DOI: 10.48011/asba.v2i1.1585

Algoritmos Genéticos para Determinação da Cinemática Inversa de um Robô de 6DoF ⋆

Matheus Alves e Farnese * **Gustavo Medeiros Freitas** **
Gustavo Pessin ***

* *Programa de Pós-Graduação em Instrumentação, Controle e Automação de Processos de Mineração - Universidade Federal de Ouro Preto (e-mail: matheus.farnese@aluno.ufop.edu.br)*
** *Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais (e-mail: gustavomfreitas@ufmg.br)*
*** *Instituto Tecnológico Vale (e-mail: gustavo.pessin@itv.org)*

Abstract: The inverse kinematics of a robotic manipulator consists on determining the robot's joints positions so that its end-effector reaches a defined position and orientation. Depending on the robot's architecture and the desired pose, the problem may have one, multiple or even no solution. This article presents the development and application of genetic algorithm techniques to determine the inverse kinematics of a robotic manipulator with six degrees of freedom. Focusing on the specific application, we investigate different genetic algorithms configurations, considering the effects of the initial population, number of generations, selection methods, crossover and mutation. The genetic algorithms are implemented in MatLab, which is integrated with RobotStudio software for controlling virtual and real robots from ABB. The proposed solution is validated through simulations with an IRB120 manipulator, controlling the robot's joints in order to achieve the desired pose.

Resumo: O cálculo da cinemática inversa de um manipulador robótico consiste em determinar a posição das juntas do robô de forma que seu efetuador alcance uma posição e orientação definida. Conforme a arquitetura do robô e a pose desejada, o problema pode possuir uma, múltiplas ou até mesmo nenhuma solução. Este artigo apresenta o desenvolvimento e aplicação de técnicas de algoritmos genéticos para determinação da cinemática inversa de um manipulador robótico de seis graus de liberdade. Buscando melhor desempenho para a aplicação específica, são investigadas diferentes configurações de algoritmos genéticos, considerando os efeitos da população inicial, número de gerações, métodos de seleção, crossover e mutação. Os algoritmos genéticos são implementados no MatLab, que é integrado ao software RobotStudio para o comando de robôs virtuais e reais da ABB. A solução proposta é validada através de simulações com um manipulador IRB120, comandando as juntas do robô de forma a alcançar a pose desejada.

Keywords: Genetic Algorithms, Robotics Manipulators, Inverse Kinematic.
Palavras-chaves: Algoritmos Genéticos, Manipuladores Robóticos, Cinemática Inversa.

1. INTRODUÇÃO

A solução da cinemática inversa é fundamental para transformar as especificações de movimento, atribuídas ao efetuador no espaço operacional, nos correspondentes movimentos espaciais conjuntos, que permitem a execução do movimento desejado (Siciliano et al., 2010). É comum encontrar pesquisas tratando desse problema, buscando identificar uma solução genérica e robusta para determinar as variáveis das juntas associadas a uma determinada posição e orientação do efetuador (Vargas, 2013).

Entretanto, a resolução da cinemática inversa não é trivial; quanto maior o número de graus de liberdade, mais complexa se torna a solução. A análise geométrica da posição do efetuador para encontrar os valores angulares das juntas se traduz em um conjunto extenso de equações não lineares. O problema analisado dessa forma pode ter múltiplas soluções ou até mesmo não possuir soluções admissíveis (Siciliano et al., 2010). Uma das formas tradicionais de resolver o problema é desacoplar as juntas de posição e de orientação, simplificando a solução. Essa técnica é aplicável apenas para manipuladores de seis graus de liberdade que utilizam punho do tipo esférico (Spong et al., 2005).

Existem outras abordagens menos convencionais para tratar o problema da cinemática inversa. Métodos algébricos, como a utilização da teoria das bases de Grobner nos polinômios não nulos da análise da cinemática inversa,

⋆ O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ), do Instituto Tecnológico Vale (ITV) e da Universidade Federal de Ouro Preto (UFOP).

podem apresentar soluções (da Silva et al., 2019). Métodos interativos também podem apresentar soluções viáveis, como por exemplo, a inversa filtrada que estima a inversa da matriz Jacobiana dinamicamente para chegar à solução do problema (Vargas, 2013). Outra forma de resolver o problema da cinemática inversa são com os subproblemas de Paden-Kahan. Esse método consiste em utilizar técnicas de análise geométrica, baseado nas rotações de cada eixo, para simplificar o problema, eliminando dependências de algumas das variáveis de juntas conforme o modelo do robô (Murray et al., 1994).

Entretanto os métodos apresentados possuem certas desvantagens que se tornam complicadores em determinadas circunstâncias. Os métodos geométricos se tornam mais complexos quanto mais elaborada for a geometria do robô. Já os métodos interativos possuem soluções únicas que dependem necessariamente do ponto de partida. Em suma, para tratar o problema da cinemática inversa de maneira genérica, os métodos tradicionais são menos eficientes por terem formulações matemáticas mais complexas (Köker et al., 2004).

Considerando isso, uma alternativa para a solução do problema são os algoritmos de aprendizado de máquina. Por exemplo, uma rede neural artificial (RNA) de retro propagação, com função de ativação sigmoidal, pode ser usada para resolver o problema da cinemática inversa (Köker, 2013). Variações desse tipo de técnica como uma RNA configurada em paralelo, podem tornar os resultados da cinemática inversa ainda mais satisfatórios, com menores erros médios das juntas e um melhor rendimento (Nunes, 2016).

Entretanto as RNAs possuem restrições, uma delas relacionada ao conhecimento prévio do problema para coleta de dados e treinamento da rede. Além dessa, o tempo de processamento e o largo consumo de recursos computacionais também podem ser fatores impeditivos. Uma das maneiras de contornar essas dificuldades é aliar essa técnica com algoritmos genéticos. Os resultados da RNA podem ser tomados, por exemplo, como população inicial para um algoritmo genético de forma a minimizar o erro de posição do efetuador (Köker, 2013).

