Proc. of the IEEE/RSJ Int. Conf. on Intelligent
Robots and Systems,
Grenoble, France, September 8-12, 1997

Evolutionary Algorithms in Kinematic Design of Robotic Systems

O. Chocron and P. Bidaud
Laboratoire de Robotique de Paris
CNRS-UPMC-UVSQ
10-12 Avenue de l'Europe, 78140 Vélizy, France.
E-mail: chocron,bidaud@robot.uvsq.fr

Abstract

This paper proposes an Adaptative Multi-Chromosome Evolutionary Algorithm (AMEA) to perform task based design of modular robotic systems. The kinematic design is optimized by the AMEA which uses both binary and real encoding (for kinematic and configuration parameters). In the problem considered for illustration, the robotic system consists in a mobile base and a manipulator arm which may be built with serially assembled link and joint modules. The manipulator has to reach a predefined set of goal frames in a 3D cluttered environment. Its design is evaluated with geometric and kinematic performance measures. Optimization results for a 3D task are given and compared with a Simple Genetic Algorithm. They clearly show the superiority of the Multi-Chromosome representation and adaptive operators in term of computing time and criteria optimization performance.

1 Introduction

Considering the task diversity in modern robotics, new solutions are required to answer the complex issue of adaptation. Modular Robotic Systems (MRS) have been proposed for adapting robot morphologies to given tasks [7][8][18][20]. MRS are minimal systems, able by recombining their modules to adopt a task based qualified kinematic structure. Thus, task diversity is answered by combination diversity.

The use of both deterministic and stochastic methods have been investigated for task based optimization of modular systems. Possible combinatorial explosions and the complexity in functions which relate the system state and task specifications (highly non-linear, discontinuous and with discrete variables in modular robotics) introduce great difficulties in the use of deterministic methods [1]. It is well known for instance that methods as hill-climbing can be easily trapped in local extrema. One can also exploit heuristics which consider each module characteristics to configure the system structure [5][9]. As a consequence, for a given problem the rule-based optimization always yield to the same solution.

Methodologies based on stochastic optimization algorithms search for best solutions with partially random operators. When considering multi-objective optimization, progressive or global methods are proposed. Progressive methods try to match iteratively the different successive constraints, to reach sequentially optimized solutions [12][17]. Unfortunately, strongly coupled constraints imply numerous backtrackings, which would slow down the search process. Global methods consist in trying to match simultaneously all constraints while optimizing some criteria [3][4]. This approach is much more complex than the previous one, but it constitutes the most promising way because of the parallel search it involves.

Our work belongs to this last category and makes use of Evolutionary Algorithms to generate optimal assemblies from an available set of components with different basic features.

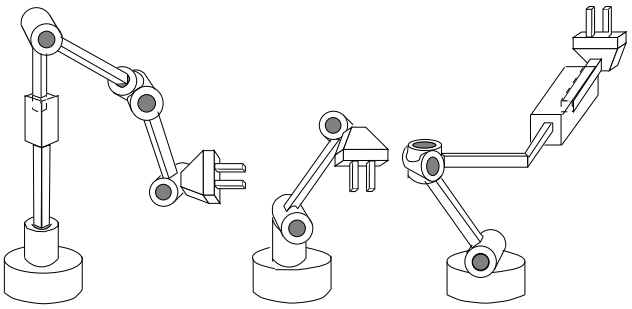


Figure 1: Modular manipulators

Previous works based on this approach yield to interesting results but under restrictions on the problem solved. Some of them suppose a particular geometry (planar or with three intersecting axes), others a fixed kinematic feature (joint type or general mobilities) or no constraints in the optimization (no obstacle avoidance) and generally no redundancies. This paper proposes a new Evolutionary Algorithm to address the problem of kinematic design for serial modular manipulators from task specifications in general and under environmental and technological constraints.

The problem consists in finding the fittest modular robotic system constructed from a serial assembly of a mobile base, links and joints as well as a set of joint parameters and a base location. Several joint types and mobility for the system (possibly redundant) are considered. The task is specified as a set of goals which are 3D end-effector configurations. The workspace is encumbered by obstacles simply represented by spheres (figure 2). The topology is evolved to perform the task in the best conditions which are evaluated through a multi-objective function relying on different functional attributes of the system.

