

UNIVERSIDADE ESTADUAL DE CAMPINAS
SISTEMA DE BIBLIOTECAS DA UNICAMP
REPOSITÓRIO DA PRODUÇÃO CIENTÍFICA E INTELLECTUAL DA UNICAMP

Versão do arquivo anexado / Version of attached file:

Versão do Editor / Published Version

Mais informações no site da editora / Further information on publisher's website:

<https://periodicos.utfpr.edu.br/rbect/article/view/8218>

DOI: 10.3895/rbect.v13n1.8218

Direitos autorais / Publisher's copyright statement:

©2020 by Universidade Tecnológica Federal do Parana. All rights reserved.

DIRETORIA DE TRATAMENTO DA INFORMAÇÃO

Cidade Universitária Zeferino Vaz Barão Geraldo

CEP 13083-970 – Campinas SP

Fone: (19) 3521-6493

<http://www.repositorio.unicamp.br>

Didactic use of genetic algorithms: a model for teaching robotics

ABSTRACT

The study of articulated robots in higher education necessarily goes through the development of their kinematic models. The inverse kinematic model is usually described algebraically, although this representation is often difficult to obtain. Thus, the use of genetic algorithms in teaching robotics can be very attractive, since they allow students to easily develop models and predict the behavior of robots before their formal development. This way, the results of this work present a relatively fast way to simulate the inverse kinematic model, allowing the designer to have a broader view of the structure of a robot, coming to identify points that must be corrected or that can be optimized. It can be concluded that the use of genetic algorithms in robotics teaching is viable, having as main advantages their easy computational implementation and precision in the representation of kinematic models.

KEYWORDS: Evolutionary algorithms. Process optimization. Computer simulation.

José Tarcísio Franco de Camargo
jffc@bol.com.br
0000-0001-5451-44 94
Centro Regional Universitário de Espírito
Santo do Pinhal, Espírito Santo do Pinhal,
São Paulo, Brasil.

Eliana Anunciato Franco de Camargo
ealcamargo@yahoo.com.br
0000-0001-6934-1653
Centro Regional Universitário de Espírito
Santo do Pinhal, Espírito Santo do Pinhal,
São Paulo, Brasil.

Estéfano Vizconde Veraszto
estefanovv@cca.ufscar.br
0000-0002-4029-4803
Universidade Federal de São
Carlos, Araras, São Paulo, Brasil.

Gilmar Barreto
gbarreto@dsif.fee.unicamp.br
0000-0003-2226-6558
Universidade Estadual de Campinas,
Campinas, São Paulo, Brasil.

Jorge Cândido
jocandido@utfpr.edu.br
0000-0002-6018-5305
Universidade Tecnológica Federal do
Paraná, Campo Mourão, Paraná, Brasil.

INTRODUCTION

The motion described by a manipulator robot can be represented by its direct and inverse kinematic models, as described in Craig (2017). Obtaining the direct kinematic model is relatively simple, since it is defined by a set of transformations among the reference systems of the degrees of freedom (DOF) or joints. Through this model it is possible to determine the position of the tool on the free end of the robot being known the positions of its joints.

The inverse kinematic model, in turn, allows the determination of the state of the joints of a robot according to the desired position for its tool. In this way, when a trajectory for the tool is defined, it is possible to determine the set of joint positions that will allow the robot to describe the desired motion (MILLER, 2017)

Obtaining the inverse kinematic model, however, tends to be more complex than obtaining the direct kinematic model, since it involves the solution of a system of non-linear equations that can admit more than one solution. Even in relatively simple cases, as for the two DOF planar robot described below, the definition of the inverse kinematic model is not trivial.

In this way, being able to predict the behavior of a robot in a relatively simple way, before the formal development of its inverse kinematic model, can become a relevant factor when teaching robotics in higher education. Through the use of genetic algorithms (GAs) it is possible to simulate the behavior of a robot, determining with relative precision the state of its joints in function of the desired position for its free end, allowing design failures to be detected, as well as the identification of possible points for optimization.

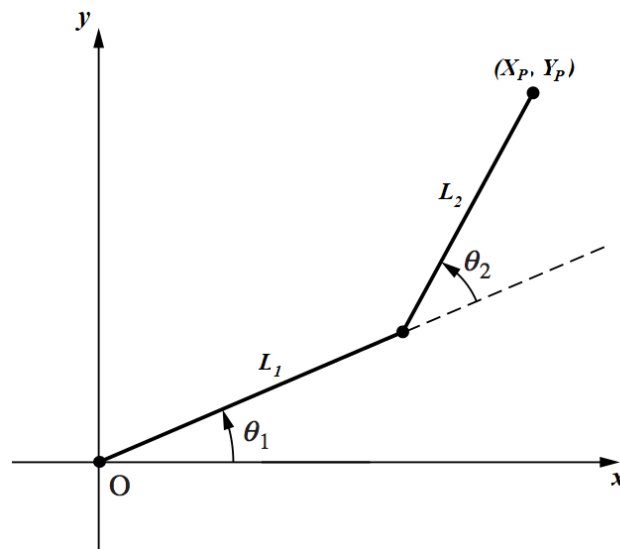
Thus, this paper aims to present the theme of GAs in the context of simulation in robotics, trying to present a generic solution capable of representing the behavior of inverse kinematic models of articulated robots in a practical, efficient and relatively simple way, with a view to explore opportunities of optimization and teaching in robotic projects.