Além da aplicação em conjunto com as redes neurais artificiais, os algoritmos evolutivos podem, por si só, apresentar soluções ao problema. Este artigo tem como foco a aplicação de um algoritmo desse gênero para resolução da cinemática inversa. O objetivo é propor, desenvolver e investigar um algoritmo genético para a solução da cinemática inversa de um manipulador antropomórfico de seis graus de liberdade (6DoF). Considerando que existem múltiplas soluções viáveis dentro do espaço de trabalho deste tipo de robô, a principal motivação é, dada uma pose de referência do efetuador, encontrar a melhor configuração das juntas que implicará em um ganho de eficiência energética e tempo de operação. A solução é desenvolvida no MatLab (MathWorks, Natick) e validada em ambiente de simulação virtual do RobotStudio utilizando o modelo IRB120 (ABB, Zurich).

O artigo está organizando em mais seis seções. A segunda seção traz uma revisão bibliográfica sobre os algoritmos genéticos e sua aplicabilidade no estudo da robótica. A terceira seção expõe os cálculos cinemáticos do manipulador utilizado nesse trabalho. A quarta seção trata da proposta, desenvolvimento e investigação sobre a utilização dos AGs para a resolução da cinemática inversa. A quinta seção trata da aplicação da solução no controle de manipulador de 6 DoF, e propõe um método de planejamento de trajetória baseado em um polinômio de quinta ordem. A sexta seção demonstra as simulações realizadas no ambiente virtual com o manipulador IRB120. Por fim, a sétima seção apresenta a conclusão deste artigo e levanta propostas de trabalhos futuros.

2. REVISÃO BIBLIOGRÁFICA

Os algoritmos genéticos (AGs) são inspirados na teoria da seleção natural, onde a partir de uma população inicial, são cruzados os melhores indivíduos a fim de encontrar o indivíduo ótimo, ou seja, a melhor solução do problema (Holland, 1992). Nesse tipo de algoritmo os indivíduos trocam informações entre si através de operadores genéticos de cruzamento e mutação, evoluindo em novas gerações de indivíduos mais aptos. Esses indivíduos são selecionados por uma função objetivo, *fitness*, que os atribui valores de custo de acordo com sua aptidão para resolução do problema. Esse processo é executado até que os critérios de parada pré-definidos sejam alcançados.

Os AGs podem ser aplicados aos mais diversos tipos de problemas; em especial a utilização destes para soluções em robótica tem se tornado mais recorrente. Um exemplo é a utilização de algoritmos genéticos com imigrantes aleatórios para propiciar a interação de robôs móveis com ambientes dinâmicos. Nesse método ocorre a substituição do pior indivíduo da população corrente e de seus vizinhos próximos por novos indivíduos aleatórios. O intuito é permitir que os indivíduos escapem dos ótimos locais presentes na superfície de *fitness*, induzidos pelas mudanças inerentes ao problema não estacionário (Tinós, 2007).

Ainda acerca de robôs móveis, outra aplicação é no controle do caminhar de robôs com pernas (Heinen, 2007). Os AGs são utilizados para otimizar os parâmetros de técnicas de controle como tabelas de ângulos e posição para autômatos, parâmetros de elipse em funções cíclicas e pesos sinápticos de redes neurais. A função *fitness* é estipulada a partir do deslocamento total do robô, além de considerar sensores que indicam o toque das patas no chão e a estabilidade do robô. Um segundo algoritmo genético é utilizado para evoluir a morfologia mais adequada do robô para permitir o caminhar que, associado ao primeiro, apresenta soluções satisfatórias.

Os AGs também podem ser aplicados para robôs com outras configurações. Uma das possibilidades é a resolução da cinemática inversa de um robô planar com 3DoF minimizando o deslocamento angular das juntas (Nunes, 2007). Uma variação dessa solução é a utilização de algoritmos genéticos contínuos. Esse método possui o mesmo objetivo dos AGs convencionais, entretanto são aplicados no nível do espaço das juntas, ao contrário do convencional onde os operadores são aplicados no nível do espaço de trabalho. Esse tipo de estratégia é útil na suavização da trajetória do robô, mas requer um tempo mais elevado para execução do algoritmo (Momani et al., 2016).

Uma abordagem que se assemelha às estratégias anteriores é a utilização de um algoritmo genético híbrido. Nesse caso dois algoritmos são utilizados: o primeiro faz uma exploração dentro do espaço de trabalho do robô a fim de identificar boas áreas, ou nichos, onde pode estar à solução. Na sequência o outro algoritmo busca a solução ótima dentro das áreas previamente determinadas. Essa estratégia contribui com a rápida convergência para soluções precisas, além de flexibilizar o uso para manipuladores diversos, bastando para tanto adequar às equações de cinemática direta (Pavan et al., 2018).

Outra vantagem dos algoritmos evolutivos na resolução da cinemática inversa é que essa pode ser resolvida considerando mais objetivos para alcançar a pose desejada. Simultaneamente o algoritmo pode minimizar a energia despendida, o tempo gasto, o número de rotações necessárias, além de minimizar o erro de posição no espaço. A redução do dispêndio de energia é baseada no consumo energético de cada motor do manipulador durante a movimentação ($W.s/^{o}$). Para a redução do tempo da trajetória é considera a especificação da velocidade de rotação de cada junta ($^{o}/s$). A quantidade de rotações considera o deslocamento angular de todas as juntas (^{o}). Já o erro de posição no espaço é baseado nas equações de distância euclidianas (mm). Dessa forma o *fitness* é definido como uma somatória de cada um desses objetivos de otimização, que ao atingir valores de ótimo global apresenta bons resultados de minimização de tempo e gasto energético (Števo et al., 2014).

