Controle de Rota de um Sistema de Armazenamento Automático Utilizando Algoritmo Genético

Júlio César De Bona[1], Vitor Tumelero Valente[2]
TCC2 - Curso de Engenharia de Controle e Automação
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - Ca*pallets* Farroupilha
Farroupilha, Brasil
[1]julio_debona@hotmail.com, [2]vitor.valente@farroupilha.ifrs.edu.br

***Resumo*—Este trabalho corresponde à disciplina de TCC2 do curso de Engenharia de Controle e Automação do campus Farroupilha do IFRS. O documento descreve as características avaliadas para a construção de um sistema de armazenamento automático, desde seu projeto até a montagem e seu funcionamento. O sistema projetado é composto por um robô que retira pequenos paletes posicionados ao longo de uma estrutura bidimensional com a ajuda de um sistema de movimentação de dois eixos controlado por dois motores. O robô traça uma trajetória para retirar todos os *pallets* necessários de uma só vez. Para realizar o cálculo da trajetória de menor distância utilizou-se algoritmos genéticos.**

I. Introdução

O sistema de armazenamento automático (*ASRS* do inglês *Automatic Storage and Retrieval System*) é um conceito introduzido na indústria de armazenamento por volta de 1950, como é indicado por Abdelkrim et al. [1]. Desde então esta tecnologia vem sendo desenvolvida para uma melhor eficiência e eficácia do sistema, otimizando a relação custo-benefício do equipamento e melhorando seu tempo de resposta. Além das melhorias em seu projeto, estuda-se diferentes aplicações para o sistema, que integra uma interface, normalmente homem-máquina, um robô cartesiano e um robô responsável pela retirada de componentes.

Ao definir o funcionamento de sistemas automáticos de armazenamento, a *ASRS MHI* [2], determina que este sistema é definido como um método de armazenamento que utiliza um ou mais robôs, cuja movimentação é limitada por uma ou mais matrizes de posições, para executar tanto o armazenamento quanto a busca de produtos. Zollinger [3], adiciona que em um *ASRS*, a posição onde um determinado produto foi armazenado fica interligada com o código do produto, assim, quando um determinado produto é solicitado por seu código, o robô identifica seu posicionamento na estrutura do armazém.

Segundo Roodbergen e Vis [4], os sistemas de armazenamento automático podem ser utilizados em aplicações diversas, cada qual com sua configuração adequada. Os principais componentes que compõem o sistema normalmente são: prateleiras, guindastes, corredores e pontos de coleta/armazenamento de produtos.

As prateleiras são tipicamente feitas de metal onde serão posicionadas as cargas que precisam ser armazenadas. Os guindastes são as estruturas altamente automatizadas, capazes de se movimentar de forma autônoma para buscar ou devolver cargas. Os corredores são o espaço vazio entre as prateleiras, por onde os guindastes podem se mover. Os pontos de coleta e armazenamento de produtos são os locais onde as cargas são colocadas tanto para serem guardadas no *ASRS* como para serem retiradas do sistema. Normalmente este ponto determina a fronteira entre o espaço do *ASRS* e o resto da empresa.

Segundo Matson e White [5], Sistemas automáticos de armazenamento são largamente pesquisados em seus diversos modelos de sistemas e, por este fator, existe uma grande variedade de opções para um *ASRS*. Entre estas variedades encontra-se a versão básica, que é o caso de um guindaste para cada corredor. Este guindaste não pode passar para outros corredores e é limitado por carregar apenas uma carga por vez, possuindo assim uma prateleira com profundidade para apenas uma carga, tornando todas as cargas diretamente acessíveis pelo guindaste.

Uma variação do sistema básico é um sistema com guindastes que possuem a capacidade de mudar de corredor, reduzindo seu número frente ao número de corredores. Esta aplicação é utilizada para casos em que a quantidade de requisições de produtos estocados não justifica a compra de um guindaste por corredor. Outra variação do sistema básico é um sistema com guindastes capazes de transportar mais do que uma carga por vez, sendo assim ele pode retirar uma carga e posicionar outra no mesmo lugar sem precisar se movimentar até o ponto de coleta e armazenamento (I/O), ou ainda retirar mais de uma carga na mesma trajetória, explica Matson e White [5].

Sistemas automáticos de armazenamento são largamente utilizados em empresas de grande porte e são desejos de empresas de pequeno porte, a fim de otimizar a estocagem e retirada de matéria-prima e peças de estoque tanto no quesito tempo quanto na confiabilidade do controle de estoque. Sarker e Babu [6], explicam que um armazenamento de componentes, ferramentas, matéria-prima e subprodutos eficiente é necessário em sistemas de manufatura modernos como indústrias automatizadas, centros de distribuição ou armazéns.

De acordo com Tompkins et al. [7], caso haja uma alta rotatividade das cargas e a variação das mesmas seja relativamente baixa, pode-se utilizar prateleiras com profundidade superior a de apenas uma carga, poupando assim prateleiras e guindastes. A forma de retirada das cargas para prateleiras de profundidade superior a um produto pode variar, sendo que

o guindaste pode ser capaz de pegar as cargas independente da sua profundidade ou um sistema interno da prateleira pode direcionar a carga para a região mais externa ou interna da prateleira.

Guindastes capazes de carregar mais do que dois *pallets* podem ser aplicados em sistemas de manufatura que possuem uma grande vazão de peças em conjunto com um sistema de manufatura altamente flexível. Visto isso, estuda-se o caso da utilização de um *ASRS* capaz de retirar diversos componentes antes de se direcionar ao ponto de I/O.

Quando o *ASRS* recebe um comando para buscar diversos produtos, este deve traçar a rota a ser realizada de uma maneira eficiente para evitar o desperdício de energia. A rota traçada leva em consideração apenas a menor distância, o que resulta em um calculo de trajetória dependente de diversas variáveis, as quais são as coordenadas dos próprios pontos de retirada dos objetos desejados. Para definir qual a melhor trajetória pode-se calcular todas as possibilidades de combinação dos pontos de retirada e selecionar a menor.

Segundo Asokan et al. [8], a utilização de técnicas de otimização heurísticas como algoritmos genéticos, têmpera simulada e colônia de formigas tem atraído muitos pesquisadores pelo fato de estas técnicas poderem ter resultados muito próximos do resultado ótimo com um tempo de processamento computacional muito reduzido em comparação a métodos matemáticos de operações com solução exata.

Com o aumento da quantidade de produtos a serem retirados torna-se inviável o cálculo de distância para todas as combinações. Visto isso, este artigo tem como objetivo o desenvolvimento de uma solução para a definição da trajetória de um *ASRS* para casos onde há a retirada de diversos produtos através da utilização de um método heurístico para a escolha da trajetória visando a convergência para uma combinação próxima à ótima, além de contruir um protótipo para realização de testes.

