولد المدارات الذكى لأزرع الروبوت

BY

Dr. Yehia Mohamed Enab;

Dep. of Automatic Control & Computer Engineering;

El-Mansoura University, EGYPT

الخلاصة — يقدم هذا البحث طريقة جديدة لتوليد المدارات التى تتبعها أزرع الروبوت النقالة أثناء الحركة . وتوصف هذه المدارات كمتسلسلة زمنية من زوايا مفصلات الزراع المختلفة . وتعتمد الطريقة الجديدة على استخدام العمليات المنطقية والاستدلالات العقلية فى تحديد قيم هذه الزوايا . وهذا يشبه الى حد كبير الطريقة التى يفكر بها الانسان الطبيعى أثناء تدريب الروبوت على القيام بمهمة معينة . ونظرا لأنّ الطريقة لا تعتمد على استخدام العمليات الحسابية المعقدة كحساب الدوال المثلثية ومعكوس المصفوفات فانها يمكن أن تستخدم خلال الزمن الحقيقى . وقد قدم البحث مجموعة من تجارب المحاكاة على الحاسب الآلى لدراسة جدوى هذه الطريقة وامكانية تطبيقها .

Keywords:

Robots, Trajectory tracking, Intelligent Inference, Computer Simulation.

ABSTRACT

The paper introduces a new trajectory tracking algorithm used for robot manipulators. This algorithm determines the different joint angles for each set point along the desired trajectory using a set of rules similar to those used in modern expert systems. These rules are designed according to the geometrical structure of the arm and the experience of operation. These rules do not require matrix inversion or trigonometric function calling as in the classical trajectory generators used with robot manipulators. The logical operations are used extensively for deciding the values of joint angles. A series of computer simulations are used to clarify the performance of the algorithm.

(1) INTRODUCTION

Robots are basically simple positional machines. In performing useful tasks, the robot arm must move its end-effector (or gripper) from one point in the space to another point. There are two important problems associated with the robot motion, "path planning" and "trajectory planning". The goal of the path planner is to find a sequence of points expressed in task coordinates (usually Cartesian) which should be traced by the end effector such that no collision occurs between the robot and the other obstacles in the work-space area. The goal of the trajectory planner is to find a time sequence of intermediate robot arm configurations (the angles and displacements of the arm joints) such that the end-effector follows the desired path provided by the path planner. Trajectory planning may include the satisfaction of a set of constraints on its position, velocity, and acceleration at a number of points along the desired trajectory either in joint or Cartesian coordinates. The configurations, possibly together with their first and second time derivatives, are then shipped off in succession to the servo mechanisms controlling the actuators that actually move the arm. The space curve traced by the end effector is called the "path" of the "trajectory".

This paper addresses the "trajectory planning problem." A new technique is introduced to generate the trajectory. The proposed algorithm can be used for the on-line trajectory generation or (tracking) by making use of the available measurements provided by the arm sensors. This occurs through the use of a properly set of rules derived from the experience of operating the specified robot. The algorithm is suitable for the robots which have unprecise kinematic model and the flexible arms which suffers from bending under the different loads. A computer simulation experiments have been carried out to test the performance of the proposed algorithm.

(2) LITERATURE REVIEW

There are two common approaches to describing and planning trajectories, "constraint satisfaction" approaches and "given paths" approaches. In the first class, the designer implicitly describes the path and trajectory followed by the arm, by specifying a set of constraints on its joint position, velocity, and acceleration at a number of configurations along the desired trajeçtory. The trajectory planner chooses a class of parameterized trajectories that satisfies the constraints. As an example Mujtaba [1] compares several different trajectories, including a cosine, a quintic, and the sum of a sine function and a linear trajectory. He restricts attention to the following constraints: zero initial and final velocity, and initial and final acceleration that is either zero or of maximum absolute value. The calculus of variations [2] and modern control theory [3], provide a more general approach to constraint satisfaction, even when the number of constraints and parameters is different. These trajectories can be executed efficiently. The main disadvantage is that the Cartesian path and the rotation curve followed by the end-effector are not given explicitly. In particular, it is often non-trivial to guarantee that the end-effector stays within the workspace area.

The "given paths" approaches address the basic shortcoming of the previous class. The trajectories planned here constrain some point on the arm, typically the end-effector, to follow a straight line path in Cartesian space from a given source configuration to a given destination [5,6,7]. The rotation curve is treated separately. There are several reasons why a programmer or spatial planning system might choose Cartesian straight line paths for the end-effector to follow. First, straight line paths are relatively predictable and easily visualised. Second, they give the shortest path, though not necessarily the fastest, between a source and a destination. Third, straight line paths occur in many applications, for example, inserting a pin, tracking a conveyor belt, or welding a seam of a rectangular plate [1]. Finally, uniform motion along straight line paths minimizes inertial forces on the hand and any object it might be carrying [6,7].