DIRECT AND INVERSE KINEMATICS FOR THE MOTION OF A ROBOT

The study of the direct and inverse kinematic models in robotics can be introduced, in a simple way, through a planar manipulator robot, as shown in Figure 1.

In the structure of Figure 1 there is a manipulator robot with 2 DOF, being its lower end attached to the referential coordinate system and its upper end (tool) free to move over the Cartesian plane. This robot consists of two arms with lengths L_1 and L_2 , with each arm being associated with a rotational joint, whose rotation angles are θ_1 and θ_2 , respectively. The free end of this robot is on the (X_P, Y_P) point of the plane.

Figure 1 – Structure of a planar robot with 2 DOF.



(Source: Elaborated by the authors)

The direct kinematic model of this manipulator, which indicates the position of the free end of the robot according to the angles of the rotational joints, can be represented quite simply by equations (1) and (2).

$$X_P = L_1 \cdot \cos \theta_1 + L_2 \cdot \cos(\theta_1 + \theta_2) \quad (1)$$

$$Y_P = L_1 \cdot \sin \theta_1 + L_2 \cdot \sin(\theta_1 + \theta_2) \quad (2)$$

In turn, the inverse kinematic model of this robot, which indicates the angles θ_1 and θ_2 of the joints in terms of the desired position for the tool, can be described, according to Craig (2017), through equations (3) and (4).

$$\theta_2 = \pm \cos^{-1} \left(\frac{X_P^2 + Y_P^2 - L_1^2 - L_2^2}{2 \cdot L_1 \cdot L_2} \right) \quad (3)$$

$$\theta_1 = \tan^{-1} \left(\frac{Y_P \cdot (L_1 + L_2 \cdot \cos \theta_2) - X_P \cdot L_2 \cdot \sin \theta_2}{X_P \cdot (L_1 + L_2 \cdot \cos \theta_2) + Y_P \cdot L_2 \cdot \sin \theta_2} \right) \quad (4)$$

In this model, the angle θ_2 can assume positive or negative values, depending on the position of the robot's elbow (up or down).

It is remarkable, by observing equations (3) and (4), that the inverse kinematic model actually has considerable complexity, even for a robot that has only 2 DOF in a plane. Thus, as the number of DOF of a robot increases, the difficulty to obtain its inverse kinematic model is also significantly increased.

One way to simulate the inverse kinematics of a robot without the explicit definition of this model is the use of GAs. These algorithms have great vocation

for the solution of optimization problems, as this model can be treated. In this way, a GA can be built in a way that, given an initial estimate for the values of θ_1 and θ_2 and a target function (get as close as possible to the coordinates X_P and Y_P), this estimate can be refined towards an optimal solution. By means of this strategy, in a very simple way, the GA can be initially fed by a random estimate for the values of θ_1 and θ_2 , evolving this solution until a certain condition of minimum positioning error. For example, the relative distance between the desired position for free end of the robot, (X_P, Y_P) , and the value for (X_P', Y_P') determined from the optimization of the estimates for θ_1 and θ_2 .

This way, through the use of GAs, it is possible to determinate the inverse kinematics of a robot without the formal specification of its model, allowing certain behaviors to be identified by the genetic algorithm. Thus, the robotics' study proposed here is driven to the use of GAs, as an alternative to conventional methods for the determination of the inverse kinematic model of a manipulator robot.

FUNDAMENTALS OF GENETIC ALGORITHMS

The use of genetic algorithms in optimization problems was initially proposed in Holland (1975), being popularized through Goldberg (1989) and Haupt (2004). Briefly, it can be said that GAs are an analogy to Charles Darwin's Theory of Evolution of Species (DARWIN, 2009), which, in turn, began with the integration of concepts between natural selection and genetics carried out by Gregor Mendel (MILLER, 2009). In summary, in a computational environment, we aim to search for the evolution of a given solution to a problem, from an initial estimate, possibly rough, to an optimal one. To do so, the optimization process requires a search space, formed by "individuals" of a "population", where the optimal solution is sought for the studied case, as well as an objective function, which leads to the pursuit towards the best possible solution (BING, 2016; GUPTA, 2016; KRAMER, 2017).