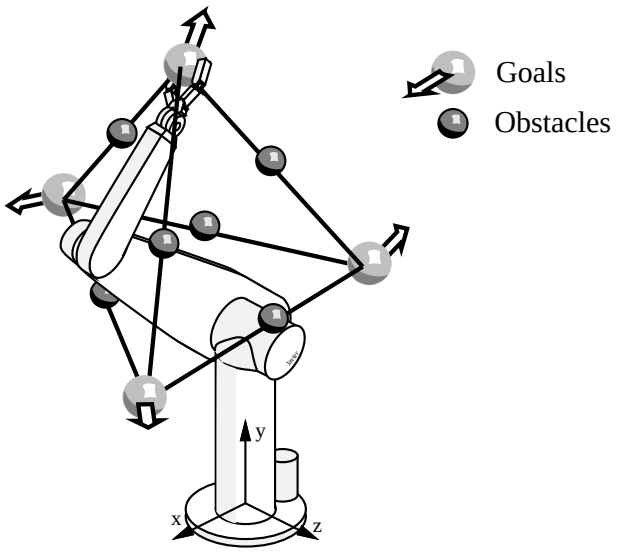


Figure 2: Task specification

In the following sections, we first introduce an adapted modular kinematics as well as the applied evaluation criteria. Then, we present a Multi-Chromosome Adaptive Genetic Algorithm and its genetic operators. Results are given for a 3D task and compared to those obtained by other GAs.

2 Problem Statement

2.1 Modular Kinematics

The modules and their assembly pattern are the design parameters for the optimization problem. The manipulator is constituted by two types of modules: joints and links. Two 1-DOF joint types (revolute (R) and prismatic (P)) and one 0-DOF joint (fixed (F)) are available, as well as up to sixteen link lengths (from 0 to 1 meter). As we consider only serial manipulators, assembly is made with successive segments. Each segment will be constituted by a joint followed by a link. Figure 3 shows the different available modules and their assembly possibilities. The basic transformation from a link (i) to its neighbour $(i+1)$ is given by an homogeneous transformation:

$$\boldsymbol{B}_i^{i+1} = R_{x,y,z}^{90} R_z^{\theta} T_x^{\tau} T_x^{\gamma} \quad (1)$$

where
- R_a^b : Rotation along a axis of angle b
- T_a^b : Translation along a axis offset b
- θ : Rotation angle if revolute joint
- τ : Translation offset if prismatic joint
- γ : Link length

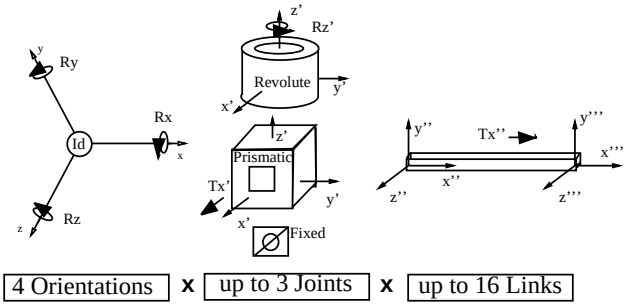
The structure equation of the mechanism may be computed as follows:

$$\boldsymbol{B}_g^e = \boldsymbol{B}_g^b \prod_{i=1}^{i=n} \boldsymbol{B}_i^{i+1} \quad (2)$$

where $\boldsymbol{B}_g^b$ is the transformation from a fixed frame attached to the ground and the mobile base frame, $\boldsymbol{B}_g^e$ specifying a configuration to reach by the end-effector.

