Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Evolutionary Trajectory Planning with Obstacles for a Mobile Manipulator

Antônio R. C. Gonçalves [*] **Elaine Guerrero-Peña** [*]
Aluizio F. R. Araújo [*] **Adrien Durand-Petiteville** [**]

[*] *Centro de Informática, Universidade Federal de Pernambuco, Av. Jornalista Aníbal Fernandes, s/n - Cidade Universitária, Recife, Pernambuco, Brazil.*
[**] *Departamento de Engenharia Mecânica, Universidade Federal de Pernambuco, Av. Prof. Moraes Rego, 1235 - Cidade Universitária, Recife, Pernambuco, Brazil.*

Abstract: Trajectory planning is a crucial issue for robotics. In recent years, researchers have used meta-heuristics, such as Multi-Objective Evolutionary Algorithms (MOEAs), to handle it. However, despite the numerous favorable features of EAs, research is needed to analyze the efficiency and effectiveness of such algorithms to find an optimal trajectory. For this reason, we present a comparative study between different Pareto-based MOEAs for trajectory planning for a mobile manipulator in an environment with obstacles. In order to generate the trajectory in the joint space, two objective functions are considered: the Cartesian velocity and the joint speed. Both functions are modified to find feasible solutions that avoid collisions. Four state-of-the-art EAs for multi-objective optimization are selected to perform the study. The planned trajectories are tested in simulation and with the actual robot TIAGo. The results suggest that Pareto-based MOEAs are suitable for offline trajectory planning, especially OSP-NSDE, which found the best solutions in the shortest time.

Keywords: Robotics, Robot Trajectory planning, Evolutionary Computation, Multi-objective Optimization, Obstacles

1. INTRODUCTION

Over decades, robotics has been developed to achieve a wide variety of tasks in production environments, such as exploring and inspecting dangerous zones, performing welding, painting, and moving objects in areas where the human physical integrity is threatened. Robots might also be able to achieve these tasks at a higher speed, reducing thus the task execution times. Among the numerous challenges, robots may have to perform their duties in the presence of obstacles in their workspace. Hence, trajectory planning considering obstacles consists in a critical component of robot acting (Saramago and Steffen Júnior, 1999).

Trajectory planning consists of computing a sequence of poses at a given time, a robotic arm has to successively reach them to achieve a desired task. It can also be seen as the calculation of the kinematic and dynamics properties of the motion. In order to compute the trajectory, one usually relies on optimization functions. Thus, the trajectory planning algorithms aim to find a minimum time, minimum energy, or minimum jerk solution. Moreover, there also recent approaches optimizing more than one function, *e.g.,* (Gasparetto et al., 2015).

Trajectory planning is generally considered as a NP-hard problem, subject to physical constraints, input torque/force constraints, and obstacle avoidance (Saramago and Steffen Júnior, 1999). Most of the existing approaches rely on exact algorithms to find an optimal solution. For this class of methods, the computational time might exponentially increase when the complexity grows, making it unsuitable to handle the complicated search space of the optimal solutions. Then, probabilistic algorithms were proposed to determine a feasible solution, not necessarily the optimal one, taking shorter convergence time than the previous approach.

Aiming to find optimal solutions, metaheuristics were used to determine the paths. In particular, single-objective evolutionary algorithms have yielded promising solutions (Silva et al., 2010; Abu-Dakka et al., 2013). In this paper, we define the trajectory planning as a multi-objective problem (MOP), such as in (Mulik, 2015; Menasri et al., 2015; Huang et al., 2018) in which we simultaneously optimize distinct utilities. Furthermore, we consider constraints that can be taken as objectives (or part of them) to plan the optimal trajectories. Despite the promising results with several Multi-Objective Evolutionary Algorithms (MOEAs), capabilities and limitations in terms of efficiency and effectiveness of MOEAs is an open research issue. This study aims to produce a comparative study of some state-of-the-art MOEAs under the viewpoint of the MOEAs and of the manipulator performance.