Os AGs são úteis também na otimização do planejamento de trajetórias de manipuladores robóticos. Um exemplo é a aplicação para otimização do tempo da trajetória quando o manipulador tem movimentos restritos, mas se conhece o caminho exato a ser percorrido. Ou seja, desde que o manipulador não seja redundante, o caminho do espaço articular relacionado pode ser escolhido a partir de um conjunto finito de poses e ser otimizado com relação ao tempo. Para tanto o problema é parametrizado em termos de velocidade e distância em relação à variação de tempo da trajetória, definindo dessa forma a representação genética dos indivíduos. A função *fitness* nesse caso é determinada como minimização de tempo, considerando para isso a área sob a curva da trajetória (Ferrentino et al., 2019).

3. CINEMÁTICA DO IRB120

O IRB120 é um robô industrial com 6 juntas rotacionais, em cadeia aberta, configuração antropomórfica e punho esférico (ABB, 2019), que será objeto de estudo deste artigo. Uma das maneiras mais usuais de calcular a cinemática direta deste tipo de manipulador é através da convenção de Denavit-Hartenberg (Spong et al., 2005). Para tanto é necessário conhecer suas dimensões e seu modelo cinemático, representados na Fig. 1.

A cinemática direta do manipulador é calculada pela convenção de DH através da equação 1, utilizando os valores apresentados na tabela da Fig 1, construída com base na arquitetura do robô apresentado na mesma figura:

$$A_{i-1}^{i} = Rot_z(\theta_i)Trans_z(d_i)Trans_x(a_i)Rot_x(\phi_i), \quad (1)$$

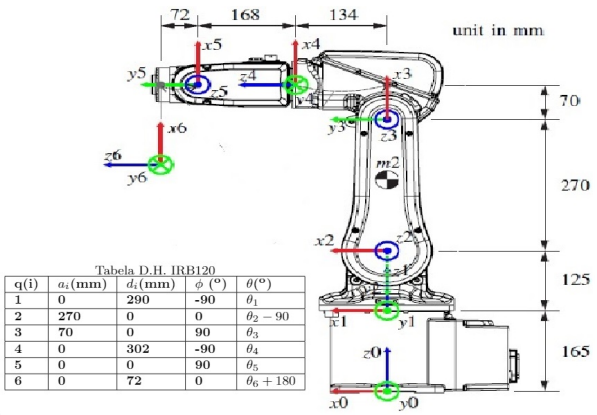

Tabela D.H. IRB120

q(i)	a_i(mm)	d_i(mm)	ϕ (°)	θ(°)
1	0	290	-90	θ_1
2	270	0	0	$\theta_2 - 90$
3	70	0	90	θ_3
4	0	302	-90	θ_4
5	0	0	90	θ_5
6	0	72	0	$\theta_6 + 180$

Figura 1. Características físicas e cinemáticas do IRB120 e respectiva tabela D.H. Adaptado (Roth, Peter M. et al., 2017).

onde Rot_x e Rot_z são as matrizes de rotação ($\in SO(3)$) ao redor dos eixos x e z, e $Trans_x$ e $Trans_z$ são as translações ($\in \mathbb{R}^3$) também ao longo dos eixos x e z.

4. ALGORITMOS GENÉTICOS PARA DETERMINAÇÃO DA CINEMÁTICA INVERSA

Uma das principais vantagens da utilização de AGs, para determinar a cinemática inversa de um robô, é que estes não necessitam de conhecimento prévio do problema a ser tratado. Isso porque, todos os indivíduos iniciais, ditos população inicial, são gerados aleatoriamente, sem, portanto, interessar qual a disposição desejada no final (Nunes, 2007). Além disso, é possível determinar o resultado baseado em mais de um objetivo de otimização (Števo et al., 2014).

4.1 Cromossomos e População Inicial

A população inicial conterá indivíduos (i) com representação genética de seis ângulos (θ_i), que serão aleatoriamente determinados, conforme mostrado pela equação 2. Essa estará condicionada que todos os ângulos estejam dentro dos limites de juntas descritos no manual do IRB120 (ABB, 2019). Tratar indivíduos que estejam fora dos limites de operação do robô pode gerar erros no resultado, retornando uma solução fora do espaço de trabalho. Além dos limites de juntas, é determinado o tamanho de cada cromossomo, e o tamanho da população, que será objeto de investigação desse trabalho.

$$i = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6). \quad (2)$$

4.2 Função Objetivo

A determinação da função objetivo do problema é uma etapa crucial e não trivial, que diz respeito à atribuição matemática de custos relacionados ao objetivo do problema a ser otimizado. Para esse caso os objetivos são três: a minimização do deslocamento angular das juntas, do erro de posição final e do erro de orientação. O intuito é encontrar a pose final que acarrete o menor deslocamento angular possível ao robô. Para os três casos podem ser

utilizados conceitos de distância euclidiana. Dessa forma as equações 3, 4 e 5 representam os objetivos para solução da cinemática inversa:

$$f_1 = \sqrt{(x - p_x)^2 + (y - p_y)^2 + (z - p_z)^2}, \quad (3)$$

$$f_2 = \sqrt{(\Phi - \Phi_d)^2 + (\Theta - \Theta_d)^2 + (\Psi - \Psi_d)^2}, \quad (4)$$

$$f_3 = \sqrt{(\theta_1 - \theta_{1i})^2 + (\theta_2 - \theta_{2i})^2 + ... + (\theta_6 - \theta_{6i})^2}, \quad (5)$$

onde f_1 corresponde à minimização do erro de distância de posição no espaço, considerando x, y e z o ponto que se deseja alcançar e p_x, p_y e p_z a posição do efetuador resultante das variáveis de juntas do indivíduo testado pelo algoritmo. Já f_2 diz respeito à minimização do erro de orientação, onde Φ, Θ e Ψ são os ângulos de Euller, *roll*, *pitch* e *yaw*, resultantes do indivíduo em avaliação e Φ_d, Θ_d e Ψ_d é a orientação desejada. Por fim f_3 é a função com o objetivo de minimizar o deslocamento angular, sendo θ_1,θ_2...θ_6 os ângulos atuais do indivíduo sendo avaliado pelo algoritmo e θ_{1i},θ_{2i}...θ_{6i} as variáveis de juntas na posição inicial. Assim sendo a função objetivo (*F.O.*) é dada pelo somatório das três equações, conforme as equação 6:

$$[F.O.] = f_1 + f_2 + f_3. \quad (6)$$

Com essa formatação o algoritmo genético passa a ser multiobjetivo, e o resultado ótimo se dará pela minimização das três funções propostas. Considerando que o consumo de energia total do robô corresponde ao somatório de energia dispendida por cada junta durante seu deslocamento, também é possível ponderar a função objetivo f_3 atribuindo um peso W_i, proporcional ao torque máximo τ_i de cada junta descrito no manual do IRB120 (ABB, 2019), de forma a melhorar sua eficiência energética, conforme equações 7, 8 e 9:

$$\tau_t = \tau_1 + \tau_2 + \tau_3 + \tau_4 + \tau_5 + \tau_6, \quad (7)$$

$$W_i = \tau_i / \tau_t, \quad (8)$$

$$f_3 = \sqrt{W_1(\theta_1 - \theta_{1i})^2 + ... + W_6(\theta_6 - \theta_{6i})^2}. \quad (9)$$

Apesar de a função objetivo somar valores de grandezas distintas, esse não é um fator impeditivo para a resolução do problema. Em suma, a interpretação dos números é realizada de maneira adimensional. Minimizando o total das três equações para um valor ótimo global, será garantido o menor deslocamento angular, com erros de posição e orientação dentro dos parâmetros aceitos pelas restrições do problema.

4.3 Restrições

Ao buscar a solução ótima é preciso garantir que os erros de posição e orientação do efetuador do robô estejam dentro de parâmetros aceitáveis, o mais próximo possível de zero. Em outras palavras, o algoritmo pode se deparar com uma solução de ótimo local e terminar sua busca entendendo que tenha encontrado a melhor solução, ainda que essa esteja distante da posição e orientação desejadas. Por esse motivo o erro de posicionamento no espaço e da orientação do robô são tratados como uma restrição para a solução. Ou seja, para cada possível solução o algoritmo checa se essa corresponde a pose desejada, sendo essa uma das condições de interrupção da busca do algoritmo. Dessa forma a evolução acontecerá até que as condições da restrição sejam atendidas, garantindo que o efetuador chegue ao ponto determinado com orientação desejada. Para o caso, são determinados limites aceitáveis de erro de posição de 1mm e erro de orientação de 1º. O objetivo de estipular a margem de erro aceitável na restrição é reduzir o custo computacional e tempo necessário para encontrar a solução perfeita onde todos os erros são iguais a 0. As implementações matemáticas das restrições de posicionamento são demonstradas pela equação 10, e de orientação pela equação 11:

$$\sqrt{x^2 - p_x^2} \leq 1, \quad \sqrt{y^2 - p_y^2} \leq 1, \quad \sqrt{z^2 - p_z^2} \leq 1, \quad (10)$$

$$\sqrt{\Phi^2 - \Phi_d^2} \leq 1, \ \sqrt{\Theta^2 - \Theta_d^2} \leq 1, \ \sqrt{\Psi^2 - \Psi_d^2} \leq 1. \quad (11)$$

Um possível questionamento seria sobre a necessidade de minimizar o erro de posição e orientação uma vez que estes já são restrições ao problema. Mas, caso a minimização não ocorra, o algoritmo busca deslocamentos angulares mínimos ignorando a importância de minimizar o erro em relação à posição e orientação final. Dessa forma, as condições de restrição nunca são atendidas, impossibilitando uma solução viável.

4.4 Implementação do Algoritmo Genético

Um dos softwares que permite o desenvolvimento de algoritmos genéticos é o MatLab. Uma das ferramentas desse software é o *Genetic Algorithm Optmization ToolBox*, conhecido como GAOT, usada para problemas de otimização em que a função objetivo ou restrição é contínua, descontínua, estocástica, não possui derivadas ou inclui simulações ou funções tipo caixa preta. Esse é um algoritmo estocástico de base populacional que pesquisa aleatoriamente por mutação e cruzamento entre os membros da população (MathWorks, 2020). O algoritmo proposto neste artigo foi desenvolvido no GAOT, que ainda possui o benefício de fácil alteração dos seus parâmetros, permitindo realizar diferentes testes comparativos sem dificuldades.

4.5 Investigação dos Algoritmos Genéticos

O objetivo do algoritmo desenvolvido é minimizar o deslocamento angular com margens de erro de posicionamento e orientação aceitáveis. Para identificar a melhor solução são analisados três fatores fundamentais: o tempo de execução do algoritmo até atingir o valor ótimo, o erro de médio de posição e o erro máximo dos ângulos de orientação, conforme equações de distância euclidiana. Além desses três, são avaliados também os critérios de parada, indicando as interrupções em valores ótimos com atendimento às restrições, por mínimos locais irreversíveis ou por limite de gerações.

Na análise são considerados os efeitos da população inicial e do número máximo de gerações. Entendendo indivíduos como "i", e gerações como "g", foram estipuladas as seguintes variações: 10i25g, 25i25g, 25i40g, 40i40g e 40i55g. Para cada combinação foram coletadas 8 amostras de resultados. Além disso, foram analisadas variações na configuração do AG, como método de seleção, *crossover* e sua taxa, mutação, *population overlapping*, elitismo e processamento

de populações em paralelo. É importante citar que todas as análises foram feitas para a mesma pose, uma vez que o problema busca minimizar o deslocamento angular, poses diferentes podem interferir nos resultados comparativos. O ponto utilizado tem coordenadas cartesianas x, y e z iguais a 400mm, 200mm e 700mm e os ângulos de orientação são iguais a 0º. Para todos os casos foi considerado como configuração inicial a posição de *Home*, onde todas as variáveis de juntas são iguais a zero.

Efeitos da População Inicial e Número de Gerações A primeira análise foi feita considerando exclusivamente o efeito do tamanho da população inicial e o número máximo de gerações sob o resultado da otimização. As configurações do algoritmo para a primeira simulação (AG1) são descritas na Tabela 1. A Fig. 2, demonstra os resultados.