Design parameters (**DP**) for a segment are:

1. Joint orientation (Id,Rx,Ry,Rz)
2. Joint type (R, P or F)
3. Link length (0; $\frac{1}{15}$; ..; $\frac{14}{15}$; 1)



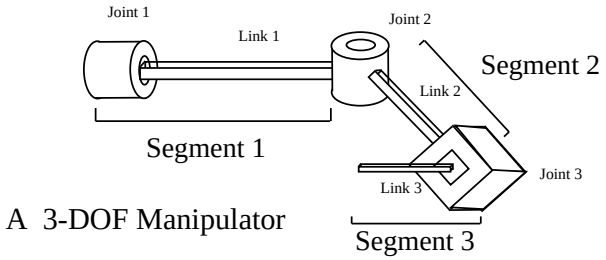


Figure 3: Module assembly

Evaluation is a key point since it drives the whole optimization process through fitness values it gives to candidate solutions. Hence, it requires a particular attention. Evaluation for optimization algorithms needs an Objective Function (F) to maximize. The F value (called fitness in EA) represents the solution worthiness and so, must be relevant to the system performances. Moreover, it has to be computable for each

solution the encoding is able to generate. The evaluation computing time is a critical point since it is performed for each individual over all generations.

In this problem, for each generated topology, the inverse kinematic problem for the whole structure (arm and base) need to be solved and this for each subgoal to satisfy. This basic problem in robotics has been widely studied by many authors. Efficient algorithms for the inverse kinematic problem are today available for manipulators without any restriction on their geometry but only for non-redundant manipulators [13][15]. Consequently, it has been decided to include the joint variables and the base location into the genetic material of each manipulator and then to solve the inverse kinematic problem simultaneously alongside the manipulator topology.

2.2 Evaluation Criteria

Six criteria have been selected here to qualify the evolving manipulators. All constraints and criteria are translated into penalty functions to be minimized. Criteria are computed for each subtask and the global mean value is used for the final evaluation. The different criteria are defined below:

- **Reachability**

The reachability is checked for all goals and a penalty R is set if the constraint is violated.

$$\boldsymbol{R} = \frac{(md - \sum_{i=1}^{n} l_i)}{md} \tag{3}$$

l_i : Length of i^{th} link (doubled if prismatic)
md : Maximum distance of task positions from base
n : Maximum number of DOF
nt : Number of goals

- **Linear and Angular distances**

Linear (L) and Angular (A) distances are used to assess the end effector configuration, according to the considered frame. The Euclidean distance is used for L and A, using the Roll Pitch Yaw convention (RPY)[19].
A and L are respectively:

$$\boldsymbol{L} = \frac{\|\boldsymbol{Xd} - \boldsymbol{Xe}\|}{\|\boldsymbol{Xd}\|} \tag{4}$$

Xd : Desired (goal) position vector
Xe : End effector position vector

$$\boldsymbol{A} = \frac{\|\boldsymbol{Rd} - \boldsymbol{Re}\|}{\|\boldsymbol{Rd}\|} \tag{5}$$

Rd : Desired angle vector
Re : End effector angle vector

Obstacle Proximity

For obstacle avoidance, we check whether each link is interacting or not with each obstacle by using a security ellipsoid (SE, defined by the link and sd). If the obstacle is inside the security ellipsoid (figure 4), they are interacting and we add a penalty amount O which is proportional to the real distance (from 0 to 1).

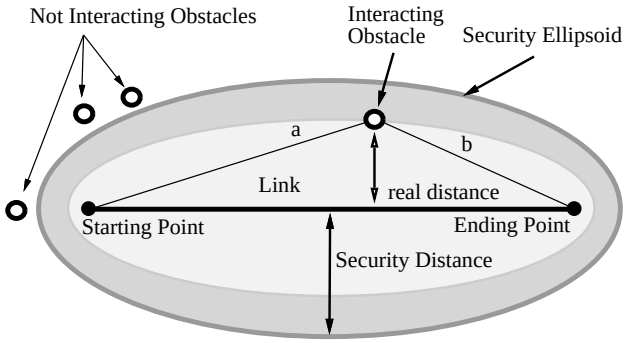


Figure 4: Obstacle avoidance ellipsoid

$$\text{if } Obstacle \notin SE_i \Rightarrow cp_{ij} = 0 \tag{6}$$

$$\text{if } Obstacle \in SE_i \Rightarrow cp_{ij} = \frac{(sd_i - d_{ij})}{sd_i} \tag{7}$$

$$\boldsymbol{O} = \sum_{i=1}^{n} \sum_{j=1}^{no} cp_{ij} \tag{8}$$

SE_{ij} : Security ellipsoid for i^{th} link
d_{ij} : Distance of i^{th} link from j^{th} obstacle
cp_{ij} : Collision penalty for i^{th} link and j^{th} obstacle
sd : Security distance
no : Number of obstacles

