Evolutionary Design of a Robotic Manipulator for a Highly Constrained Environment

Sébastien Rubrecht, Ekta Singla, Vincent Padois, Philippe Bidaud, Michel de Broissia

Chapter 1
Evolutionary Design of a Robotic Manipulator for a Highly Constrained Environment

S. Rubrecht, E. Singla, V. Padois, P. Bidaud, M. de Broissia

Abstract This paper presents the design of a manipulator working in a highly constrained workspace. The difficulties implied by the geometry of the environment lead to resort to evolutionary-aided design techniques. As the solution space is likely to be shaped strangely due to the particular environment, a special attention is paid to support the algorithm exploration and avoid negative impacts from the problem formulation, the fitness function or the evaluation. In that respect, a specific genome able to encompass all cases is set up and a constraint compliant control law is used to avoid the arbitrary penalization of robots. The presented results illustrate the methodology adopted to work with the developped evolutionary-aided design tool.

1.1 Introduction

In the field of robotic manipulator design, the classical methods [1] turn out to be inefficient when the problem is highly constrained, as the expressions of the constraints (obstacles) cannot be formalized into a classical design formulation. Thus, it is hard to check if a solution complies with the constraints. Moreover, the solution space may be very large, and as the validations are time consuming, it is relevant

S. Rubrecht, M. de Broissia
Bouygues Travaux Publics
1, av. E. Freyssinet, 78062 St Quentin en Yvelines
e-mail: {s.rubrecht, m.debroissia}@bouygues-construction.com

E. Singla, V. Padois, P. Bidaud
Université Pierre et Marie Curie
Institut des Systèmes Intelligents et de Robotique
4 place Jussieu, 75005 Paris
e-mail: {singla, padois, bidaud}@isir.upmc.fr

to use performance indicators and to consider the problem as a multiobjective optimization.

The presence of multiple objectives in a problem gives rise to a set of optimal solutions, instead of a single optimal solution. This set of solutions is known as the set of Pareto-optimal solutions and rely on the notion of *Pareto-dominance* [5] to treat simultaneously and independently each performance indicator.

In a typical minimization problem where the fitness f is composed of n functions f_i $(1 \leq i \leq n)$, a solution $\mathbf{x}$ is dominating an other solution $\mathbf{x}'$ if

$$\exists\, i \text{ such as } f_i(\mathbf{x}) < f_i(\mathbf{x}') \tag{1.1}$$

and

$$\forall j \neq i, f_j(\mathbf{x}) \leq f_j(\mathbf{x}') \tag{1.2}$$

Based on this principle, the solution of a multiobjective optimization is a set of non-dominated solutions (Pareto-optimal solutions) to the problem . In the absence of any further information, one of these Pareto-optimal solutions cannot be said to be better than the other.

Evolutionary Algorithms (EAs) have been widely used in robotics design optimization ([2],[3]) as they are very well adapted for optimization over vast, non continuous search space. This field of application of EA is a growing trend and is mentionned as *evolutionary aided design* in the introductory chapter of this book. One of the first robot design problems using evolutionary algorithm was carried out by Sims [4], generating creatures competing in walking, jumping, swimming, etc.

Since 1990, a large number of MultiObjective Evolutionary Algorithms (MOEAs) has been proposed ([5], [6], [7], [8], [9], [10]). The primary reason for this is their ability to find multiple Pareto-optimal solutions in one single simulation run. Since EAs work with a population of solutions, a simple EA can be extended to maintain a diverse set of solutions. Evolutionary algorithms are now widely used, from the whole system structure design to robots reconfiguration [11], controller design, and in various domains such as cooperative robotics [12] and mini-invasive surgery [13]. Amidst several works presented for optimal designs of fundamental robots, Snyman et al. utilized in [19] Evolutionary Algorithms for the design of a 3R industrial robot while aiming at minimizing joint torque over an entire given trajectory. Another eminent contributions by Ceccaralli and Lanni ([20]) and Carbone et al. in [21] involved the formation of the robot design problem as multiobjective optimization problems. However, the complexity associated with cluttered environments and larger number of degrees of freedom is left unaddressed.