Tabela 1. Configurações do AG1.

Método de seleção	**Torneio**
Crossover	**Heurístico**
Taxa de *crossover*	**0,5**
Mutação	**AdaptFeasible**

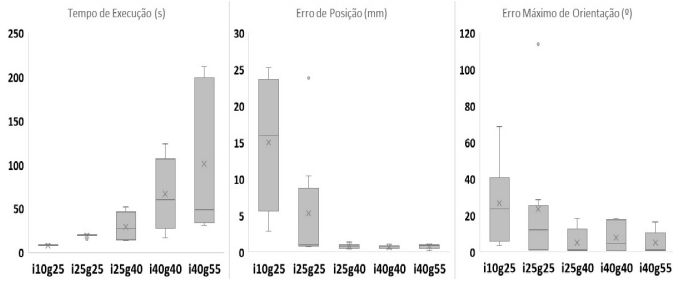

Figura 2. Efeitos da População Inicial e do Número de Gerações.

É possível perceber que para os números menores de indivíduos e gerações o algoritmo converge rapidamente para a região da solução ótima, entretanto não há gerações o suficiente para encontrar o ótimo global. Em consequência, as restrições não são atendidas, e os erros de posição e orientação são grandes. Em contrapartida, ao aumentar o número de indivíduos e gerações o tempo se eleva, mas os erros de posição e orientação alcançam níveis satisfatórios. Existe ainda o problema no qual o algoritmo fica preso em mínimos locais, sem alcançar um valor de ótimo global que atenda as restrições. Nas 40 amostras expostas nesses gráficos 32,5% foram interrompidas ao atingir o limite de gerações, 27,5% ficaram presas em mínimos locais e 40% convergiram a um valor de ótimo global. Para o caso de 40 indivíduos e 55 gerações o tempo médio dos testes que convergiram foi de 40s. Esse tempo foi considerado elevado pensando em uma possível aplicação que exija uma tomada de decisão em tempo real. Por esse motivo a opção foi por não aumentar mais o número de indivíduos e gerações.

Efeitos do Elitismo e Sobreposição de Populações Para a sequência dos testes foram mantidas todas as configurações do AG anterior e acrescentado uma taxa de 25% de sobreposição de população e elitismo para 2 indivíduos. A Tabela 2 expõe a configuração do genético para a segunda simulação (AG2) e os gráficos da Fig. 3 mostram os resultados alcançados.

Tabela 2. Configurações do AG2.

Método de seleção	**Torneio**
Crossover	**Heurístico**
Taxa de *crossover*	**0,5**
Mutação	**AdaptFeasible**
Sobreposição de População	**0,25**
Elitismo	**2**

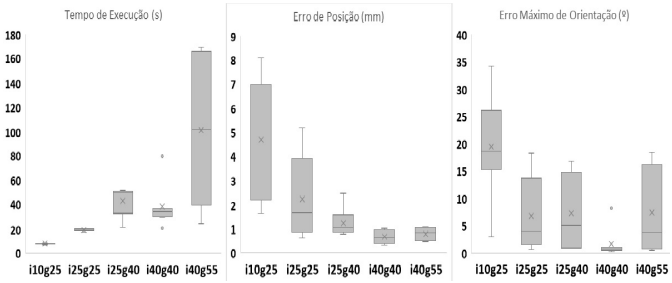

Figura 3. Efeitos do Elitismo e da Sobreposição de Populações.

Para esse caso o aumento do tempo e minimização do erro conforme a elevação do número de indivíduos e gerações se manteve. Mas diferente do que foi observado para as configurações anteriores, os erros de posição e de orientação foram menores para menos indivíduos e gerações. Mas ainda que os erros tenham sido menores nos testes com poucas gerações, estes continuaram sendo interrompidos sem encontrar o ótimo global. O ponto negativo foi que para os indivíduos que convergiram ao ótimo global o tempo de execução foram significativamente maiores. Este fator pode ser prejudicial para aplicações que necessitem de uma tomada de decisão mais ágil. Das 40 simulações apresentadas nessa seção 37,5% foram interrompidas ao atingir o limite de gerações, 25% ficaram presas em mínimos locais e 37,5% convergiram para um valor de ótimo global.

Foi avaliada ainda uma variação da taxa de *crossover* para 0.8, mantendo as demais configurações. Essa variação não trouxe melhoras aos resultados. Não houve redução dos níveis de erro e os tempos de execução foram elevados. Apenas 35% dos testes convergiram a um valor de ótimo global, ao passo que 27,5% ficaram presos em ótimos locais e 37,5% foram interrompidos pelo limite de gerações.

Efeito de Populações em Paralelo Foram considerados também os efeitos do processamento de populações em paralelo. A Tabela 3 apresenta as configurações desta simulação (AG3) e a Fig. 4 mostra os gráficos de tempo e erros obtidos neste teste.

Tabela 3. Configurações do AG3.

Método de seleção	**Torneio**
Crossover	**Heurístico**
Taxa de *crossover*	**0,5**
Mutação	**AdaptFeasible**
Populações em Paralelo	**Sim**

Nesse caso os tempos de execução aumentaram muito, sem uma melhora dos índices de erros. O tempo médio dos testes com 25 indivíduos e 40 gerações foi de 166s e apenas 2 destes convergiram ao valor de ótimo global. Portanto

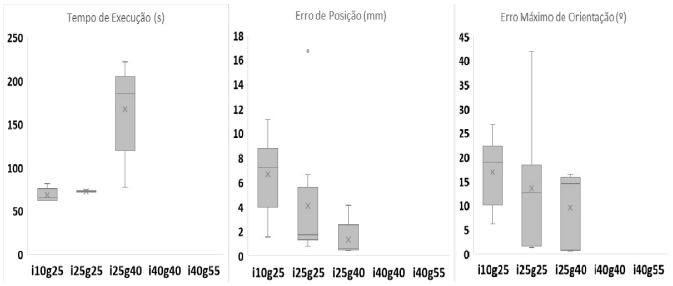

Figura 4. Efeitos do Processamento de População em Paralelo.

a análise foi interrompida precocemente e já considerada inviável.