O trabalho está organizado em algumas seções da seguinte forma: A seção II contém a revisão bibliográfica que destaca os principais conceitos envolvidos no tema do projeto. A seção III apresenta uma fundamentação teórica dos principais tópicos do projeto. A seção IV relata as definições e o desenvolvimento do projeto, explicando suas características e metodologias escolhidas. A seção V apresenta os resultados obtidos no trabalho e por fim, na seção VI conclui-se o trabalho com as devidas observações.

II. Revisão Bibliográfica

A utilização da tecnologia *ASRS* tem como foco grandes armazéns que utilizam *pallets* como produtos a serem estocados, porém a utilização desta tecnologia para outras aplicações com o intuito de melhorar a qualidade do trabalho vem aumentando gradativamente. Em paralelo, existe a preocupação em realizar serviços de qualidade para evitar retrabalho ou insatisfações por conta de clientes. A automação de um serviço manual também contribui para a redução de custos e muitas vezes para a agilidade de produção.

Em uma biblioteca tem-se diversos livros que devem ser armazenados em seus devidos locais para que seja possível realizar a fácil retirada quando necessário, porém este armazém de livros pode ser de proporções bastante grandes, igualmente ao seu número de usuários, o que faz com que uma grande quantidade de livros devam ser guardados em seus devidos locais diariamente. Com o intuito de facilitar este trabalho, Farooq et al. [9], desenvolveu um sistema automático de armazenamento de livros, o qual realiza o escaneamento da capa do livro a ser guardado e a partir de um banco de dados realiza a comparação dos caracteres encontrados na capa. O *ASRS* utiliza dois quadros, um responsável por movimentar os eixos X e Y e outro responsável por movimentar o eixo Z. Para a movimentação dos eixos utilizou-se de três motores DC. A posição do objeto nas esteiras do sistema foi monitorada a partir de sensores infravermelhos, porém não há realimentação de posição na estrutura do *ASRS* o que limitou o projeto no quesito custo de implementação, caso seja um projeto de grande porte, além que não utilizar o nível de automação que poderia ser utilizado.

Na falta de espaço para a utilização de *ASRS* convencionais, com diversos corredores no interior do sistema, os quais são ocupados por sistemas de guindaste, a solução indicada por Kosler et al. [10], é a utilização de um *ASRS* tridimensional compacto, que não utiliza corredores em seu projeto, uma vez que ocorre a rotatividade de toda coluna de *pallets* para a retirada e armazenamento dos mesmos. Os produtos rotacionam até que o item solicitado se encontre na posição de retirada, onde há um corredor com um guindaste apenas para a retirada de materiais. Este sistema pode ser aplicado para casos em que não se faz necessária uma alta taxa de retirada ou armazenamento de materiais, pois o sistema conta com apenas um guindaste de saída.

Os sistemas *ASRS* são de grande complexidade quando se trata de seu sistema operacional de armazenamento, considerando a variedade de produtos que devem ser armazenados com rotatividade na mesma posição. Uma vez que este sistema é de grande valor de implementação, necessita de grande planejamento para evitar problemas de comportamento durante sua operação. Pensando nisso, Brezovnik et al. [11], optou por realizar o desenvolvimento deste sistema através de um algoritmo de otimização baseado em inteligencia coletiva, inspirada no comportamento coletivo de insetos sociais e outras sociedades de animais, implementando-o para realizar o planejamento estrutural e o controle de processos do *ASRS*. Este algoritmo de otimização é capaz de economizar muito espaço através da utilização de um design não convencional, porém a avaliação de viabilidade econômica do sistema não foi realizada.

Neste mesmo contexto de algoritmos de aprendizado, Asokan et al. [8], desenvolveu um estudo de caso para a redução da distância percorrida por um *ASRS* no armazenamento de produtos utilizando dois métodos otimização e levando em consideração diversos parâmetros como validade, multa por atraso e tempo de produção. Os métodos utilizados foram algoritmos genéticos e enxame de partículas, gerando um

resultado bastante satisfatório com a convergência para a menor distância possível por ambos os métodos, porém na média de iterações o algoritmo de otimização por enxame de partículas resultou em uma distância menor do que o algoritmo genético, o que prova que a sua convergência é mais confiável para esta aplicação.

Em contrapartida, Bortfeldt [12], opta por utilizar algoritmos genéticos para resolver o problema do empacotamento ortogonal bidimensional depois de realizar um estudo comparativo com onze outros métodos heurísticos, comparando-os e concluindo que algoritmos genéticos possuem melhor performance, o que se deve ao fato de que outros métodos heurísticos não consideram a relação mutua das variáveis com tanta extensão.

III. Fundamentação Teórica

Nesta seção serão referenciados conceitos que se tratam da utilização de um algoritmo genético e das variadas possibilidades de construção de um *ASRS* em escala reduzida.

A. Algoritmos Genéticos

Como citado por Holland and Goldberg [13], os algoritmos genéticos são baseados na teoria da evolução das espécies de Darwin e nos estudos de genética, tendo como base a seleção dos indivíduos mais aptos a reprodução e sobrevivência. Cada indivíduo possui seu código genético, o qual é repassado parcialmente ou completamente para a próxima geração, caso o indivíduo se reproduza.

Segundo Pacheco et al. [14], estes princípios são copiados para a criação de algoritmo computacionais, os quais buscam o melhor resultado para um problema através de uma população de soluções com indivíduos que são comparados a cromossomos artificiais. Estes cromossomos representam as possíveis soluções do problema e ao longo das gerações são submetidos ao processo evolucionário que envolve avaliação, seleção, recombinação e mutação. A partir de algumas gerações, a população deverá conter indivíduos que satisfaçam o problema de melhor forma que as primeiras gerações.

Ainda definido por Pacheco et al. [14], os algoritmos genéticos podem ser caracterizados por uma série de definições, as quais foram utilizadas como base no desenvolvimento do mesmo para a aplicação em questão. As definições são:

1) Problema a Ser Otimizado
2) Representação das Soluções
3) Codificação do Cromossomo
4) Avaliação
5) Seleção
6) Operadores Genéticos
7) Inicialização da População
8) Parâmetros e Critério de Parada

Seguindo as definições encontradas na bibliografia, projeta-se o algoritmo a ser utilizado na ordem dos índices anteriormente definidos.

