Proceedings of the 1996 IEEE
International Conference on Robotics and Automation
Minneapolis, Minnesota - April 1996

OPTIMAL PLANNING OF ROBOT CALIBRATION EXPERIMENTS BY GENETIC ALGORITHMS

Hanqi Zhuang*, Jie Wu** and Weizhen Huang**
* Department of Electrical Engineering
** Department of Computer Science and Engineering
Florida Atlantic University, Boca Raton, FL 33 431

Abstract[1]

In this paper, techniques developed in the science of genetic computing are applied to solve the problem of optimally selecting robot measurement configurations, which is an important element in successfully completing a robot calibration experiment. Genetic algorithms are customized for a type of robot measurement configuration selection problem in which the robot workspace constraints are defined in terms of robot joint limits. Simulation studies are conducted to examine the effectiveness of the genetic algorithms for the application.

I. INTRODUCTION

Robot calibration is a process by the robot accuracy is enhanced through modification of its control software [1]. An important step towards a successful calibration task is the planning of a robot experiment, whose main purpose is to find a set of optimal robot measurement configurations for robot calibration. The objective of this work is to apply the available genetic computing techniques to the problem of optimally selecting robot measurement configurations in a robot calibration experiment.

Qualitatively, optimal selection of robot configurations can be stated as the problem of determining a set of robot measurement configurations within the reachable robot joint space so that the effect of measurement uncertainties on the estimation of robot kinematic parameters is minimized [2-5].

Genetic algorithms (GAs) are heuristic search algorithms based on the mechanics of natural selection and natural genetics. They mimic a natural evolution process, in which those highly-fit individuals will have better chance to survive competition within a generation. Individuals also exchange genes to form new and potentially better offsprings in the new generations. GAs search along different directions in the hope to find better and better solutions in the search process [6-9].

In this work, GAs are applied for the problem of optimal selecting robot measurement configurations. We tune the GA parameters to solve a type of robot measurement configuration selection problems in which the robot workspace constraints are defined in terms of robot joint limits. Simulation studies are conducted for both cases to examine the efficiency of GAs.

II. Problem Statement

In order to state the problem of robot measurement configuration selection, we first introduce the concept of robot kinematic error parameter observability.

A. Observability of Error Parameters

The goal of kinematic identification is to estimate an independent kinematic parameter vector ρ that accounts in the least-squares sense for positioning and orientation errors of the robot. A common approach to the problem has been to define such a set of variables in terms of additive changes $d\rho$ to the robot nominal link parameter vector ρ^0 in a given kinematic model. Linearization of robot forward kinematic equations about ρ^0 at the ith particular joint measurement configuration $\boldsymbol{q}_i$ provides the Identification Jacobian matrix $J_i = J(\boldsymbol{q}_i, \rho^0)$, which relates $\boldsymbol{y}_i \in \mathrm{R}^k$, end-effector pose errors at the ith configuration to $d\rho$, the vector of independent kinematic parameter errors [1]

$$\boldsymbol{y}_i = J_i d\rho \tag{2.1}$$

Let $\boldsymbol{y} = [\boldsymbol{y}_1^T, \boldsymbol{y}_2^T, ..., \boldsymbol{y}_s^T]^T$, where s is the number of measurements, and define an aggregated Identification Jacobian matrix $\boldsymbol{J}$ that is obtained by stacking J_i one on top of the other, . The overall measurement equation for least squares estimation of $d\rho$ is then

$$\boldsymbol{y} = \boldsymbol{J} d\rho \tag{2.2}$$

It is said that the kinematic error parameter vector $d\rho$ is observable if and only if $\boldsymbol{J}^T\boldsymbol{J}$ is full rank.

The condition number of $\boldsymbol{J}$ is used as an observability index in this work,

$$\mathrm{Cond}(\boldsymbol{J}) = \sigma_{max}/\sigma_{min} \tag{2.3}$$

where σ_{max} and σ_{min} are maximum and minimum singular values of $\boldsymbol{J}$.