This paper details some of the key issues in the design of a robotic serial arm in a highly constrained environment with a special attention to keep the best conditions for the EA to explore the solution space. In that scope, the way to set up the problem (genome choice), the algorithm itself (type and genetic operators), the evaluation step (trajectories, control law) and the indicators retained are fundamental elements. The work in [22, 23, 24] come close to the presented approach. In general, these approaches employ modular robots to cater the task specifications and

recommend specific encoding systems to employ evolutionary algorithms to handle varying number of degrees of freedom. However, the work presented in these papers is limited to some specific exemplary workspaces and trajectories. In section 1.2, the environment and context of this design is presented. Section 1.3 details the resolution method, from the problem analysis to the implementation. Section 1.4 exposes the first results. Finally, the last section presents concluding remarks and the future work to be done on this subject.

1.2 Case Study

This research work is lead within the framework of a project dedicated to Tunnel Boring Machine (TBM, see Fig. 1.1). The usual tasks are maintenance operations in hostile conditions: hyperbaric atmosphere, high temperature, and even operation immerged in mud.

The geometry of the problem is a typical excavation room geometry (diameter 10 m, depth 1 m). The missions defined for fitness are trajectories tracking all around the upper part of the cutter head, focusing on key points to clean or inspect. Transmission arms are obstacles to take into account. The basis of the robot is fixed near the top of the excavation room, at the exit of the airlock.

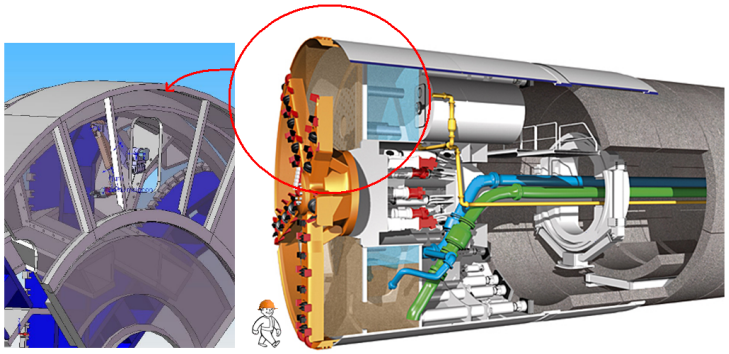

Fig. 1.1 Example of a manipulator in a TBM.

The robots (EA individuals) are composed of elements taken from a pool of robot segments inspired (shapes and joint limits) from the real robot segments of the Maestro manipulator (Cybernetix [1], see Fig. 1.2). In particular, only 1 degree of freedom (DOF) rotational joints are allowed.

[1] http://www.cybernetix.fr/

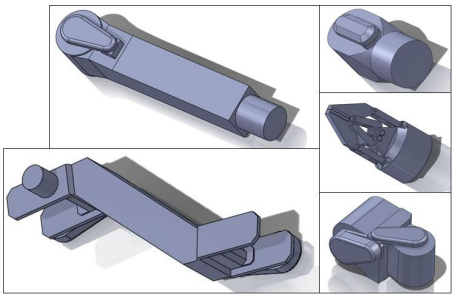

Fig. 1.2 Example of robot segments of the Maestro manipulator from which the elements of the individuals are inspired.

1.3 Genetic Algorithm and Implementation

In this section, our implementation of the genetic algorithm is introduced. This implementation relies on SFERES [15] which provides a general framework for evolution based optimization.

1.3.1 Genetic Algorithm

The efforts made to approach the Pareto-optimal front involves two (possibly conflicting) objectives. First is the *convergence* — minimizing the distance between the final pareto front and the optimal front and second is the *diversity* — maximizing the difference in the generated solutions in terms of objectives or parameter values. To consider both items, the popular technique of Nondominated Sorting Genetic Algorithm II (NSGA-II) [10] is considered suitable for our design problem. This technique possesses the features of elitism and parameter-less sharing. Elitism is the process of selecting better solutions out of the *combined population* of parent and child generations and, therefore, avoid the degrading of any good solution.

For a problem with the population size as N, NSGA-II works on $2N$ solutions at each iteration. These solutions are sorted with respect to their non-domination and are arranged into different Pareto optimal fronts. This is termed as *non-dominated sorting*. To send N solutions to the next iteration, a new list is formed. Since each Pareto front contains equally good candidates, therefore, unless there is less space than the number of elements in a front, all the elements of each front are kept adding to the new list. For further sorting at a particular level, say $r-th$ front, *crowded distance sorting* is utilized. Based on this, the upper ranked elements of the $r-th$ Pareto front are included in the new list. This sorting is based on the maximum distance available around an element, in the objective function space, within which there exists no other element. This helps maintaining some significant *diversity* in the resulting solution, by selecting widely spread population.