It is proposed to compare the trajectories planned for a 7-DoF manipulator. The optimization takes place in the joint space in which the arm reaches a final pose starting from a given initial posture in a 3D environment with obstacles. First, only a Cartesian velocity function and a joint speed function are used to generate a trajectory. Then, both functions are modified to avoid collisions with obstacles inserted in the environment and to produce (often locally) optimal feasible solutions. The chosen MOEAs belong to the class of algorithms based on Pareto dominance criterion, which are characterized as fast-convergence MOEAs. Thus, efficient models are selected: NSGA-II (Deb et al., 2002), NSDE (Angira and Babu, 2005), SPEA2 (Zitzler et al., 2001), GDE3 (Kukkonen and Lampinen, 2005), and OSP-NSDE (Guerrero-Peña and Araújo, 2019). The solutions found by the algorithms are analyzed taking into consideration Hypervolume, Spacing, Pareto front, and the value of the objective functions. Other measures are used to analyze the trajectory quality found by the MOEAs: times to go through trajectories, Cartesian and joint distances, and the error between the planned trajectory and the trajectory executed by the robot. The trajectories computed by the algorithms are tested both in a simulated environment and on the robotic platform TIAGo (Pages et al., 2016) developed by PAL Robotics.

This paper is organized as follows. Section 2 presents the robot trajectory planning problem. Section 3 describes the evolutionary algorithms for the trajectory planning. Section 4 introduces the simulation setup. Section 5 and 6 show the experimental results and the real application. Finally, Section 7 outlines some conclusions.

2. TRAJECTORY PLANNING PROBLEM FOR ROBOT MANIPULATOR

2.1 Trajectory Planning Problem without Obstacles

Trajectory planning aims to generate a set of configurations that must be reached by the robotic arm for a given time, in order to achieve the target avoiding possible collisions. The trajectory can be expressed either in the joint space (configuration of all joints) or in the workspace (configuration of the end-effector) and it consists of a sequence of poses, velocities, or accelerations. In this work, the trajectory obtained optimizing a number of conflicting objective functions. Hence, meta-heuristics are used to determine a compromising optimal planning in which the solutions comprise trajectories represented by a curve whose derivatives are continuous to a certain order (Gasparetto et al., 2015).

In this work, trajectories are computed by evolutionary algorithms. They produce a set of randomly initiated individuals to be successively submitted to selection and variety operators to generate diversity and to hopefully improve the fitness of part of the population throughout the generations until a stop criterion is reached.

Each individual in a population represents a trajectory, a candidate solution to the problem, which is formed by a set of angular configurations that the manipulator must take to move from an initial to a final pose (Pires et al., 2007)):

$$\begin{bmatrix} \left\{q_1^{(\Delta t,T)}, \ldots, q_i^{(\Delta t,T)}, \ldots, q_{n_k}^{(\Delta t,T)}\right\} \\ \left\{q_1^{(2\Delta t,T)}, \ldots, q_i^{(2\Delta t,T)}, \ldots, q_{n_k}^{(2\Delta t,T)}\right\} \\ \vdots \\ \left\{q_1^{((n_c-2)\Delta t,T)}, \ldots, q_i^{((n_c-2)\Delta t,T)}, \ldots, q_{n_k}^{((n_c-2)\Delta t,T)}\right\} \end{bmatrix}^T \tag{1}$$

where n_k is the number of manipulator's joints, $q_i^{(t,T)}$ is angle value at instant t of the i^{th} joint at the T^{th} generation, with $i \in [1, \cdots, n_k]$, n_c is the number of configurations used to define the trajectory (including the initial and the final postures), and Δt is the sampling time between two consecutive configurations.

The computation of the sequence of poses, as defined by Eq. (1), can be treated as a minimization problem. The objective functions can be displacement time, the linear or angular distances, the sum of speeds, the sum of accelerations, and the energy consumed in motion by the manipulator, or a combination of two or more functions. In this paper, we face a multi-objective problem defined by two objective functions to be optimized: the Cartesian velocity function $f\dot{p}$ (Eq. 2) and the joint speed function $f\dot{q}$ (Eq. 3), such as:

$$f\dot{p} = \sum_{j=3}^{n_c} \left\{d\left(p_j, p_{j-1}\right) - d\left(p_{j-1}, p_{j-2}\right)\right\}^2 \tag{2}$$

$$f\dot{q} = \sum_{j=1}^{n_c} \sum_{i=1}^{n_k} \left(\ddot{q}_i^{(j\Delta t,T)}\right)^2 \tag{3}$$

where $d(\cdot,\cdot)$ is the Cartesian distance between two subsequent points (p_j) from the end-effector in the workspace, and $\ddot{q}_i$ is the acceleration of the i^{th} joint. The function $f\dot{p}$ (Eq. 2) minimizes changes of the actuator's speed, while $f\dot{q}$ (Eq. 3) minimizes changes of the joints' speed over time.