1 This work is partialy funded by NSF though grant #DMI-9409716

B. Problem Formulation

Generally, the problem of optimal robot calibration experiment planning can be stated as follows: Determine m robot measurement configurations within the reachable robot joint space so that the effect of measurement uncertainties on the parameter estimation errors is minimized. Note that m should be sufficiently large. For instance, when full poses of the robot end-effector can be measured, a necessary condition is $6m > t$, where t has been defined as the number of independent kinematic parameters of the manipulator.

The problem stated above is difficult to solve since the mathematical relationship among parameter errors, measurement noise and measurement configurations is yet to be properly modeled. However, if an error-model-based technique is adopted for kinematic parameters identification, the optimal configuration selection problem may be solved through investigation of the conditioning of the Identification Jacobian since the upper bound of the parameter error norm is proportional to the condition number of the Identification Jacobian. In the following discussion, the error model based technique is used to illustrate the concept of the proposed method. The configuration selection method is equally applicable to other robot calibration techniques in which linear transformations relating pose measurements to the unknown kinematic parameters are available (that is, measurements and parameters are related by a matrix $\boldsymbol{J}$).

Let us define the problem of optimal robot measurement configuration selection as follows:

Problem: Determine m robot measurement configurations in the reachable robot joint space such that Cond($\boldsymbol{J}$) is minimized.

III. OVERVIEW OF GENETIC ALGORITHMS

A. A Simple Genetic Algorithm

A GA typically has the following elements:

* A genetic representation (or an encoding) for the feasible solutions to the optimization problem.
* A population of encoded solutions.
* A fitness function that evaluates the optimality of each solution.
* Genetic operators that generate a new population the existing population.
* Control parameters.

The GA may be viewed as an evolutionary process wherein a population of solutions evolves over a sequence of generations (refer to Figure 3.1). During each generation, the fitness of each solution is evaluated, and solutions are selected for reproduction based on their fitness. Selection embodies the principle of Survival of the fittest. 'Good' solutions are selected for reproduction while 'bad' solutions are eliminated. The 'goodness' of a solution is determined from its fitness value. The selected solutions then undergo recombination under the action of the crossover and mutation operators.

The mechanics of a simple genetic algorithm, in its simplest form, is just copying chromosomes, swapping partial chromosomes and occasionally changing the value of a randomly selected bit in a chromosome. Each chromosome of 0s and 1s is the encoded version of a solution to the optimization problem. Using genetic operators, crossover and mutation, the algorithm creates the subsequent generation from the chromosomes of the current population. This generational cycle is repeated until termination criteria are reached. For example, best fitness values do not change much after a number of generations, or a predefined number of generations have been processed.

```
{
initialize the parameters of the GA;
randomly generate the old_population;
while convergence not achieved
    {
    clear the new_population;
    evaluate the fitness of each individual in the
      old_population;
    copy individuals with highest fitness to the
      new_population;
    perform crossover based on crossover rate;
    perform mutation based on mutation rate;
    place offsprings from crossover and mutation to
      the new_population;
    replace the old_population by the
      new_population;
    }
}
```

Figure 3.1 Basic structure of a genetic algorithm

For the design of coding schemes and genetic operators, readers are referred to [6,8,9].

IV. APPLICATION TO CONFIGURATION SELECTION PROBLEM

To apply genetic algorithms to the optimal robot measurement selection problem, one has to consider the following issues:

1. The fitness function.
2. The coding scheme.
3. The reproduction, crossover and mutation operators.

In this application, we use some of the

reproduction, crossover and mutation techniques discussed above. Namely, we use rank-based reproduction scheme combined with linear scaling to control number of offsprings for each chromosome. We then use stochastic remainder selection procedure to refine the results produced by the rank-based procedure. That is, a population of chromosomes are first ranked based on the their fitness values. The ranked chromosomes are then designated to a number of copies after scaling and stochastic remainder selection. Furthermore, we use one-point crossover and one-point mutation schemes to recombine genetic materials of the chromosomes from the mating pool.

