Multiobjective Design Optimization of Counterweight Balancing of a Robot Arm using Genetic Algorithms*

Carlos A. Coello Coello Alan D. Christiansen Arturo Hernández Aguirre

Department of Computer Science
Tulane University
New Orleans, LA 70118

Abstract

In this paper, we present a hybrid approach to optimize the counterweight balancing of a robot arm, which uses a combination of a genetic algorithm (GA) with the min-max multiobjective optimization method to get the Pareto optimal set of solutions. This set corresponds to several possible robot designs from which the most appropriate has to be chosen by the designer. Our approach is compared to a more traditional min-max search technique in which a combination of random and sequential search was used to generate the Pareto optimal solutions. Our results show how the GA is able to get solutions with a lower deviation from the ideal vector.

Keywords: genetic algorithms, multiobjective optimization, robot arm optimization, design optimization, counterweight balancing

1 Introduction

The use of industrial robots in different fields of technology is becoming more common every day, making it more important to be able to improve their efficiency in terms of energy consumption and working accuracy. The proper balancing of a robot manipulator is one way to improve such efficiency. There are two main methods of balancing a robot manipulator [5]: 1) by spring mechanisms, and 2) by counterweights. The second approach, which is the one selected for this work, has been frequently used in the literature for establishing better mass distributions of mechanisms and its use on robot manipulators involves the minimization of driving forces or torques as well as the support reactions at joints. Since these two criteria have to be satisfied at the same time, a multiobjective optimization approach has to be taken. The lengths and masses of balancing mechanisms of the robot arm are used as design variables, and several constraints derived from the allowable movements of the arm are imposed. The optimization model used for this work is based on the rigid-body dynamics of the PUMA-560 robot [6] [1]. We used a hybrid approach to solve this problem, in which we combined a GA with the min-max method to get the Pareto optimal set, which corresponds to several possible robot designs from which the decision maker has to choose the most appropriate. Our approach is compared to a more traditional min-max technique in which a combination of random and sequential search is used to generate the Pareto optimal solutions. This problem has a highly non-convex search space, which implies the presence of several local minima.

*This work was supported in part by EPSCoR grant: NSF/LEQSF (1992-93)-ADP-04.

2 Statement of the Problem

Koski and Osyczka [5] present a multiobjective optimization model of a PUMA-560 robot arm based on its rigid-body dynamics. By using angular coordinates for the PUMA-560 robot, it is possible to calculate the generalized torques at each joint applying:

$$M_{ti} = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_i}\right) - \frac{\partial L}{\partial \theta_i} \tag{1}$$

where θ_i is the rotation at joint i and $\dot{\theta}_i$ is the corresponding angular velocity. The term

$$L = T - V \tag{2}$$

1082-3409/95 $04.00 © 1995 IEEE

represents the Lagrangian function of the mechanical system. Here, T is the total kinetic energy of the system and V is the total potential energy. The application of eq. (1) to a fully articulated robot arm results in the following nonlinear second-order system of differential equations

$$A\ddot{\theta} + B\dot{\theta}^2 + c - m = 0 \tag{3}$$

The manipulator is an isostatic structure, and thus it is possible to get explicit expressions for all forces and moments in the system. The friction in the joints as well as the flexibility of the arm are not included in our design model. For the application of optimization methods, a two-member robot arm, which corresponds to the two links of the PUMA-560 robot in a plane motion, is considered. This arm is assumed to move in the xy-plane only. In the model used by Koski and Osyczka, only the counterweight masses m_4 and m_5, as well as their distances from the joints x_1 and x_2 are treated as design variables, whereas all the other quantities are fixed. Unfortunately, due to lack of space, we could not include the complete mathematical expressions for the torques and the reactions, but they may be found in Koski and Osyczka [5]. The first two criteria are chosen as follows:

$$\begin{aligned} f_1(\bar{x}) &= \max_{\theta_1} \max_{\theta_2} \max_{\dot{\theta}_i, \ddot{\theta}_i} M_{t1}; \\ f_2(\bar{x}) &= \max_{\theta_1} \max_{\theta_2} \max_{\dot{\theta}_i, \ddot{\theta}_i} M_{t2} \end{aligned} \tag{4}$$