In this context, the use of GAs implies a stochastic process, where possible solutions are grouped into a population, being all of which evaluated simultaneously, with higher scores attributed to the best individuals, i.e., to the best solutions. Thus, possible solutions to the problem are treated as individuals within a population of solutions.

In turn, the evolution of the population towards optimized solutions passes through events where individuals combine with each other, in "crossover" processes, or suffer "mutations", similarly to what occurs in biological populations. Such evolution gives rise to new generations that should represent better solutions to the problem addressed.

THE BINARY GENETIC ALGORITHM

The computational implementation of a GA is relatively simple, but it is interesting to code the individuals of a population in a binary representation, for the proper application of the algorithm proposed here. Thus, from the initial population of individuals that constitutes a space of search towards the best

solution of the problem, as long as a certain evolutionary criterion is not reached, the following steps must be repeated.

1. Each of the individuals of the population is evaluated, assigning to the same grades that represent their respective “fitness” for the solution of the treated problem. Such grades are obtained from the objective function, which represents the north for the best solution. The higher the score of an individual, the closer it is to the optimal solution.
2. The best individuals of the population are selected, so that they can be combined in pairs determined by sortition, in a process called “crossover”. Through these crossings, individuals, in pairs, exchange part of their bits, giving rise to a new generation for the population.
3. Each individual of the new generation is subjected to an eventual “mutation”. In this process, bits of a particular individual can change their value, upon occurrence of a low probability event.
4. “Elitism” is applied in the new generation. This implies bringing the best individuals of the current generation to the new one, thus preserving the best solution obtained so far.
5. This procedure is repeated again from the initial step until the expected evolutionary criteria are met, that is, the population or one of its individuals reaches the limits of the optimal solution.

The fitness of an individual x_i of the population can be represented by a function $f(x_i)$, which indicates how close this individual is to the optimal solution to the studied problem. Thus, in a population composed by N individuals, each of them will have its own fitness defined through $f(x)$.

The analysis and comparison of the fitness of the individuals from a population will establish the probability $p(x_i)$ that an individual i will have to generate descendants, through the crossover process. In the case where this probability is directly proportional to the numerical value of $f(x)$, then it can be calculated by Equation (5).

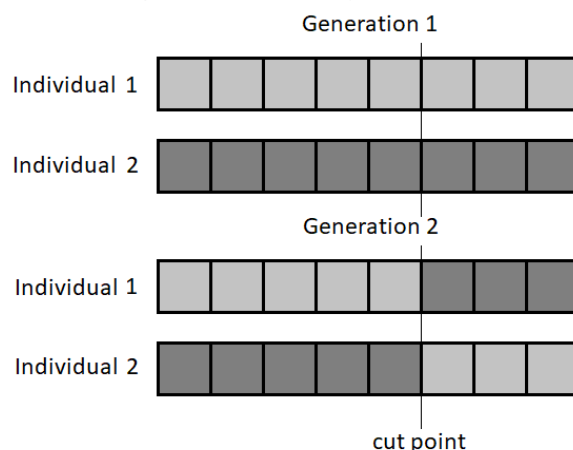
$$p(x_i) = \frac{f(x_i)}{\sum_{k=1}^N f(x_k)} \quad (5)$$

If the probability increases as the numerical value of the objective function tends to zero, as is the case of the model discussed in this paper, then the probability of selecting an individual shall be calculated as shown in Equation (6).

$$p(x_i) = \frac{1 - \left(\frac{f(x_i)}{\sum_{k=1}^N f(x_k)} \right)}{(N-1)} \quad (6)$$

The crossover procedure, to which individuals with better fitness will be subjected, can be understood from Figure 2.

Figure 2 – Crossover procedure between two individuals of a population.

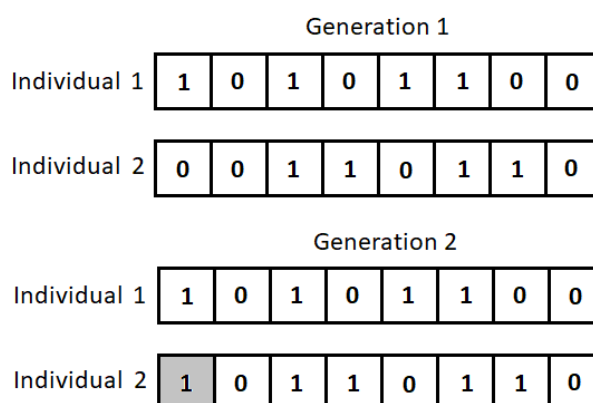


(Source: Elaborated by the authors)

Prior to the crossover, a “cut point” must be defined at random, which will indicate the region to be exchanged between the two individuals. Through Figure 2 it can be observed that, from the cut point, there is the exchange of information between the pair.

In turn, the mutation procedure is represented by Figure 3.

Figure 3 – Mutation occurred in one of the individuals of the population.