- **Involved modules**

The involved module criterion I includes total link lengths and number of joints. we use I in order to minimize total mass and structure complexity with regard to the end effector configuration. I is defined as:

$$\boldsymbol{I} = \frac{\sum_{i=1}^{n} l_i + \frac{rt_i}{20}}{ed} \tag{9}$$

rt_i : Set to 1 if i^{th} joint is R, 0 if P
l_i : Length of i^{th} link (x2 if P)
ed : End effector distance from base frame

Dexterity

The dexterity D is evaluated using the Yoshikawa manipulability index w [21]. This index is proportional to the manipulability ellipsoid volume and so,

represents the distance of the pose from a kinematic singularity. This distance is maximized by minimizing D defined for each pose :

$$w = \sqrt{Det(JJ^t)} \tag{10}$$

$$\boldsymbol{D} = \frac{1}{1+w} \tag{11}$$

w : manipulability index
J : Jacobian Matrix

Objective Function

All evaluated criteria are minimized since they are positive penalty functions. A positive (and bounded by 1) objective function F (called fitness) is designed from these criteria and used by the EA to evaluate candidate solutions. F is defined as:

$$\boldsymbol{F} = e^{-(k1*L+k2*A+k3*I+k4*O+k5*D)} \tag{12}$$

k_i : Weight of i^{th} criterion (let free)

The optimization goal is then to find a set of design parameters (DP) which maximizes the objective function (F) according to the priorities (k_i).

3 A Multiple Chromosome - Evolutionary Algorithm

Genetic Algorithms (GA) are stochastic algorithms which simulate the evolution of a population. Individuals are candidate solutions for the optimization problem and they are encoded into chromosomes to be manipulated by genetic operators [11]. We started with the Simple GA introduced by Goldbeg [10], using the basic operators (Selection, Crossover and Mutation). Many improvements have been brought since by the Evolutionary Computing community and some of them were used here [2][16]. Instead of solving the sub-problem of the inverse kinematics during the evaluation phase as we proposed in a Two-level GA [6] (TGA), we search simultaneously for topology and configurations by using a single genome for both subsets of parameters. This avoid interrupting the global evolution for the determination of the configurations, and greatly accelerate the evolution process.

The basic problem in gathering all the genotype (structure and configurations) on a single binary chromosome is the inefficiency of crossover for long strings (more than 300 bits for our example). In fact, longer the chromosome is, less effective the crossover is because it annihilate most of the genes ordering for changing just one gene value. When the algorithm tries to reach a task configuration, others are likely to be discarded whatever their own quality.

To prevent this problem, we propose to distribute informations over several chromosomes. Each chromosome gathers some highly linked informations which so, are not disturbed by a global crossover. For instance, each configuration has its own chromosome and is not concerned with what happens to others. Each chromosome undergoes its own crossover and is independent with regard to this operator. Moreover, as a chromosome is constituted by naturally linked parameters (as a configuration), each chromosome can be locally evaluated and a decision can be made by the genetic operators concerning its manipulations.

In fact, all chromosomes are globally linked via the topology chromosome, but some of them are more particularly dedicated to unrelated parts of the objective function.

The Adaptive Multiple Chromosome Evolutionary Algorithm (AMEA) takes advantage of this fact and allows non-disruptive crossover for the topology and the configurations. Moreover, the AMEA (figure 5) is much more faster than TGA since it includes only one evolution loop.