2.2 Trajectory Planning Problem with Obstacles

In this work, obstacles are represented by spheres in the robot's workspace. In order to deal with obstacles in the environment, this study adopts the technique employed by Menasri et al. (2015) to detect the occurrences of collisions of the robot with an obstacle.

Firstly, a sphere of center O is used to represent an obstacle and a set of control points $\{Q_1, \cdots, Q_i, \cdots, Q_{n_k}\}$ must be defined in Cartesian Space as the centers of each joint of the robotic arm. Next, for each new arm configuration and for each pair (Q_i, Q_{i+1}) of consecutive control points, the orthogonal projection N of O onto the line (Q_iQ_{i+1}) is computed. The point N belongs to the segment $[Q_iQ_{i+1}]$ if $-\pi/2 \leq \theta \leq \pi/2$ and $|\boldsymbol{Q_{i+1}O}| \cdot \cos(\theta) \leq |\boldsymbol{Q_{i+1}Q_i}|$, where θ is the angle formed by the vectors $\boldsymbol{Q_{i+1}O}$ and $\boldsymbol{Q_{i+1}Q_i}$. If N belongs to the segment $[Q_iQ_{i+1}]$, then $\Delta_{O|N}$, the closest distance between the current link and the obstacle center, is calculated. If $\Delta_{O|N} < \delta_{coll}$, where δ_{coll} is a user defined threshold, then we consider there is a collision (Fig. 1).

The number of poses in a trajectory (individual) that would collide into an obstacle is then evaluated. This number of collisions ($N_{collision}$) becomes a penalty for each objective function (Eq. 2 and 3), as follows:

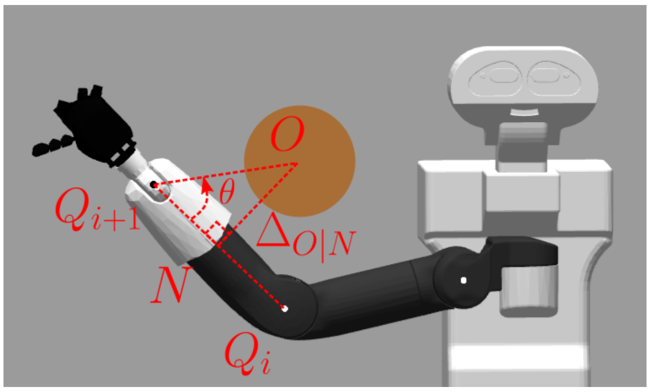


Fig. 1. Example of collision detection parameters where Q_i and Q_{i+1} are two control points, the yellow sphere of center O represents an obstacle, N is the orthogonal projection of O onto the line (Q_iQ_{i+1}), and θ is the angle between $\boldsymbol{Q_{i+1}Q_i}$ and $\boldsymbol{Q_{i+1}O}$

$$f_1 = f\dot{p} * (1 + N_{collision}) \tag{4}$$
$$f_2 = f\dot{q} * (1 + N_{collision}) \tag{5}$$

3. EVOLUTIONARY ALGORITHMS UNDER STUDY

Five Pareto-based Evolutionary Algorithms are selected to perform the study: Non-dominated Sorting Genetic Algorithm II (NSGA-II: Deb et al. (2002)), Non-dominated Sorting Differential Evolution (NSDE: Angira and Babu (2005)), Strength Pareto Evolutionary Algorithm 2 (SPEA2: Zitzler et al. (2001)), Generalized Differential Evolution 3 (GDE3: Kukkonen and Lampinen (2005)), and Non-dominated Sorting Differential Evolution improvement with Prediction in the Objective Space (OSP-NSDE: Guerrero-Peña and Araújo (2019)). The Pareto dominance principle (Deb et al., 2002) is used to assign fitness to solutions. However, such a principle may not work with efficacy if a MOEA loses diversity. Thus, a diversity mechanism is used to maintain or insert population variety when selecting parents or survivors that are equally ranked. The diversity maintenance in the Pareto algorithm preserves exploration and tends to avoid concentration of population, much exploitation, in several niches of the search space. Hence, a MOEA simultaneously seeks the fittest and most spread population.