The genetic operators are generally used at various rates, in our case values are taken around

- Mutation rate: 10%
- Cross over rate: 13%
- Generations: 500
- Individuals: 150

1.3.2 Genome

The design process focuses exclusively on the robot morphology using the elementary segments shown in Fig. 1.2. In that framework, each robot is described as a concatenation of segments. A segment is composed of a link having a joint (rotational or prismatic[2] or not. Two frames are associated to each elementary segment. The first one represents the three possible joint axes: every link is oriented along its z axis. The second one represents the three possible orientations of the next segment.

According to this description, there are 11 elementary segments:

- 3 with a rotational joint about the x axis, the following segment being oriented along x, y or z (called **rxx**,**rxy**, and **rxz** respectively)
- 3 with a rotational joint about the y axis (**ryx**, **ryy** and **ryz**)
- 2 with a rotational joint about the z axis (**rzx** and **rzz**)
- 3 segments without joint (**ex**, **ey** and **ez**)

rzy is not mentionned as it is the same as **rzx** rotated by $\frac{\pi}{2}$ *rads* around z axis.

In addition we define 10 possible lengths for the segments between 0.05 m and 1.05 m. The association table is presented in Table. 1.1.

As an example, a portion of a robot is represented on Fig. 1.3 (left). Each robot is defined by a chromosome of 16 genes, each one representing a segment or not: the genes from 100 to 109 does not match anything (segment "None") Actually, as we do not want every robot to have 16 DOFs, we define genes that do not match anything (segment "none").

When a fixed segment appears in the genotype of an individual (gene from 190 to 219), a segment combination is done, thus offering the possibility to have segments which orientation differs from the x, y and z axes (Fig 1.3 right).

1.3.3 Trajectory tracking

The aim of the fitness function is to qualify the ability of an individual (robot) to carry out a maintenance mission in the TBM. An efficient way to check the motion skills of a robot is to simulate a trajectory tracking in the 3D environment. So, a

[2] Prismatic joints are not used within the framework of the considered application.

Table 1.1 Genome. Each gene is a number composed of 3 digits: the 2 first are the joint type, the last being the link length. A gene value is between 100 and 219.

Gene **ABC**	**AB**: Joint type
	C: length (m)

Joint number - **AB**	10	11	12	13	14	15
Joint type	None	rxx	rxy	rxz	ryx	ryy

Joint number - **AB**	16	17	18	19	20	21
Joint type	ryz	rzx	rzz	ex	ey	ez

Length number - **C**	0	1	2	3	4
segment length (m)	0.05	0.15	0.25	0.35	0.45

Length number - **C**	5	6	7	8	9
segment length (m)	0.55	0.65	0.75	0.85	0.95

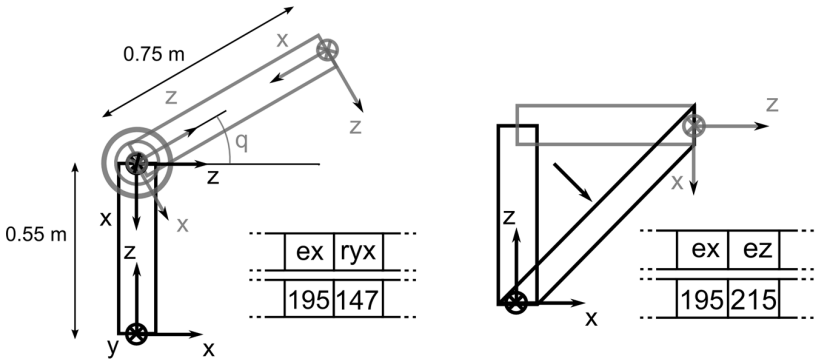

Fig. 1.3 Genotypes examples: portion of robot and combination of 2 segments. q is the rotational joint angle.

relevant 3D trajectory has been defined (sequence of 361 3D points for a total length of approximately 8.5m, which comes out to a mean distance of 24mm between 2 points) and the fitness function is a trajectory tracking. Dynamics is not computed as it does not impact on the indicators retained (see 1.3.5). Each individual has to track the same trajectory. The simulator uses the Kinematic and Dynamic Library (KDL) which is part of the OROCOS project [14].

