Dual-Arm Robot Motion Planning Based on Cooperative Coevolution

Petar Ćurković and Bojan Jerbić

Faculty of Mechanical Engineering and Naval Architecture, Department of Robotics and Manufacturing Systems Automation, University of Zagreb, Ivana Lučića 5, 10000 Zagreb, Croatia
{petar.curkovic,bojan.jerbic}@fsb.hr

Abstract. This paper presents a cooperative coevolutionary approach to path planning for two robotic arms sharing common workspace. Each arm is considered an agent, required to find transition strategy from given initial to final configuration in the work space. Since the robots share workspace, they present dynamic obstacle to each other. To solve the problem of path planning in optimized fashion, we formulated it to multi-objective optimization domain and implemented co-evolutionary algorithm to simultaneously optimize four conflicting objectives. End-effector trajectory length, end-effector velocity distribution, total rotate angle and number of collisions are the objectives to be optimized. Simulation results for two 2-R type robots are presented.

Keywords: Co-evolution, path planning, multi-objective optimization.

1 Introduction

Over the last two decades, evolutionary algorithms have been applied in a variety of fields, namely: robotics, scheduling, construction engineering, speech recognition, space engineering, image processing etc. An augmentation of evolutionary computation, coevolutionary computation, drives the inspiration from natural processes of coevolution between different (animal) species. The main difference between coevolutionary and standard evolutionary algorithms is in the nature of individuals' fitness evaluation. While the former uses a static fitness function for evaluation of individuals from a single population, the latter employs non-stationary fitness function for evaluating individuals from multiple populations, based on their interactions with individuals from other populations [1].

Coordinated path planning for multiple robots is difficult because the problem is NP-complete, whose complexity grows exponentially as the number of DOF increases [2] [3]. Required is to plan not only for the paths of individual robots, but also for the order of their consecutive movements, in order to prevent the robots from colliding with one another. The path planning problem for multiple manipulators sharing common work space have been studied for some time now [4].

L.M. Camarinha-Matos, P. Pereira, and L. Ribeiro (Eds.): DoCEIS 2010, IFIP AICT 314, pp. 169–178, 2010.

2 Contribution to Technological Innovation

Current industrial control approaches are on the edge of becoming obsolete. While the processes inducted by the globalization have opened the door for a worldwide market, at the same time, competition is additionally boosted. In this context, traditional, deterministic automation approaches are cost ineffective, inflexible and unreliable in the conditions of short-termed business opportunities.

In this line of thought, employing dual-arm robots in decentralized control fashion, for solving complex assembly tasks receives recent attention in the first line in research community. Such robots should be able to decompose complex assembly tasks to several simpler ones, simultaneously change end effectors, reduce facilities costing, total assembly time and spare area.

However, theory about motion planning of this type of robots is not perfect. The problem is twofold; from the one hand, feasible collision free paths between two intermediate points should be calculated prior the movement of each robot begins. From the other hand, it is difficult to achieve synchronization of the two hands in context of industrial communication protocols, i.e. fast Ethernet, due to its stochastic nature and time delays.

This work was motivated bearing in mind that the two hands of the robots could be to some extent autonomous, being controlled by two controllers in decentralized fashion. Each controller should be governed by one evolutionary algorithm, and the communication between the two controllers should take place at the time controllers are checking for collisions, for the case when the objective is to find collision free paths.

Some of the technological benefits of the proposed approach: assembly speedup and efficiency increase due to asynchronous and parallel computation. Scalability – it should be possible to add additional agents and represent them in form of new co-evolving population. Modularity and cost reduction, it is easier and most cost effective to rearrange system that is based on decentralized control principles.

Problem of synchronization of the two hands at the level of execution of complex tightly coupled assembly tasks is beyond the scope of this paper and considered for future work.

3 Related Literature

Many important contributions to the problem of path planning in recent years have been made, each one possessing its own merits and disadvantages. Comprehensive survey can be found in [5]. Practical multi-robot motion planning problems are often decoupled in the sense that the robots trajectories are planned for only one robot at a time in priority order. Then in the second phase, velocities are modulated so that collisions between the robots are avoided. If a problem of path planning is represented as an optimization problem, robust optimization techniques, such as evolutionary algorithms have proven suitable for finding solution for such formulated problems.