Algorithms such as NSGA-II, NSDE, and GDE3, employ the crowding distance diversity mechanism for the selection of the next-generation population. Other two MOEAs, SPEA2 and OSP-NSDE use a density estimation technique and an enhanced archive truncation method, which prevents boundary solutions from being removed. MOEAs can also change the variation operators used to generate offspring. NSGA-II and SPEA2 employ polynomial mutation and SBX crossover. A tournament is made to select potential parents. In the case of SPEA2, only members of the external archive participate in the mating selection process. Moreover, NSDE and GDE3 apply Differential Evolution (DE) operators. In NSDE, all individuals in the population are parents. GDE3 establishes rules for selection, even in the case of comparisons involving infeasible solutions. Finally, in OSP-NSDE, the offspring is generated by DE operators followed by the polynomial mutation. OSP-NSDE also employs an Objective Space Prediction (OSP) strategy to guide population movements. OSP is strategically triggered when several premises based on the Approximated Hypervolume metric are satisfied.

Usually, Pareto-based MOEAs are the fastest and simplest computable approaches. Therefore, the selected algorithms are expected to find an optimal trajectory in an acceptable time.

4. SIMULATION SETUP

This section describes the TIAGo manipulator and parameters setup for the MOEAs under study. The decision-maker used to choose the optimal trajectory among the approximated solutions of the Pareto front is presented. Finally, the metrics used to evaluate the MOEAs, and the solutions for the robot are shown.

4.1 TIAGo Robot

TIAGo is a robotic platform composed of a mobile base embedding a 7-DOF robotic arm. Each joint can be controlled in terms of position or velocity, and the angular values are measured by encoders. Finally, the forward kinematic model computes the end-effector position as function of the seven joints values. The forward kinematics of the TIAGo manipulator can be found in the documentation provided by the Pal Robotics (S.L., 2016).

4.2 Algorithms setup

In this study, an individual for the MOEAs is represented by Equation (1), seven vectors each one with seven joint angles, i.e., seven intermediate poses that when added to the initial and final poses ($n_c = 7 + 2$) form a trajectory. An individual has a dimension $D = (n_c - 2) \times n_k = 49$, where $n_k = 7$ is the number of TIAGo's arm joints.

All considered MOEAs have a population of 100 individuals and run for $150,000$ function evaluations (FEs). NSGA-II, SPEA2, and OSP-NSDE settings include mutation probability of $p_m = 1/D$ and distribution index $\eta_m = 20$ for the mutation operator. Canonical rand/1 strategy, with mutation step size $F = 0.5$ and crossover rate $CR = 0.9$ are used for NSDE, GDE3 and OSP-NSDE. For OSP-NSDE, the variation percentage was set to $\lambda = 0.2$, Hypervolume variation percent was set to $\beta = 0.9$, and the initial forecast horizon was set to $p = 50$. The other parameters of the MOEAs were set as in their cited publications.

4.3 Decision Making

MOEAs produce an approximate Pareto Front (PF), from which we must choose a single solution (trajectory). Often, the knee-point in the PF is the chosen solution if the decision-maker does not have a particular preference. Thus, the distance to the extreme line knee-point method is used to select (x^*), the closest solution to a given reference point, the origin (Liang et al., 2018):

$$x^* = \min_{1 \le i \le NP} (d(x_i)) \mid d_i = \sqrt{f_{1_N}(x_i)^2 + f_{2_N}(x_i)^2} \tag{6}$$

where f_{1_N} e f_{2_N} are the normalized objective functions, and NP is the number of solutions in PF.

4.4 Metrics to Assess the Experiments

Average values to assess performance of the MOEAs:

- Machine time assesses the suitability the MOEAs finding the trajectories on-line. This performance measure also includes the calculation of the time spent by the decision-maker;
- The number of individuals at the Pareto Front determines the representativeness of the algorithm output sample space;
- Hypervolume (HV) (Zitzler et al., 2007) estimates the region dominated by the approximated PF from a reference point (nadir point);
- Spacing (Schott, 1995) calculates the average distance between individuals on the Pareto Front.

We evaluate the planned trajectory considering the trajectory distance covered. This metric is calculated by linear interpolation of settings sampled every 100 ms. The metric value (meters) is equal to the sum between the Euclidean distances of subsequent positions of the end-effector.

5. EXPERIMENTAL RESULTS

We ran three groups of simulations to validate the performance of the MOEAs: obstacle-free, one-obstacle, and two-obstacle environments (Fig 3)0. In such MATLAB-simulated 3D environment, we found the optimal trajectory for each MOEA in each environment. In all case, the robot configuration started at its rest position $pos_{init} = \{0.07, 0, 0, 0, 0, 0, 0\}$ and stoped at the final posture $pos_{end} = \{2, 0, 0, 0, 0, 0, 0\}$. The Kruskal-Wallis test with 99.9% confidence level was run to determine if there are significant differences between the MOEAs performances. Table 1 shows the performance of the algorithms in terms of the mean values and their standard deviations The best average results for each case are highlighted. All MOEAs under comparisons were executed 10 independent times on a computer with 3.2GHz x 8 CPU, 8GB RAM, and Ubuntu 16.04 LTS.

