2024 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)
July 15-19, 2024. Boston, MA, USA

Stiffness-Aware Genetic Algorithm for Robotic Path Finding Optimization

Alex Pasquali, Miriam Massini Alunni, Davide Chiaravalli and Gianluca Palli

***Abstract*—This work presents a genetic algorithm to evaluate a suitable path for robotic manipulation of elastic objects. The goal is to find a path, given an initial and final position, that accounts for the distance covered by the robot while minimising the forces perceived. These forces are generated by flexible objects that are difficult to model analytically. The proposed model-free approach considers these elastic forces in the fitness functions, making the genetic algorithm stiffness-aware. Furthermore, a dynamic exploration strategy allows for the convergence to effective solutions in a finite number of iterations. A simulated analysis of the suitable parameter configuration is performed for the robotic platform employed in the experiments. These parameters are then used in the validation of the method, demonstrating positive outcomes of the proposed approach in terms of convergence and effectiveness.**

***Index Terms*—Genetic Algorithms, Robotic Path Finding, Optimization, Trajectory Planning, Three-dimensional Paths, Industrial Robotics, Flexible Materials, Genetic Operators, Evolutionary Optimization**

I. Introduction

Genetic algorithms (GAs) represent a class of optimization methods inspired by the principles of natural selection and genetics [1]. They stem from the imitation of evolutionary mechanisms [2] found in biology, applying concepts of selection, crossover, and mutation to generate and progressively improve solutions suitable for the given problems [3]. The process of a genetic algorithm begins with a population of randomly generated solutions, often referred to as individuals, and are evaluated based on their fitness, that is a measure of how well they solve the problem.

In recent decades, GAs have attracted growing interest in the scientific community for their ability to tackle non-convex and nonlinear optimization problems [4]. Indeed, with their focus on mimicking genetic processes, GAs have proven to be effective in solving multi-objective [5], constrained [6], and adaptive optimization problems [7]. However, it is important to note that several aspects need to be carefully designed in order to obtain truly effective GAs, such as a solution domain representation consistent with the evolutionary optimization paradigm, and a proper selection of crossover and mutation operators [8].

Among the many applications of genetic algorithms, one of the relevant fields is trajectory and path optimization [6], [9], which finds applications in various contexts, such as robotics, logistics, transportation design, and route planning [10]. Particularly, the search for optimal paths in three-dimensional environments holds fundamental importance in numerous sectors, as the ability to find effective solutions to navigation and movement problems is crucial for the success of many practical applications [11].

In the context of robotic path finding, several computational approaches have been proposed and studied in the literature. Methods based on traditional algorithms, such as heuristic search algorithms [12] and local optimization methods [13], have provided partial solutions often limited in terms of efficiency.

In this paper, we present a novel stiffness-aware genetic algorithm to perform an efficient evaluation of suitable paths for robotic manipulation of elastic objects. In particular, this approach tackles a dual-objective optimization: the minimization of forces perceived by the robot, and the minimization of the distance covered. The proposed approach exploits a decreasing exploration ratio to properly generate mutated paths at each task cycle and obtain new populations improving the solution. By exploiting force measurements on each path key point in real-time during the execution of each task, the system can produce an effective solution within a finite number of iterations.

Moreover, since the fitness function is solely based on sensor measurement, the whole evaluation is performed in a model-free manner. Indeed, the lack of an analytical model of flexible objects allows for increased flexibility and generalizability due to the known complexity associated with their modelling and parameter estimation [14], [15].

The case study proposed demonstrates the effectiveness of the algorithm in terms of space exploration to find the proper path by decreasing the forces involved. This is done by a previous analysis of the possible parameters configuration by performing simulated experiments. Once the best parameter configuration is selected we employ them to perform different experiments in the real robotic setup.

The remainder of the paper is organized as follows: Sec. II introduces the mathematical formulation of the proposed genetic algorithm, explains the description, the definition of the fitness function and finally the analysis and selection of the algorithm parameters; Sec. III illustrates the experimental setup and discusses the results of our experiments and analyses performed; Sec. IV provides the concluding remarks.

Alex Pasquali, Miriam Massini Alunni, Davide Chiaravalli and Gianluca Palli are with DEI - Department of Electrical, Electronic and Information Engineering, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy.

This work was supported by the Horizon Europe project *IntelliMan - AI-Powered Manipulation Systems for Advanced Robotic Serivice, Manufacturing and Prosthetics* [grant number 101070136].