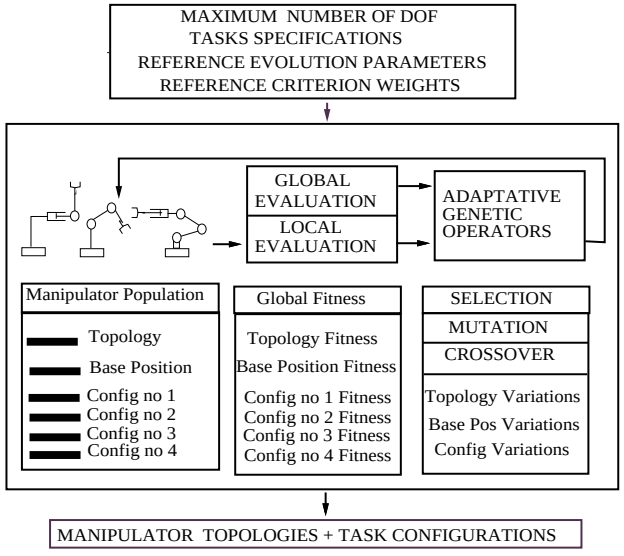


Figure 5: Adaptive Multiple-Chromosome EA

3.1 Encoding

Encoding is the representation of candidate solutions (genotype) and is directly manipulated by the EA and its operators. Binary coding is used for topologies and real coding for base position and configurations (figure 6). These choices have been made because a modular topology is better described by integer values (non-valued parameters) and real values avoid the loss of precision in configurations involved by discretization.

Each segment (figure 3) of the manipulator is coded

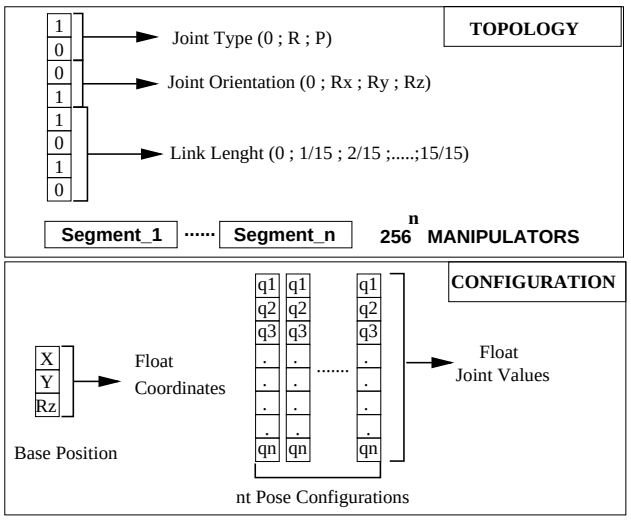


Figure 6: Heterogeneous Genome

with 8 bits. The first two bits represent the relative orientation of joint axis (Id, Rx, Ry, Rz) and the next two bits define the joint type. The four remaining bits encode the link length (16 lengths from 0 to 1). The whole manipulator encoding is obtained by the concatenation of the binary representation of segments. This gives a n-octet long binary string for a n-DOF manipulator. The base and joint configurations are encoded with real valued genes and put in separate chromosomes. Real values are adimensioned (from 0 to 1) and coded with 6 digit floats to allow 10^6 possible values.

The search space for the topology includes 256^n individuals and there are $256^n * 10^{6n*nt+18}$ solutions for a n-dof manipulator and nt goal frames.

This kind of mixed genetic material allows to keep the advantages brought by the binary representation, according to the fundamental theory of GA [6][11]. But also, it improves the precision for joint values and so for end effector configurations. Moreover, the real encoding is more effective than the binary for necessary small modifications in the later stages of optimization when the perfect configurations are very close.

3.2 Adaptive Genetic Operators

Genetic operators are successively applied to the chromosomes (representing kinematic structure or configurations) in an iterative way to simulate the evolution process of successive generations. Fundamental operators are **Selection, Crossover and Mutation**.

The **selection** picks out individuals in the population according to their fitness, to form the mating pool which will be used to breed the new generation. The remainder stochastic sampling with replacement based on the roulette wheel selection is used [10]. Selection pressure is relevant to the philosophy of the evolution. If a quick convergence, toward a single solution is needed we insist on exploitation, but if an extensive search, with numerous different solutions is necessary we advantage exploration. Increasing or decreasing the selection pressure is made by smoothing or emphasizing the fitness discrepancies. We decided for an adaptive selection pressure which will act upon the Objective Function via a hardness coefficient $\boldsymbol{\lambda}$ which is controlled by the fitness standard deviation $\boldsymbol{\sigma}_f$. If the solutions are widely scattered on the fitness landscape (large σ_f), we need to select more thoroughly and so, a hard selection pressure is required (large λ). In the other way, if most solutions are tightly clustered near a mean value, we let other individuals a chance to survive and so, the fitness function is softened by decreasing λ. Thus, this adaptive scaling will help the selection operator to keep a diversified population. Meanwhile, this strategy has to be modified to force convergence toward high fitnesses. So, we use also the mean fitness to harden the fitness function for high mean fitness. Thus, λ ranges from zero to the reference hardness λ_o.