Table 1 shows the performance of the algorithms for the three experiments, considering the Hypervolume and Spacing metrics, PF size, and the average value of objective functions. The obtained results suggest that for the obstacle-free environment, we notice that best Spacing is obtained by GDE3, while OSP-NSDE reaches the best average HV. Analyzing the number of solutions obtained by the compared algorithms, one can see that NSDE finds a more significant amount of trade-off solutions, while OSP-NSDE is the worst in this regard. However, this algorithm reaches the best mean values in the objective functions. In the results for environments with obstacles, NSDE obtained the best spacing value and found the highest number of solutions in the Pareto font. For one-obstacle environments, OSP-NSDE reached the best HV and the lowest joint velocity value (f_2). For the two-obstacle case, OSP-NSDE reached the best Cartesian velocity (f_1) value. In summary, the averages of objective function values suggest that OSP-NSDE has smoother trajectories than the other MOEAs in most cases. To assess the trajectory quality for the robot, the best trajectories found by each algorithm were compared regarding the Cartesian distance covered by the end-effector and joint distance in time (see Fig. 2). In the obstacle-free experiment (Fig. 2 a) and d)), OSP-NSDE converges quicker for the best overall values. A similar result is obtained for trajectory planning in environments with two obstacles (Fig. 2 (c, f)). When the trajectory is designed for one-obstacle environment, we can observe that OSP-NSDE also yields the shortest Cartesian distance (Fig. 2 (b)), however, NSGA-II produces the best result (Fig. 2 (e)) in terms of joint distance. In summary, OSP-NSDE obtains the best results within the shortest time (between 50 s and 100 s) for most tests.

6. TRAJECTORY EXECUTION IN A SIMULATOR AND IN A REAL ROBOT

Ten trajectories were obtained for each experiment with each tested algorithm. These trajectories were simulated in robotics simulator Gazebo to verify if they would not provoke collisions. The simulation reads the sensor data with a 100 ms sampling rate. The samples are used to calculate the distance covered by the end-effector (Table 2). During the simulation, each trajectory execution time, difference between the trajectory final and starting times, was also measured. For the simulation, the trajectories were designed to run within 16 seconds and are equally divided between 8 partial trajectories of 2 seconds each. The real runtime results of planned routes can be seen in Table 2. All the execution times were shorter than 16 s due to the condition of sending the next pose having been reached before the planned time. For all three experiments, OSP-NSDE obtained the shortest distance traveled by the end-effector.

Besides the tests in a simulated prototype, experiments were performed as a real robot (TIAGo), to verify the feasibility of applying this type of approach for an actual robotics. The best trajectory obtained by OSP-NSDE for the obstacle-free [1], one-obstacle [2] (see Fig. 4 (a)), and two-obstacle [3] (see Fig. 4 (b)) cases are shown in videos as the sequence of TIAGo postures. All videos of the experiments with TIAGo robot can be found on Youtube.

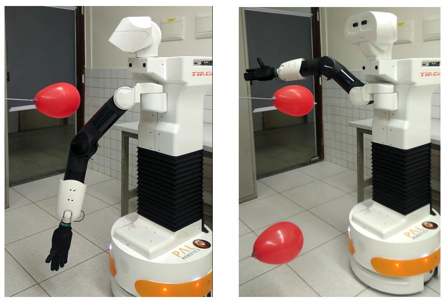
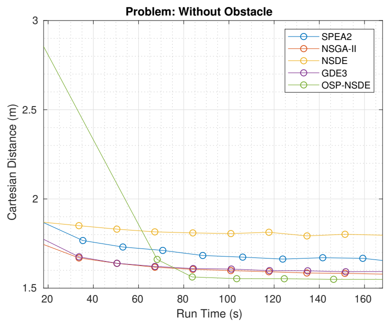

(a) (b)

Fig. 4. Experiments for TIAGo. a) One-Obstacle Environment ; b) Two-Obstacles Environment

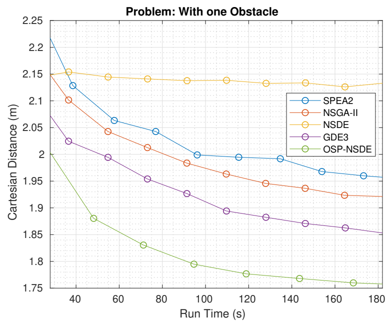
7. CONCLUSION

Our results suggest that the tested MOEAs are suitable for offline trajectory planning. OSP-NSDE presented the best