Recently, Bejczy et. al. [8] introduced a general theoretical frame for task space motion planning where tools from differential geometry are used. An optimization technique [9] is used to control the kinematics of a robot, even if bending owing to flexibility of the robot links occurs.

(3) THE ALGORITHM

3.1 Problem Formulation

From the kinematical point of view, the position of the end effector is described in Cartesian coordinates by choosing a fixed frame $O_oX_oY_oZ_o$ associated with the robot base (see Fig. 3.1). This position can be represented by the vector $\underline{P} = [p_x \; p_y \; p_z]^T$ pointing from the origin of the base frame to the origin of the hand frame (usually located at the center of the fully closed fingers). Usually the components of this vector can be determined from the knowledge of the generalized joint displacements which are the components of the configuration vector $\underline{Q} = [q_1 \; q_2 \; \dots \; q_n]^T$ (where n is the number of degrees of freedom for the manipulator). In general the position of the gripper can be described as

$$\underline{P} = \underline{f}(\underline{Q}) \qquad (1)$$

where $\underline{f}$ is (3x1) vector transformation which is a highly nonlinear and contains many calls for trigonometric functions.

Robot task is usually described as a time sequence of the position vector $\underline{P}$. Thus, it is required to determine the inverse of the above transformation at each set point on the trajectory to find the corresponding configuration vector $\underline{Q}$ i.e.,

$$\underline{Q} = \underline{f}^{-1}(\underline{P}) \qquad (2)$$

In general f^{-1} is very tedious to be computed on line and its solution is not unique. This is the main reasons that the known methods of inverse kinematics [10] are not used directly to solve this problem.

3.2) The Methodology

Figure 3.2 shows the block diagram of the "Intelligent Trajectory Generator (ITG)" eystem. The basic idea in the proposed system is to construct a set of rules in the general format "IF Antecedent THEN Consequent" of "IF Situation THEN Action", to determine the joint angles required to feed the robot controller to displace the gripper position from its actual one to the desired one. The antecedent or situation is a logical expression contains relational and logical operators. The set points coordinates along the desired trajectory and the sensor measurements are feeded into the ITG. Sensors may be a set of Light Emitting Diodes (LED's) which can be fixed at the gripper, elbow, and the other important points along the robot body. A vision system is used to detect the three-dimensional position of these important points. The classical position sensors such as "digital position encoder" which give the joints angles can still be used without the camera system. In this case the kinematical model of the arm should be used to calculate the position of the important points mentioned previously. In general, the choice of these points are decided by the rules designer according to his experience of operating the arm.

The value of the antecedent is TRUE or FALSE. The action is to decrease, increase, or maintain the value of certain joint angle(s). As an example of the rule used in the intelligent trajectory generator is

IF ($x_d > x_g$) AND ($y_e < 0$) THEN ($q_1 = q_1 + S$).

where the subscript d means desired coordinate, g means gripper coordinate and e means elbow coordinate. Here the angle q_1 is increased by a certain value (S) if the logical value of the situation is TRUE.

The value of S can be determined in either two ways:

i) A fixed step is used. The desired position is reached after few iterations of applying the rules. The accuracy of positioning increases as S becomes small but the number of iterations will be increased and vise versa. When using one step a head concept in which the joint angles are determined in one iteration then, if S is very small the arm moves slowly and it may be unable to follow the desired trajectory. If S is large the arm moves faster but it may precede or vibrate around the desired trajectory points. In fact the value of S controls the smoothness of the trajectory tracking. The suitable value of S can be chosen such that a compromise occurs between the velocity of tracking and its accuracy. It is easy to recognize that the suitable fixed value of S is trajectory dependent.

ii) A variable step is used. This step can be chosen to be proportional to a certain defined error. As an example $S=f*|x_d - x_g|$, where f is constant proportionality factor (chosen imperically), and |..| is the absolute operator. In this case the value of the émpirical constant f is less dependent on the trajectory, instead it is more dependent on the geometrical dimensions of the arm links which are constant for the same robot. This fact is clarified experimentally as will be shown later.

Another important aspect of using these rules is the method of scanning them. For each set point along the desired trajectory there should be a RULE 0 to applied at first. This rule examines if the desired set point is inside or outside the work space area. If this point belongs to the work space area the other rules are scanned one by one, only one rule will be matched at each time.