1) Problema a Ser Otimizado: O problema de otimização aplicado ao algoritmo genético no projeto em questão é definido como sendo a obtenção da menor distância a ser percorrida por um *ASRS* na retirada de diversos objetos partindo do ponto de I/O e retornando ao mesmo ponto no fim da retirada dos objetos. Sendo assim, calcula-se a trajetória pela equação da distância euclidiana 1.

$$\begin{aligned} Distancia = &\sqrt{(x_1^2 + y_1^2)} + \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ &+ ... + \sqrt{((x_n - x_{n-1})^2 + (y_n - y_{n-1})^2)} \end{aligned} \quad (1)$$

A melhor trajetória pode facilmente ser calculada testando todas as possibilidades com a ajuda de um software matemático com uma rotina básica de programação, uma vez que sabe-se que a quantidade de combinações possíveis é equivalente ao fatorial da quantidade de posições. Porém quando a quantidade de objetos se torna elevada, verifica-se que a quantidade de iterações necessária aumenta exponencialmente como demonstrado no gráfico da Fig. 1, inviabilizando o cálculo de todas as possibilidades.

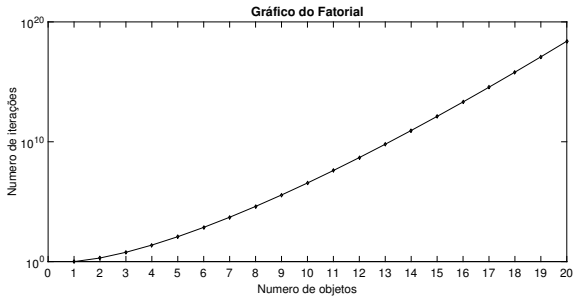

Fig. 1. Gráfico de relação quantidade de objetos por quantidade de iterações

A partir do gráfico pode-se observar que acima de um determinado valor de objetos, a quantidade de iterações atinge valores astronômicos, obrigando assim a avaliar uma forma alternativa à de calcular todas as possibilidades para a definição de melhor trajetória. Para isso será realizado o cálculo heurístico em questão, que deve apresentar uma convergência para o valor ótimo de distância a partir da emulação da evolução das espécies.

2) Representação das Soluções: Neste tópico define-se como as soluções serão representadas para efeito de cálculo, podendo definir a partir disto se o resultado é bom ou ruim para a aplicação. Para o problema a ser resolvido, as soluções são compostas por um conjunto de posições compostas por coordenadas ”X”e ”Y”, que definem o posicionamento do objeto.

Sabendo o tipo de variáveis presentes na solução pode-se definir o indivíduo, ou cromossomo, como sendo uma variável que contém um vetor de posições e cada posição possui um vetor de coordenadas. As posições deverão sofrer alterações de ordenação ao longo das gerações, porém nunca devem se repetir em um cromossomo, assim como as coordenadas nunca devem ser alteradas para sua determinada posição.

3) Codificação do Cromossomo: A codificação do cromossomo se trata da transformação de um indivíduo para uma solução avaliável que possa ser comparada com as soluções de outros indivíduos. Para a definição da solução da trajetória do robô aplica-se a equação 1 que possui como resposta um valor real facilmente comparável com os outros valores. Este valor de distância é atrelado à mesma variável que possui as posições dos objetos, fazendo com que cada indivíduo possua seu valor de distância.

4) Avaliação: A avaliação é realizada a partir do resultado que cada indivíduo da geração fornece, sendo que para este caso o resultado se trata da distância da trajetória correspondente ao indivíduo. A avaliação permite classificar o indivíduo como sendo um indivíduo de bom ou mal resultado dentro da gama de resultados obtidos na geração. Para separar estes indivíduos, organiza-se o vetor de indivíduos de cada geração colocando os cromossomos de melhor valor, que possuem a menor distância, nas primeiras posições do vetor e os de pior valor nas últimas, gerando assim um vetor de qualidade de indivíduos decrescente.

5) Seleção: A seleção é subdividida em alguns tópicos, sendo que cada um deles define um tipo de seleção. Um desses tópicos pode ser chamado de seleção de acasalamento, ou recombinação, e possui o intuito de gerar descendentes a partir de uma combinação de dois indivíduos aleatórios ou de todos os indivíduos a fim de dar a todos a chance igual de acasalar, como explica Justesen [15].

Outro tópico apresentado por Justesen [15], é o de seleção ambiental, ou elitismo, aplicada após a avaliação de todos os indivíduos da geração, determinada pela famosa regra da sobrevivência dos mais fortes ou aptos, fazendo com que apenas uma pequena parcela de cromossomos da geração atual sobreviva para passar a próxima geração.

Por último, o tópico de mutação descrito por Deb [16], define que os indivíduos devem sofrer uma mutação com uma probabilidade definida de modo que pelo menos um indivíduo sofra mutação por geração.

6) Operadores Genéticos: Segundo Dreier [17], os operadores genéticos são baseados por um lado em métodos de ascensão heurística como a recombinação e o elitismo mas por outro lado em um método de exploração randômico chamado de mutação.

A recombinação se trata da manipulação das variáveis de dois indivíduos, chamados de pais, de certa forma que estas variáveis criem um novo indivíduo, que será pertencente à próxima geração. Este novo indivíduo, chamado de filho, deve ser aplicável ao sistema e gerar um resultado para ser avaliado.

O problema em questão é comparado ao problema do caixeiro viajante, onde um homem deve passar por diversas cidades para entregar correspondências, porém ele deve saber qual a menor distância a ser percorrida. Com isso, Silvanandam e Deepa [18], define que o método de recombinação PMX (Partially Mapped Crossover) pode ser aplicado a este caso da forma que todos os destinos dos pais devem ser encontrados nos filhos, porém em ordem diferente.

Para executar esta operação, como explica Jawahar et al. [19], primeiramente são selecionados dois pais, e em seguida dois pontos de cruzamento do vetor de posições são definidos aleatoriamente. A partir disto, o segundo passo é criar dois filhos que recebem os valores de posição que estão no intervalo de cruzamento dos pais, preservando a ordem, porém os valores sofrem uma troca da seguinte forma: O filho 1 recebe as posições do pai 2, e o filho 2 recebe as posições do pai 1. A Fig. 2 representa de forma ilustrativa o processo.

No terceiro passo se faz necessário completar os índices dos vetores de posição dos filhos, os quais não possuem valores de posição, com as posições que não estão dentro dos intervalos de cruzamento de nenhum dos pais. No exemplo ilustrado, verifica-se que as posições 'B', 'C' e 'I' se aplicam ao requerido, e com isto, estas posições são levadas aos filhos na mesma posição dos respectivos pais.

Ao passo 4, para completar os vetores de posição dos filhos, verifica-se onde estão as posições que se encontram dentro do intervalo de cruzamento de um pai, porém não se encontram entre o intervalo de cruzamento do outro. A partir do índice da posição do pai que não se encontra no intervalo definido, realiza-se a verificação do mesmo índice do filho inverso. Caso o índice ainda esteja livre, a posição é repassada a ele. Caso contrário, é repassada para o primeiro índice livre.