$$\lambda = \lambda_o(\tilde{f} + \sigma_f) \quad and \quad fm = f^{\lambda} \tag{13}$$

λ_o : Reference selection pressure
$\tilde{f}$: Population mean fitness
fm : Modified individual fitness

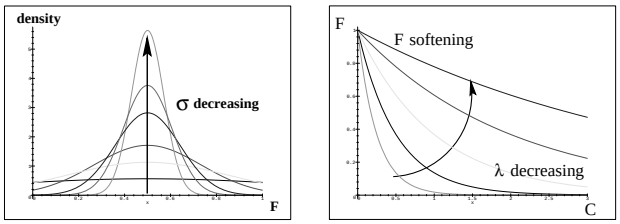


Figure 7: Selection Adaptation to Diversity

The **crossover** (or crossing-over) allows the building blocks (highly fit pieces of genetic code), useful but dispatched among the population, to be gathered inside the same individual by the effect of chromosomal brewing (sexual reproduction). A uniform crossover is chosen instead of the simple one point crossover for binary strings because of the decreasing crossover efficiency for long chromosomes. For real encoding, the uniform crossover consist in exchanging genes (floats) between both parents chromosomes. To prevent the configuration crossover to be too disruptive, we modify the permutation probability (initially 0.5) according to the local fitness of each configurations to disadvantage the highly fitted parameter migration.

$$\hat{f}_{\alpha i} = \frac{f_{\alpha i}}{\tilde{f}_{\alpha}} \tag{14}$$

$$p_{\alpha\beta i} = \frac{0.5}{\hat{f}_{\alpha i} * \hat{f}_{\beta i}} \tag{15}$$

$\hat{f}_{\alpha i}$: Relative local fitness, robot α config i
$\tilde{f}_{\alpha}$: Mean local fitness, robot alpha
$p_{\alpha\beta i}$: Permutation probability for i^{th} config

Mutation allows new building blocks to appear in population and lost ones to reappear. This mutation operator will randomly change a bit value in the population chromosomes for binary strings. Adaptation is made by tuning the mutation probability according to each fitness. For real valued chromosomes, a normally distributed variable z is added to the float number. The real mutation also adapt itself to the evolution by modifying the standard deviation σ according to the fitness of each individual.

- Binary Mutation

$$pm = \frac{pm_o}{F} \tag{16}$$

- Real Mutation

$$\sigma_i = \frac{\sigma_o}{f_i} \ , \ z = N(0, \sigma^2) \tag{17}$$

pm_o : Minimum binary mutation probability
σ_o : Minimum mutation standard deviation
σ_i : Local mutation standard deviation
f_i : Local Fitness value
z : Normal random variable

3.3 Evolution Parameters

Once genetic operators are designed, we have to tune the reference evolution parameters before the optimization. Evolution parameters are:

- λ_o : Reference selection pressure
- pc_o : Reference crossover probability
- pm_o : Reference mutation probability
- S : Size of population
- T : Maximum number of generations

Without any theorical certitudes about genetic operator influence, their occurrence probabilities have to be adjusted for each optimization problem. For an adaptive algorithm, where these influences will be modified during the process we need to set reference values to prevent the adaptation to become instable and meaningless. All adapting parameters are based upon references values which have been tuned and used for the non-adaptive case.

λ_o=1, pc_o=0.6,σ_o=0.001, pm_o=0.001, S=20, T=50

4 Simulation Results

We used three different algorithms for the same 3D task specifications. These algorithms are the Two-level GA (TGA) , the Multi-Chromosome Evolutionary Algorithm (MEA) and the Adaptive Multi-Chromosome Evolutionary Algorithm (AMEA).