3.3) Single Link Manipulator

Figure 3.3 shows a single link manipulator. The inputs to the intelligent trajectory generator (ITG) are the desired coordinates (x_d, y_d) and the gripper coordinates (x_g, y_g). The output of the engine is the joint angle q. The first thing to be done by the ITG is to test if the desired location lays within the work space area or not. The robot work space area is a circle of radius equal to the arm length. The rule designer can use his common sense and experience with arm operation to formulate the suitable rules. As an example, it is easy to decide that the angle q should be incremented (counterclockwise rotation) if the recent configuration is under the desired one and q should not be varied if these two configurations are identical (see Fig. 3.3). Also, q should be decreased (clockwise rotation) if the arm configuration is over the desired one. All the possible combinations between the different types of coordinates are studied and the rules used in determining the joint angle q are given in Table 1.

TABLE 1

RULE 0: (work space area constraint)

IF $|(x_d^2 + y_d^2) - L^2| > w$ THEN print "out of reach"

RULE 1: IF $(|x_d - x_g| \le w)$ AND $(|y_d - y_g| \le w)$

THEN $q(t+1) = q(t)$

RULE 2: IF $\{[(x_d \le x_g)$ AND $y_d \ge 0]$ OR $[(x_d > x_g)$ AND $y_d \le 0]\}$

THEN $q(t+1) = q(t) + S$

RULE 3: IF $\{[(x_d > x_g)$ AND $y_d \ge 0]$ OR $[(x_d \le x_g)$ AND $y_d \le 0]\}$

THEN $q(t+1) = q(t) - S$

where w is a small positive number and t is the discrete time. The value of S may be chosen as a constant positive step. Also S can be varying step given as $S = f * |x_d - y_g|$, and f is a constant proportionality factor chosen by the designer.

3.4) Two Links Manipulator

Figure 3.4 shows two links manipulator drawn with two configurations I and II. For the trajectory shown and the arm in configuration I, the deviation in the x-coordinate between the desired and gripper positions can be reduced either by increasing the joint angle q_1 or the joint angle q_2 or both. To reduce the error in the x coordinate with the arm in configuration II, the angle q_1 should be increased and the angle q_2 should be decreased (which is different from the above case). It is evident that the main difference between the two arm configurations is that y_g in configuration I is higher than y_e, while the inverse situation occurs in configuration II.

In fact it is recommended to use only the minimum amounts of angle modifications to satisfy the goal in order to reduce the amount of work given by the servo mechanism of the manipulator. Another reason for reducing the number of modified joint angles is to reduce the error which can be accumulated as a result of successive angle modifications. A similar analysis can be derived for reducing the deviation in the y-coordinate.

All the different possible configurations have been investigated and Table 2 shows the rules used for modifying the joint angles suitably. In constructing these rules q_1 is used to minimize the deviation in x-coordinate, and q_2 is used to minimize the error in y-coordinate. The equality conditions of both the gripper and desired set points are excluded hence the error between the corresponding coordinates will be zero and no updating for the joint angles occurs.

TABLE 2

```
RULE 0: (work space area constraint)
     IF $|L_1-L_2| < (x_g^2+y_g^2)^{.5} < |L_2+L_1|$   THEN go to RULE 1
                                                  ELSE print "out of reach"
RULE 1:  IF $[(x_d>x_g)$ AND $(y_d>0)]$ OR $[(x_d<x_g)$ AND $(y_d<0)]$
         THEN $q_1 = q_1 - f*|x_d-x_g|$
RULE 2:  IF $[(x_d>x_g)$ AND $(y_d<0)]$ OR $[(x_d<x_g)$ AND $(y_d>0)]$
         THEN $q_1 = q_1 + f*|x_d-x_g|$
RULE 3:  IF $[(y_d>y_g)$ AND $(x_g>x_e)]$ OR $[(y_d<y_g)$ AND $(x_g<x_e)]$
         THEN $q_2 = q_2 + f*|y_d-y_g|$
RULE 4:  IF $[(y_d>y_g)$ AND $(x_g<x_e)]$ OR $[(y_d<y_g)$ AND $(x_g>x_e)]$
         THEN $q_2 = q_2 - f*|y_d-y_g|$
```

3.5) Three-Links Manipulators

Figure 3.5 shows the three-links manipulator used. Such structure is familiar in many industrial robots such as PUMA robot and experimental ones such as HRA934 hydraulic robot arm. The suitable rules can be derived directly from the rules used for the single link and two links manipulators with the following modifications:
i) The projection of the arm on the horizontal plane $O_oX_oY_o$ is dealed as a single link arm. The angle q_1 is modified to reduce either the x or y coordinate using the rules given in Table 1.
ii) The plane containing the three links can be considered as the plane containing the two links arm studied previously. The rules given in Table 2 can be used now. The x coordinate is replaced by $x_g^2+y_g^2$ and the y coordinate is replaced by z_g. According to this modification the angle q_2 is used to reduce the deviation between the quantity $x_g^2+y_g^2$ and the quantity $x_d^2+y_d^2$ and q_3 is used to reduce the deviation in the z coordinate.