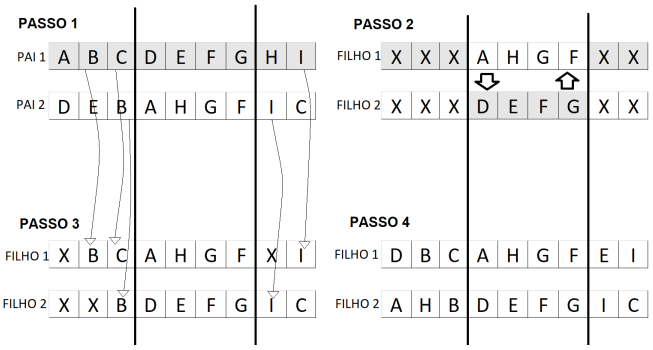

Fig. 2. Aplicação do algoritmo de recombinação PMX

Ao fim dos 4 passos, dá-se por encerrado o procedimento de recombinação de variáveis, gerando dois filhos que unem características de dois pais diferentes. Após a definição deste procedimento, explica-se o método da mutação genética.

De acordo com Dornberger et al. [20], a mutação pode ser realizada alterando aleatoriamente o local de busca de um item partindo do pressuposto que são conhecidas mais de uma posição para o mesmo item e que a nova posição escolhida randomicamente possui o mesmo item.

O conceito de mutação para Jian et al. [21], é diferente e aplicável ao projeto desenvolvido, onde define-se aleatoriamente duas posições do vetor de posições do pai e estas posições são trocadas ao serem repassadas para o filho, como segue a Fig. 3.

O último operador genético a ser demonstrado é definido como seleção, ou elitismo, e define-se após ordenar todos os resultados da geração em ordem decrescente (do melhor resultado para o pior) e então passar o primeiro indivíduo,

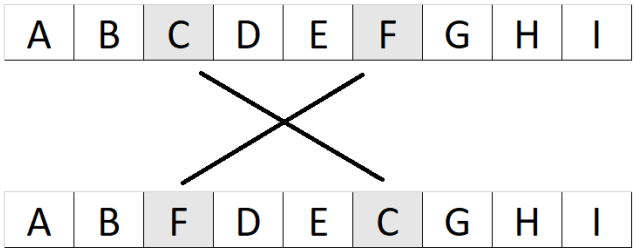

Fig. 3. Aplicação do algoritmo de mutação

correspondente ao melhor resultado da geração, diretamente para a próxima geração, como explica Liu et al. [22]. Este operador garante que não haverá a depreciação do resultado ao longo das gerações.

7) Inicialização da População: A população deve ser criada a partir de indivíduos aleatórios, os quais pertencerão a primeira geração. Como forma alternativa de dar início à população, Pacheco [14], acrescenta que esta população pode ser criada a partir de "sementes", correspondentes à resultados parciais que podem ser retirados de outras execuções do algoritmo com parâmetros em comum. Esta inserção de dados prévios pode acelerar a convergência e providenciar um resultado mais próximo do esperado.

8) Parâmetros e Critério de Parada: Alguns parâmetros devem estar explícitos ao iniciar o projeto de um algoritmo genético. Os parâmetros bem definidos são essenciais para o bom funcionamento do método heurístico em questão. Estes métodos são definidos por Pacheco [14]:

1) Tamanho da população - Corresponde ao número de indivíduos que são criados a cada iteração;
2) Taxa de crossover ou recombinação - Probabilidade de ocorrer a recombinação entre dois indivíduos;
3) Taxa de mutação - Probabilidade de ocorrer a mutação de um indivíduo;
4) Número de gerações - Quantidade de ciclos de evolução do algoritmo;
5) Total de indivíduos - Quantidade total de tentativas realizadas (tamanho da população x número de gerações).

É definido como critério de parada o que representa o fim do algoritmo genético, que pode ser representado por qualquer um dos dois últimos parâmetros descritos. Este algoritmo representado nada mais é do que um ciclo de evoluções que é reproduzido até alcançar o seu critério de parada.

B. Componentes Robóticos

Com o intuito de realizar testes com resultados obtidos no algoritmo genético em uma estrutura física realizou-se o projeto de um *ASRS* em escala reduzida, com um pequeno número de posições. Para o projeto desta estrutura e de seu funcionamento, buscou-se os melhores componentes para a aplicação dentro dos valores de custo disponíveis.

1) Controlador: Os microcontroladores se subdividem em diversos modelos. Alguns modelos são largamente utilizados devido ao fato de serem de fácil implementação e/ou possuir uma linguagem de programação simplificada, podendo ser citado neste caso o microcontrolador utilizado na plataforma de prototipagem *Arduino Mega*, o qual é fabricado pela empresa *Atmel* e possui uma arquitetura de 8 bits, o que significa que o seu poder de processamento está entre os menores dos microcontroladores comercializados na atualidade. Há outros modelos de microcontroladores que possuem melhor processamento, com arquitetura de 32 bits (conhecida por arquitetura *ARM*), que são muito utilizados no mercado para diversas aplicações embarcadas na área de eletrônica e robótica e ainda possuem um custo reduzido.

2) Motores: Dentre os motores que podem ser aplicados a um *ASRS* de escala reduzida encontra-se o motor de passo, o qual é rotacionado por uma série de bobinas eletromagnéticas chaveadas eletronicamente. O número de polos magnéticos que um motor de passo possui define a sua quantidade de passos por revolução, o que o torna um motor de precisão pelo fato de ser acionado passo por passo, porém sua operação ainda é um malha aberta pelo fato de não haver realimentação da posição.

Outro motor que poderia suprir as necessidades da aplicação é o motor DC com um *encoder* acoplado ao seu eixo. Este motor é capaz de proporcionar precisão no movimento devido ao controle realizado pela contagem dos pontos do *encoder*, que varia sua precisão em função da quantidade de pontos por revolução. O motor DC é de fácil acionamento, necessitando de apenas uma ponte H, que é capaz de inverter o sentido da rotação do motor a partir de um sinal enviado pelo controlador. Este tipo de motor normalmente opera em alta velocidade de rotação, necessitando de um sistema de redução de velocidade e consequentemente aumentando seu torque, como pode ser visto em (2).

$$\frac{Velocidade_{entrada}}{Torque_{entrada}} = \frac{Velocidade_{saida}}{Torque_{saida}} \tag{2}$$

3) Mecanismo de Movimentação: Todos os sistemas que envolvem a utilização de motores necessitam de um acoplamento que transfira a energia cinética proveniente do motor para o que se deseja movimentar. Com base nisso deve-se optar por um mecanismo de movimentação que ao ser acoplado ao motor possa realizar o esforço solicitado sem danificar a estrutura ou o próprio motor.