(Source: Elaborated by the authors)

Mutation is a random event of low probability, which may occur to reverse the value of one or more bits of individuals in a given population. When applying the mutation procedure in a GA, care must be taken to do not make this process an event with high frequency, which could cause degeneration of the solution represented by the group.

In an GA, “elitism” aims to preserve the best characteristics of the current generation, transporting it to the next generation. Specifically, the fittest individual (or those who are most fit) passes directly from the current generation to the next generation, without undergoing any modifications.

GENETIC ALGORITHMS AND THE KINEMATICS OF THE PLANAR ROBOT

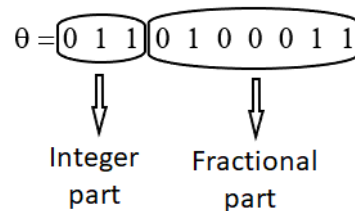
This section describes the adopted procedure regarding the use of a GA to solve the inverse kinematic problem of a planar robot. The solution to the model described at the beginning of this paper requires the calculation of joint angles, θ_1 and θ_2 , which satisfy the positioning of the free end of the robot at a point on the Cartesian plane.

The objective function to be used in the optimization process will be the distance between the position obtained with the values θ_1 and θ_2 applied to equations (1) and (2), (X_p', Y_p') , and the position (X_p, Y_p) arbitrarily specified for the end of the robot. This function is represented by Equation (7).

$$d = \sqrt{(X_p - X_p')^2 + (Y_p - Y_p')^2} \quad (7)$$

For the implementation of the GA, as previously described, it is necessary to code the variables θ_1 and θ_2 in a binary format. Considering that the values of these angles will be constrained to the interval between 0 and 2π , we opted for a coding in ten bits, being the three most significant bits reserved to the integer part and the other seven bits reserved to the fractional part of the number. Figure 4 illustrates this condition.

Figure 4 – Binary representation adopted for the angles θ_1 and θ_2 .



(Source: Elaborated by the authors)

The computational implementation of the GA also requires that an individual be represented by the following data structure.

Structure **Individual**

```
{
  Real  $\theta_1$ ;
  Real  $\theta_2$ ;
  Real fitness;
  Real selection probability;
};
```


In this structure, θ_1 and θ_2 define the solution represented by the individual, “fitness” synthesizes the grade assigned to this solution and “selection probability” represents the probability of the individual being selected for the crossover procedure.

Fitness will be calculated by Equation (7), being remembered that, because it represents a distance, the less its numerical value, the greater will be the fitness of the individual. This implies the use of Equation (6) for the calculation of the selection probability of an individual.

In turn, a population with N individuals will describe the space for the search and evolution towards the optimal solution. This population is described as a vector with N individuals in the computational implementation of this algorithm.

Individual Current_Generation[N];

Thus, from an initial generation with N individuals, the following algorithm can be used to determine future generations, until an optimal solution is reached.

1. Select, at random, “ N ” individuals for the 1st. generation of the population. If there are individuals whose values of θ_1 and θ_2 are outside the permitted limits ($0 \leq \theta < 2\pi$), replace these individuals for others.
2. Calculate the fitness of each individual, through equations (1), (2) and (7).
3. Calculate the selection probability of each individual, through Equation (6).
4. Find the individual with higher fitness in this generation.
5. While the higher fitness does not meet the criterion of stop:
 - a. To start a new generation, repeat “ $N/2$ ” times:
 - i. Select, by sortition, based on the selection probability, two individuals of the current generation.
 - ii. Perform the crossover of the two selected individuals.
 - iii. Store the two individuals generated by crossover in the new generation.
 - iv. End.
 - b. Submit all individuals of the new generation to an eventual mutation process.
 - c. If in the new generation there are individuals whose values of θ_1 and θ_2 are outside the permitted limits ($0 \leq \theta < 2\pi$), replace these individuals for the fittest of the current generation.

- d. Select the less fit individual of the new generation and replace it with the fittest individual of the current generation.
 - e. Calculate the fitness of each individual of the new generation.
 - f. Calculate the selection probability of each individual of the new generation.
 - g. Select the fittest individual of the new generation.
 - h. Make the new generation be the current generation.
 - i. End.
6. End of the algorithm.

Some points of this algorithm are highlighted at next. A considerable advantage of this is the fact that it does not require a predetermined solution so that it can evolve towards an optimal solution. In this way, Step (1) of the algorithm allows us to create a random initial generation as a starting point. As the only constraint, consider that the values of θ_1 and θ_2 of this first generation must be within the allowed range ($0 \leq \theta < 2\pi$).

Step (2) defines the individual fitness from the values of θ_1 and θ_2 . Through equations (1) and (2), the values of these angles will determine the position (X_p, Y_p) represented by this solution. In turn, through Equation (7) the distance between this position and the one desired for the free end of the robot (X_p, Y_p) can be calculated. This distance represents the fitness of the individual, being better the lower is its numerical value.