Efeito dos Métodos de Seleção, crossover e Mutação Foi observado também o comportamento do algoritmo genético quando variado o método de seleção e *crossover*. Para a quarta simulação (AG4) os três parâmetros foram alterados, mantendo a taxa de *crossover*, a sobreposição de população e o elitismo conforme a Tabela 4. Os gráficos das Fig. 5 mostram os resultados dessa variação.

Tabela 4. Configurações do AG4.

Método de seleção	**Roleta**
Crossover	**Aritmético**
Taxa de *crossover*	**0,5**
Mutação	**Gaussiana**
Sobreposição de População	**0,2**
Elitismo	**2**

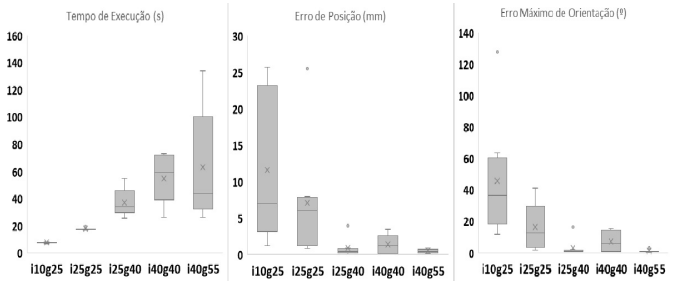

Figura 5. Efeito dos Métodos de Seleção, *crossover* e Mutação.

Nessa hipótese foi possível notar uma melhora na taxa de convergência para valores ótimos. Apesar de 47,5% dos algoritmos terem sido interrompidos pelo limite de gerações, 45% deles convergiram a um valor de ótimo global. Consequentemente apenas 7,5% ficaram presos em valores de ótimo local. Portanto essa configuração demonstra maior robustez para escapar de ótimos locais.

Outra configuração analisada foi considerando o método de seleção e mutação uniformes e o método de *crossover* como Single Point. Entretanto essa configuração teve o pior desempenho entre os testes. Isso porque para qualquer número de indivíduos e/ou gerações, o algoritmo ficou preso em valores de ótimo local.

Efeito do Doping da População Inicial Considerando que normalmente um robô antropomórfico não utiliza todo seu espaço de trabalho e que possivelmente as principais áreas de trabalho são conhecidas, foi utilizada uma estratégia de dopar a população inicial com um indivíduo conhecido, dentro de um espaço de trabalho restrito. Ou seja, o primeiro indivíduo da população inicial foi alterado de forma que as variáveis de junta fossem propositalmente mais próximas às da pose do robô no ponto desejado. Para realizar essa quinta simulação (AG5) foram consideradas as melhores configurações avaliadas nos testes anteriores, conforme Tabela 5. Os resultados estão demonstrados nos gráficos da Fig. 6.

Tabela 5. Configurações do AG5.

Método de seleção	**Roleta**
Crossover	**Aritmético**
Taxa de *crossover*	**0,5**
Mutação	**Gaussiana**
Sobreposição de População	**0,2**
Elitismo	**2**

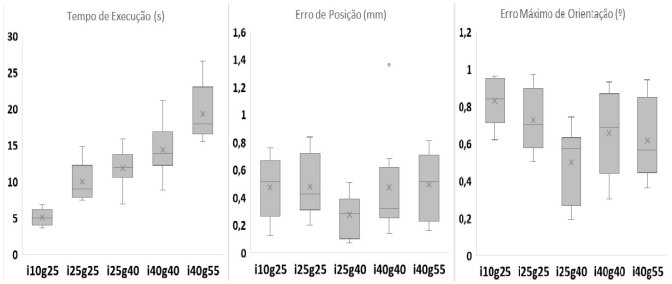

Figura 6. Efeitos do Doping da População Inicial.

Os resultados alcançados com uma população inicial dopada foram excelentes. Em 100% dos testes houve convergência para um valor de ótimo global, em tempos significativamente baixos e com valores de erros dentro do limite aceito. Para populações com 10 indivíduos o algoritmo convergiu em um tempo médio de 5s, com erros atingindo a ordem de 0,12mm para posição 0,62º para orientação. O melhor tempo de evolução foi de 3,57s.

5. APLICAÇÃO E COMANDO DO MANIPULADOR

Para emular a aplicação dos algoritmos desenvolvidos na seção anterior, foi criado um *socket* de comunicação TCPIP entre uma interface de usuário (GUI) do MatLab e o ambiente de desenvolvimento e simulação da ABB, o RobotStudio. Essa interface permite que a solução gerada seja trafegada até a plataforma de simulação do robô, cujo controlador interpreta a linguagem estruturada em módulos, denominada RAPID. Nesse software é determinado ao robô um movimento no espaço das juntas através da função *MoveAbsJ*. Essa função é utilizada para mover o robô a uma configuração definida por valores absolutos das juntas, levando o efetuador à pose desejada ao longo de um caminho não linear (ABB, 2018).O movimento ocorre conforme as curvas de velocidade e aceleração pré-definidas pelo fabricante.

Para essa aplicação, o cálculo da cinemática inversa através de métodos heurísticos permite a determinação da posição das juntas do robô conforme uma pose desejada do manipulador no espaço. Entretanto, a execução do comando *MoveAbsJ* é realizada utilizando as rampas de aceleração e desaceleração definidas no controlador do robô da ABB (IRC5). Outra estratégia para o controle cinemático do robô consiste em determinar a trajetória executada pelas juntas por meio de um polinômio de quinta

ordem (Spong et al., 2005). Com esse polinômio, é possível definir a posição, velocidade e aceleração de cada junta para cada instante de tempo utilizando as equações 12, 13 e 14:

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5, \quad (12)$$

$$\frac{\partial \theta}{\partial t} = a_1 + 2a_2(t) + 3a_3(t^2) + 4a_4(t^3) + 5a_5(t^4), \quad (13)$$

$$\frac{\partial^2 \theta}{\partial t} = 2a_2 + 6a_3(t) + 12a_4(t^2) + 20a_5(t^3), \quad (14)$$

onde $i = 1, ..., n$ indica a junta do robô, t é o tempo definido do percurso em segundos, e a_0, a_1, a_2, a_3, a_4, a_5 e a_6 são os coeficientes do polinômio, considerando as posições iniciais e finais (calculadas pelo AG) das juntas, tempos iniciais e finais (definidos pelo operador), e assumindo nulas as velocidades e acelerações iniciais e finais das juntas.