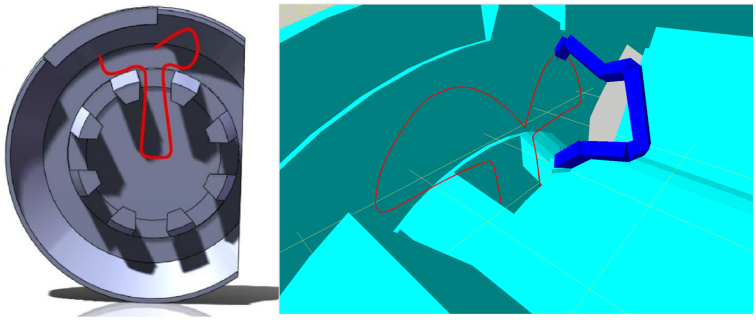

Fig. 1.4 Section view of the TBM trajectory and manipulator example.

1.3.4 Control Law

The control law of the robots is in charge of computing at each simulation iteration the joint velocities to reach the current point in the sequence of 3D points composing the trajectory. In our case, two specifications led us to design our own control law to handle the problem of tracking a given trajectory by any manipulator in a cluttered environment. First, a guideline of this work is to compensate the impact of cluttered environment by supporting the Evolutionary Algorithm exploration. Consequently, the control law should neither be penalizing in terms of configurations (e.g. to deal properly with singular configurations or proximity to constraints) nor in terms of robots (redundant or not). Second, the framework of evolutionary aided design requires meaningful fitnesses to be efficient: so, the constraints violations (such as collisions between the robot and the environment), which would never occur in real conditions, are not accepted.

1.3.4.1 Control framework

The huge number of individuals evaluations (trajectory trackings of manipulators) prevents from using prediction or planification techniques in the control strategy (for obvious computation time reasons), so the control law is reactive. Regarding the framework, a velocity kinematics framework has been retained rather than a purely kinematics one. Actually, as the robot is not known a priori, a general inversion method is needed to be applied at each simulation iteration. In the kinematic framework, the model linking the joints and the operational positions is not linear and this operation is complex and time consuming.

1.3.4.2 CCC

The control law briefly described here - Constraint Compliant Control (CCC) - is detailed in [25]. The CCC is an iterative velocity kinematics control law. It is a strict prioritized multiobjective law ([16], [28]) with 3 hierarchical levels:

1. The first level gathers the terms relative to *passive avoidance*: to satisfy the considered constraints (collisions) formulated as inequalities, the robot motion is stopped along the directions of the critical constraints. The critical constraints are determined through an iterative process;
2. The second level gathers the operational tasks, in our case the trajectory tracking;
3. The third level gathers the active avoidance terms: it tends to get the robot away from the constraints. Most of the time, as the environment is cluttered, these terms cannot be all satisfied, which justifies the existence of the first term.

The main property of the CCC is that it never violates its constraints, even if they are not compatible with the trajectory tracking. As a result, indicators related to these constraints are useless in the design process. Moreover, the CCC resorts to the Damped Least Square (DLS) inverse [17], to avoid inconvenient behaviors around kinematic singularities. As a consequence, the compliance with the constraints and the approximation around singularities directly impacts the trajectory tracking error rather than imposing dedicated indicators. It appears more relevant as it limits the number of indicators without creating meaningless weighted sums of scores based on non realistic behaviors.

1.3.4.3 Practical implementation

In practice, a computationally efficient implementation of the CCC does not perfectly ensure collision avoidance (one constraint per segment may not always be sufficient, see [25]). So, the number of collisions per segment per iteration is included in the set of indicators. Anyway, the use of the CCC is justified as only a small portion of the evaluated robots collides, which minimizes the impact of this indicator on the complexity of the problem (see 1.3.5).