The probability of selecting a particular individual for crossover, pointed in Step (3), derives from its fitness, that is, from the distance that its respective solution represents. This probability, calculated through Equation (6), takes into account the fitness of the other individuals, having a greater probability of selection the one who is at a shorter distance from the objective (X_p, Y_p). Given its relative behavior, the sum of the probabilities of each individual should be equal to 1.

Finding the individual with the highest fitness, as provided in Step (4), allows us to verify how far the simulation is from its stopping criterion. This criterion, pointed out in Step (5), can be adjusted in several ways. In the simulation developed in this work, it was chosen to establish as a criterion of stopping for the evolution of the population the highest individual fitness, with numerical value inferior to a pre-established limit. Other stop criteria to be considered may be a given number of generations, loss of diversity of a population or convergence to a given solution after a certain number of generations (HOLLAND, 1975).

The selection of individuals for crossover by sortition, indicated in Step (5.a.i), can be interpreted through the format of a "lottery", based on the fitness and on the probability of an individual being selected. In this model of sortition, a set of "lottery tickets" is attributed to each individual of the population, which is proportional to its probability of selection. Thus, through the draw of the "winning ticket", it is defined the individual ("owner of the ticket") that will be selected for the crossover.

The crossover, indicated in Step (5.a.ii), is the event that will trigger the birth of a new generation, different from the current. Moreover, this new generation may pass through a mutation process, as provided in step (5.b), increasing the diversity of the population. As previously indicated, mutation is an event that should be used with caution in GAs, since high mutation rates may lead to degeneration of the population and, therefore, the loss of the solution that it represents.

Crossover and/or mutations are procedures that may eventually give rise to “degenerate” individuals, that is, whose values of θ_1 and θ_2 are outside the permitted limits. This kind of occurrence may be circumvented through the replacement of the degenerate individuals by the fittest individuals from the current generation, as indicated in Step (5.c). In this same sense, with or without degenerate individuals, it is convenient to preserve the fittest individual of the current generation (elitism). To do so, Step (5.d) proposes to replace the less fit individual of the new generation with the fittest of the present generation.

Once the new generation is defined, the fitness and the selection probability of each individual must be recalculated. Finally, the new generation is made the current generation and, if the stop criterion has not been reached, the procedures are repeated for the creation of another generation.

RESULTS AND DISCUSSION

The model presented throughout this text was implemented using “C” language. The working environment where it was developed is “Code::Blocks”, which is free to use and friendly enough for undergraduate students in engineering courses. The basic code developed in classroom was made available in [educapes Portal](http://educapes.capes.gov.br/handle/capes/561163), under the address <http://educapes.capes.gov.br/handle/capes/561163>.

It was simulated a planar robot having his arm, L_1 , length of five units and his forearm, L_2 , four units. The angles θ_1 and θ_2 , which define the solution represented by an individual, were encoded in ten bits, being three bits dedicated to the representation of the integer part of the angle and seven bits destined to the representation of the fractional part. It should be stressed that the angles are restricted to the interval $0 \leq \theta < 2\pi$.

In the developed implementation, a population of ten individuals was used. The cut point for the crossover is set at random, each time this operation is performed, being limited to a maximum of 60% of the bits, counted from the least significant bit. In turn, the probability of mutation was limited to 1%, in order to avoid population degeneration.

The simulation in question consisted of the displacement of the free end of the robot, according to a linear trajectory described by Equation (8).