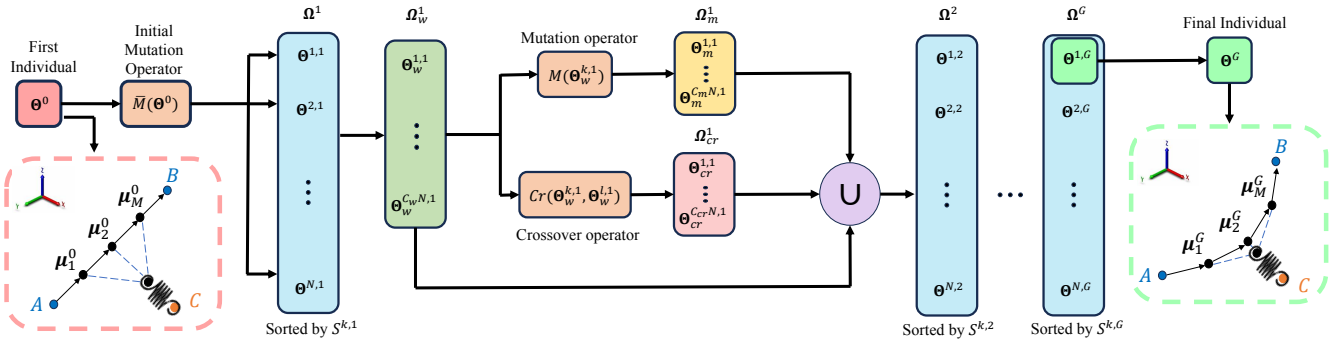


Fig. 1: Scheme of the genetic algorithm where $\boldsymbol{\Theta}$ represents the individuals and $\boldsymbol{\Omega}$ represents the generations.

II. Methodology

This section introduces the proposed GA for the path finding optimization problem in robotic systems with external elastic forces applied. Each individuals are represented by mean position ($\boldsymbol{\mu}$) and standard deviations (σ), organized into populations across generations ($\boldsymbol{\Omega}$). The algorithm incorporates mutations, crossovers, initialization, and a parameter selection process that involves exploring combinations to optimize algorithm performances Fig. 1. Simulated experiments with various springs and constants contribute valuable insights for informed decision-making on parameter choices.

A. Individuals, Population and Generations

The genetic representation utilized within our optimization framework comprises two key elements: the mean workspace position ($\boldsymbol{\mu}$) and the standard deviation (σ), which guide the exploration-exploitation trade-off during the evolutionary search. The population structure ($\boldsymbol{\Omega}$) organizes these genetic parameters across generations. The genetic representation of an individual $\boldsymbol{\Theta}$ is structured as follows:

$$\boldsymbol{\Theta}^{k,i} = \{\boldsymbol{\mu}^{k,i}, \sigma^{k,i}\}, \quad k = 1, \ldots, N.$$

In this equation, $\boldsymbol{\Theta}^{k,i}$ denotes the genetic parameters for individual k in generation i. It comprises the mean position $\boldsymbol{\mu}^{k,i}$ in solution space and the standard deviation $\sigma^{k,i}$, which guides the exploration-exploitation trade-off during the evolutionary search. A population $\boldsymbol{\Omega}$ is represented as follows:

$$\boldsymbol{\Omega}^{i} = \{\boldsymbol{\Theta}^{1,i}, \ldots, \boldsymbol{\Theta}^{N,i}\} \quad i = 1, \ldots, G.$$

$\boldsymbol{\Omega}^{i}$ represents the set of all individuals in the i-th generation. The $\boldsymbol{\mu}^{k,i}$ represents the mean position vector of individual k in generation i. Each row corresponds to a point in the solution space (x, y, z), with M dimensions.

$$\boldsymbol{\mu}^{k,i} = \begin{bmatrix} x_1^{k,i} & y_1^{k,i} & z_1^{k,i} \\ \vdots & \vdots & \vdots \\ x_T^{k,i} & y_T^{k,i} & z_T^{k,i} \end{bmatrix}, \quad x, y, z \in \mathbb{R}.$$

The parameters N, G, and T represent the population size, the number of generations, and the dimension of the solution space, respectively.

$$N, G, T \in \mathbb{N}^*.$$

The $\sigma^{k,i}$ denotes the standard deviation (exploration ratio) associated with individual k in generation i, ensuring non-negativity as a requirement for variance.

$$\sigma^{k,i} \in \mathbb{R}, \quad \sigma^{k,i} \geq 0.$$

B. Fitness Function Definition

The algorithm needs a properly defined fitness function that quantifies the quality of the solutions along with the selection of appropriate genetic operators. For these reasons our fitness function $S^{k,i}$ takes into account spatial distribution $\Phi^{k,i}$ and force $\Psi^{k,i}$ factor.