(4) EXPERIMENTAL WORK

A series of computer simulations were performed to test the availability and performance of the proposed "intelligent trajectory tracking algorithm". These simulations were performed using two and three link manipulators. Only the position of the end effector is tracked, the orientation of the gripper is assumed to be constant through the whole trajectory. It is assumed that the initial configuration of each manipulator is fully extended along the vertical direction.

The performance index used to evaluate the validity of the algorithm is the Euclidean distance (E_i) representing the deviation between the desired gripper position and the actual one at i th set point along the trajectory. The average deviation along the trajectory E_{av} is defined as

$$E_{av} = (1/n_t) \sum_{i=1}^{n_t} E_i$$

where n_t is the number of set points along the desired trajectory.

The intelligent trajectory generator is represented by a set of IF...THEN... statements and the logical operators. The program is written in Turbo-basic on AT-IBM compatible personal computer.

Case study 1: Two links manipulator

The actual robot arm is replaced by its kinematical model. This model is represented by the following two equations:

$$x_g = [L_2 \cos(q_1+q_2) + l_1 \cos(q_1)]$$

$$y_g = [L_2 \sin(q_1+q_2) + l_1 \sin(q_1)]$$

Also, the elbow coordinates are given by the following two equations

$$x_e = l_1 \cos q_1$$

$$y_e = l_1 \sin q_1$$

The desired trajectory is represented by two arrays x(n),y(n), where n is the number of points along the trajectory and the i^{th} elements of these arrays correspond to the cartesian coordinates of the i^{th} trajectory set point. The link lengths are 100,100 mm.

Experiment 1

The goal of this experiment is to study the effect of large sudden change between two consecutive set points along the trajectory. Fig. 4.1 shows a trajectory ABCD of this type. The value of the proportionality constant f is chosen to be 0.001. The actual trajectory (derived using the rules in Table 2) is superimposed on the desired one (see Fig. 4.1). There are 100 set points between A and B, and so between the two points C and D. There are no intermediate set points between B and C. Point B is 100 mm away from C.

It is evident from Fig. 4.3 that the large sudden change does not affect tracking quality. However, it is not recommended to use such type of trajectories hence actually the control angles vary with relatively large steps. This may be unrealizable when using most types of dynamic controllers.

Experiment 2

In this experiment a nonlinear trajectory is used. Fig. 4.2 shows the desired trajectory which is an ellipse with major axis length of 130 mm and minor axis length of 80 mm. The center of the ellipse is located at point (20,-30). The orientation of the major axis is (-45°). The chosen number of points along the trajectory is 180. They are uniformly distributed along it with angular displacement of 2° between each two consecutive points. The actual trajectory generated using ITG is shown in dotted line in the same figure.

It is clear that the proposed algorithm has the ability to closely follow nonlinear paths. There is no need for linearizing such type of paths between certain points as required by many trajectory generators.

Experiment 3

The interest of this experiment is to study the effect of increasing the number of set points on the quality of tracking. The same path used in experiment 3 is also used here. The number of set points is varied from 100 to 1000. Fig. 4.3 shows the variation of the average deviation E_{av} versus n_t. As may be logically expected, E_{av} is inversely proportional to n_t. However, increasing n_t requires additional memory for storing the desired path set points coordinates, which may be unsuitable for most microprocessor based systems. A compromise between these two factors should be made.

Experiment 4

Our interest in this simulation is to establish the dependence of tracking quality and the proportionality gain factor f. In other words, what is the optimal value of f such that the average deviation E_{av} is minimum .? Two elliptic trajectories are used in this experiment. The first, namely T1 is that described previously, and the second trajectory T2 has center at (-30, 40) mm, major axis equals 125 mm, minor axis equals 100 mm, and orientation of -45°. Figure 4.4 shows the variation of E_{av} against f. It is evident that the suitable range for f is the same for both trajectories. The conclusion derived is that the optimal range for f is trajectory independent (at least for the same class of trajectories). This is because the decision of the intelligent controller depends on local measurements irrespective of the global shape of the trajectory. This means that, f is constant for the same manipulator, therefore, f needs to be calculated only once for the same robot arm.

Experiment 5

The goal of this simulation is to study the deviation at each set point between the desired and the actual trajectories. The same trajectory used in experiment 2 is also used here. Fig. 4.5 shows this variation against the radial angle for different values of n_t=180, 360, 720. As shown in this figure, the maximum deviation occurs always around the same radial angles, namely at angles 0, 180°, 360°. This can be explained by the nature of the trajectory at these angles at which the gradient of the path has maximum variation. however, this deviation remains within acceptable limits even for relatively small values for n_t.