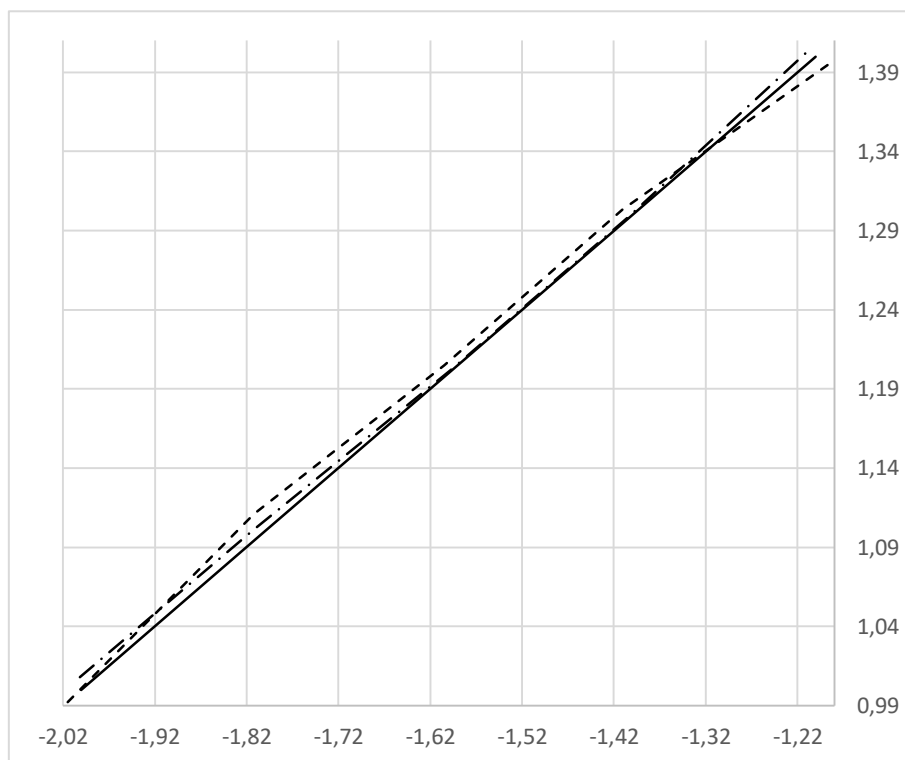
$$Y_p = 0.5 \times X_p + 2 \quad (8)$$

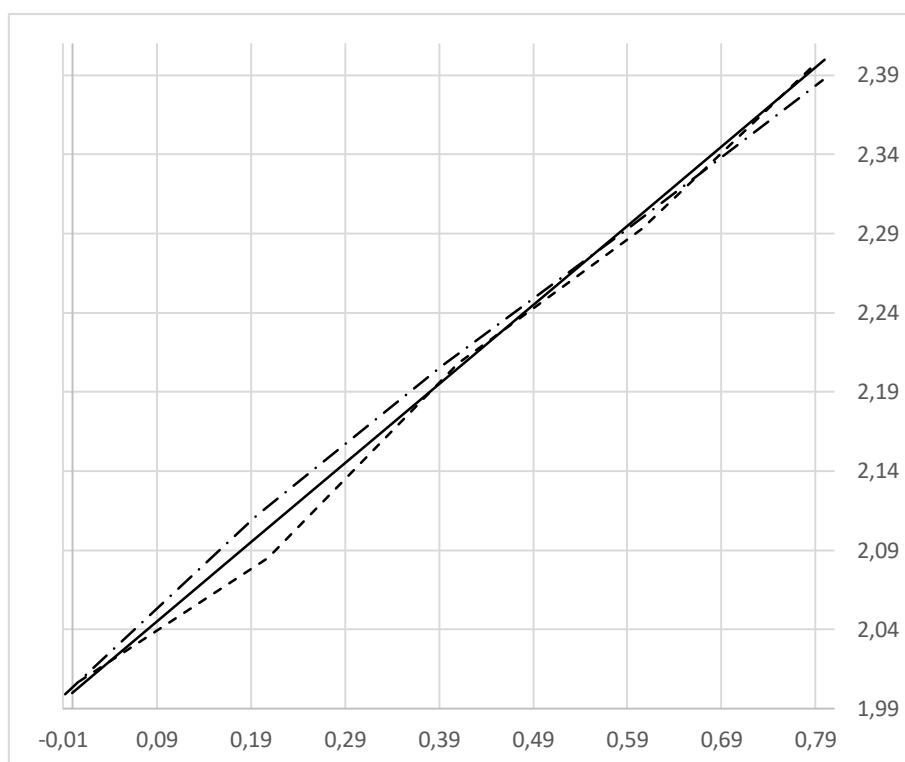
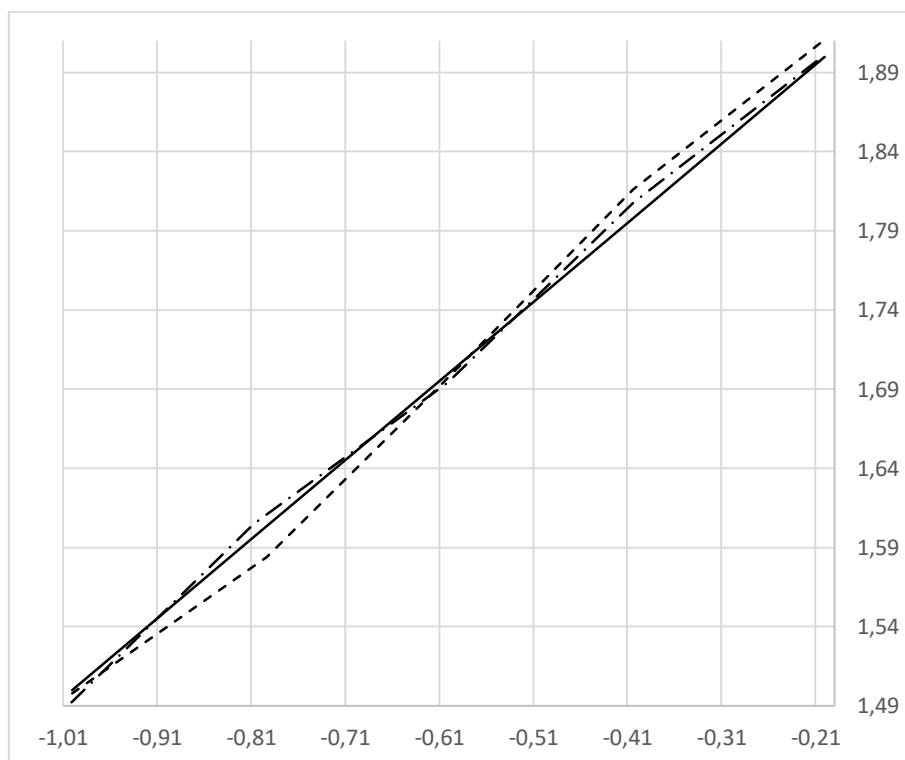
This trajectory was limited to the interval $-2 \leq X_p \leq 2$, with a calculation step equal to 0.2 units.