$$S^{k,i} = S^{k,i}(\Phi^{k,i}, \Psi^{k,i}) = \sum_{j=1}^{T} S_j^{k,i} =$$

$$= \sum_{j=1}^{T} ((\Phi_j^{k,i})^2 \cdot \Psi_j^{k,i}) \cdot (1 + |\Phi_j^{k,i} - \Psi_j^{k,i}|).$$

The term $(\Phi_j^{k,i})^2 \cdot \Psi_j^{k,i}$ combines the spatial distribution and force exchange for point j, while $(1+|\Phi_j^{k,i} - \Psi_j^{k,i}|)$ modulates this contribution based on the absolute difference between spatial distribution and force score.
$\Psi^{k,i}$ accounts for the forces acting on the robot along the path. It calculates the magnitude of the applied forces relative to a maximum allowable force F_{max}. This factor aims to minimize the impact of external forces on the path.

$$\Psi_j^{k,i} = \Psi_j^{k,i}(\mu_j^{k,i}) = 1 + \frac{\sqrt{F_x^2 + F_y^2 + F_z^2}}{F_{max}}, \quad F_{max} > 0.$$

To obtain the spatial distribution factor $\Phi^{k,i}$ we can define the points $\boldsymbol{A}$ and $\boldsymbol{B}$ which represent the start and end points of the path, respectively, in three-dimensional space.

$$\boldsymbol{A} = [x_A, y_A, z_A], \quad \boldsymbol{B} = [x_B, y_B, z_B].$$

Let's now consider the matrix $\boldsymbol{P}^{k,i}$ collecting the sequence of points that constitute the path for the individual $\boldsymbol{\Theta}^{k,i}$. The

function $d(\boldsymbol{P}_j^{k,i})$ calculates the Euclidean distance between two generic consecutive points of a path.

$$\boldsymbol{P}^{k,i} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{\mu}^{k,i} \\ \boldsymbol{B} \end{bmatrix}, \quad d(\boldsymbol{P}_j^{k,i}) = |\boldsymbol{P}_j^{k,i} - \boldsymbol{P}_{j+1}^{k,i}|.$$

$\Phi^{k,i}$ evaluates the spatial distribution of points along the path. It measures the deviation of the distances between consecutive points from an ideal spacing D. This factor penalizes deviations from the ideal spacing.

$$D = \frac{|\boldsymbol{A} - \boldsymbol{B}|}{M+1},$$

$$\Phi_j^{k,i} = \left(1 + \frac{1}{2}|d(\boldsymbol{P}_j^{k,i}) - D| + \frac{1}{2}|d(\boldsymbol{P}_{j+1}^{k,i}) - D|\right)^2.$$

C. Mutations, Crossovers and Initialization

Regarding the concept of mutation, it is possible to define the function $M(\boldsymbol{\Theta}^{k,i})$ that represents the mutation operator. It generates a new individual $\boldsymbol{\Theta}^{n,i+1}$ based on the current individual $\boldsymbol{\Theta}^{k,i}$. The operations needed to perform a mutation are the following:

$$\boldsymbol{\Theta}^{n,i+1} = M(\boldsymbol{\Theta}^{k,i}) = \{\boldsymbol{\mu}^{n,i+1}, \sigma^{n,i+1}\},$$

$$\boldsymbol{\zeta}^{k,i} = [\zeta_1^{k,i}, \dots, \zeta_M^{k,i}], \quad \zeta_t^{k,i} = \frac{S_t^{k,i}}{S^{k,i}},$$

$$m = \arg\max_t(\zeta_t^{k,i} \cdot U(0,1)_t), \quad t = 1, \dots, T,$$

$$\begin{cases} \boldsymbol{\mu}_t^{n,i+1} = \boldsymbol{\mu}_t^{k,i} & \text{if } t \neq m \\ \boldsymbol{\mu}_t^{n,i+1} = \boldsymbol{\mu}_t^{k,i} + \boldsymbol{U}(-1,1)_{1\times 3} \cdot \sigma^{k,i} & \text{if } t = m, \end{cases}$$

$$t = 1, \dots, T,$$

$$\sigma^{n,i+1} = \delta \cdot \sigma^{k,i}, \quad \delta \in (0,1].$$

Once the mutation probabilities $\boldsymbol{\zeta}^{k,i}$ are computed, representing the probability of each point $\boldsymbol{\mu}_t^{k,i}$ being mutated, the selection of the point $\boldsymbol{\mu}_m^{k,i}$ is performed through an index m. The mutation of $\boldsymbol{\mu}_m^{k,i}$ is done by adding a random vector $\boldsymbol{U}(-1,1)_{1\times 3} \cdot \sigma^{k,i}$. Finally, also the standard deviation $\sigma^{n,i+1}$ of the new individual is updated by a scaling factor δ.