Um mecanismo muito utilizado em robôs cartesianos é o mecanismo de polia e correia, no qual a correia é associada ao eixo que se movimenta nas duas extremidades. Com isto, ao mesmo tempo que a polia é rotacionada pelo motor, a correia é movimentada levando consigo o eixo ao qual esta está presa.

Como outro mecanismo de movimentação, tem-se o sistema de parafuso fixo, o qual é acoplado a um motor elétrico. A movimentação ocorre por meio de uma porca que é fixada ao sistema e se movimenta ao longo do parafuso. A movimentação deste mecanismo varia com o passo do parafuso, que é a distância entre cada filete da rosca, sendo que a relação de movimentação é o passo do parafuso para cada rotação do motor. Este sistema de movimentação sofre perdas de atrito, assim como qualquer outro tipo de transmissão, como

demonstra Cunha et al. [23], sendo neste caso para cada ponto de contato da rosca com o parafuso, por isso é aconselhável mantê-lo bem lubrificado com a intensão de diminuir este atrito.

IV. Projeto

Com o projeto fundamentado pode-se descrever como foram realizados os procedimentos para a construção e o desenvolvimento do sistema em um todo, desde o robô em escala reduzida até o algoritmo heurístico produzido.

A. Projeto Mecânico

O sistema mecânico projetado possui como base sistemas utilizados em meio industrial, porém com escala reduzida, realizando a retirada e armazenamento de pequenos *pallets*. Seguindo o projeto realizado por Farooq et al. [9], a estrutura deve consistir em dois sistemas de movimentação, sendo que um sistema controla as dimensões XY responsáveis por movimentar o robô pelo corredor, chamado de movimentador, enquanto o outro é responsável por realizar a movimentação na direção dos *pallets*, chamado de auxiliar.

A estrutura do sistema movimentador consiste em um quadro responsável por realizar a movimentação nos dois eixos. O sistema é operado por dois motores acoplados a polias, as quais movimentam as suas respectivas correias.

Um motor é responsável pela movimentação de subida e descida do robô e se localiza fixo na parte superior da estrutura, enquanto o outro é responsável pela movimentação ao longo do comprimento da estrutura e se movimenta juntamente com a guia no sentido da altura, sendo que para ambos os casos a correia é apoiada por um rolete na lateral inversa. A estrutura básica pode ser visualizada na Fig. 4.

O sistema movimentador deve ser capaz de alcançar 5 posições para *pallets* em seu comprimento e 5 posições para *pallets* em sua altura. Considerando que cada posição possui aproximadamente 102 mm de largura e altura, a estrutura projetada deve alcançar 536 mm em cada dimensão, tendo em vista que a instalação dos componentes elétricos e seus suportes devem ser considerados no tamanho da estrutura, as dimensões totais da mesma são de aproximadamente 800 mm de comprimento e altura. A largura da região que suporta os *pallets* é de 100 mm.

O sistema de transferência da energia cinética proveniente do motor para a movimentação do robô, realizado através de um conjunto de polias e correias, é acoplado aos motores e *encoders*. Entre o motor e o *encoder* que se mantém fixos, na parte superior do projeto, exite uma polia acoplada diretamente ao motor e a um eixo árvore. Na outra extremidade do eixo árvore é acoplada outra polia, que por sua vez é acoplada ao *encoder*. O eixo árvore existe para que o sistema realize o mesmo esforço em ambos os lados dos eixos horizontais, os quais são movimentados pelo conjunto de correias em questão, evitando desta forma a torção das guias.

Na Fig. 5 é apresentado o sistema auxiliar apoiado por dois eixos guia os quais são acoplados a outras duas guias lineares através de apoios em ambas as extremidades. O sistema

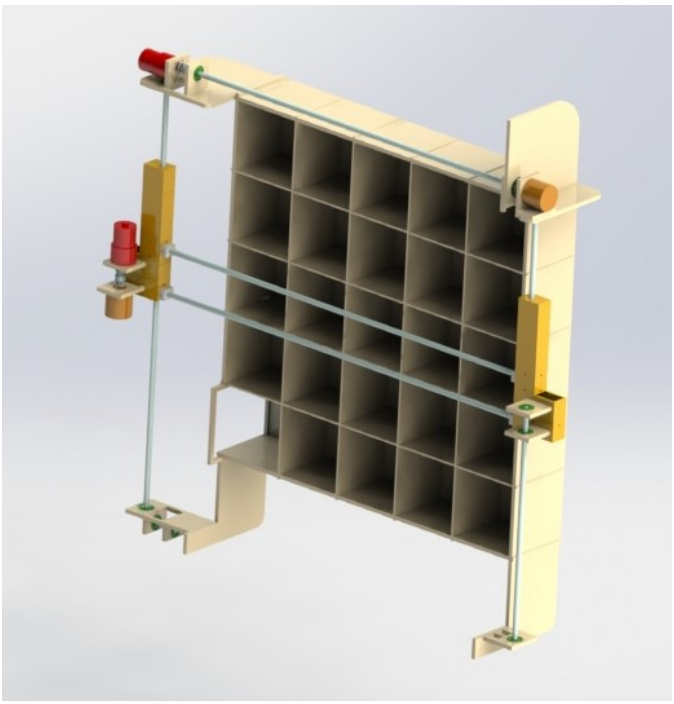

Fig. 4. Projeto mecânico do sistema movimentador

auxiliar é composto por um mecanismo de biela manivela acionado por um servo motor, o qual é responsável por realizar a retirada dos *pallets*.

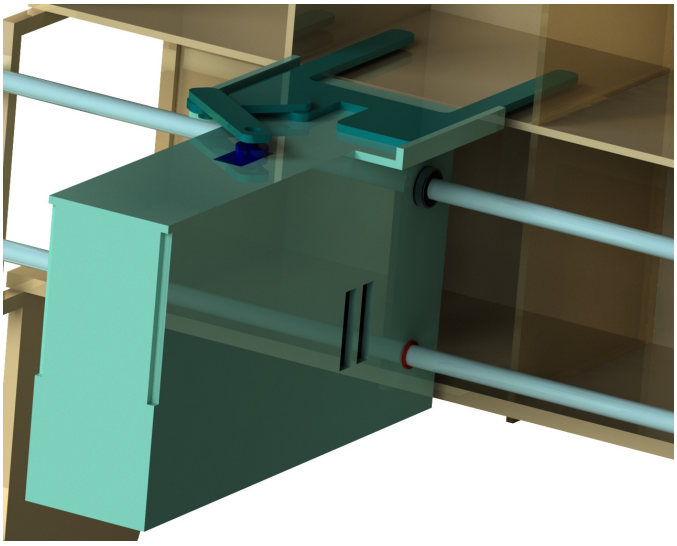

Fig. 5. Projeto do sistema auxiliar

Pode-se observar na Fig. 5 dois retângulos vazados com um suporte no centro o qual representa a posição onde as correias são presas. O mesmo rasgo se repete na outra lateral do sistema auxiliar permitindo que a correia seja presa no mesmo, passe pela polia acoplada ao motor de um lado e pelo rolete no outro lado, atravessando assim o sistema auxiliar.