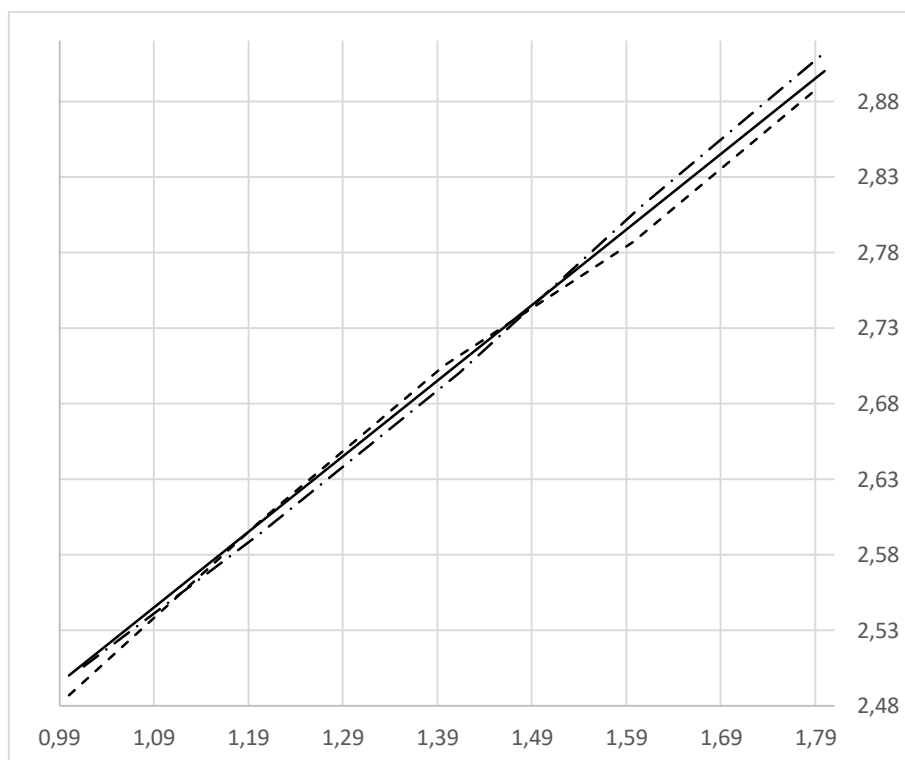
The stop criterion used to finish the calculation of the angles θ_1 and θ_2 associated with a position (X_p', Y_p') consists of obtaining a distance of less than 0.020 units in relation to (X_p, Y_p) .

Figure 5 shows subplots for the trajectory generated by the simulation. The solid line represents the desired path, while dashed lines show the tracks obtained through GA.

Figure 5 – Subplots for the generated paths in the simulation.







(Source: Elaborated by the authors)

The existence of two solutions is justified by the fact that the free end of the planar robot can reach a certain point in two distinct ways: with its “elbow” pointing up or down. Figure 5 also allows us to observe that the trajectories obtained from the simulation are significantly close to the desired trajectory.

Regarding the simulation itself, the results were obtained by predicting, in the algorithm, the presence of crossover, mutations and elitism. However, for testing purposes, the simulation was also performed using only crossover and elitism and only mutations and elitism. These variations led to some interesting empirical observations, which are presented bellow.

- When applied only crossover and elitism, without mutations, it was noted that the algorithm hardly converged toward the stopping criterion. Such an occurrence can be explained by the fact that the crossover used reached only the least significant part of the bits and did not allow the exchange of these in a ratio higher than 60%. Thus, with the most significant bits preserved, the algorithm tends not to converge. Such a situation can be circumvented if a crossover with more than one cut point is used.
- When used mutation and elitism, without crossover, it was noticed that the algorithm presented slow convergence, but the stopping criterion was achieved most of the time. This occurrence can be explained by the fact that mutation, although a phenomenon of low probability, can occur in all bits of the individual. Thus, the occurrence of mutations in the most significant bits of the individual, associated with elitism, allowed the algorithm to show convergence.