Regarding the concept of crossover, it is possible to define the function $Cr(\boldsymbol{\Theta}^{k,i}, \boldsymbol{\Theta}^{l,i})$ that represents the crossover operator. It generates a new individual $\boldsymbol{\Theta}^{c,i+1}$ based on two current individuals $\boldsymbol{\Theta}^{k,i}, \boldsymbol{\Theta}^{l,i}$. The operations needed to perform a crossover are:

$$\boldsymbol{\Theta}^{k,i} = \{\boldsymbol{\mu}^{k,i}, \sigma^{k,i}\}, \quad \boldsymbol{\Theta}^{l,i} = \{\boldsymbol{\mu}^{l,i}, \sigma^{l,i}\},$$

$$\boldsymbol{\Theta}^{c,i+1} = Cr(\boldsymbol{\Theta}^{k,i}, \boldsymbol{\Theta}^{l,i}) = \{\boldsymbol{\mu}^{c,i+1}, \sigma^{c,i+1}\},$$

$$\begin{cases} \boldsymbol{\mu}_t^{c,i+1} = \boldsymbol{\mu}_t^{k,i} & \text{if } \zeta_t^{k,i} \cdot U(0,1) \leq \zeta_t^{l,i} \cdot U(0,1) \\ \boldsymbol{\mu}_t^{c,i+1} = \boldsymbol{\mu}_t^{l,i} & \text{otherwise} \end{cases}$$

$$t = 1, \dots, T,$$

$$\sigma^{c,i+1} = \delta \cdot \sigma^{k,i} = \delta \cdot \sigma^{l,i}, \quad \delta \in (0,1].$$

The crossover operator selects points from either $\boldsymbol{\mu}^{k,i}$ or $\boldsymbol{\mu}^{l,i}$ based on the comparison of their respective mutation probabilities $\zeta_t^{k,i}$ and $\zeta_t^{l,i}$ with random values $U(0,1)$. Additionally, the standard deviation $\sigma^{c,i+1}$ of the new individual is updated by a scaling factor δ.

The initialization of the algorithm is as follows:

$$\boldsymbol{\Omega}^1 = \{\boldsymbol{\Theta}^{1,1}, \dots, \boldsymbol{\Theta}^{N,1}\},$$

$$\boldsymbol{\Theta}^0 = \{\boldsymbol{\mu}^0, \sigma^0\}.$$

The initial individual is represented as $\boldsymbol{\Theta}^0$, characterized of the initial mean vector $\boldsymbol{\mu}^0$ and the initial standard deviation σ^0. The initial mean vector $\boldsymbol{\mu}^0$ is computed using a linear interpolation function $L_I(\dots)$ based on the starting point $\boldsymbol{A}$ and the ending point $\boldsymbol{B}$ of the path to be performed, and T. The initial standard deviation σ^0 is chosen as the distance between two consecutive points of the linear path $\boldsymbol{\mu}^0$:

$$\boldsymbol{\mu}^0 = L_I(\boldsymbol{A}, \boldsymbol{B}, T), \quad \sigma^0 = \frac{|\boldsymbol{A} - \boldsymbol{B}|}{M+1}.$$

The scaling factor δ is computed as:

$$\delta = \sqrt[G]{\frac{\sigma^G}{\sigma^0}}, \quad 0 \leq \sigma^G \leq \sigma^0,$$

where σ^G is the desired final value of the standard deviation. Each individual $\boldsymbol{\Theta}^{k,1}$ in the population is initialized using the initial mutation operator $\bar{M}$ applied to $\boldsymbol{\Theta}^0$ (Fig. 1).

$$\boldsymbol{\Theta}^{k,1} = \bar{M}(\boldsymbol{\Theta}^0) = \{\boldsymbol{\mu}^{k,1}, \sigma^{k,1}\}, \quad k = 1. \dots, N.$$

All the components of the mean vector $\boldsymbol{\mu}^{k,1}$ are updated by adding to $\boldsymbol{\mu}_t^0$ a random value from a uniform distribution multiplied by the initial standard deviation σ^0.

$$\boldsymbol{\mu}_t^{k,1} = \boldsymbol{\mu}_t^0 + U(-1,1) \cdot \sigma^0, \quad t = 1, \dots, T.$$

Finally, the standard deviation $\sigma^{k,1}$ of each individual is updated by the scaling factor δ:

$$\sigma^{k,1} = \delta \cdot \sigma^0.$$

D. Algorithm Parameters: Description

This part describes how the population is partitioned and manipulated during different generations of the genetic algorithm. The coefficients C_w, C_m, and C_{cr} represent proportions within the population allocated as selected, mutated, and crossed individuals.