Case Study II: Three link manipulator

The actual robot arm is replaced by its kinematical model. This model is represented by the following three equations:

$$x_g = [L_2 \cos(q_2+q_3) + l_3 \cos(q_2)] \cos(q_1)$$
$$y_g = [L_2 \cos(q_2+q_3) + l_3 \cos(q_2)] \sin(q_1)$$
$$z_g = [L_2 \sin(q_2+q_3) + l_3 \sin(q_2)] +l_1$$

Also, the elbow coordinates are given by the following three equations

$$x_e = l_2 \cos q_2 \cos q_1$$
$$y_e = l_2 \cos q_2 \sin q_1$$
$$z_e = l_2 \sin q_2 + L_1$$

The desired trajectory is represented by three arrays $x(n), y(n), z(n)$, where n is the number of points along the trajectory and the i th elements of these arrays correspond to the cartesian coordinates of the i th trajectory set point. The link lengths are 100,100,100 mm.

Experiment 6 Motion along straight line segment

Suppose that the robot end effector is required to travel along a straight line segment in the task space (Cartesian coordinates). The starting point is A(50,30,220) at time $t_o =$ 0 sec, and the goal point is (250,200,400) at time $t_f=10$ sec. The end effector must be at reset at points A and B. This implies that the velocity $\underline{v}(t_o)=\underline{v}(t_f)=0$. It is also prefered to have a smooth robot motion that implies that the acceleration and its time derivative (jerk) should be continuous along the trajectory and such that the acceleration $\underline{a}(t_o)=\underline{a}(t_f)=0$. The parametric equation of the straight line is given as

$$\frac{x(t)-x_o}{x_f - x_o} = \frac{y(t)-y_o}{y_f - y_o} = \frac{z(t)-z_o}{z_f - z_o} = \lambda(t)$$

where $\lambda(t)$ is the motion parameter. In order to satisfy the given constraints about the position, velocity, and the acceleration, a fifth order polynomial representation of the line segment AB is selected. This implies that

$$\lambda(t) = a_5 t^5 + a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

With the condition $\lambda(t_o)=0$ we have $a_0=0$. The condition $\underline{v}(t_0)=0$implies $a_1=0$, also the condition $\underline{a}(t_o)=0$ implies $a_2=0$. The polynomial $\lambda(t)$ is reduced to

$$\lambda(t) = a_5 t^5 + a_4 t^4 + a_3 t^3$$

Applying the three conditions $\lambda(t_f)=1$, $\underline{v}(t_f)=0$, and $\underline{a}(t_f)=0$ give

$$a_5 t_f^5 + a_4 t_f^4 + a_3 t_f^3 = 1$$
$$5\, a_5 t_f^4 + 4\, a_4 t_f^3 + 3\, a_3 t_f^2 = 0$$
$$20\, a_5 t_f^3 + 12\, a_4 t_f^2 + 6\, a_3 t_f = 0$$

After few mathematical treatment, the solution becomes

$$a_5 = 6/t_f^5 \quad , \quad a_4 = -15/t_f^4 \quad , \quad a_3 = 10/t_f^3$$

and the motion parameter $\lambda(t)$ is given as

$$\lambda(t) = (6/t_f^5)t^5 - (15/t_f^4)t^4 + (10/t_f^3)t^3$$

It should be noticed that $\underline{v}(t)$ has a very desirable velocity profile, it reaches maximum value at $t_f/2$ where the acceleration becomes zero. We also have $\underline{a}(t)>0$ for $t<t_f/2$ and $\underline{a}(t)<0$ for $t>t_f/2$.

For the straight line segment mentioned previously, the trajectory is divided into 100 points. The intelligent trajectory generator is used to track the path. The actual path is found to be very close to the desired one. The average error of tracking was about .4 mm.

CONCLUSION

The paper introduced an intelligent trajectory tracking algorithm used for robot manipulators. The rules used to modify the joint angles depend on the geometrical structure of the arm and experience of operating it. The algorithm can be used to track linear and nonlinear trajectories. This algorithm determines the different joint angles for each set point in one iteration (one step ahead). This enables the algorithm to be used for the on-line applications. The algorithm does not require matrix inversion or trigonometric function calling. Instead, logical and relational operations are used extensively for deciding the values of joint angles. The arm kinematical model is not required if optoelectronic sensor feedback is used to estimate the position of certain points along the robot body.. Work can be expanded to include orientation tracking. A similar algorithm is currently developed to solve the dynamical control problem for the robot manipulators.