Area	TGA	MEA	AMEA
Coding	all binary	Binary/Float	Binary/Float
Evolution	Separate	Simultaneous	Simultaneous
Operators	Static	Static	Adaptive

For the manipulator evolution example, the global task area is represented by a tetrahedron (figure2). Four goals have been defined on its vertices and six obstacles on the middle of its edges. The manipulator evolves to reach these goals, while optimizing all performance criteria and avoiding obstacles. All criteria weight are set to 1 and the security distance is set to 0.1 meter. Evolution parameters are those described in the previous section and the evolved topologies includes until 8 degrees of freedom, allowing redundant manipulators. The total search space includes more than $1.8 * 10^{229}$ solutions (n=8, nt=4) and we explore only $20 * 50 = 1000$ solutions during the evolution.

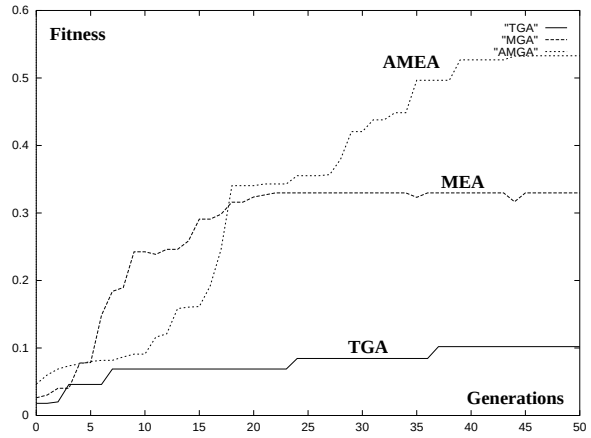


Figure 8: Best fitness comparison

Figure 8 shows the best individual evolution with the TGA, the MEA and the AMEA. The results have to be scaled because of the difference of time processing between the TGA (6 hours) and the new algorithms (3 minutes) to perform the 20*50 evolution (SUN/SPARC5). We clearly see that not only the multi-chromosome algorithms are 120 times quicker, but also that the optimized best solution is far better.

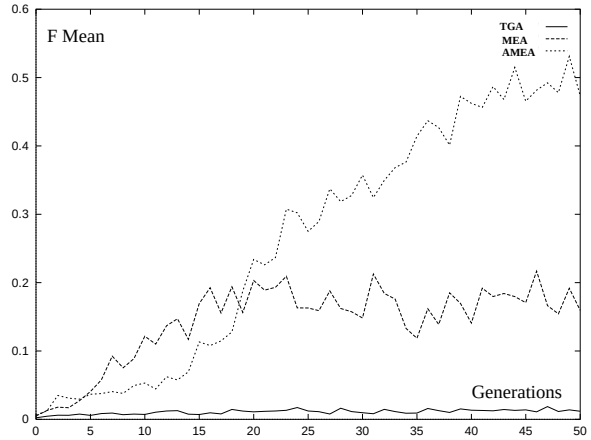


Figure 9: Mean fitness comparison

Figure 9 shows the mean fitness evolution for TGA, MEA and AMEA. The differences here are more striking

as we see that the population is much more improved with new EAs. This shows that rhe adaptive algorithm continues the improvement of the whole population as long as the best fitness is not reached for all. In fact, the TGA and MEA will always keep a constant gap between best and mean fitness because of the constant mutation which perturbate the population while the AMEA will decreases the mutation noise with increasing fitness. This is similar to the Simulated Annealing temperature which decreases with time, allowing less and less variations and finally drives all individuals toward a single solution.

5 Conclusion

The adaptive EA we presented is an adaptation of a previous algorithm (TGA) which includes a representation based upon the particularity of the 3D modular kinematics utilized. The Multi-Objective optimization is performed in quite a short time with regard to the total search space and the complex links which exist between the design parameters and the objective function.