$$C_w, C_m, C_{cr} \in [0,1], \quad C_w + C_m + C_{cr} = 1.$$

These coefficients are multiplied by N to determine the number of individuals assigned to each operation category:

$$C_w \cdot N, C_m \cdot N, C_{cr} \cdot N \in \mathbb{N} \quad C_w \geq \max(C_m, C_{cr}).$$

$$\boldsymbol{\Omega}^i = \{\boldsymbol{\Theta}^{1,i}, \dots, \boldsymbol{\Theta}^{N,i}\} \text{ with } S^{1,i} \leq \dots \leq S^{N,i},$$

the population $\boldsymbol{\Omega}^i$ in generation i-th is sorted based on its fitness scores. Thereafter, the population is divided into three

TABLE I: Parameter configuration that include the number of individuals (N), selection rate (C_w), mutation rate (C_m), crossover rate (C_{cr}), total generations (G), and unique experiment names.

N	C_w	C_m	C_{cr}	G	Exp. Name
8	50.00%	50.00%	0.00%	124	Exp-01
	50.00%	25.00%	25.00%	124	Exp-02
	75.00%	25.00%	0.00%	247	Exp-03
	75.00%	12.50%	12.50%	247	Exp-04
16	50.00%	50.00%	0.00%	62	Exp-05
	50.00%	37.50%	12.50%	62	Exp-06
	75.00%	12.50%	12.50%	122	Exp-07
	75.00%	18.75%	6.25%	122	Exp-08
32	75.00%	12.50%	12.50%	59	Exp-09
	75.00%	18.75%	6.25%	59	Exp-10
	87.50%	6.25%	6.25%	118	Exp-11
	87.50%	9.37%	3.12%	118	Exp-12
64	87.50%	6.25%	6.25%	55	Exp-13
	87.50%	9.37%	3.12%	55	Exp-14
	93.75%	3.12%	3.12%	110	Exp-15
	93.75%	4.68%	1.56%	110	Exp-16

subgroups $\mathbf{\Omega}_w^i$, $\mathbf{\Omega}_m^i$, and $\mathbf{\Omega}_{cr}^i$, representing selected, mutated and crossed individuals respectively (Fig. 1).

$$\mathbf{\Omega}_w^i = \{\mathbf{\Theta}_w^{1,i}, \ldots, \mathbf{\Theta}_w^{C_w \cdot N, i}\},$$

$$\mathbf{\Theta}_w^{k,i} = \mathbf{\Theta}^{k,i}, \quad k = 1, \ldots, C_w \cdot N,$$

where individuals $\mathbf{\Theta}_w^{k,i}$ in $\mathbf{\Omega}_w^i$ remain unchanged as they are directly selected for the next generation.

$$\mathbf{\Omega}_m^i = \{\mathbf{\Theta}_m^{1,i}, \ldots, \mathbf{\Theta}_m^{C_m \cdot N, i}\},$$

$$\mathbf{\Theta}_m^{k,i} = M(\mathbf{\Theta}_w^{k,i}), \quad k = 1, \ldots, C_m \cdot N,$$

where the $\mathbf{\Omega}_m^i$ comprises the mutated individuals from the selected ones.

$$\mathbf{\Omega}_{cr}^i = \{\mathbf{\Theta}_{cr}^{1,i}, \ldots, \mathbf{\Theta}_{cr}^{C_{cr} \cdot N, i}\},$$

$$\mathbf{\Theta}_{cr}^{k,i} = Cr(\mathbf{\Theta}_w^{k,i}, \mathbf{\Theta}_w^{l,i}), \quad k = 1, \ldots, C_{cr} \cdot N,$$

$$l \in [1, C_w \cdot N] \text{ random choice with } l \neq k,$$

in which $\mathbf{\Omega}_{cr}^i$ represents the crossed individuals $\mathbf{\Theta}_{cr}^{k,i}$. The new component is selected by the crossover between two individuals selected from the selected subgroups.

Finally, the next generation population $\mathbf{\Omega}^{i+1}$ is formed by merging the individuals from all three subgroups:

$$\mathbf{\Omega}^{i+1} = \{\mathbf{\Theta}^{1,i+1}, \ldots, \mathbf{\Theta}^{N,i+1}\} = \mathbf{\Omega}_w^i \cup \mathbf{\Omega}_m^i \cup \mathbf{\Omega}_{cr}^i.$$

The process is repeated until the latest generation $\mathbf{\Omega}^G$, where the final individual $\mathbf{\Theta}^G = \mathbf{\Theta}^{1,G}$ represents the final solution for the algorithm (Fig. 1):

$$\mathbf{\Omega}^G = \{\mathbf{\Theta}^{1,G}, \ldots, \mathbf{\Theta}^{N,G}\} \text{ with } S^{1,G} \leq \cdots \leq S^{N,G}.$$