Rana and Zalzala [6] [7] propose an evolutionary planner to evolve near time-optimal collision-free trajectories for multi-arm robot manipulators. In this study, planning is carried out in joint space of the manipulator, and the path is represented as a string of via points connected by cubic polynomial splines. Davidor [8] also applies evolutionary algorithms to the trajectory generation by searching the inverse kinematics solutions to pre-defined end effector robot paths. Pires, Machado and Oliveira [9] employ multi objective genetic algorithm to evolve joint-space strings of manipulator configurations. Five indices, namely manipulator joints travelling distance, joint velocity, cartesian distance, cartesian velocity and energy are used to qualify the evolving trajectories. Toyoda and Yano [10] used multi-purpose genetic algorithm to optimize movement of multi-joint robotic arms in presence of stationary obstacles. Optimum solutions with smooth trajectories and minimal joint rotation were obtained. Venegas and Marcial-Romero [11] present preliminary results of Constructive Solid Geometry based approach to path planning of multiple robot arms. They used two phase genetic algorithm to obtain a plan for the robotic arms by using a strategy that combines the exploration of the free collision space while looking for the target position from each previously explored area.

Majority of papers dealing with evolutionary – based path planning consider either single agent operating in an environment without presence of obstacles, or in an environment containing static, point obstacles. The problem then boils down to finding suitable set of interior points, to be interpolated to formulate the polynomial of given order representing the trajectory.

The method presented in this paper considers concurrent development of robot trajectories for two 2R type robots sharing common work space. Each robot is considered an agent that is to find appropriate strategy for moving from given initial to final configuration. Since one agent presents dynamical obstacle to the other, and vice versa, often it is necessary for the agents to detour away from optimal, shortest paths, to find feasible strategies. It is also necessary to check for collisions between the links of the two robots in each consecutive time step.

Cooperative coevolutionary algorithms (CEAs) offer great potential for concurrent multiagent domains. Two populations, each representing set of configurations of one robot in joint space, co-evolve to minimize number of collisions between interacting individuals, at the same time minimizing distance traveled, total joint rotation angle, and equalizing end-effector velocity profile. Best collaborators are sought and preserved to achieve memory effect in subsequent populations, and to bias coevolutionary search for optimal strategies.. The paper is organized as follows: In section 2, impact on technological innovation is discussed. Section 3 presents related literature and recent work, with focus on implementation of evolutionary algorithms to path planning. Problem and proposed algorithm are discussed in the section 4. Section 5 presents simulation and result for one given scenario. In section 5 some coevolutionary modifications are described. Finally, we discuss results and give insights for future work in sections 6 and 7.

4 Problem and Algorithm Formulation

In this paper, two *2-R* type robots with two links and two joints are considered. The end-effector is considered to move in the horizontal plane. The configuration spaces of the two robots are: $C_1 = q_{11} \times q_{21} \in R^2$ for the first robot and $C_2 = q_{12} \times q_{22} \in R^2$ for the second. The two manipulators are to move from given initial to a given final configuration. The lengths of all links are set to 1 *m*, with distance between the robots of 2.1 *m*. The links are free to rotate in the range $[0, \pi]\, rad$.

4.1 Individual Representation

An individual in a population is encoded as real-valued vector in the joint space:

$$\left[\left\{q_{11}^{(\Delta t,G)},..,q_{ij}^{(\Delta t,G)}\right\},\left\{q_{11}^{(2\Delta t,G)},..,q_{ij}^{(2\Delta t,G)}\right\},..,\left\{q_{11}^{((n-2)\Delta t,G)},..,q_{ij}^{((n-2)\Delta t,G)}\right\}\right] \quad (1)$$

Where *i* denotes the robot *i*=1,2, *j* is the number of *DOF,* Δt is sampling time between two consecutive configurations, *q* is angle between the link and positive *x* axis, *G* is current generation. At the beginning of the evolutionary process, joint values are randomly initialized, whereby the initial and final configurations are not encoded into the string because they remain constant throughout the search process. Without the lost of generality, adopted is normalized sampling time with $\Delta t = 0.1\, s$.