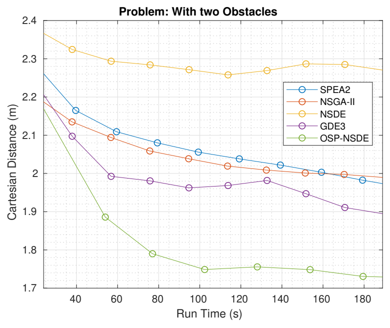
[1] Video of the experimentation without obstacles.
[2] Video of the experimentation with one obstacle.
[3] Video of the experimentation with two obstacles.



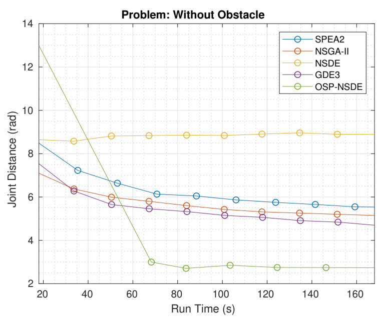
(a) OSP-NSDE reaches the minimum value in 83.77 s.



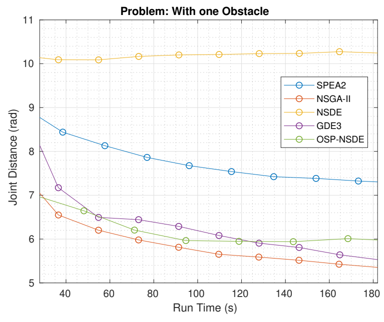
(b) OSP-NSDE reaches the minimum value in 71.21 s.



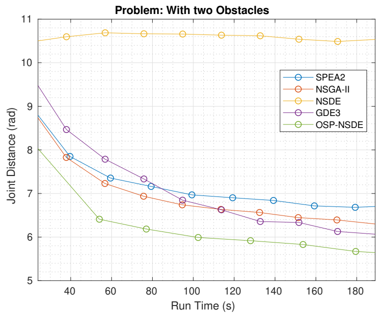
(c) OSP-NSDE reaches the minimum value in 54.02 s.



(d) OSP-NSDE reaches the minimum value in 68.25 s.



(e) NSGA-II reaches the minimum value in 146.2 s.



(f) OSP-NSDE reaches the minimum value in 102.4 s.

Fig. 2. Convergence analysis of the MOEAs under study for the three experiments. a-c) Cartesian distance (m) vs. Run Time (s); d-f) Joint distance (rad) vs. Run Time (s).

Table 1. Performance evaluation of the MOEAs chosen for the three experiments, considering the Hypervolume and Spacing metrics, PF size, and the average value of objective functions.