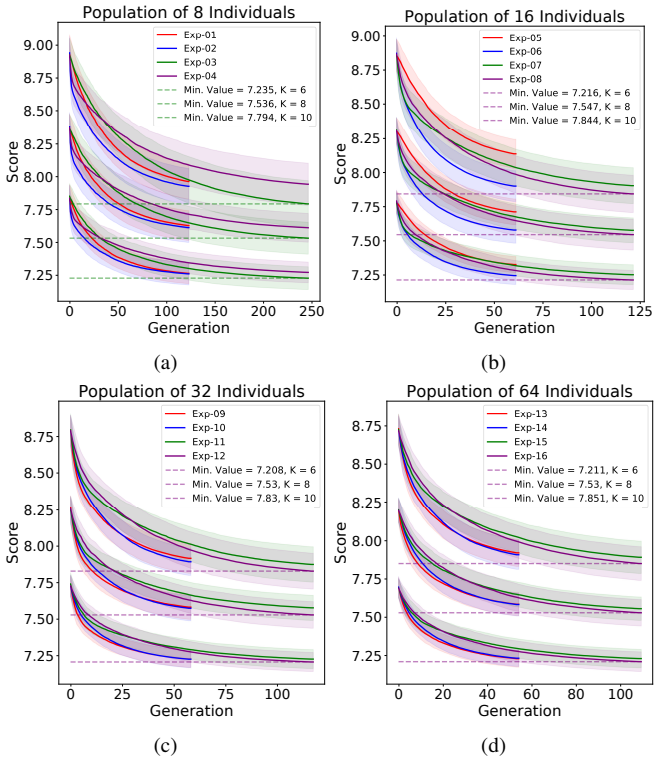


Fig. 2: Shows the score means and standard deviations of experiments with different numbers of individuals (N) and factors (C_w, C_m, C_{cr}) to achieve the minimum fitness function score. Each experiment is conducted for three different spring constants: $K = 6$, $K = 8$, $K = 10$ from bottom to top in each image.

E. Algorithm Parameters: Analysis and Selection

Finally, our goal is to identify parameter configurations that optimize both the convergence speed and solution quality of the algorithm. To achieve this, we need to carefully select parameters such as the population size (N), crossover rate (C_{cr}), mutation rate (C_m), and selection rate (C_w).

Higher values of the crossover rate C_{cr} and mutation rate C_m encourage exploration of the solution space, leading to a more diverse population and increasing the likelihood of discovering novel solutions. Conversely, lower values of these rates promote exploitation, focusing the search on refining promising solutions. The selection rate C_w determines the number of individuals that directly have access to the next generation, higher values of that rate favouring the selection of fitter individuals and potentially accelerating convergence.

Once we have defined the parameters for our algorithm, the next crucial step is to select the most appropriate ones. Tab. I outlines our parameter selection process, where various combinations are explored to evaluate their impact on the algorithm's performance. In our approach, we set a maximum limit of 500 paths to explore, guiding us in determining the parameter choices.

TABLE II: Parameter configuration of Exp-12, that include the number of individuals (N), selection rate (C_w), mutation rate (C_m), crossover rate (C_{cr}), total generations (G).

N	C_w	C_m	C_{cr}	G	Exp. Name
32	87.50%	9.37%	3.12%	117	Exp-12

Additionally, we conduct a simulated experiment (Fig. 2) involving various springs fixed at a specific point in space ($\boldsymbol{C} = [0.6, 0, 0]$) with initial and final point of the path ($\boldsymbol{A} = [0, 0, 0]$, $\boldsymbol{B} = [0.6, 0.6, 0.6]$), with different spring constants ($K = 6, 8, 10$), maximum force $F_{max} = 30N$ and the final exploration ratio $\sigma^G = 0.01$. Each experiment is repeated 200 times (for each configuration from Exp-01 to Exp-16). These experiments provide valuable insights into how the algorithm behaves under different parameter settings, aiding us in making decision regarding their selection.

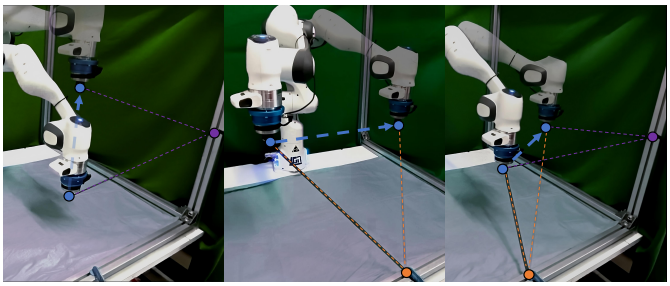
The graphs in Fig. 2, compare various parameter combinations, focusing on achieving the minimum fitness function score. Each experiment was conducted with different numbers of individuals (N) and factor scores (C_w, C_m, C_{cr}). We observe that, overall, the experiment Exp-12 with $N = 32$ individuals demonstrates better convergence and solution quality compared to others. This is observable in terms of convergence speed and consistency in reducing the fitness function score. The choice of the suitable parameters was guided by a comprehensive assessment of performance, with a specific focus on the trade-off between exploration and exploitation. In particular, this parameter configuration effectively balanced the exploration of the fitness space while maintaining stable convergence towards high-quality solutions.