4.2 Operators of the Coevolutionary Algorithm

The algorithm starts with random initialization of two populations, each representing set of configurations for one robot. The performance of each robot's configuration depends on the current state of the robots from other population, since the two must coordinate the motion, to achieve continuous collision free movement. In the canonical CCEA, each individual from the first population should be evaluated by all individuals from other population, what is extremely time-consuming. To speed-up the evolution process, modified co-evolution is considered. In the modified version, each individual is evaluated by a finite set of the top collaborators from the other population, based on scores from previous generation. In this study, we evaluated the top 10 % of the populations, which is a modification of the approach proposed by the Sims [12], where "…the most "interesting" results occurred when the all vs. best competition pattern was used". The sizes of both populations were same and set to 60 individuals. In what concerns the selection operator, the successive generations are reproduced on the basis of roulette wheel selection. Standard single point crossover operator is used. The mutation operator replaces one allele with a given probability using the equation, where rand gives random number from the given interval:

$$q_{ij}^{(\Delta t,G+1)} = q_{ij}^{(\Delta t,G)} + rand \in (0, \pi/5] \Big| q_{ij}^{(\Delta t,G+1)} < \pi \quad (2)$$

4.3 Fitness Criteria

Fitness criteria should take into account number of collisions between the two robotic arms, trajectory length, velocity profile, and total rotate angle. The most important criterion is the collision number, since collisions should be avoided at all costs. To check for collisions, in each time step Δt, linear system is solved that describes current positions of all links of the two robots.

4.4 Collision Penalty

Collision penalty depending on collision between Robot 1 and Robot 2 in corresponding configurations is given by:

$$C_1 = \sum_{k=1}^{k=n-2} C_k, C_1 \to \min \tag{3}$$

where:

$$C_k = \begin{cases} 1 \text{ if R1 and R2 collide in i}^{th} \text{ generation} \\ 0 \text{ otherwise.} \end{cases} \tag{4}$$

It is important to note that the evaluation of the eq. 3 needs representatives from other co-evolving population and here is where cooperative coevolution makes contribution.

4.5 Total Distance of the End-Effector Movement

$$C_2 = \sum_{k=1}^{k=n} dist\left(p_j, p_{j-1}\right), C_2 \to \min \tag{5}$$

where p_j is the robot's intermediate end-effector position. In the case where no obstacles are present in the environment, optimal value of the function is length of the straight line connecting initial and final end-effector position. For all other cases, $C_2 > C_{2optimal}$.

4.6 Total Rotation Angle

Since the robots are redundant systems even in this simple form of 2 *DOF,* resulting in possibilities of reaching the same point in the space in elbow-up and elbow-down configurations, criterion of minimizing the total distance is not enough. Additionally, it is necessary to minimize the total rotation angle, to ensure no oscillations between the elbow-up and elbow-down configurations occur. Following expression

defines total angle for one joint of one robot. Each robots' total angle should be minimized:

$$C_3 = \sum_{i=1}^{n} \left| \alpha_i - \alpha_{i-1} \right| \rightarrow \min \tag{6}$$

where α_i is the angle between the limb of the robot and positive horizontal axis.

4.7 End-Effector Velocity Distribution

To ensure even distribution of passing points along the robots' trajectory, the distance between two adjoining points in unit time should be equal:

$$C_4 = \left\{ dist\left(p_j, p_{j-1} \right)_{\max} - dist\left(p_j, p_{j-1} \right)_{\min} \right\} \rightarrow \min \tag{7}$$

Optimal value for C_4 is equal to 0, what means that all passing points are equally distant from each other.

4.8 Objective Function Calculation

Objective (fitness) function for each candidate is calculated as weighted linear combination of above equations.

$$F = f\left(w_1 \cdot C_1 + w_2 \cdot C_2 + w_3 \cdot C_3 + w_4 \cdot C_4 \right) \rightarrow \min \tag{8}$$

Where values of weight factors w_i are constants. Values of weight constants have significant impact on the overall behavior of the algorithm. Namely, since objective criteria are in conflict, i.e. shortest distance criterion conflicts collision penalty criterion, proper tuning of these parameters is very tedious and time demanding. We are considering implementing non-linear functional relationships in weight parameters in the future. To address this problem in this paper, we implement dynamic weight factors and show the effectives of proposed approach.

5 Simulation Results

Several experiments were conducted to test the performance of the proposed algorithm. It was observed that, beside the parameters of the fitness function, the success ratio of the algorithm depends on the initial and final configurations of the robots. The easiest scenario is when configurations of the robots result with no collisions. Most difficult scenarios occurred for configurations when significant detouring from shortest paths was necessary, (to ensure collision free motion).