Experiment 1: Trajectory without obstacle					
Algorithms	**Spacing**	**HV**	**PF Size**	$f_1(m^4/s^4)$	$f2(rad^4/s^4)$
SPEA2	1.41E+03(6.08E+02)	5.19E+08(3.00E+07)	1.95E+01(1.01E+01)	1.24E+05(5.65E+04)	6.18E+01(1.54E+01)
NSGA-II	4.06E+03(4.48E+03)	5.03E+08(3.02E+07)	1.91E+01(6.30E+00)	1.75E+05(6.41E+04)	6.40E+01(1.52E+01)
NSDE	7.93E+03(3.81E+03)	5.27E+08(1.60E+07)	**4.16E+01(1.39E+01)**	1.81E+05(4.95E+04)	6.15E+01(1.05E+01)
GDE3	**1.53E+04(1.17E+04)**	5.21E+08(1.21E+07)	1.31E+01(6.31E+00)	1.73E+05(4.00E+04)	6.30E+01(6.76E+00)
OSP-NSDE	5.66E+03(1.44E+04)	**5.66E+08(6.57E+07)**	3.20E+00(1.62E+00)	**1.70E+04(3.54E+04)**	**4.13E+01(3.16E+01)**
Experiment 2: Trajectory with one obstacle					
Algorithms	**Spacing**	**HV**	**PF Size**	$f_1(m^4/s^4)$	$f2(rad^4/s^4)$
SPEA2	1.06E+03(1.97E+03)	5.83E+08(3.65E+07)	9.64E+01(7.95E+00)	6.13E+04(2.90E+04)	5.76E+01(1.45E+01)
NSGA-II	5.73E+02(5.11E+02)	6.06E+08(4.79E+07)	9.59E+01(1.16E+01)	**4.29E+04(2.93E+04)**	5.35E+01(1.73E+01)
NSDE	**5.39E+03(2.48E+03)**	5.18E+08(4.94E+07)	**1.00E+02(0.00E+00)**	2.29E+05(4.41E+04)	8.05E+01(1.21E+01)
GDE3	1.75E+03(1.05E+03)	6.12E+08(5.26E+07)	9.46E+01(1.51E+01)	4.75E+04(2.19E+04)	5.50E+01(1.83E+01)
OSP-NSDE	2.03E+03(1.23E+03)	**6.25E+08(5.36E+07)**	9.27E+01(1.83E+01)	4.60E+04(1.76E+04)	**5.03E+01(1.79E+01)**
Experiment 3: Trajectory with two obstacles					
Algorithms	**Spacing**	**HV**	**PF Size**	$f_1(m^4/s^4)$	$f2(rad^4/s^4)$
SPEA2	7.58E+02(3.05E+02)	5.30E+08(1.54E+07)	9.99E+01(3.16E-01)	9.15E+04(1.95E+04)	6.96E+01(1.16E+01)
NSGA-II	9.94E+02(4.38E+02)	5.36E+08(2.28E+07)	9.98E+01(6.32E-01)	7.32E+04(3.63E+04)	6.60E+01(1.49E+01)
NSDE	**7.40E+03(2.73E+03)**	4.78E+08(1.79E+07)	**1.00E+02(0.00E+00)**	3.08E+05(9.81E+04)	8.80E+01(1.05E+01)
GDE3	1.62E+03(1.34E+03)	**5.55E+08(1.87E+07)**	9.79E+01(5.28E+00)	7.07E+04(3.53E+04)	**5.97E+01(1.25E+01)**
OSP-NSDE	1.85E+03(1.19E+03)	5.45E+08(1.36E+07)	9.66E+01(5.68E+00)	**6.76E+04(2.09E+04)**	6.34E+01(1.15E+01)

Table 2. Trajectory execution time and Trajectory distance covered in the simulator

Algorithms	**Without obstacle**		**With one obstacle**		**With two obstacles**	
	Execution Time	**Cartesian Distance**	**Execution Time**	**Cartesian Distance**	**Execution Time**	**Cartesian Distance**
SPEA2	1.13E+01(2.62E-01)	2.05E+00(2.48E-01)	1.12E+01(1.14E-01)	2.60E+00(9.27E-01)	1.06E+01(8.92E-01)	4.06E+00(1.43E+00)
NSGA-II	1.12E+01(1.80E-01)	2.14E+00(2.97E-01)	1.16E+01(5.39E-01)	3.30E+00(9.84E-01)	1.02E+01(5.24E-01)	3.84E+00(9.78E-01)
NSDE	1.09E+01(1.93E-01)	1.96E+00(1.35E-01)	1.01E+01(2.21E-01)	3.05E+00(8.87E-01)	1.06E+01(7.12E-01)	3.72E+00(1.29E+00)
GDE3	1.09E+01(2.29E-01)	1.93E+00(1.17E-01)	9.89E+00(3.18E-01)	2.95E+00(8.53E-01)	1.05E+01(4.75E-01)	4.11E+00(9.38E-01)
OSP-NSDE	1.05E+01(2.65E-01)	**1.77E+00(2.43E-01)**	9.74E+00(2.22E-01)	**1.79E+00(1.12E-01)**	1.00E+01(3.12E-01)	**2.58E+00(6.97E-01)**