where notation $\dot{\theta}_i$, $\ddot{\theta}_i$ is associated with the chosen angular velocity profile. The construction of joints, especially with the choice of bearings, depends largely on the reaction forces at the joints. Thus, it seems reasonable to choose the maximum values of the joint forces as two additional criteria. By using the fixed trapezoidal velocity profiles and every feasible position of the arm, these criteria can be expressed in the form

$$\begin{aligned} f_3(\bar{x}) &= \max_{\theta_1} \max_{\theta_2} \max_{\dot{\theta}_i, \ddot{\theta}_i} R_1; \\ f_4(\bar{x}) &= \max_{\theta_1} \max_{\theta_2} \max_{\dot{\theta}_i, \ddot{\theta}_i} R_2 \end{aligned} \tag{5}$$

The multiobjective optimization problem [5] becomes:

$$\min \ (f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x}), f_4(\bar{x}))^T \tag{6}$$

subject to

$$\begin{aligned} \theta_i^l \le \theta_i \le \theta_i^u \quad & i = 1, 2; \\ x_i^l \le x_i \le x_i^u \quad & i = 1, 2, 3, 4 \end{aligned} \tag{7}$$

The numerical design data for the problem under consideration is given below [5]. These values are close to those for the first two links of the PUMA-560 robot [1].

$$\begin{gathered} m_1 = 17\ kg, \quad m_2 = 6\ kg, \quad m_3 = 2\ kg, \\ L_1 = L_2 = 0.43\ m, \quad e_1 = 0.07\ m, \quad e_2 = 0.14\ m, \\ \theta_1^l = -40^\circ, \quad \theta_1^u = 220^\circ, \quad \theta_2^l = -140^\circ, \quad \theta_2^u = 140^\circ, \\ \dot{\theta}_{1\ max} = 2\ \tfrac{rad}{s}, \quad \dot{\theta}_{2\ max} = 4\ \tfrac{rad}{s}, \\ \ddot{\theta}_{1\ max} = 10\ \tfrac{rad}{s^2}, \quad \ddot{\theta}_{2\ max} = 20\ \tfrac{rad}{s^2}, \\ x_1^l = x_2^l = 0, \quad x_1^u = x_2^u = 0.2\ m, \quad x_3^l = x_4^l = 0, \\ x_3^u = 35\ kg, \quad x_4^u = 15\ kg, \\ J_1 = 0.2619\ kg - m^2, \quad J_2 = 0.0924\ kg - m^2 \end{gathered}$$

3 The Classical Min-Max Method

In the classical min-max method, an optimal solution is a vector of decision variables which minimizes some global criterion. A function describing this global criterion is a measurement of how close the decision maker can get to the ideal vector—i.e., the vector that contains the optimal solutions of every objective function assuming that these were treated independently—, which we'll denote by f^0. The use of weighting coefficients has been introduced before [4] in conjunction with this method to rank the importance of the candidate criterion, so that the min-max problem can be restated as follows

$$f(x^*) = \min_x \max_i \omega_i \left| \frac{f_i^0 - f_i(x)}{f_i^0} \right| \quad i = 1, \ldots, k \tag{8}$$

where ω_i is the weighting coefficient representing the relative importance of the ith criterion. Koski and Osyczka [5] took this approach to solve the counterweight balancing problem presented in this paper, by using the Computer Aided Multicriteria Optimization System (CAMOS) [2]. They used a method which combines random and sequential search to generate the Pareto-optima.

4 Use of the Genetic Algorithm

Our approach consisted on using a genetic algorithm (GA) to obtain both, the ideal vector and the Pareto-optimal solutions. First, we ran GAs to optimize each objective separately. Then, with this vector, we introduced the following fitness function:

$$t_1 = \left|\frac{f_1^0 - f_1}{f_1^0}\right| \quad t_2 = \left|\frac{f_2^0 - f_2}{f_2^0}\right| \quad t_3 = \left|\frac{f_3^0 - f_3}{f_3^0}\right|$$

$$t_4 = \left|\frac{f_4^0 - f_4}{f_4^0}\right| \quad t_5 = \left|\frac{f_1^0 - f_1}{f_1}\right| \quad t_6 = \left|\frac{f_2^0 - f_2}{f_2}\right|$$

$$t_7 = \left|\frac{f_3^0 - f_3}{f_3}\right| \quad t_8 = \left|\frac{f_4^0 - f_4}{f_4}\right| \tag{9}$$

$$\begin{gathered} if\ (t_1 > t_5)\ z_1 = t_1\ else\ z_1 = t_5; \\ if\ (t_2 > t_6)\ z_2 = t_2\ else\ z_2 = t_6 \\ if\ (t_3 > t_7)\ z_3 = t_3\ else\ z_3 = t_7; \\ if\ (t_4 > t_8)\ z_4 = t_4\ else\ z_4 = t_8 \\ z = w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 \\ fitness = \frac{1}{z} \end{gathered}$$