REFERENCES

[1] M. Brady, et. al. (editors), "Robot motion : planning and control," MIT Press, 1982.
[2] M. S. Mujtaba, "Discussion of trajectory calculation methods," in Exploratory study of computer integrated assembly systems, Binford, T. O. et. al., Standford University, Artificial Intelligence Laboratory, AIM 285.4, 1977.
[3] R. Courant, and D. Hilbert, "Methods of mathematical physics," Jon Wiley, New York, 1937.
[4] A. F. Bryson, and Y. C. Ho, "Applied optimal control,"Blaisdell, New York, 1969.
[5] D. E. Whitney, "The mathematics of coordinated control of protheses and manipulators," J. Dynamic Systems, Measurement, Control, (Dec 1972), pp. 303-309.
[6] R. P. Paul, "Manipulator path control," Proc. IEEE Int. Conf. Cybernetics and Society, New York, September, 1975, pp. 147-152.
[7] R. H. Taylor, "Planning and execution of straight-line manipulator trajectories," IBM J. Research and Development 23 (1979), pp. 424-436.
[8] A. K. Bejczy, T. J. Tarm, and Z. F. Li, "Task driven feedback control of robot arms; a step towards intelligent control," Proc. of 25th IEEE Conf. on Decision and Control, Athens, Greece.
[9] Van Den Bosch P.P. J., W. Jongkind, H. R. Van nauta lemke, and D. Zoetekouw," Kinematic control by means of optimization," in Theory of Robots, edited by P. Kopacek, et al., IFAC Proceedings Series, 1988, no.3, pp. 107-112.
[10] C. S. Lee, "Robot arm kinematics," in Tutorial on Robotics, by C. S. Lee, et. al. (editors), IEEE Computer Society Press/North-Holland, 1983, pp. 47-65.