III. Experiments

This section describes the experimental setups and the various tests conducted. To assess the proposed algorithm, a robotic framework has been developed that can emulate elastic behaviour by using a collaborative robot and an elastic cords. Particularly we replicate the Exp-12 parameters condition as shown in Tab. II, in addition the final value of the exploration ratio is selected as $\sigma^G = 0.01$.

A. Experiment Setup

The robot used for the task is a collaborative 7DoF Panda Robot from Franka Emika. It is equipped with an external force/torque sensor from Nordbo, which is securely attached to the robot's end-effector, allowing it to measure the forces exerted on the robot during the motion path. Additionally, the sensor is provided with a metallic element designed to hold the elastic cord, while the other end is attached to a fixed frame. Throughout the experiment, the robot moves from different initial points $\boldsymbol{A}$ to different final points $\boldsymbol{B}$, with the path dividing into 5 points T, and during the motion, the robot is affected by the external force applied by the elastic cords. Our goal is to validate the proposed genetic algorithm to find a suitable path able to minimize the imposed fitness function.



(a) **Vertical** (b) **Horizontal** (c) **Two elastic cords**

Fig. 3: Shows the three different experiment setups: the vertical, the horizontal and the two elastic cords path finding. The figure highlights the initial, the final points and the elastic cords.

TABLE III: Fitness function values and improvements of the first ($S^{1,1}$) and the final ($S^{1,G}$) generation about the three different experiments set up.

Experiment	$S^{1,1}$	$S^{1,G}$	Improvement (%)
Vertical	9.944	8.399	15.53%
Horizontal	9.606	7.722	19.61%
Two elastic cords	11.153	9.711	12.93%

For this purpose, we performed three different experiments that imply several setups (Fig. 3), the Tab. III shows how the fitness function values change for each experiment.

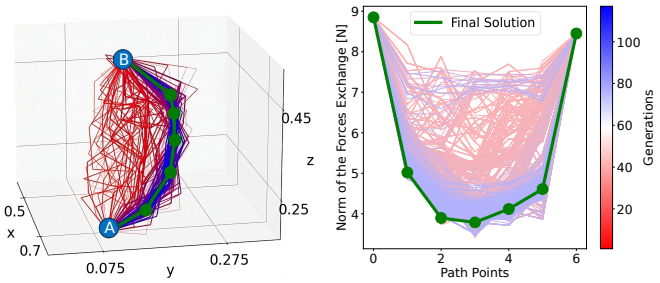
Vertical path finding: In this experiment, the elastic cord is affixed to the metallic element secured to the robot, while its other end is tethered to a vertically fixed frame. The robot follows a trajectory from an initial point ($\boldsymbol{A} = [0.63, 0.11, 0.13]$) to a final point ($\boldsymbol{B} = [0.56, 0.15, 0.50]$), and the imposed maximum force is $F_{max} = 12N$.

Horizontal path finding: In this second test, the end of the elastic cord not connected to the robot is constrained to a fixed horizontal frame ($\boldsymbol{A} = [0.5, -0.25, 0.35]$, $\boldsymbol{B} = [0.5, 0.25, 0.35]$, $F_{max} = 12N$).

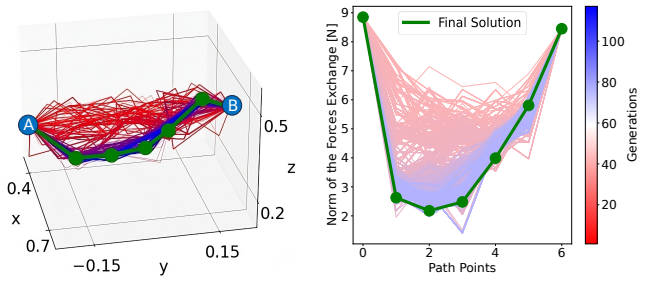
Two elastic cords path finding: This final experiment is a combination of the previous tests. There are two elastic cords: one is attached to the vertically frame and the other is fixed to the horizontal one ($\boldsymbol{A} = [0.45, 0.20, 0.13]$, $\boldsymbol{B} = [0.46, 0.34, 0.43]$, $F_{max} = 18N$).