The weights w_i were also chosen such that $w_1 + w_2 + w_3 + w_4 = 1$.

For all the tests, we used binary tournament selection, double-point crossover, and a population size of 100 chromosomes. Instead of doing several runs with random values for the crossover and mutation probabilities, we used a nested loop in which these two values ranged from 0.1 to 0.9 at increments of 0.1, over 50 generations. This implies that 81 runs were necessary for each design. This procedure showed to be very reliable in terms of finding "good" solutions with the GA, when using a floating-point representation. Execution time becomes an issue, since each run of the GA takes about 2.5 hours on a Sun Workstation with four 90 MHz HyperSparc CPUs. However, the independence of each process made it possible to run them simultaneously on different machines, to improve the performance of the GA.

5 Comparison of Results

We generated the ten Pareto-optimal designs presented by Koski and Osyczka [5]. The ideal vector is $f^0 = (112.75, 30.21, 374.82, 195.21)$ according to Koski. The GA produce a better ideal vector: $f^0 = (92.03, 29.59, 374.80, 195.19)$. To evaluate our results, we used as a parameter the maximum deviation from the optimum, which is defined by

$$L_p(f) = \sum_{i=1}^{4} w_i \left|\frac{f_i^0 - f_i(x)}{\rho_i}\right| \tag{10}$$

where $\rho_i = f_i^0$,or $f_i(x)$, depending on which gives the maximum value for $L_p(f)$.

The comparison of our results with those found by Koski and Osyczka [5] are shown in Table 1. The first eight rows corresponds to the optimal solution vector, and therefore in those cases the deviation $L_p(f)$ is computed by directly comparing the two results, taking the lower as the optimal and the difference of the other one with respect to the first as the deviation. We can clearly see how the GA provided better results in all cases.

From these results we can see that the set of weights $w_1 = 0.1$, $w_2 = 0.1$, $w_3 = 0.4$ and $w_4 = 0.4$ gives the best compromise solution overall. Other interesting aspects to notice from the results to this problem is that there is a great variation in the ranges of the solutions, and that when the mass of the counterweight is close to zero, the variables x_1 and x_2 (joint distances) may assume any value we want, because they won't influence the solution in a significant way.

6 Future Work

We are considering to use several other approaches to multiobjective optimization that have been proposed within the GA community. For example, we want to try the weighted sum approach proposed by Hajela and Lin [3] which includes the weights of each objective in the chromosome, and promotes their diversity in the population through fitness sharing. This allows the simultaneous generation of a family of Pareto optimal designs corresponding to different weighting coefficients in a single run of the GA. Finally, because of the intensive CPU time-consuming nature of this problem, it would be desirable to explore the use of other techniques that can reduce the number of function evaluations, such as the approximation of functions by low order polynomials over some small region. In this case a computationally expensive function is evaluated at a sufficient number of points to construct a low order polynomial approximation. Then, an iterative optimization algorithm is used for finding the minimum of the approximate function. At the point obtained the optimization model is replaced by a new approximate model, and the process continues until the improvement in the objective function can't be distinguished.

7 Conclusions

A GA-based min-max approach has been proposed for a complex multiobjective optimization problem: a robot arm balancing. This problem has four objective functions to be minimized, and is highly non-convex.