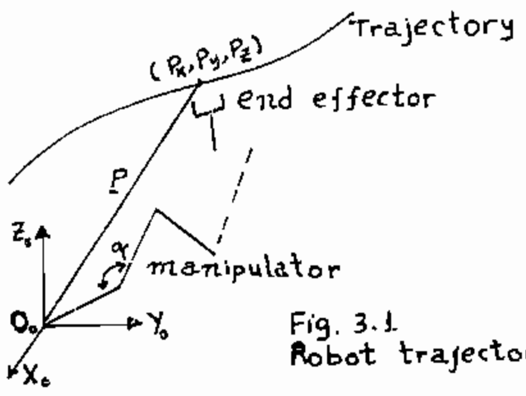


Fig. 3.1 Robot trajectory

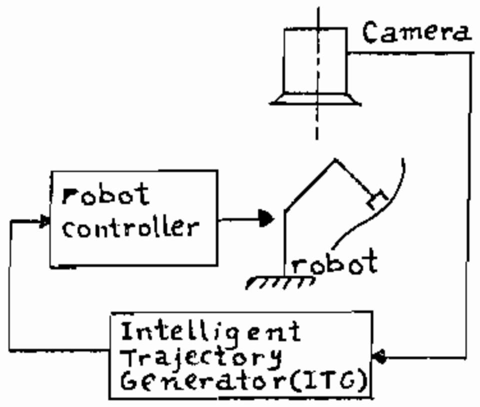


Fig. 3.2 System block diagram

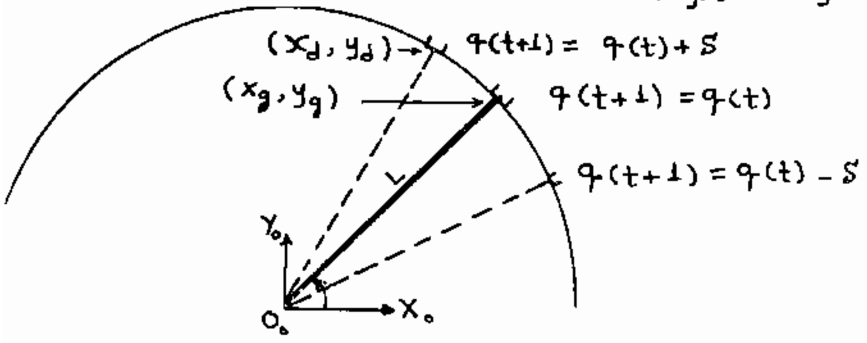


Fig. 3.3 Single arm manipulator

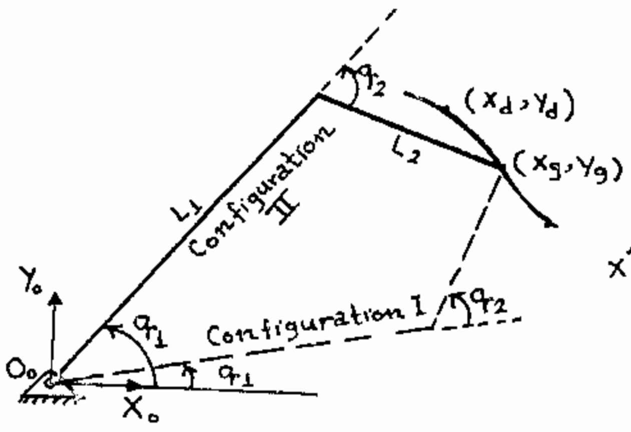


Fig. 3.4 Two links manipulator

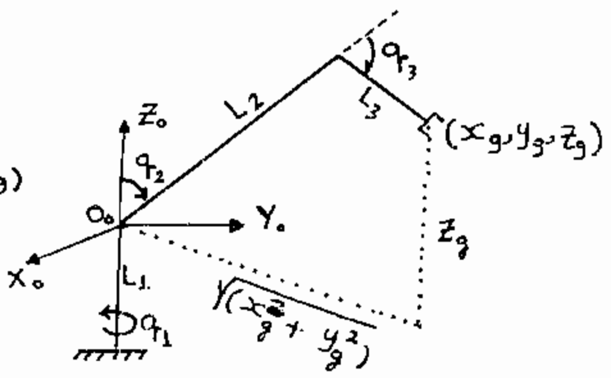


Fig. 3.5 Three links manipulator

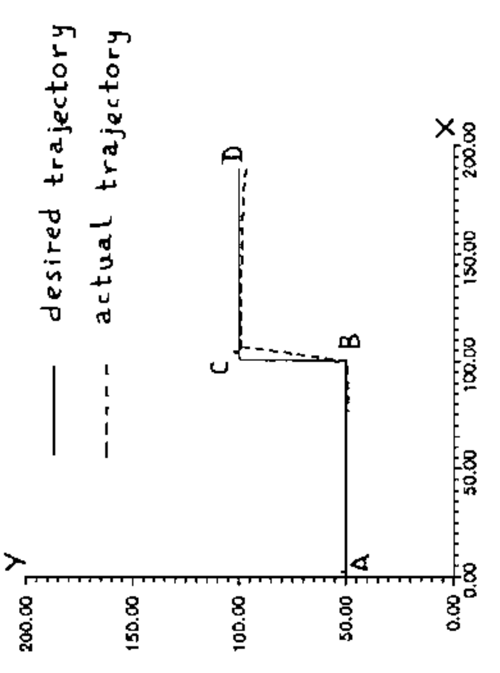


Fig. 4.1 Trajectory sudden change

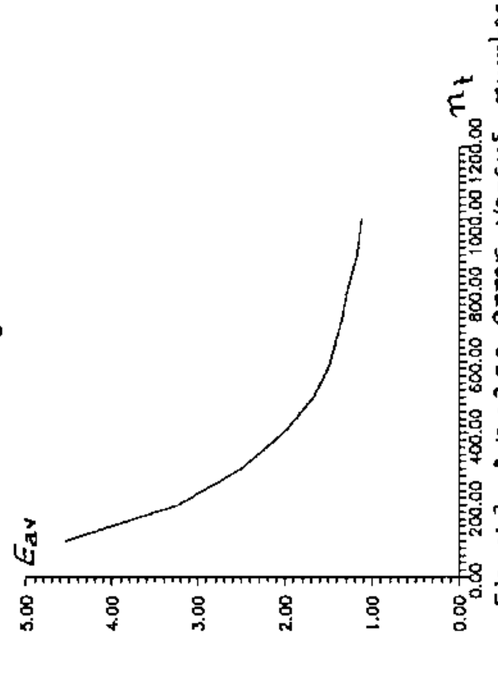