We used none database or robotic design experience to give rise to solutions from random seed. Evolved topologies showed rather good performance criteria as we asked to the genetic optimization and allowed to access to several solutions which represent some making sense options for this kind of design. The method is very flexible and adaptable as we can chose through the fitness function any criteria to optimize whatever its form and relations with the encoded solutions. It is important to notice that the whole manipulator and its configurations have been optimized in parallel. This guarantee a global optimization of both structure and behavior (configurations). For now, the behavior is reduced to joint static values, but it can quickly evolve to kinematic or dynamic laws which could be evaluated through a simulation process for complex tasks

Designing mechanisms for more complex robots (parallel manipulators, walking robots) demand some important adaptation for all the parts of an EA, specially with representation and might need to make some hybrid algorithms, using some existing technics in AI. Eventually, the lack of optimality this method involves might be compensated by hybridization with efficient classical optimization technics or even human interaction.

References

[1] J.S. Arora, O.A. Elwakeil and A.I Chahande, "Global optimization methods foe engineering applications : a review" *Journal of Structural Optimization*, 1995

[2] T. Back, *Evolutionay Algorithms in Theory and Practice* 1996

[3] P. Chedmail and E. Ramstein, "Robot Mechanism Synthesis and Genetic Algorithms" *Proc. IEEE of ICRA*,1996

[4] I. M. Chen and J. W. Burdick, "Determining Task Optimal Modular Robot Assembly Configurations" *to be published*,1996

[5] M. Chew, S. N. T. Shen, G. F. Issa, "Kinematic Structural Synthesis of Mechanism Using Knowledge-Based Systems" *Journal of Mechanical Design, ASME Transactions*,1995

[6] O. Chocron and P. Bidaud, "Genetic Design of 3D Modular Manipulators" *accepted paper IEEE ICRA*,1997

[7] T. Fukuda and S. Nakagawa, "A Dynamically Reconfigurable Robotic System (Concept of a system and optimal configuration)" *IECON'87*,pp 588-595,1987

[8] S. Farritor, S. Dubowsky, N. Rutman and J. Cole, "A Systems-level Modular Design Approach to Field Robotics" *Proc. IEEE ICRA*, 1996

[9] T. Fukuda, S. Nakagawa, "Dynamically Reconfigurable Robotic System" *Proc. IEEE of IROS*, 1988

[10] D. E. Goldberg, *Genetic Algorithms in search, Optimization and Machine Learning* Addison Wesley, 1989

[11] J. H. Holland, *Adaptation in natural and artificial systems* Ann Arbor: The University of Michigan Press, 1975

[12] J. O. Kim and P. Khosla, "A Multi-Population Genetic Algorithm and its Application to Design of Manipulators" *Proc. IEEE of IROS, Raleigh, NC*, 1992

[13] D. Manocha, Y. Zhu, "A Fast Algorithm and System for the Inverse Kinematics of General Serial Manipulators" *in Proc. IEEE ICRA*, 1994

[14] T. Matsumaru "Design and Control of the Modular Robot Systems: TOMMS" *Proc. IEEE of ICRA*,1995

[15] C. Mavroidis, F.B. Ouezdou and P. Bidaud, "Inverse kinematics of six-degree of freedom 'general' and 'special' manipulators using symbolic computation" *Robotica (1994)*,volume 12, pp421-430 1994 Cambridge University Press

[16] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* Springer-Verlag, 1992

[17] C. Paredis and P. Khosla, "Kinematic Design of Serial Link Manipulators from Task Specifications" *International Journal of Robotics Research*, Vol. 12, No. 3, pp. 274-287, June 1993.

[18] C. J. J. Paredis, et al., "A Rapidly Deployable Manipulator System," *in Proc. of IEEE ICRA*, Vol. 2, pp. 1434-1439, Minnesota, Minneapolis, April 22-28, 1996.

[19] L. Sciavicco, B. Siciliano, *robotica industriale: modellistica e controllo di manipulatori* McGraw-Hill, serie di tecnologia, 1995

[20] Yim, M., "A Reconfigurable Modular Robot with Many Modes of Locomotion" *Proc. of the JSME Int. Conf. on Advanced Mechatronics*, pp. 283-288 Tokyo Japan 1993

[21] T. Yoshikawa, *Fundations of robotics: Analysis and control* The MIT Press, 1990