O objetivo da implementação do sistema auxiliar é retirar

o(s) *pallet(s)* e carrega-lo(s) até o ponto de I/O, os quais possuem dimensões determinadas de 80mm por 10mm, já o suporte responsável por pega-los possui 100mm de alcance, fazendo com que este componente seja capaz de carregar até dez *pallets* por vez.

O sistema movimentador opera com ajuda de um conjunto de rolamentos lineares e buchas deslizantes autolubrificantes. Componentes os quais são responsáveis por realizar a transferência da movimentação tanto das guias lineares verticais para as guias lineares horizontais como das guias lineares horizontais para o sistema auxiliar.

Segundo Tompkins et al. [7], os sistemas de armazenamento automático podem se mover a uma velocidade máxima aproximadamente seis vezes menor no sentido vertical quando comparado com a velocidade do sentido horizontal, porém este calculo é realizado para a operação de produtos pesados. Para a aplicação em questão estes valores devem ser testados com o sistema já implementado.

B. Projeto Elétrico

O *ASRS* necessita de um sistema de acionamento para os motores, levando em consideração que o controlador é um dispositivo projetado apenas para sinais, jamais para potência. Este sistema de acionamento então precisa ser capaz de transformar um pequeno sinal de 5 V e corrente na ordem de alguns mA em 12 V e até 3 A para alimentar o motor.

O circuito a ser utilizado como *driver* de potência é conhecido como ponte H e é projetado utilizando transistores de potência. Para a aplicação, utilizou-se um *drive* comercializado, conhecido como L298. O componente pode acionar cargas com tensão de 5 até 50 V e corrente de até 2 A, segundo a empresa STMicroeletronics, [24].

Os motores precisam ser controlados de uma forma que seja possível descobrir qual a sua posição e a quantidade de voltas completas e assim saber qual a distância percorrida pelo sistema correspondente, de modo a permitir que o sistema robotizado complete a movimentação determinada e chegue ao local correto.

O motor a ser utilizado no projeto se trata de um motor DC, o qual opera em uma tensão de 12 V e uma corrente de até 600 mA. Devido ao sistema de redução mecânica, o motor atinge uma velocidade máxima com carga de 150 rpm. A velocidade do motor não foi projetada para obter-se o melhor rendimento possível, pois este não era o foco do projeto.

O motor DC possui um *encoder* acoplado ao seu eixo. Ao contrário de um motor de passo, por exemplo, que pode perder passos sem que isso seja identificado pelo sistema de controle, este é um sistema controlado em malha fechada, o que oferece maior confiabilidade para o sistema.

O *encoder* é responsável por monitorar a quantidade de pulsos passados e é conectado ao controlador para que este realize o processamento deste sinal com o intuito de relacionar o deslocamento do sistema com a quantidade de pulsos contados.

O controlador utilizado deve ser capaz de realizar a leitura de pelo menos dois *encoders* ao mesmo tempo, além de realizar o condicionamento dos seus valores e o controle dos motores. Isso obriga o microcontrolador utilizado a ter uma taxa de processamento alta, devido ao fato de os *encoders* exigirem um monitoramento veloz. Por isto optou-se pela utilização do controlador com maior capacidade de processamento.

O controlador utilizado se trata de um módulo STM32 F103, com um microprocessador de arquitetura 32-bit do tipo ARM Cortex-M3, clock de 72 MHz, 20 kb de memória RAM e 64 kb de memória flash. O microprocessador faz parte do módulo que possui pinos de acesso às saídas e entradas digitais e analógicas que facilitam a sua utilização.

Para o referenciamento do sistema utilizou-se dois fins de curso nos extremos próximos ao ponto de I/O, permitindo assim que para cada ciclo o sistema seja referenciado em ambos os eixos, evitando a acumulação de erros ao longo da execução.

C. Projeto do Algoritmo Genético

O algoritmo genético projetado deve ser capaz de identificar uma trajetória que se aproxime de forma acentuada da menor distância possível através de diversos indivíduos buscando melhoria a cada geração.

As variáveis são definidas como sendo um vetor de posições, e estas posições são definidas em coordenadas ”X”e ”Y”correspondentes ao endereço dos produtos. Na Fig. 6 pode-se visualizar um exemplo de definição de variáveis para uma trajetória com seis posições.

Posições	5		1		4		6		2		0		3	
Coordenadas	3	4	2	0	1	4	0	4	4	4	2	3	3	1

Fig. 6. Exemplo de definição de variáveis

Na programação desenvolvida em linguagem C, a qual pode ser compilada e executada pelo microcontrolador utilizado, definiu-se as posições através de um vetor de *structs*. Estas *structs* possuem um vetor de coordenadas de duas posições, além da própria variável de posição.

Sabendo como serão definidas as variáveis, definiu-se que as populações de indivíduos serão definidas pelo dobro da quantidade de posições existentes na trajetória. Por exemplo, se a trajetória for composta por 6 destinos, cada geração será composta de 12 indivíduos.

Os indivíduos precisam ser definidos de forma que a ordem das suas posições varie para cada caso. Com isto, define-se cada indivíduo da primeira geração através de uma função randômica da linguagem C, garantindo que a sequência de posições dos indivíduos sejam diferentes.

Após a definição de todos os indivíduos da primeira geração, calcula-se a distância da trajetória, partindo do ponto de I/O e retornando para o mesmo. O cálculo é realizado através da distância entre pontos.

Com as distâncias calculadas, ordena-se os indivíduos de mais atrativo para menos atrativo, ou seja, em ordem crescente de distância, deixando a menor no topo, representando o melhor resultado. Este melhor resultado deverá obrigatoriamente

ser passado diretamente para a próxima geração, representando o elitismo.

A próxima geração é definida por indivíduos provenientes da geração anterior. Estes indivíduos podem sofrer mutação ou recombinação com fatores variáveis, e sempre deve ocorrer o elitismo de ao menos um componente do grupo.