B. Paths Improvement

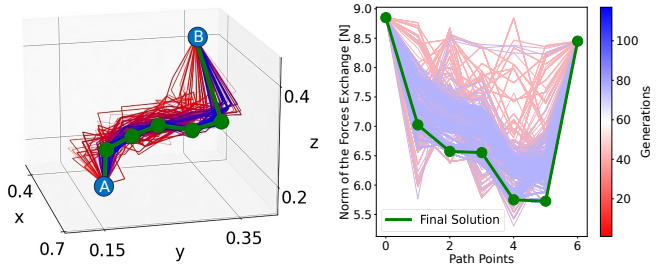
After conducting a series of experiments, we can now proceed to analyze and discuss the results obtained by the algorithm concerning the final path and the forces exchanged with the elastic cords. In the experiment focused on vertical path finding (as shown in Fig. 4a), it is observable that the path tends to orient itself towards the attachment point of the elastic cord while still staying relatively close to the path that would directly connect the starting and ending points. This observation can also be made regarding the forces exchanged between the robot and the elastic cord, which show



(a) Vertical path finding



(b) Horizontal path finding



(c) Two elastic cords path finding

Fig. 4: Shows the evolution, during generations, of the paths (on the left side) and the forces exchange between the robot and the elastic cords (on the right side), for each experiment.

a noticeable decrease in their values during the path between points A and B.

The same observation can also be drawn for the Horizontal path finding experiment, as shown in Fig. 4b. The graphs in Fig. 4c represent the scenario with two elastic cords, in this case, the path has to compensate for two external forces. Finally, the solution obtained from these experiments performs well concerning the fitness function and is congruent with expectations.

Regarding the improvement of the fitness function, we obtain the minimum value in the experiment focused on two elastic cords path finding, which results in a 12.93% improvement. It is plausible that the situation occurred due to the difficulty of mediating two elastic forces which are generated from different points. In such cases, adopting a trade-off path may be the correct solution to compensate for both. The experiment focused on horizontal path finding, instead, has the maximum value, resulting in a 19.61% improvement. These values are presented in Tab. III.

IV. Conclusion

In conclusion, our study proposes a GA based approach that optimizes the path finding of robotic systems when dealing with elastic stiffness. Our experiments and analyses show that the approach can efficiently navigate solution spaces while minimizing forces exchanged between the robot and elastic cords in various experimental scenarios. The parameter selection process also identifies proper configuration that maximize convergence and solution quality.

Moreover, this method has several advantages. It is model-free, requires a finite number of iterations, and is suitable for manipulating deformable objects in repetitive operation tasks.

References

[1] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*. IEEE, 2019, pp. 380–384.

[2] P. A. Vikhar, "Evolutionary algorithms: A critical review and its future prospects," in *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)*. IEEE, 2016, pp. 261–265.

[3] R. Leardi, "Genetic algorithms in chemistry," *Journal of Chromatography A*, vol. 1158, no. 1, pp. 226–233, 2007, data Analysis in Chromatography. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021967307007054

[4] K. Gallagher and M. Sambridge, "Genetic algorithms: a powerful tool for large-scale nonlinear optimization problems," *Computers & Geosciences*, vol. 20, no. 7-8, pp. 1229–1236, 1994.

[5] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering System Safety*, vol. 91, no. 9, pp. 992–1007, 2006, special Issue - Genetic Algorithms and Reliability. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832005002012

[6] A. Pasquali, K. Galassi, and G. Palli, "A fast score-based method for robotic task-free point-to-point path learning," in *2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2023, pp. 1159–1164.

[7] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[8] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Information sciences*, vol. 237, pp. 82–117, 2013.

[9] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[10] K. Galassi and G. Palli, "Robotic wires manipulation for switchgear cabling and wiring harness manufacturing," in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2021, pp. 531–536.

[11] R. Chelouah and P. Siarry, "A continuous genetic algorithm designed for the global optimization of multimodal functions," *Journal of Heuristics*, vol. 6, pp. 191–213, 2000.

[12] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robotics and Autonomous Systems*, vol. 86, pp. 13–28, 2016.

[13] J. Y. Hwang, J. S. Kim, S. S. Lim, and K. H. Park, "A fast path planning by path graph optimization," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 33, no. 1, pp. 121–129, 2003.

[14] A. Caporali, P. Kicki, K. Galassi, R. Zanella, K. Walas, and G. Palli, "Deformable linear objects manipulation with online model parameters estimation," *IEEE Robotics and Automation Letters*, pp. 1–8, 2024.

[15] K. Galassi, A. Caporali, and G. Palli, "Cable detection and manipulation for dlo-in-hole assembly tasks," in *2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2022, pp. 01–06.