Fig. 1 shows evolved motions for two robots sharing work space. In the above case, all lengths of robot links are 1 *m*, and the distance between bases of the robots is set to 1.5 *m.* The algorithm successfully evolved trajectories for given start and end

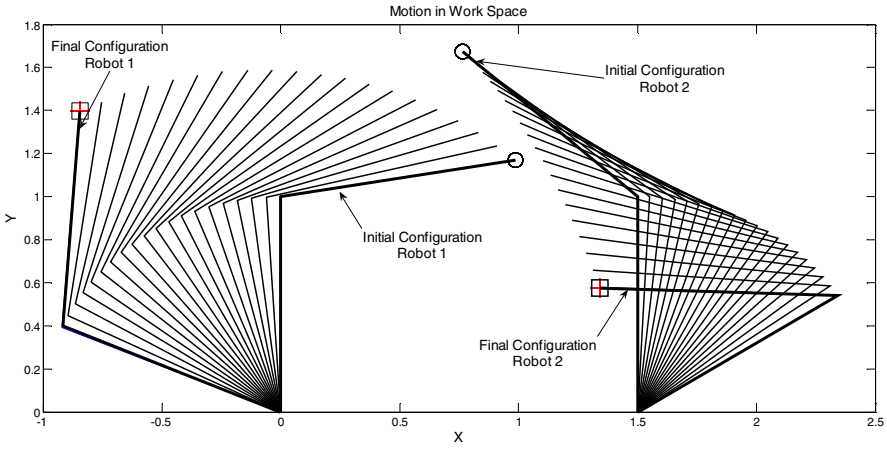

Fig. 1. Trajectories evolved for two robots simultaneously. Trajectories are optimized in terms of length, end-effector velocity distribution, total rotation angle, and number of collisions.

configurations of the robots. Trajectory of the left robot could be further optimized in terms of length, since it is obvious that end-effector performs arc motion, while linear motion would result in reduced trajectory length.

Operation of the coevolutionary algorithms is very complex and theoretical background is still developing in the research community [13] [14]. For example, it is not possible to employ simple elitist function to the coevolutionary algorithm, as it is in the case of standard algorithm. The reason is that each individual from one population is, in canonical case, evaluated by each individual from coevolving population. The consequence is that performance of the individuals from first population depends on the structure of the other population(s). That means that individual, that had high fitness value and was good in one generation, may suddenly become not so good in the next generation. The fitness vs. time function is not monotonously growing in that case.

6 Modified Coevolution

Standard evaluation each individual by each individual from other populations is time consuming. To speed up the evolutionary process, we save top 10% individuals from both populations and evaluate them by each individual from other population. By doing so, we hope to only increase the speed of the convergence, without hindering the ability of the algorithm for finding solutions near Pareto front. Other issue is biasing coevolutionary search towards optimal solutions. Taking in consideration the nature of the problem, we introduce dynamic fitness function. Since it is very difficult to find proper combination of weight factors, we employ following procedure:

The search starts with weight parameters having initial values chosen either by random or by some previous experience. Afterwards, number of collisions is monitored for the pair of individuals we call *best collaborators* – the pair receiving the

highest fitness value. Since the most important criterion is to find collision free trajectories, weight w_1 is increased and simultaneously all other weight values start to decay. This way, importance is given to part of the fitness function responsible for finding collision free paths. After best collaborators have no more collisions, opposite process starts and other three weight factors start to increase, whereas w_1 starts to decay, as shown by Fig 2.

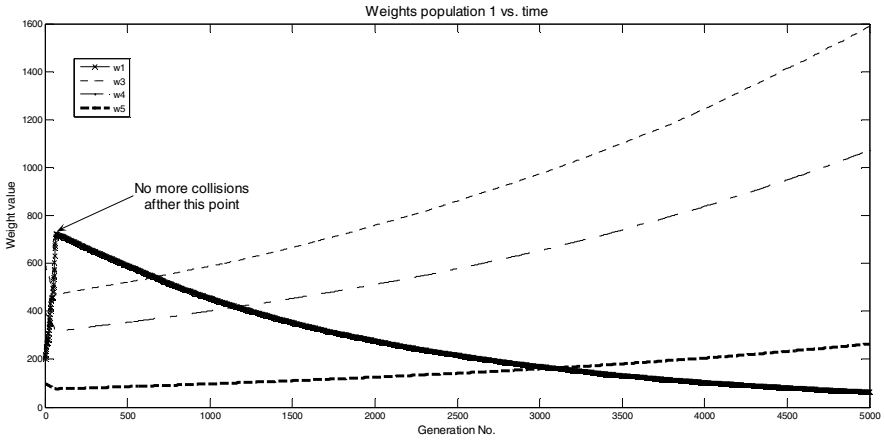

Fig. 2. Non-stationary weight factors over generations

If, in any moment, best collaborators start to collide, the first process begins, increasing the importance of avoiding collisions.