1.3.5 Indicators

The indicators are the scores obtained by the robot through the fitness function. They are voluntarily simple and composed of a single magnitude (no weighted sums representing a priori tradeoffs between different magnitudes). The trajectory tracking quality but also intrinsic parameters are rated, such as the number of DOFs. All the indicators, listed below, are to be minimized:

- **Maximum linear error along the trajectory tracking.** There are no strategic points on which to compute the error with a higher weight w.r.t to others: the current design being a preliminary design, the trajectory should be equally tracked;
- **Number of DOFs.** The number of DOFs is a technological difficulty (manufacturing, energy, control), even if a more redundant robot has, in general, better reachability skills in a cluttered environment.
- **Robot total length.** The shorter robot able to perform the trajectory tracking has usually better adaptability to other tasks.
- **Number of collisions per segment per iterations.** As mentionned previously, despite the use of the CCC, the number of collisions is added as a fourth indicator to minimize. However, the CCC considerably reduces the number of collisions w.r.t. usual control laws. As a result, almost every robot obtain 0 (no collision), and the size of the problem is not much increased by the presence of this indicator. Actually, a robot failure on this indicator is most often due to a control failure rather than because of antagonism between indicators. In order not to penalize the individuals for which this failure occurs, it has been decided not to take this indicator as a *constraint* (in the Evolutionary Algorithm sense : criteria which, if not respected, disqualifies the individual).

1.4 Results

Even if the design process is still under progress, the results presented here are conclusive regarding our particular problem.

1.4.1 Design with simple trajectory

Preliminary designs have been realized with a simple trajectory to set up the process properly. The robot in Fig. 1.5 is a solution obtained with only 3 indicators:

- Maximum linear error of the trajectory tracking;
- Number of DOFs;
- Number of collision per segment per iteration.

The retained manipulator possesses 5 DOFs and tracks the path with a maximum error of 80mm (shown on Fig. 1.5). However, the robot cannot be considered acceptable as the link lengths are too large, with a total robot length of 6.40m. Such results encouraged to include the link sizes as one of the indicators of the optimization process.

In order to obtain more reasonable robots, the total length of the robot had been added to the set of objective functions. The resulting robot for the presented case is much shorter (1.60 m), as shown in Fig. 1.6. The number of DOFs is 5 and the

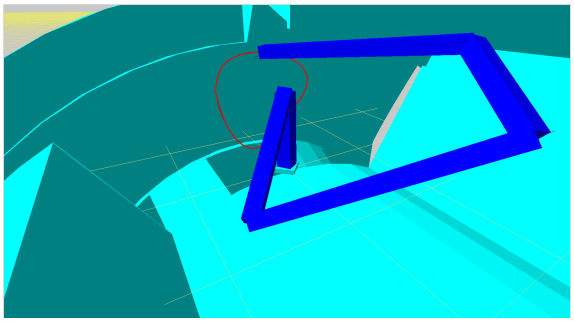

Fig. 1.5 Robot 1, indicators: linear tracking error, number of DOFs and collisions per segment per iteration.

maximum trajectory tracking error is 90*mm*, which remains acceptable and tends to prove that the robot total length is not antagonistic with other indicators in this case.

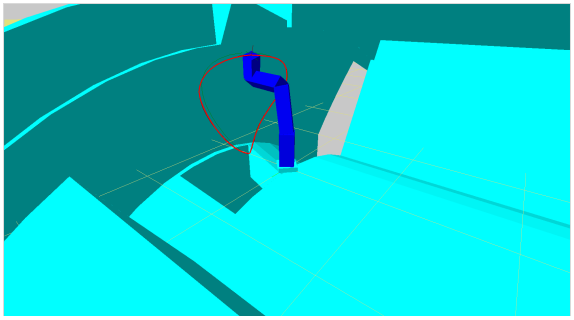

Fig. 1.6 Robot 2, indicators: linear tracking error, number of DOFs, collisions per segment per iteration and robot total length.

1.4.2 Design with complex trajectory

As the solutions fit the specifications for the simple trajectory, a similar work had been carried out with the trajectory representing a maintenance mission (inspection of the cutter head). This path includes the complications of navigating the robot deep into the narrow space, available between the cutter head and the robot base (see 1.3.3 and Fig. 1.4). Using the 4 indicators of linear error, number of DOFs, total length and number of collisions per segment per iteration, turned out to be sufficient to obtain appropriate robots.

The tracking of one of the final robots is represented in a sequence of simulation pictures, shown in Fig. 1.7. This robot has 5 DOFs with a maximal linear error of 12 cm and a length of 2.80 m.

The selection of a suitable robot out of all the possible solutions of the final front is another task in the complete design process, which is not a part of this paper. It is worth mentioning here that since collision avoidance is an inherent part of the chosen motion controller, the working of the resulting robots would certainly be free from any collision when using such a controller.