As curvas de posição, velocidade e aceleração das 6 juntas relacionadas ao exemplo apresentado na Seção 6 são apresentados na Fig 7.

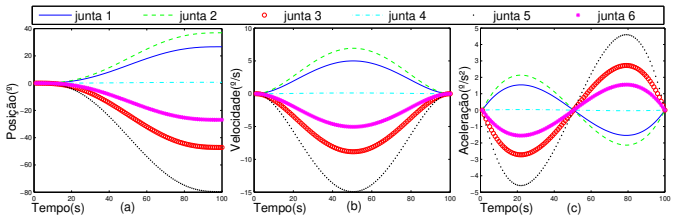

Figura 7. (a) Posições, (b) velocidades e (c) aceleração das juntas durante a trajetória.

O planejamento de trajetória utilizando um polinômio de quinta ordem permite incorporar restrições de acelerações. Essas restrições são necessárias para evitar variações instantâneas de comando das juntas, provocando movimentos bruscos do robô.

6. SIMULAÇÃO EM AMBIENTE VIRTUAL

Para verificar a efetividade do método de computação evolutiva para resolver a cinemática inversa, foi utilizada a configuração do AG5 integrado ao ambiente virtual do RobotStudio. Entretanto, buscando demonstrar que o método estocástico é capaz de encontrar todas as possíveis soluções do problema, foi desconsiderado o efeito do doping da população inicial. Caso contrário, o indivíduo propositalmente inserido interferiria no resultado, levando sempre à mesma configuração das juntas.

Foram realizadas simulações considerando os pontos cartesianos $x = 200mm$, $y = 350mm$ e $z = 750mm$. Os ângulos de orientação desejados Φ_d, Θ_d e Ψ_d como 30°, 45° e 60° respectivamente. Essas coordenadas foram inseridas também no ambiente virtual de simulação do RobotStudio, que cria um frame na posição desejada com a devida orientação. Dessa forma é possível comprovar que as variáveis de juntas determinadas pelo AG levaram o robô à pose desejada. A Fig. 8 mostra as diferentes configurações das juntas do robô, resultando na mesma pose do efetuador.

Os resultados, além de minimizarem o deslocamento das juntas para alcançar a pose final, foram precisos e coincidem com as 4 possíveis configurações (apresentadas na

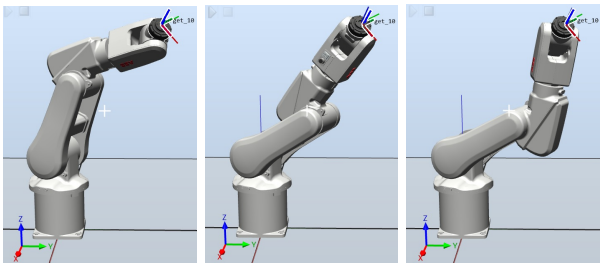

Figura 8. (a) Robô pra frente com cotovelo para cima (b) Robô para frente com cotovelo para baixo (c) Robô para trás com cotovelo pra baixo.

Fig. 9) para a cinemática inversa de um manipulador de 6 DoF com punho esférico (Siciliano et al., 2010). Em cinco simulações, o robô alcançou 3 das 4 configurações básicas. A quarta configuração, que seria robô para trás com o cotovelo para cima, está fora do espaço de trabalho.

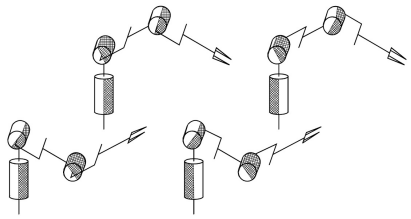

Figura 9. Possíveis soluções para a cinemática inversa de um manipulador com 6DoF e punho esférico (Siciliano et al., 2010).

Ainda que as poses tenham sido alcançadas com deslocamento angular minimizado, a ponderação da função objetivo conforme equação 9 acarreta num menor deslocamento das juntas de maior torque. Conforme mostrado na Fig 10(c), esse método leva a uma configuração final do robô voltada para frente ao partir da posição de *Home*, fazendo com que o deslocamento total do efetuador, indicado pelo traço vermelho, seja reduzido. Esse fato representa, além de uma melhor eficiência energética, um menor tempo para alcançar a pose final.

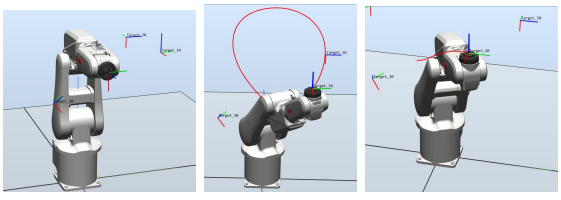

Figura 10. (a) Posição Inicial *Home* (b) [F.O] sem ponderação, robô voltado para trás (c) [F.O] com ponderação, robô voltado para frente.

7. CONCLUSÃO

Neste artigo foi investigada a utilização de algoritmos genéticos na solução da cinemática inversa de um manipulador robótico de 6DoF com punho esférico. Os resultados demonstram que os AGs podem apresentar soluções precisas ao problema. Foi verificado que não é necessário um

conhecimento prévio da atividade robotizada para aplicação da solução, uma vez que a população inicial é gerada de maneira aleatória. Ainda é possível afirmar que não são necessários muitos indivíduos na população inicial para que o algoritmo convirja para resultados ótimos, desde que o AG esteja configurado corretamente. Além disso, foi possível concluir que a precisão da pose final está diretamente relacionada ao número de gerações, portanto quanto menor o erro desejado, mais gerações são necessárias.