We now concentrate on issues regarding fitness function and coding scheme.

A. The Fitness Function

For the measurement configuration selection problem, one may use either condition number or the observability index of the Identification Jacobian as the fitness function. While these mathematical indices reflect to a certain degree the estimation errors if the selected joint variables are used for parameter estimation in a later stage, the relationship between the values of the condition number or the observability index to the goodness of the solution is not certain. One cannot claim that a set of measurement configurations that produce a small condition number is certainly better than the other set of measurement configurations with a large condition number in terms of the effectiveness of robot calibration in a later stage. The problem is that we do not have a mathematical tool up till now that can characterize the problem better than the cited performance indices. Moreover, the variables of these performance indices are the measurement configuration, which can be represented by joint variables of the robot. The physical significance of these variables goes without saying. While we are striving to discover a better fitness function for the problem at hand, in this work we use the condition number of the Identification Jacobian defined in Section 2 as the fitness function of the problem. The correlation between this type of objective functions and the efficiency of robot calibration were demonstrated by experimentation in [3].

B. The Coding Scheme

Coding is a process that converts the values of control variables to strings. One motivation of the coding is the attempt to associate high fitness values with similarities among strings in the population. Next, we give an example of our coding scheme. Assume that the robot has three joints, revolute-revolute-prismatic (RRP), whose joint variables are θ_1, θ_2 and d_3. One may use a 4-bit binary number for each joint variable. Thus we need 12 bits to code each set of joint variables (a measurement configuration). Assume further that 4 measurement configurations are used to compute the Identification Jacobian. In total we need 4x12 = 48 binary bits to form a binary string to represent the variable vector of the fitness function.

The above scheme can be extended to any robot with any number of degrees of freedom and any number of measurement configurations. The number of bits, s_L, for each string can be calculated by the following equation:

$$s_L = mnb$$

where b is the number bits for each joint variable, n is the number of degrees of freedom of the robot and m is the number of measurement configurations planned for the robot calibration experiment.

One has no choice in n, as the number of degrees of freedom for a given robot is fixed. One does not have much choice on m either, since normally twice as much as the necessary number of measurements need to be taken for a reliable estimation of robot parameters. For instance, if the robot has 30 kinematic parameters and each complete pose measurement provides 6 estimation equations. The minimum number of measurements required for the estimation of the 30 parameters is 30/6 = 5. A good choice of m is thus about 10. As to b, the number of bits for each joint variable, one does not have much choice too. For instance, a joint can move from 0 degree to 90 degree. If one wants to explore the fitness value for angles that are one degree apart, the number of bits is then 6.

Some of the strings may not represent a feasible angle. For instance, string 111111 (representing a joint displacement of 127 degrees) is not a viable string as the corresponding angle exceeds the robot joint limit of 90 degrees. This problem can be avoided by normalize the joint angles. That is, the smallest and largest string values represent the smallest and largest values of the joint angle, respectively. In between while the string value is increased or decreased, the joint value will also be proportionally increased or decreased.

V. EXPERIMENT DESIGN AND RESULTS

The main objective of this experimental study is to find suitable GA parameters for solving the problem of robot measurement configuration selection. In order to achieve this objective, experiments must be carefully designed.

Because SCARA robots such as Adept one, IBM 7545 and Intelledex are most popular in the electronic manufacturing industry, we choose an Intelledex robot as the manipulator for testing our algorithms. To examine the effectiveness of the proposed techniques, an independent evaluation step will be employed to check if so-called best-fit offsprings are really superior for our application. This is achieved by using criteria that are different from the fitness function of the GAs.

A. Environment Simulation

The Intelledex is a RRP-type robot. The Modified Complete and Parametrically Continuous (MCPC) convention was used to model the robot, and its parameters are listed reference [10]. Furthermore, the joint limits for θ_1, θ_2 and d_3 are set according to the manufacturing specifications.