Method	w_1	w_2	w_3	w_4	f_1	f_2	f_3	f_4	x_1	x_2	x_3	x_4	$L_p(f)$
Koski	0.25	0.25	0.25	0.25	138.88	38.93	510.18	268.92	0.186	0.198	7.95	4.06	0.3809
GA	0.25	0.25	0.25	0.25	133.16	41.87	375.73	195.92	0.200	0.200	0.029	0.045	0.2170
Koski	0.3	0.3	0.2	0.2	139.91	37.98	612.36	298.39	0.171	0.184	16.9	5.66	0.4737
GA	0.3	0.3	0.2	0.2	102.45	41.87	532.12	195.92	0.200	0.200	20.46	0.045	0.2431
Koski	0.35	0.35	0.15	0.15	152.99	37.74	667.45	336.62	0.194	0.182	19.6	7.59	0.5540
GA	0.35	0.35	0.15	0.15	96.99	40.70	581.09	209.56	0.200	0.200	25.055	0.853	0.2438
Koski	0.4	0.4	0.1	0.1	152.76	38.85	800.85	344.61	0.130	0.193	32.9	7.84	0.5793
GA	0.4	0.4	0.1	0.1	94.71	40.20	615.27	215.86	0.200	0.1778	27.689	1.237	0.2298
Koski	0.2	0.2	0.3	0.3	136.76	38.91	505.85	264.17	0.190	0.197	8.05	3.82	0.3711
GA	0.2	0.2	0.3	0.3	133.15	41.87	375.76	195.92	0.200	0.200	0.033	0.045	0.1742
Koski	0.15	0.15	0.35	0.35	139.62	38.63	457.88	245.80	0.200	0.200	0.039	0.044	0.2917
GA	0.15	0.15	0.35	0.35	133.14	41.87	375.79	195.91	0.200	0.200	0.039	0.044	0.1315
Koski	0.1	0.1	0.4	0.4	141.63	39.46	408.89	228.29	0.103	0.114	0.138	2.08	0.1915
GA	0.1	0.1	0.4	0.4	133.16	41.87	375.72	195.91	0.200	0.200	0.03	0.044	0.08862
Koski	0.5	0.1	0.2	0.2	99.44	41.46	592.53	202.09	0.172	0.093	26.5	0.45	0.2036
GA	0.5	0.1	0.2	0.2	98.91	41.88	553.41	195.84	0.200	0.200	23.244	0.04	0.1749
Koski	0.1	0.5	0.2	0.2	153.03	35.75	645.41	335.46	0.198	0.157	17.0	7.84	0.4584
GA	0.1	0.5	0.2	0.2	133.25	41.84	375.83	196.25	0.200	0.200	0.0	0.065	0.2533
Koski	0.4	0.2	0.2	0.2	121.99	38.42	606.99	258.65	0.148	0.182	20.6	3.6	0.3788
GA	0.4	0.2	0.2	0.2	98.91	41.87	553.44	195.91	0.200	0.200	23.243	0.044	0.2090

Table 1: Pareto-optimal and minimal solutions for the robot arm considered.

Furthermore, the complex calculations involved consume a lot of CPU time, and make necessary the development of heuristic techniques that need the least possible number of function evaluations. The great variation of the results obtained show that this problem would be very difficult to solve with pure random search, or with brute-force techniques. Also, to find a reasonable heuristics seems a difficult task given the factors previously mentioned, and the possible presence of local minima. The GA has showed to be very consistent in this application, finding better compromise solutions for all the instances of the problem under consideration.

References

[1] B. Armstrong, O. Khatib, and J. Burdick. The explicit dynamic model and inertial parameters of the PUMA 560 arm. In *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pages 510–518, San Francisco, California, apr 1986.

[2] H. A. Eschenauer, A. Osyczka, and E. Schäfer. Interactive multicriteria optimization in design process. In Hans Eschenauer, Juhani Koski, and Andrzej Osyczka, editors, *Multicriteria Design Optimization. Procedures and Applications*, chapter 3, pages 71–114. Springer-Verlag, Berlin, 1990.

[3] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4:99–107, 1992.

[4] P. Hajela and C. J. Shih. Multiobjective optimum design in mixed integer and discrete design variable problems. *AIAA Journal*, 28(4):670–675, apr 1990.

[5] J. Koski and A. Osyczka. Optimal counterweight balancing of robot arms using multicriteria approach. In Hans Eschenauer, Juhani Koski, and Andrzej Osyczka, editors, *Multicriteria Design Optimization. Procedures and Applications*, chapter 5.1, pages 151–167. Springer-Verlag, Berlin, 1990.

[6] Charles P. Neuman and John J. Murray. The complete dynamic model and customized algorithms for the puma robot. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-17(4):635–644, jul 1987.