1.5 Conclusions and Future Works

1.5.1 Conclusions

The work presented here finds its justification in the maintenance of TBM in hostile conditions. The environment being very constrained, evolutionary design offers many advantages, but attention must be paid to preserve a good exploration as the solution space is not well shaped. A simple and exhaustive genotype has been set up to cover easily all robot possibilities, including segment directions not only along the absolute frame axes. The constraints compliant control law resorts to passive constraint avoidance to make the behavior more realistic. It avoids individuals penalization, makes indicators more meaningful and reduces the impact of the collision based indicator on the problem size. The results obtained are suitable solutions to our problem.

1.5.2 Future Works

As mentioned previously, a sensitivity analysis is needed to estimate the impact of our work toward diversity maintenance in the population. In addition, more sophisticated indicators are currently being tested, such as manipulability [18]. Finally, implementation of recent works in evolutionary design will be carried out, such as:

- variation of the genetic operators individual by individual according to their position and repartition along the Pareto front [26];
- evolution of the indicator along the process to prevent from bootstrap [27];

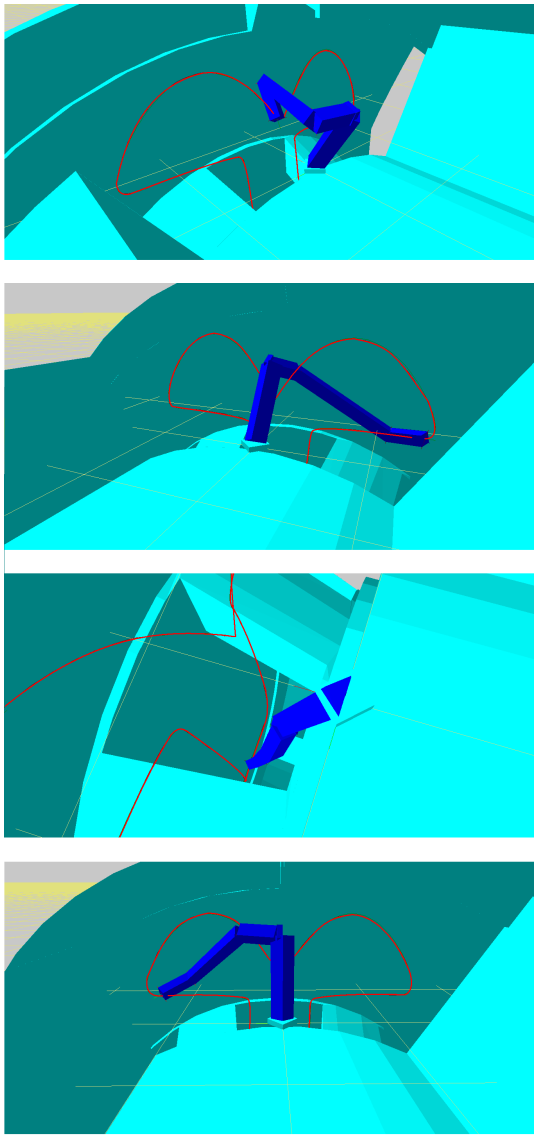

Fig. 1.7 Sequence of the complex trajectory tracking. Robot 3, indicators: linear tracking error, number of DOFs, collisions per segment per iteration and robot total length.

1.6 Acknowledgements

This work is a part of the TELEMACH project and is supported by the French National Research Agency (ANR), Interactive Systems and Robotics Program 2007 (PSIROB 07).