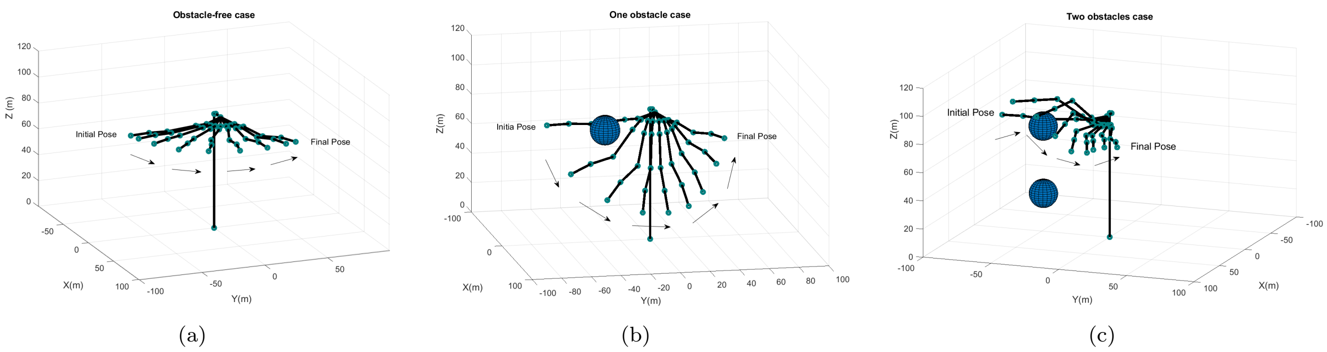


Fig. 3. Intermediate poses generated by trajectory planning. a) Obstacle-free Environment ; b) One-Obstacle Environment; c) Two-Obstacle Environment

performance according to the evaluation metrics, including the response time, due to the acceleration of its convergence caused by the OSP mechanism (Guerrero-Peña and Araújo (2019)). Even though, all the tested MOEAs demands execution tome reduction to be employed as a online planning strategy.

ACKNOWLEDGEMENTS

We thank the financial support for this project granted by CAPES, CNPq and FACEPE. We also thank Pal Robotics for supporting the use of TIAGo.

REFERENCES

Abu-Dakka, F.J., Rubio, F., Valero, F., and Mata, V. (2013). Evolutionary indirect approach to solving trajectory planning problem for industrial robots operating in workspaces with obstacles. *European Journal of Mechanics-A/Solids*, 42, 210–218.

Angira, R. and Babu, B. (2005). Non-dominated sorting differential evolution (NSDE): An extension of differential evolution for multi-objective optimization. In *IICAI*, 1428–1443.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2), 182–197.

Gasparetto, A., Boscariol, P., Lanzutti, A., and Vidoni, R. (2015). Path planning and trajectory planning algorithms: A general overview. In *Motion and operation planning of robotic systems*, 3–27. Springer.

Guerrero-Peña, E. and Araújo, A.F.R. (2019). Multi-objective evolutionary algorithm with prediction in the objective space. *Information Sciences*.

Huang, J., Hu, P., Wu, K., and Zeng, M. (2018). Optimal time-jerk trajectory planning for industrial robots. *Mechanism and Machine Theory*, 121, 530–544.

Kukkonen, S. and Lampinen, J. (2005). GDE3: the third evolution step of generalized differential evolution. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, 443–450 Vol.1. DOI: 10.1109/CEC.2005.1554717.

Liang, J., Zhu, X., Yue, C., Li, Z., and Qu, B. (2018). Performance analysis on knee point selection methods for multi-objective sparse optimization problems. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, 1–8. IEEE.

Menasri, R., Nakib, A., Daachi, B., Oulhadj, H., and Siarry, P. (2015). A trajectory planning of redundant manipulators based on bilevel optimization. *Applied Mathematics and Computation*, 250, 934–947.

Mulik, P. (2015). Optimal trajectory planning of industrial robot with evolutionary algorithm. In *2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC)*, 0256–0263. IEEE.

Pages, J., Marchionni, L., and Ferro, F. (2016). TIAGo: the modular robot that adapts to different research needs. In *International workshop on robot modularity, IROS*.

PAL Robotics S.L. (2016). TIAGo ikfast moveit! inverse kinematics plugin. https://github.com/pal-robotics/tiago_ikfast_arm_plugin.

Pires, E.S., de Moura Oliveira, P.B., and Machado, J.T. (2007). Manipulator trajectory planning using a MOEA. *Applied Soft Computing*, 7(3), 659–667.

Saramago, S.F. and Steffen Júnior, V. (1999). Dynamic optimization for the trajectory planning of robot manipulators in the presence of obstacles. *Journal of the Brazilian Society of Mechanical Sciences*, 21(3), 372–383.

Schott, J.R. (1995). Fault tolerant design using single and multicriteria genetic algorithm optimization. Technical report, DTIC Document.

Silva, M.A., Coelho, L.d.S., and Freire, R.Z. (2010). Biogeography-based optimization approach based on predator-prey concepts applied to path planning of 3-DOF robot manipulator. In *2010 IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, 1–8. IEEE.

Zitzler, E., Brockhoff, D., and Thiele, L. (2007). The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In *Evolutionary multi-criterion optimization*, 862–876. Springer.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). SPEA2: Improving the strength pareto evolutionary algorithm. *TIK-report*, 103.