Such way, after individuals are evolved, which generally do not collide with each other, they are fine-tuned afterwards. If stationary fitness function is used, it is very difficult to evolve satisfactory strategies for the robots. There are numerous possibilities to improve this process of tuning fitness function. For example, we monitor only best collaborators i.e. one individual from the first and one individual from the second population to determine when to change fitness function. At the other hand, gradients of the growth of weight factors are chosen after several experiments were conducted. It is very important parameter, whose behavior should be determined with more care.

7 Discussion

As it is already mentioned, there are many possibilities to improve presented approach. There are two critical factors: speed of the convergence and completeness of the solution i.e. number of satisfying configurations developed by the algorithm before stopping criterion is matched. In what concerns speed of the convergence, it is proposed to evaluate a finite representative set from each population. This way, number of evaluations of the algorithm is decreased, but some, possibly good combinations of individuals might get lost. In what concerns completeness of solution, we propose dynamic weight factors for the fitness function. In such way, we are able to

adapt the evolutionary process according to current state in the population. It is rather rough idea, how the process should look like, since only two individuals are monitored, and based on their interaction it is decided on the values of the weight factors. Simple method for changing fitness factors is adopted, namely, they are increased or decayed by adding or subtracting some fixed increment. The values of the increments remain fixed over the evolutionary process, but are different for each weight factor.

8 Conclusion and Future Work

Our long-term goal is to develop a framework based on described principles and implement it to a pair of real robots, probably of SCARA configuration. To do so, many important questions should be answered. First of all, algorithm should be further optimized in terms of speed of execution, although off-line planning with combination of machine learning is possible. No real physical properties of the robot are taken into consideration in this work, i.e. maximal acceleration etc. what is another important issue. At the level of execution of the algorithm, problem of synchronization of two hands based on Ethernet protocols is issue for itself and beyond the scope of this paper, but certainly an issue to be cleared before implementation to real robots would be possible.

Acknowledgments. The author kindly acknowledge Croatian Ministry of Science, Education and Sports for the Grant No. 120-1201948-1941, *Autonomous Multiagent Assembly.*

References

1. Paredis, J.: Coevolutionary Computation. Artificial Life Journal 2(4) (1996)
2. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
3. Canny, J.: The Complexity of Robot Motion Planning. MIT Press, Boston (2006)
4. Erderman, M., Lozano-Perez, T.: On multiple moving objects. In: Proceedings of the IEEE Conference on Robotics and Automation, San Francisco, California, USA, pp. 1419–1424 (1986)
5. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Boston (1991)
6. Rana, A.S., Zalzala, M.S.: An Evolutionary Algorithm for Collision Free Motion Planning of Multi-arm Robots. In: IEEE Genetic Algorithms in Engineering Systems: Innovations and Applications (1995)
7. Rana, A.S., Zalzala, M.S.: An Evolutionary Planner for Near Time-Optimal Collision-Free Motion of Multi-Arm Robotic Manipulators. In: UKACC International Conference on Control (1996)
8. Davidor, Y.: Genetic Algorithm and Robotics; A Heuristic Strategy for Optimization. World Scientific, Singapore (1991)
9. Solteiro Pires, E.J., Tenreiro Machado, J.A., Moura Oliveira, P.B.: Robot Trajectory Planning Using Multi-objective Genetic Algorithm Optimization. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 615–626. Springer, Heidelberg (2004)

10. Toyoda, Y., Yano, F.: Optimizing Movement of A Multi-Joint Robot Arm with Existence of Obstacles Using Multi-Purpose Genetic Algorithm. Ind. Eng. Man. Sys. 3, 78–84 (2004)
11. Venegas Montes, H.A., Raymundo Marcial-Romero, J.: An Evolutionary Path Planner for Multiple Robot Arms. In: Evo Workshops. LNCS. Springer, Heidelberg (2009)
12. Sims, K.: Evolving 3D Morphology and Behavior by Competition. In: Artificial Life IV Proceedings. MIT Press, Cambridge (1994)
13. Bucci, A.: Emergent Geometric Organization and Informative Dimensions in Coevolutionary Algorithms. PhD thesis, Brandeis University (2007)
14. Panait, L., Luke, S., Harrison, J.F.: Archive-based Cooperative Coevolutionary Algorithms. In: Proceedings of the GECCO (2006)