Os índices de recombinação e mutação devem ser determinados pelo usuário para que o software possa calcular da forma mais próxima da desejada os possíveis coeficientes de cada fator, tendo que garantir que haverá um número par de indivíduos para realizar a recombinação, e os indivíduos que não forem definidos por elitismo e nem por recombinação, deverão então ser definidos pela mutação.

Em se tratando da quantidade de gerações a ser utilizada é definido que a equação que rege esta variável deve levar em consideração a quantidade de posições que devem compor a trajetória, sendo demonstrada em (3). Esta equação foi obtida de forma empírica.

$$NúmeroGerações = \frac{NúmeroPosições^2}{2} \qquad (3)$$

O software desenvolvido deve produzir diversas gerações a fim de garantir que o sistema irá operar adequadamente, isto é, irá se aproximar do mínimo valor de distância possível. Os processos de cálculo de distância dos indivíduos e organização por ordem crescente desta distância são realizados a cada geração, sempre garantindo a aprovação do melhor resultado à próxima geração.

Durante a execução de alguma tarefa do sistema, seja ela de armazenamento ou de retirada de produto, o mesmo se mantém ocupado, não permitindo que o software realize outro pedido de execução. Ao encerrar a tarefa solicitada, a interface volta a operar normalmente, aguardando uma nova solicitação.

A comunicação entre o microcontrolador e o usuário é realizada através de uma porta serial, utilizando o hyperterminal como comunicador. Utiliza-se este método para simular um controle realizado por outro equipamento, podendo ser um CLP, outro controlador ligado a uma IHM ou até mesmo um teclado matricial.

Para fins de testes, realizou-se a criação de quatro sequências de quatro, seis, dez e dezoito posições que são utilizadas de forma que a ordem é recebida pelo algoritmo genético de forma randômica. Estes dados pré estabelecidos foram utilizados para facilitar os testes e para que não seja necessário informar todas as coordenadas para cada teste a ser realizado.

Criou-se um modelo de fluxograma simplificado para o melhor entendimento da ordem dos processos ao longo da operação do protótipo, podendo então visualizar o esquemático a partir da Fig. 7.

Os sinais que variam de um a quatro correspondem a quais posições pré estabelecidas o usuário selecionou (quatro, seis, dez ou dezoito posições). Considerando que caso o usuário envie um sinal diferente das opções definidas, o software irá ignorar o sinal.

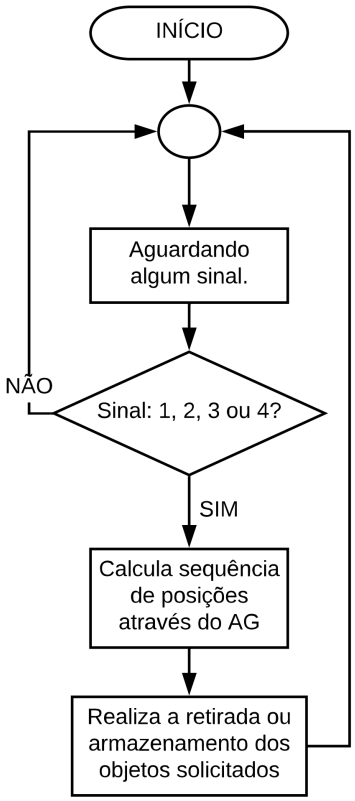

Fig. 7. Modelo de fluxograma do processo

V. RESULTADOS

Esta seção avalia o rendimento do sistema projetado, realizando o estudo de desempenho do algoritmo genético para diferentes casos e aplicações. Comparou-se a aplicação do algoritmo desenvolvido com a realização de métodos matemáticos para a obtenção do resultado, podendo ser visualizado a partir da Tabela I.

Tabela I
ESTUDO DE DESEMPENHO DO AG

PARÂMETROS AG				
Qntd. Posições	4	6	10	18
Qntd. Gerações	8	18	50	162
RESULTADOS				
Média 1k Execuções	12,67	15,83	21,15	28,19
Melhor Valor AG	12,42	15,05	18,89	20,65
Pior Valor AG	13,06	16,37	23,42	35,08
Melhor Valor ABS	12,42	15,05	18,89	19,41

Para todos os testes determinou-se que seriam utilizados fatores de recombinação, mutação e elitismo de 60%, 30% e 10%, respectivamente, tendo estes valores como base, porém com ajustes realizados pelo software para satisfazer uma quantidade possível dentro dos limites operacionais do algoritmo.

Através dos resultados obtidos, pode-se visualizar que a média ao longo de mil resultados é satisfatório pois se aproxima do melhor valor absoluto, demonstrando que o sistema

converge para o resultado esperado dentro da quantidade de gerações estipuladas.

Tendo feito o estudo de desempenho do algoritmo, realiza-se o estudo de viabilidade do mesmo em comparação com um ciclo matemático que testa todas as possibilidades. Na forma matemática, o melhor resultado possível é garantido, em quanto no meio desenvolvido, obtém-se um valor aproximado do melhor possível. Visualiza-se os resultados deste estudo por meio da Tabela II

Tabela II
ESTUDO DE VIABILIDADE DO AG

PARÂMETROS AG				
Qntd. Posições	4	6	10	18
Qntd. Gerações	8	18	50	162
RESULTADOS				
Qntd. Iterações AG	64	216	1.000	5.832
Qntd. Iterações Mat.	24	720	$3,6e6$	$6,4e15$
Tempo Ex. AG[seg]	$1,6e{-}4$	$5,6e{-}4$	$2,6e{-}3$	$1,5e{-}2$
Tempo Ex. Mat.[seg]	$6,2e{-}5$	$1,9e{-}3$	$9,3$	$1,6e10$

Com os resultados de quantidade de iterações obtidos verifica-se de forma rápida que para a possibilidade que passa por quatro posições o sistema não é viável, pois a quantidade de cálculos necessária para chegar ao resultado utilizando algoritmos genéticos é maior do que para calcular todas as possibilidades.

Para o caso de seis posições o resultado ainda não é muito expressivo, porém a quantidade de cálculos para o algoritmo gerado já é menor do que a quantidade de cálculos que seria realizada para todos os casos, de forma matemática.

Para os casos de dez e dezoito posições, a vantagem da utilização do algoritmo genético fica clara, onde uma quantidade expressiva de cálculos é necessária para a resolução do sistema de forma matemática.

Realizou-se um estudo do tempo necessário para a execução do algoritmo genético e comparou-se esta resposta a uma estimativa do tempo para o cálculo de forma matemática. Esta estimativa foi realizada pela comparação da quantidade de iterações proporcionalmente.