To simulate the robot pose measurement process, we randomly generate a number of robot measurement configurations within the robot workspace to form an initial population. Each chromosome of the population, representing a set of robot measurement configurations, is evaluated by its fitness function, the condition number of the Identification Jacobian at these robot measurement configurations. This population is continuously updated by three basic genetic operations. After the algorithm converges, each candidate solution obtained by the GA is then decoded to a set of robot measurement configurations. This set of configurations are then used to "calibrate" the robot, again by simulation.

To be able to evaluate the performance of GAs, we use the best, medium and worst fitness values as three measures. Due to the randomness of the genetic algorithms, results will be different from one run to another, even if the values of the GA parameters are identical. Therefore we run a genetic algorithm a number of times for each set of parameter values. The average fitness values of the multiple runs are used to evaluate the performance of the GA algorithms under different parameter settings.

B. GA Parameter Tuning

The GA we used in the simulation study have the following characteristics. A concatenated binary coding technique is adopted. Each joint variable is assigned a p-bit substring. Each measurement configuration is represented by np bits, where n is the degrees of freedom of the robot. Because a chromosome consists of m measurement configurations, the length of a chromosome is nmp bits. For instance, if 5 bits are used for each joint angle, 3 joint angles form a configurations, and 4 measurements are needed for computing the Identification Jacobian and its condition number, the chromosome length will be 60.

The reproduction procedure adopted in the simulation studies can be described as follows. After all the chromosomes in the old generation are ranked based on their fitness values, a scaling procedure is introduced to prevent super strings from being overly prosper. The stochastic remainder approach is then used to determine how many copies of each chromosome to be made to form a mating pool.

Parameters in GA are population size, crossover rate, mutation rate, and maximum number of copies for the best chromosome. Other parameters related to the problem are number of bits to represent a joint variable in a chromosome and number of measurement configurations in each chromosome. These two parameters effectively change the chromosome length. We applied the linear scaling procedure and limited the maximum number of copies to the best chromosome in the reproduction stage. In addition, we also test if using the elitist strategy improves the performance of GA.

There exits a number of ways to select GA parameters. A logic approach is to treat the selection problem as an optimization problem and then to apply a search technique to find best parameters for the GA. Grefenstetee designed a meta level GA to tune the parameters of GAs [9]. A population of GAs with different parameters are competing one other. Based on their performance, the meta level GA will adjust parameters of the low-level GAs. The process continues until some convergence criteria are satisfied. Grefenstetee discovered that although the optimal parameter values of GA found by the meta level GA produce better performances, the improvement is marginal, comparing with the set of parameter values suggested by De Jone [9].

In our simulation studies study, we decided to use the cyclic search (or successive search) method. Starting from the initial parameters similar to those suggested by De Jong and Grefenstetee, GA parameters are changed one at a time, until all parameters are changed. This process continues until no significant improvement can be obtained.

The initial parameters for GA are given in Fig. 5.1. Note that this set of parameters were chosen mainly based on De Jone's recommendation [8]. The identified values of GA parameters for our application are listed in Fig. 5.2.

The set of parameter values given in Fig. 5.2 is very close to the set of initial parameter values listed in Fig. 5.1. There are two reasons that can explain this phenomenon. The first reason is that genetic algorithms are very robust in the sense that changing its parameter values usually will not its performance significantly. The second reason is that the initial set of parameter values is not selected arbitrarily. On the contrary, they are chosen after taking into consideration of the suggestions made by researchers in the area of genetic algorithms and by preliminary simulations.

Population size (Popsize) = 50,
Crossover rate (P_c) = 0.5,
Mutation rate (P_m) = 0.01,
Maximum number of copies for the best chromosome (R_n) = 1.5,
Number of bits for each joint angle (N_b) = 6,
Elitist (Elit) = Yes.

Fig. 5.1 Initial parameters of GA

Population size (Popsize) = 70,
Crossover rate (P_c) = 0.6,
Mutation rate (P_m) = 0.01,
Maximum number of copies for the best chromosome (R_n) = 2,
Number of bits for each joint angle (N_b) = 8,
Elitist (Elit) = Yes.

Fig. 5.2 Final parameters of GA

We plot also the best, median and worst fitness values in Fig. 5.3, which were obtained using the set of GA parameter values that are similar to those given in Fig. 5.2.

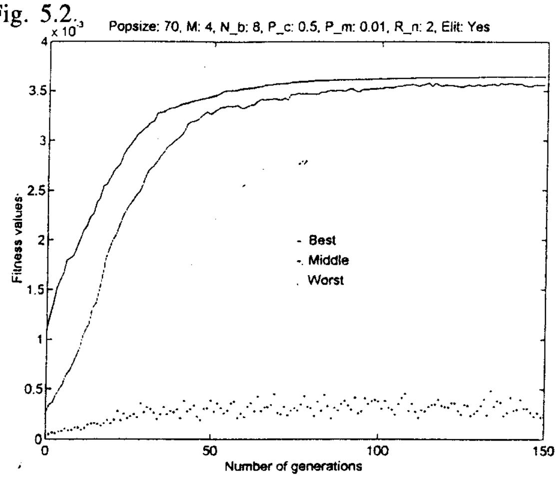

Fig. 5.3 The best, median and worst fitness values

C. Verification Results

To simulate the calibration process, it is assumed that the measurements obtained by a "measuring device" are contaminated with measurement noise. We simulate a moderately accurate measuring device, which can guarantee a position accuracy of 0.001 cm. We further assume that the measurement errors by the joint sensors are small enough therefore they can be ignored, and that the robot parameters are not known exactly. We start with the nominal robot parameters and iteratively solve for the "actual" parameters, using a number of "contaminated" robot end-effector pose measurements. After the algorithm converges, the identified robot kinematic parameters are then fed to a verification routine. In this routine, a number of different robot poses are generated by the identified robot parameters, and are measured by the simulated "measuring device". The discrepancy between the computed robot pose and the "measured" robot pose, referred to as pose error, is computed for every robot pose generated at the verification stage. The set of GA parameters that produces the best result is considered to be the winner.

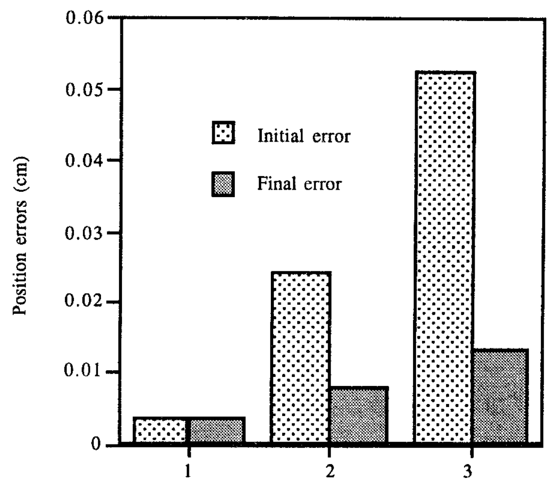

Fig. 5.4 Position errors with GA

1. Error from using best chromosome
2. Error from using median chromosome
3. Error from using worst chromosome

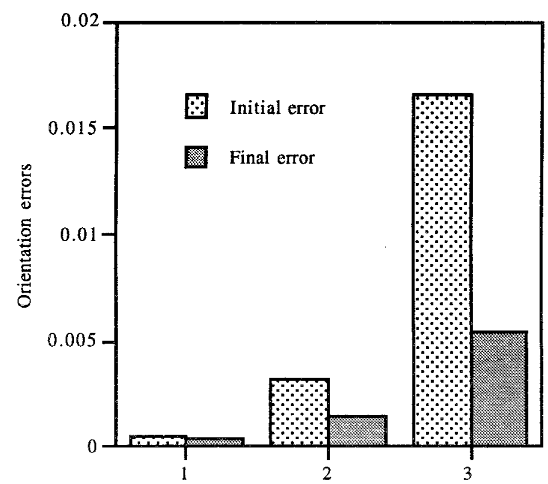

Fig. 5.5 Orientation errors with GA

1. Error from using best chromosome
2. Error from using median chromosome
3. Error from using worst chromosome

The verification results are given in Figs. 5.4- 5.5.

From these figures, the following comments can be made:

1. Using a set of well-selected robot measurement configurations can improve significantly the efficiency of robot calibration. This is evident from the simulation results. Performance of the best chromosome are 20 times better than that of the worst chromosome and 10 times better than that of the median chromosome in terms of both position errors and orientation errors.
2. The performance of the genetic algorithms is not much better than that of the random search procedure. This is because when the size of the population is large, there is a very good chance that some good chromosomes are in the initial population. This suggests that if only a single good chromosome is needed, one may not need to use genetic algorithms. On the other hand, the average performance of the population after a number of generations is improved significantly. One can utilize this property to choose a set of measurement configurations that is most suitable for implementation.

VI. CONCLUSIONS

In this paper, the genetic computing technique has been applied to the problem of optimally selecting robot measurement configurations for robot calibration experiments. Sets of reasonably good parameters for GAs have been determined through simulations. Verifications have also been conducted to explore the advantages and limitations of the genetic algorithms. It has been demonstrated that to seek a single set of optimal measurement configurations, the performance of genetic algorithms is only slightly better than that of the random search algorithm. On the other hand, if multiple optimal solutions are needed, the genetic algorithms have an edge over random search.

The classical genetic algorithm is a very useful tool to solve certain type of search problems. However, the original GA may not be able to solve a type of optimal robot calibration experiment planning problem, in which the measurable robot workspace is defined in the Cartesian space. The major problem is due to the evolution process implemented by the crossover and mutation operations in GA. Although two parents may represent legal points in a measurable workspace of the robot, their children may not.

This is a realistic problem for practical applications because in most application cases, a measurable workspace is only a subset of the entire robot workspace. If some of the identified optimal measurement configurations are not within the measurable workspace, then the optimal solution is not useful. One way to remedy this problem is to check the validity of the children after crossover and mutation before placing them in the new generation pool. However, this is not a desirable approach theoretically, although it may work in practice.

Another way of avoiding this problem is to devise a genetic algorithm that always produce legal children. Readers are referred to [11] for a more detailed account of this issue.

REFERENCES

[1] Mooring, B. W., Z. S. Roth, and M. R. Driels, *Fundamentals of Manipulator Calibration*, John Wiley & Sons, 1991.

[2] Driels, M. R., and U. S. Pathre, "Significance of Observation Strategy on the Design of Robot Calibration Experiments," *J. Robotic Systems*, Vol. No. 2, 1990, pp. 197-223.

[3] Borm, J. H., and C. H. Menq, "Determination of Optimal Measurement Configurations for Robot Calibration Based on Observability Measure," *Int. J. Robotics Research*, Vol. 10, No. 1, Feb. 1991, pp. 51-63.

[4] Khalil, W., M. Gautier and C. Enguehard, "Identifiable Parameters and Optimum Configurations for Robot Calibrations," *Robotica*, Vol. 9, pp. 63-70.

[5] Zhuang, H., K. Wang, and Z.S. Roth, "Optimal selection of measurement configurations for robot calibration using simulated annealing," *Proc. IEEE Int. Conf. R&A*, San Diego, May 1994, pp. 393-398.

[6] Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Learning*, Addison-Wesley, 1989.

[7] Holland, J.H., *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.

[8] De Jong, K.A., *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*, Doctoral Diseertation, University of Michigan, 1975.

[9] Grefenstette, J.J, "Optimization of Control Parameters for Genetic Algorithms," *IEEE Trans Systems, Man, and Cybernetics*, Vol. 16, No. 1, pp 122-128.

[10] Wu, W-C, *Camera-Aided SCARA Arm Calibration*, M.S. Thesis, Florida Atlantic University, Boca Raton, Florida, 1994.

[11] Huang, W., *Optimal Planning of Robot Calibration Experiments by Genetic Algorithms*, M.S. Thesis, Florida Atlantic University, Boca Raton, Florida, 1995.