Em termos de convergência, os resultados demonstram que a melhor configuração do AG foi com seleção por roleta, *crossover* aritmético com taxa de 0,5 e mutação gaussiana. Mas não é possível afirmar que não existam configurações melhores; para isso é necessário desenvolver e investigar outras hipóteses com diferentes combinações. Outra constatação é que estratégias como a utilização de funções de elitismo e sobreposição de população também contribuem na conversão do resultado para valores ótimos globais. Mas a estratégia que obteve melhores resultados foi o doping da população inicial. Quando se conhece o problema e é possível restringir o espaço de trabalho do efetuador, alterar a população inicial com indivíduos pertencentes ao espaço que se deseja trabalhar é uma estratégia que garante a evolução para valores ótimos em taxas de tempo reduzidas.

Um dos principais benefícios do método apresentado é a facilidade de adaptação da solução, o que permite aplicá-la a diferentes processos robotizados, bastando adaptar a cinemática do robô e seus limites de juntas. Em especial, existem processos robotizados dentro da empresa de mineração Vale que podem ser aprimorados utilizando as técnicas apresentadas. A montagem robotizada de carro de grelha móvel no processo de pelotização pode ser estruturada conciliando técnicas de visão computacional e algoritmos genéticos para determinação das poses necessárias para realizar a montagem. A lavagem robotizada de equipamentos fora de estrada também pode ser otimizada utilizando a solução proposta para aumentar a flexibilidade durante a reprogramação da planta (Cota et al., 2017).

Entretanto ainda são necessários avanços para aumentar o grau de confiabilidade da solução. O próximo passo deste trabalho seria a resolução da cinemática inversa considerando a inserção de um trilho, equivalente ao sétimo eixo do robô. Outras estratégias de geração de trajetória e controle do braço manipulador também serão investigadas, evitando colisões e viabilizando a operação em ambientes com obstáculos.

REFERÊNCIAS

ABB, C. (2018). Rapid instructions, functions and data types. https://library.abb.com/pt. Acessado: 20 junho 2019.

ABB, R. (2019). Product specification: IRB120. https://new.abb.com/products/robotics/industrial-robots/irb-120/irb-120-data. Acessado: 11 maio 2019.

Cota, E., Torre, M.P., Ferreira, J.A.T., Fidêncio, A.X., Rodrigues, G.B., Rocha, F.A.S., Azpúrua, H., Freitas, G.M., and Miola, W. (2017). Robótica na Mineração. In *ABM Proceedings*. Editora Blucher.

da Silva, S.R.X., Schnitman, L., and Filho, V.C. (2019). Análise da eficiência computacional para solução do problema da cinemática inversa de robôs seriais utilizando a teoria de bases de gröbner. In *Anais do 14º Simpósio Brasileiro de Automação Inteligente*. Galoa.

Ferrentino, E., Cioppa, A.D., Marcelli, A., and Chiacchio, P. (2019). An evolutionary approach to time-optimal control of robotic manipulators. *Journal of Intelligent & Robotic Systems*, 99(2), 245–260.

Heinen, M.R. (2007). *Controle inteligente do caminhar de robôs móveis simulados*. Master's thesis, Universidade do Vale do Rio dos Sinos.

Holland, J.H. (1992). Genetic algorithms. *Scientific American*, 267(1), 66–73.

Köker, R. (2013). A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization. *Information Sciences*, 222, 528–543.

Köker, R., Öz, C., Çakar, T., and Ekiz, H. (2004). A study of neural network based inverse kinematics solution for a three-joint robot. *Robotics and Autonomous Systems*, 49(3-4), 227–234.

MathWorks (2020). Global optimization toolbox. URL `https://www.mathworks.com/help/gads/index.html?s_tid=CRUX_lftnav`. Acessado: 04 março 2020.

Momani, S., Abo-Hammou, Z.S., and Alsmad, O.M. (2016). Solution of inverse kinematics problem using genetic algorithms. *Applied Mathematics & Information Sciences*, 10(1), 225–233.

Murray, R.M., Li, Z., and Sastry, S.S. (1994). *A Mathematical Introduction to Robotic Manipulation*. CRC Press.

Nunes, L.E.N.d.P. (2007). *Geração e otimização de trajetórias de um manipulador robótico utilizando algoritmos genéticos*. Ph.D. thesis, Universidade Estadual Paulista.

Nunes, R.F. (2016). *Mapeamento da cinemática inversa de um manipulador robótico utilizando redes neurais artificiais configuradas em paralelo*. Master's thesis, Universidade Estadual Paulista.

Pavan, K.K., Murali, M.J., and Srikanth, D. (2018). Generalized solution for inverse kinematics problem of a robot using hybrid genetic algorithms. *International Journal of Engineering & Technology*, 7(4.6), 250.

Roth, Peter M., Vincze, Markus, Kubinger, Wilfried, Müller, Andreas, Blaschitz, Bernhard, and Stolc, Svorad (2017). Proceedings of the oagm arw joint workshop vision, automation and robotics.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2010). *Robotics: modelling, planning and control*. Advanced textbooks in control and signal processing. Springer London.

Spong, M.W., Hutchinson, S., and Vidyasagar, M. (2005). *Robot Modeling and Control*. JOHN WILEY & SONS, INC.

Števo, S., Sekaj, I., and Dekan, M. (2014). Optimization of robotic arm trajectory using genetic algorithm. *IFAC Proceedings Volumes*, 47(3), 1748–1753.

Tinós, R. (2007). Comportamento auto-organizável em algoritmos genéticos aplicados a robôs móveis em ambientes dinâmicos. *Sba: Controle & Automação Sociedade Brasileira de Automatica*, 18(1), 13–23.

Vargas, L.V. (2013). *Inversa Filtrada: Uma solução alternativa para a cinemática inversa de manipuladores robóticos*. Master's thesis, Universidade Federal do Rio de Janeiro.