References

1. G. Pahl and W. Beitz, Engineering Design, *Springer-Verlag*, Berlin, 1984.
2. 0. Chocron and P. Bidaud, Genetic design of 3d modular robot topologies, *Proc. IEEE International Conference on Robotics and Automation*, 1997, pp. 223-228.
3. C. Leger, Automated synthesis and Optimization of robot configuration: an evolutionary approach, Ph.D. dissertation, Canegie Mellon University, 1999.
4. K.Sims, Evolving virtual creatures, *Computer Graphics (Siggraph '94 Proceedings)*, 1994, pp. 15-22.
5. K. Deb, Multiobjective Optimization Using Evolutionary Algorithms, *Chichester*, U.K.: Wiley, 2001.
6. C. M. Fonseca and P. J. Fleming, Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization, *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993, pp. 416-423.
7. J. Horn and N. Nafploitis and D. E. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization, *Proceedings of the First IEEE Conference on Evolutionary Computation*, 1994, pp. 82-87.
8. N. Srinivas and K. Deb, Multiobjective function optimization using nondominated sorting genetic algorithms, *Evol. Comput.*, vol. 2, no.3, 1995, pp. 221-248.
9. E. Zitzler and L. Thiele, Multiobjective optimization using evolutionary algorithms: A comparative case study, Parallel Problem Solving From Nature, textit Springer-Verlag, 1998, pp. 292-301.
10. K. Deb and A. Pratap and S. Agarwal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm : NSGA-II, *IEEE Transactions on evolutionary computing*, vol.6, no. 2, 2002, pp. 182-197.
11. J. Han and W.K. Chung and Y. Youm and S.H. Kim, Task based design on modular robot manipulator using efficient genetic algorithm, *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997, pp. 507-512.
12. C. C. Sotzing and W. M. Htay, and C. B. Congdon, Gencem: A genetic algorithms approach to coordinated exploration and mapping with multiple autonomous robots, *IEEE Congress on Evolutionary Computation*, 2005, pp. 2317-2324.
13. D. Salle and P. Bidaud and G. Morel, Optimal Design of High Dexterity Modular MIS Instrument for Coronary Artery Bypass Grafting, *Proceedings of the IEEE International Conference on Robotics and Automation*, vol.2, 2004, pp. 1276-1281.
14. H, Bruyninckx, Open Robot Control Software: the OROCOS project, *IEEE Int. Conf. Robotics and Automation*, 2001, pp. 2523-2528.
15. S. Landau and S. Doncieux and A. Drogoul and J.-A. Meyer, Sferes: un framework pour la conception de syst'emes multi-agents adaptatifs, *Technique et Science Informatique*, 21(4), 2002.
16. A. Liegeois, Automatic supervisory control of the configuration and behavior of multibody mechanisms, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7(12), 1977, pp. 868–871.
17. Y. Nakamura and H. Hanafusa, Inverse kinematics solutions with singularity robustness for robot manipulator control, *Trans. ASME Journal of Dynamic System, Measures and Control* vol.108, 1986, pp. 163–171.

18. T. Yoshikawa, Manipulability of robotic mecanisms, *The International Journal of Robotics Research*, vol. 4, n. 2, 1985, pp. 3-9.
19. J.Snyman and F.van Tonder, Optimum design of a three-dimensional serial robot manipulator, *Structural and Multidisciplinary Optimization*, vol.17, no.2, pp. 172–185, 1999.
20. M.Ceccarelli and C.Lanni, A multi-objective optimum design of general 3R manipulators for prescribed workspace limits, *Mechanism and Machine Theory*, vol.39, no.2, pp. 119–132, 2004.
21. G.Carbone, E.Ottaviano, and M.Ceccarelli, An optimum design procedure for both serial and parallel manipulators, *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 221, no.7, pp. 829–843, 2007.
22. P.Chedmail, E.Ramstein, and N. CMAO-Productique, Robot mechanism synthesis and genetic algorithms, in *1996 IEEE International Conference on Robotics and Automation, 1996. Proceedings.*, vol. 4, 1996, pp. 3466–3471.
23. G.Yang and I.Chen, Task-based optimization of modular robot configurations: minimized degree-of-freedom approach, *Mechanism and machine theory*, vol.35, no.4, pp. 517–540, 2000.
24. O.Chocron, Evolutionary design of modular robotic arms, *Robotica*, vol.26, no.03, pp. 323–330, 2008.
25. S. Rubrecht, V. Padois, P. Bidaud and M. de Broissia, Constraint Compliant Control, *In proceedings of the 12th conference on Advances in Robot Kinematics*, 2010.
26. O. Chocron, Evolutionary design of modular robotic arms, *Robotica*, vol. 26, 2008, pp. 323-330.
27. J-B. Mouret and S. Doncieux, Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity, *IEEE Congress on Evolutionary Computation*, 2009, pp. 1161-1168.
28. Siciliano, B and Slotine, J-J. A general framework for managing multiple tasks in highly redundant robotic systems. *IEEE International Conference on Advanced Robotics*, 1991, pp. 1211-1216.