Fig. 4.3 Average error versus number of trajectory set points

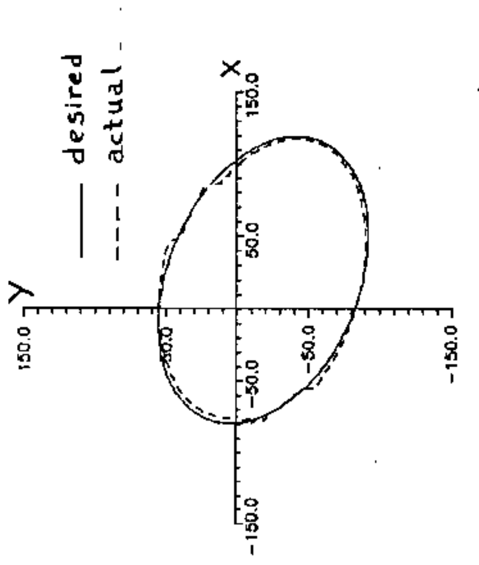


Fig. 4.2 non-linear trajectory

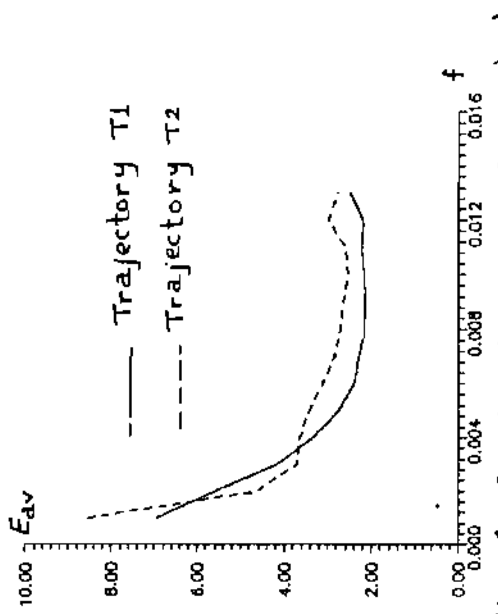
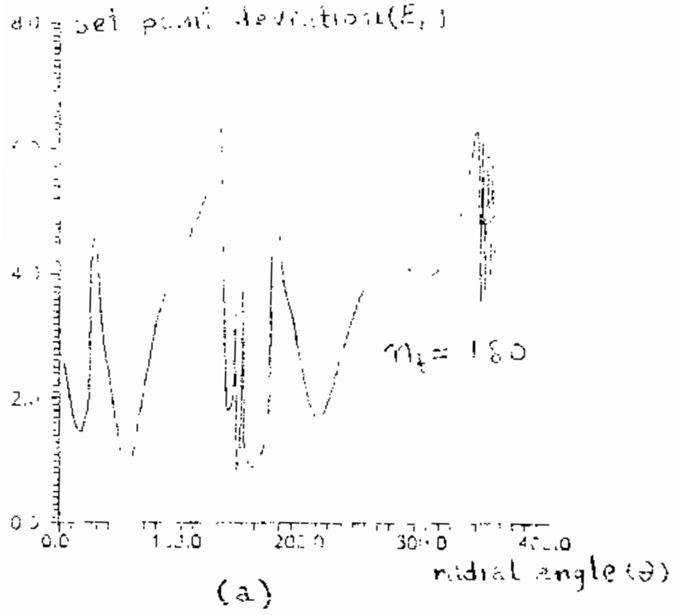
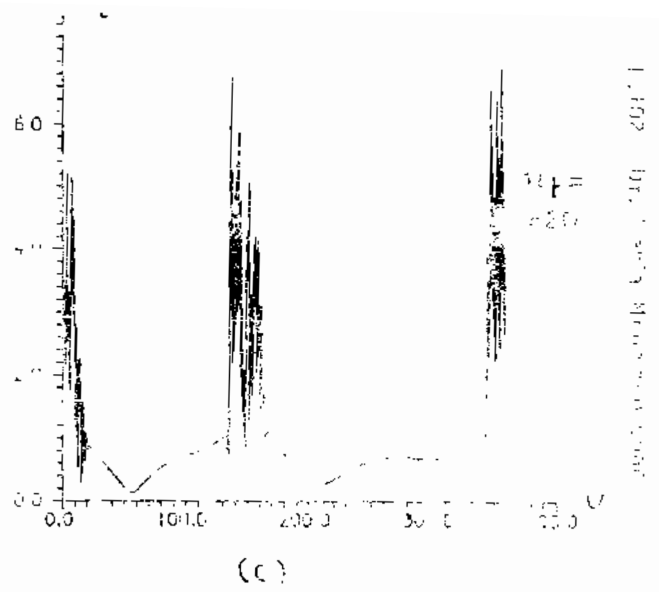


Fig. 4.4 Average error versus constant proportionality factor





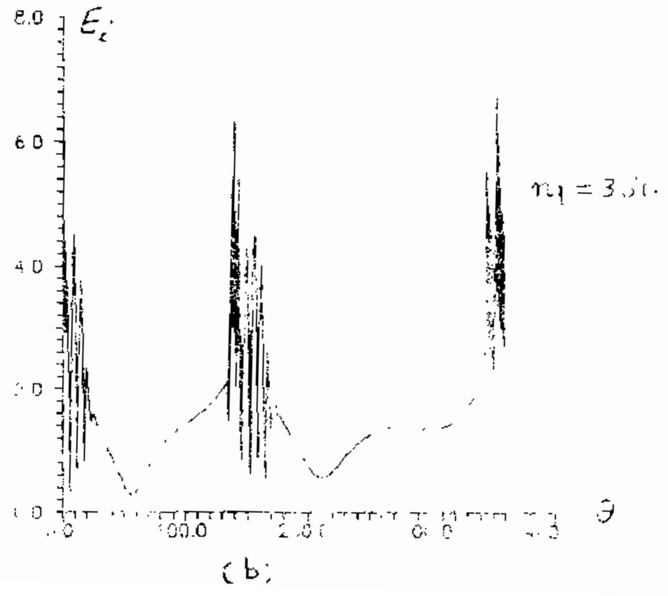


Fig [illegible] Set point deviation versus radial angle for different elliptic trajectories