- The combination of crossover, mutation and elitism causes the algorithm to present its best results, with convergence occurring, most of the times, after a few dozen iterations.

In concern the use in robotics classes, it was adopted a problem based learning methodology (PBL). Students had previous knowledge about genetic algorithms, which were studied in Artificial Intelligence discipline. This way, when presented to the direct/inverse kinematic problem, they were instigated to develop alternative approaches to the algebraic solution. The use of a genetic algorithm came from insights discussed among students and the teacher (moderator). At next, the implementation was discussed by the group, that decided to use C language to build the prototype.

CONCLUSIONS

The results of this study demonstrate that the use of genetic algorithms for the solution of the inverse kinematic problem of robotics is feasible, especially in situations where the explicit determination of the inverse model is costly. A considerable advantage for the use of a GA in this type of problem is its relatively simple computational implementation. The presented algorithm is also parallelizable, being able to be fragmented in a cluster of computers, reducing the calculation time to obtain more precise solutions. Regarding the precision of the algorithm, it depends heavily on the number of digits used for the binary encoding of an individual. In the example presented here, encoding through ten bits ensured that the error of the calculated trajectory was less than two hundredths of a unit. Another significant advantage in the use of GAs in optimization problems is their high flexibility, which makes them easily adaptable to other robot models or even other types of optimization problems.

Uso didático de algoritmos genéticos: um modelo para o ensino de robótica

RESUMO

O estudo de robôs articulados no ensino superior passa necessariamente pelo desenvolvimento de seus modelos cinemáticos. O modelo cinemático inverso é geralmente descrito algebricamente, embora essa representação seja frequentemente difícil de obter. Assim, o uso de algoritmos genéticos no ensino de robótica pode ser muito atraente, pois permite que os alunos desenvolvam facilmente modelos e prevejam o comportamento dos robôs antes de seu desenvolvimento formal. Desta forma, os resultados deste trabalho apresentam uma maneira relativamente rápida de simular o modelo cinemático inverso, permitindo que o projetista tenha uma visão mais ampla da estrutura de um robô, chegando a identificar pontos que devem ser corrigidos ou que possam ser otimizados. Pode-se concluir que o uso de algoritmos genéticos no ensino de robótica é viável, tendo como principais vantagens sua fácil implementação computacional e precisão na representação de modelos cinemáticos.

PALAVRAS-CHAVE: Algoritmos evolucionários. Otimização de processos. Simulação computacional.

REFERENCES

- BING, X.; MENGJIE, Z.; BROWNE, W. N. A Survey on Evolutionary Computation Approaches to Feature Selection. **IEEE Transactions on Evolutionary Computation**, v. 20, n. 4, p. 606-626, 2016.
- CRAIG, J. J. **Introduction to Robotics: Mechanics and Control**. 4th.edition, New York: Pearson, 2017.
- DARWIN, C. A origem das Espécies e a Seleção Natural. São Paulo: Madras, 2004.
- GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization and Machine Learning**. Boston: Addison-Wesley Longman Publishing Co., 1989.
- GUPTA, A.; YEW-SOON, O.; LIANG, F. Multifactorial Evolution: Toward Evolutionary Multitasking. **IEEE Transactions on Evolutionary Computation**, v. 20, n. 3, p. 343-357, 2016.
- HAUPT, R. L.; HAUPT, S. E. **Practical Genetic Algorithms**. 2nd. Edition, New Jersey: John Wiley & Sons, 2004.
- HOLLAND, J. H. Adaptation in natural and artificial systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Ann Arbor: The University of Michigan Press, 1975.
- KRAMER, O. **Genetic Algorithm Essentials**. New York: Springer, 2017.
- MILLER, Frederic P.; VANDOME, Agnes. F.; McBREWSTER, John. **Gregor Mendel**. London: Alphascript Publishing, 2009.
- MILLER, M. R.; MILLER, R. Robots and Robotics: Principles, Systems, and Industrial Applications. USA: Mc Graw Hill Education, 2017.

Recebido: 25 abr. 2018.
Aprovado: 11 mar. 2020.
DOI: 10.3895/rbect.v13n1.8218
Como citar: CAMARGO, J. T. F. *et al.* . Didactic use of genetic algorithms: a model for teaching robotics. **Revista Brasileira de Ensino de Ciência e Tecnologia**, Ponta Grossa, v.13 , n. 1, p. 79-95, jan./abr. 2020. Disponível em: <<https://periodicos.utfpr.edu.br/rbect>>. Acesso em: XXX.
Correspondência: José Tarcísio Franco de Camargo - jtfc@bol.com.br
Direito autoral: Este artigo está licenciado sob os termos da Licença Creative Commons-Atribuição 4.0 Internacional.