Pode-se visualizar que para os casos de quatro e seis posições, ambos os métodos demoram menos de um segundo para realizar a execução. No caso de dez posições, o método matemático demora quase dez segundos, enquanto o algoritmo apenas alguns mili segundos. Por último, no caso de dezoito posições, o algoritmo genético ainda leva apenas quinze mili segundos, enquanto o método matemático leva um tempo astronômico que pode ser calculado como 522 anos.

VI. CONCLUSÃO

Pode-se perceber que os resultados obtidos foram bastante satisfatórios para casos em que a quantidade de posições a serem acessadas ultrapasse o valor de cinco posições. Para casos com um valor inferior a cinco, o algoritmo desenvolvido se mostra ineficiente, porém, ainda eficaz.

O meio de desenvolvimento do algoritmo genético se mostrou capaz de satisfazer a necessidade, podendo calcular uma resposta aproximada dentro dos limites estipulados, sendo que existe uma infinidade de formas de criação deste algoritmo.

O microcontrolador escolhido se mostrou capaz de desenvolver os cálculos necessários dentro do período necessário para que a leitura dos *encoders* ou o controle dos motores não sejam prejudicados.

Um método de controle para *ASRSs* que operam com suporte para vários produtos foi proposto e se mostrou capaz de realizar a operação podendo gerar diminuição nos custos de energia além de agilizar o processo de armazenamento ou retirada de produtos onde a demanda é elevada.

Sistemas de armazenamento automático estão mais comuns no setor da indústria a cada dia que passa, o que se deve claramente ao crescimento da automação industrial que é uma tendência mundial. Com o aumento da utilização destes dispositivos novas demandas são criadas e diferentes formas de controle também. Este projeto desenvolveu uma solução pertinente que pode ser aplicada a diversos nichos da indústria.

REFERÊNCIAS

[1] B. Abdelkrim, S. Zaki, and G. Noureddine, "Performance analysis for multi-aisle automated storage/retrieval systems using visual petri net developer," in *Computational Intelligence in Robotics and Automation, 2003. Proceedings. 2003 IEEE International Symposium on*, vol. 3. IEEE, 2003, pp. 1475–1481.

[2] (2017, Oct.) As/rs groupcharts2004. [Online]. Available: http://www.mhi.org/as-rs

[3] H. A. Zollinger, "Planning, evaluating and estimating storage systems," in *Proceedings of the seminar on advanced material handling, Purdue University, West Lafayette, IN*, 1975, pp. 29–30.

[4] K. J. Roodbergen and I. F. Vis, "A survey of literature on automated storage and retrieval systems," *European journal of operational research*, vol. 194, no. 2, pp. 343–362, 2009.

[5] J. O. Matson and J. A. White, "Operational research and material handling," *European Journal of Operational Research*, vol. 11, no. 4, pp. 309–318, 1982.

[6] B. R. Sarker and P. S. Babu, "Travel time models in automated storage/retrieval systems: A critical review," *International Journal of Production Economics*, vol. 40, no. 2-3, pp. 173–184, 1995.

[7] J. A. Tompkins, J. A. White, Y. A. Bozer, and J. M. A. Tanchoco, *Facilities planning*. John Wiley & Sons, 2010.

[8] P. Asokan, J. Jerald, S. Arunachalam, and T. Page, "Application of adaptive genetic algorithm and particle swarm optimisation in scheduling of jobs and as/rs in fms," *International Journal of Manufacturing Research*, vol. 3, no. 4, pp. 393–405, 2008.

[9] U. Farooq, H. Khan, M. U. Asad, A. Mateen, A. Iqbal, and H. Mahmood, "Design and development of an image processing based automatic library storage system," in *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, vol. 4. IEEE, 2010, pp. 292–297.

[10] R. B. De Koster, T. Le-Duc, and Y. Yugang, "Optimal storage rack design for a 3-dimensional compact as/rs," *International journal of production research*, vol. 46, no. 6, pp. 1495–1514, 2008.

[11] S. Brezovnik, J. Gotlih, J. Balič, K. Gotlih, and M. Brezočnik, "Optimization of an automated storage and retrieval systems by swarm intelligence," *Procedia Engineering*, vol. 100, pp. 1309–1318, 2015.

[12] A. Bortfeldt, "A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces," *European Journal of Operational Research*, vol. 172, no. 3, pp. 814–837, 2006.

[13] J. Holland and D. Goldberg, "Genetic algorithms in search, optimization and machine learning," *Massachusetts: Addison-Wesley*, 1989.

[14] M. A. C. Pacheco *et al.*, "Algoritmos genéticos: princípios e aplicações," *ICA: Laboratório de Inteligência Computacional Aplicada. Departamento de Engenharia Elétrica. Pontifícia Universidade Católica do Rio de Janeiro. Fonte desconhecida*, p. 28, 1999.

[15] P. D. Justesen, "Multi-objective optimization using evolutionary algorithms," *University of Aarhus, Department of Computer Science, Denmark*, 2009.

[16] K. Deb, "Multi-objective optimization using evolutionary algorithms: an introduction," *Multi-objective evolutionary optimisation for product design and manufacturing*, pp. 1–24, 2011.
[17] F. Dreier, "Genetic algorithm tutorial," 2002.
[18] S. Sivanandam and S. Deepa, *Introduction to genetic algorithms*. Springer Science & Business Media, 2007.
[19] N. Jawahar, P. Aravindan, and S. Ponnambalam, "Optimal random storage allocation for an as/rs in an fms," *The International Journal of Advanced Manufacturing Technology*, vol. 14, no. 2, pp. 116–132, 1998.
[20] R. Dornberger, T. Hanne, R. Ryter, and M. Stauffer, "Optimization of the picking sequence of an automated storage and retrieval system (as/rs)," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 2817–2824.
[21] L. Jian, W. Xin, W. Weize, L. Changlong, Z. Ting, and Z. Yang, "Route optimization based on genetic algorithms of stacker for automated storage and retrieval system," in *Intelligent Systems (GCIS), 2010 Second WRI Global Congress on*, vol. 1. IEEE, 2010, pp. 243–248.
[22] S.-a. Liu, Q. Wang, and L. Jin, "An optimization model for storage location problem in the automated storage and retrieval system," in *Control and Decision Conference (CCDC), 2011 Chinese*. IEEE, 2011, pp. 3904–3909.
[23] L. B. da Cunha, G. Niemann, J. E. Shigley, A. S. Hall Jr, A. R. Holowenko, H. G. Laughlin, S. Melconian, C. R. Moura, R. P. Carreteiro, G. E. Stemmer *et al.*, *Elementos de máquinas*. LTC, 2005.
[24] (2018, Mai.) Stmicroeletronics - datasheet l298 - dual full-bridge driver. [Online]. Available: https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf