# *UAV Path Planning*: uma abordagem por otimização multiobjetivo e *Differential Evolution*

Diego Geraldo[1], Armando Zeferino Milioni[1] e Mônica Maria de Marchi[2]
1. Instituto Tecnológico de Aeronáutica – Pça Mal. Eduardo Gomes, 50. 12228-900, São José dos Campos, SP
2. Instituto de Estudos Avançados – Rodovia dos Tamoios, km 5,5. 12228-001, São José dos Campos, SP

***Resumo* — Neste artigo, o problema de geração de trajetórias para Veículos Aéreos Não Tripulados (em inglês, *UAV Path Planning*) é modelado como um problema de otimização multiobjetivo, para o qual se busca a minimização simultânea da distância total percorrida e da exposição às ameaças de solo (e.g, radares e mísseis solo-ar). Ferramentas do campo da Otimização Evolutiva Multiobjetivo foram utilizadas na abordagem do problema, especificamente, o algoritmo denominado *Generalized Differential Evolution* 3. A abordagem multiobjetivo, associada ao uso de uma heurística baseada em população de soluções, permite a geração de um conjunto de boas opções de trajetórias, em uma única execução do código. Como consequência, o decisor passa a ter conhecimento das diferentes possibilidades antes de fazer a escolha da trajetória mais adequada ao cenário em questão.**

***Palavras-Chave* — Veículos Aéreos Não Tripulados, Trajetórias, Otimização Multiobjetivo.**

## I. INTRODUÇÃO

Veículos Aéreos Não Tripulados, também conhecidos como UAV (do inglês, *Unmanned Aerial Vehicles*), são aeronaves capazes de operar sem a presença de pilotos a bordo. Os UAVs podem ser empregados com diversas finalidades, em contextos civis ou militares, na realização de tarefas como: levantamento aerofotogramétrico, monitoração ambiental e meteorológica, busca e salvamento, vigilância e reconhecimento.

As aeronaves não tripuladas podem ser controladas remotamente ou voar de maneira autônoma, seguindo, por exemplo, uma trajetória planejada anteriormente.

O problema de construção de uma trajetória (em inglês, *path planning*), envolve a construção de um caminho que ligue um ponto de início a um destino, e que apresente características de otimalidade segundo critérios específicos [1].

Trajetórias destinadas a UAVs devem satisfazer, basicamente, a três critérios [2]: (i) atendimento às restrições físicas e cinemáticas da aeronave (e.g, limites de velocidade, raio mínimo de curva, combustível disponível); (ii) preservação da integridade física do UAV (e.g, desviar de áreas perigosas, evitar colisões com obstáculos ou outras aeronaves); e (iii) atendimento aos requisitos operacionais estabelecidos a partir do tipo de missão (e.g, menor distância, menor consumo de combustível, menor risco).

Diego Geraldo, diegog@ita.br; Armando Zeferino Milioni, milioni@ita.br; Monica Maria de Marchi, monica@ieav.cta.br, Tel. +55-12-3947-5335.

A literatura demonstra que a busca pela solução ótima de um caso genérico do problema em estudo é considerada de natureza NP-completa [3]. Por consequência, diferentes heurísticas foram empregadas como abordagem alternativa, dentre elas o algoritmo A* [3] [4], Programação Linear Inteira Mista [5] e *Minimax* [6].

Os planejadores de trajetórias baseados nesses métodos, entretanto, apresentam alguns óbices, pois trabalham com versões muito simplificadas do problema original, assumindo, por exemplo, que os UAVs voam mantendo altitude constante, promovendo a discretização do espaço de busca, a linearização de modelos tipicamente não lineares, e outras simplificações [7] [8].

Em contrapartida, a literatura indica que as abordagens realizadas por meio de algoritmos evolutivos permitiram modelagens mais complexas e próximas da realidade, sem, contudo, despender demasiado esforço computacional.

O problema de geração de trajetórias com finalidades militares pode ser considerado, por natureza, de otimização multiobjetivo. Em contextos militares, a presença do risco é inerente, e trajetórias mais curtas são desejáveis, uma vez que minimizam o tempo de permanência da aeronave em ambiente hostil. Entretanto, quando outros fatores passam a ser considerados, como exposição a ameaças de solo (e.g, radares e mísseis solo-ar), nem sempre o caminho mais curto será também o de menor risco. Nessas situações, diz-se que os objetivos almejados (i.e, minimizar comprimento da trajetória e exposição às ameaças) passam a ser conflitantes, sendo necessário estabelecer uma relação de compromisso entre eles.

Frente a um problema típico de múltiplos objetivos conflitantes, a literatura existente, predominantemente, utiliza abordagens como a Soma Ponderada [9, p. 50], a fim de tornar possível a aplicação de algoritmos evolutivos destinados à otimização de uma única função-objetivo, como Algoritmos Genéticos (AG) [8] [10] [11] [12] e *Differential Evolution* (DE) [13] [14].

Contudo, a definição *a priori* de pesos pode se tornar uma tarefa difícil, pois o decisor deverá confiar exclusivamente em suas experiências anteriores a fim de definir as importâncias relativas entre os objetivos. Além disso, após ser apresentado à solução ótima correspondente ao conjunto de pesos arbitrados, "o usuário sempre gostaria de saber qual outra trajetória ótima teria sido gerada se um vetor de pesos ligeiramente diferente houvesse sido definido" [15, p. 2].

Os anos da última década testemunharam o crescimento significativo de um ramo da Computação Evolutiva, denominado *Evolutionary Multiobjective Optimization* (EMOO), que oferece métodos exclusivamente voltados à

abordagem de problemas com dois ou mais objetivos conflitantes. A principal vantagem da abordagem por ferramentas de EMOO é que tais métodos são capazes de gerar não apenas uma, mas diversas soluções ditas Pareto-ótimas, em uma única execução do algoritmo. O fato de se ter um conjunto de soluções à disposição para decidir implica em maior flexibilidade, pois o decisor passa a ter conhecimento das diferentes possibilidades antes de fazer a escolha da trajetória mais adequada ao cenário para o qual foi planejada.

Alguns trabalhos existentes na literatura já abordaram o problema de *UAV Path Planning* com ferramentas de EMOO [1] [7] [15] [16]. Um ponto comum a esses trabalhos é a utilização de AG como heurística de busca e otimização, com destaque para o *Nondominated Sorting Genetic Algorithm* (NSGA-II) [17]. Apesar de bastante popular, diversos estudos empíricos indicam que os resultados obtidos por esse algoritmo podem ser superados por outros algoritmos multiobjetivo que empregam operadores de DE no processo evolutivo [18] [19] [20].

Motivados por essas constatações empíricas, e pelo que foi discutido até então, neste trabalho, apresentamos uma modelagem multiobjetivo ao problema de geração de trajetórias para UAVs, e empregamos um algoritmo evolutivo baseado em *Differential Evolution* como mecanismo otimizador. Em particular, optamos pelo *Generalized Differential Evolution* 3 (GDE3), cuja descrição breve será fornecida mais à frente.

O restante deste artigo está organizado da seguinte maneira: a Seção II apresenta alguns fundamentos de otimização multiobjetivo; na Seção III, introduções à heurística de DE e ao GDE3 são fornecidas; a modelagem do problema é apresentada na Seção IV, a Seção V traz os resultados dos experimentos realizados; e por fim, na Seção VI, são apresentadas as conclusões do trabalho.

## II. OTIMIZAÇÃO MULTIOBJETIVO

Nos casos em que dois ou mais objetivos conflitantes passam a ser analisados simultaneamente, a melhora em um dos critérios resultará na piora de pelo menos um dos outros objetivos. Diferentemente dos problemas de otimização mono-objetivo, para os quais, em geral, busca-se encontrar apenas uma solução ótima, no campo da otimização multiobjetivo, o que se busca é encontrar um conjunto de soluções de compromisso, de maneira que cada uma delas atenda aos objetivos em níveis aceitáveis [21] [22].

Formalmente, e sem perda de generalidade, podemos definir um problema de otimização multiobjetivo como sendo [9]:

$$\begin{aligned} \text{Min} \quad & \vec{Y} = [f_1(\vec{X}), f_2(\vec{X}), \dots, f_M(\vec{X})] \\ \text{s.a:} \quad & g_j(\vec{X}) \geq 0, \qquad j = 1,2,\dots,J \\ & h_k(\vec{X}) = 0, \qquad h = 1,2,\dots,K \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \qquad i = 1,2,\dots,n. \end{aligned} \tag{1}$$

com:

$$\vec{X} = [x_1, \dots, x_n] \in D$$
$$\vec{Y} = [y_1, \dots, y_m] \in S$$

Em (1), $\vec{X}$ é um vetor $n$-dimensional de variáveis de decisão, sujeitas a restrições laterais inferiores $x_i^{(L)}$ e superiores $x_i^{(U)}$; $\vec{Y}$ é um vetor $M$-dimensional de funções-objetivo; $D \subseteq R^n$ denota o espaço de variáveis do problema; e $S \subseteq R^M$ representa o espaço de objetivos do problema. O restante desse trabalho assume a minimização de $M$ funções-objetivo.

Nos problemas multiobjetivo com restrições, uma solução que atenda a todas as restrições é dita viável, enquanto outra que viole pelo menos uma delas é denominada inviável.

A existência de mais de uma solução ótima em problemas multiobjetivo leva à necessidade de se estabelecer uma noção diferente de otimalidade. A definição mais comumente utilizada para esse fim é a de *otimalidade de Pareto* [21].

**Definição 1 – Dominância Fraca de Pareto:** um vetor de variáveis de decisão $\vec{X}_1 \in D$ domina fracamente outro vetor de variáveis $\vec{X}_2 \in D$, também escrito como $\vec{X}_1 \preccurlyeq \vec{X}_2$, se, e somente se:

$$\forall_i \in \{1,2,\dots,M\} : f_i(\vec{X}_1) \leq f_i(\vec{X}_2) \tag{2}$$

**Definição 2 – Dominância de Pareto:** um vetor de variáveis de decisão $\vec{X}_1 \in D$ domina outro vetor de variáveis $\vec{X}_2 \in D$, também escrito como $\vec{X}_1 \prec \vec{X}_2$, se, e somente se:

$$\begin{aligned} &\forall_i \in \{1,2,\dots,M\} : f_i(\vec{X}_1) \leq f_i(\vec{X}_2) \ \wedge \\ &\exists j \in \{1,2,\dots,M\} : f_j(\vec{X}_1) < f_j(\vec{X}_2) \end{aligned} \tag{3}$$

**Definição 3 – Otimalidade de Pareto:** Seja $\vec{X} \in D$ uma solução arbitrária. Então:

(i) a solução $\vec{X}$ é dita não dominada em relação ao conjunto $D' \subseteq D$, se, e somente se, não houver outra solução em $D'$ que domine $\vec{X}$;

(ii) a solução $\vec{X}$ é dita Pareto-ótima, se, e somente se, $\vec{X}$ for não dominada em relação a todo o espaço de variáveis $D$.

Em palavras, esta definição diz que uma solução $\vec{X}^*$ é Pareto-ótima se não existir outro vetor $\vec{X}$ capaz de melhorar uma função-objetivo, sem causar uma piora em pelo menos um dos outros critérios [23].

O conjunto de soluções ótimas no espaço de variáveis é denominado Conjunto de Pareto, e a correspondente imagem no espaço objetivo, Fronteira de Pareto.

O principal objetivo dos algoritmos de otimização multiobjetivo é identificar soluções pertencentes (ou próximas) ao Conjunto de Pareto. No entanto, para muitos problemas, o número de soluções Pareto-ótimas pode ser infinito, o que torna inviável, ou mesmo impossível, a consecução de tal tarefa. Portanto, partindo de uma abordagem mais prática, pode-se assumir que os algoritmos multiobjetivo buscam identificar um conjunto finito de soluções que representem o Conjunto de Pareto tão bem quanto possível [24].

## III. DE e GDE3

A heurística de DE [25] foi proposta em 1995, com a finalidade de resolver problemas mono-objetivo de domínios estritamente contínuos.

Tal qual um típico algoritmo evolutivo, o DE trata-se de um método baseado em populacão de soluções, que utiliza operadores de recombinação e mutação para gerar novas soluções, e mecanismo de seleção que permite manter a população com tamanho constante. A ideia básica do DE é a de auto-adaptação da mutação às características da função-objetivo e da distribuição corrente da população pelo espaço de busca.

A cada geração, para cada solução "pai" $\vec{X}_{i,G}$, um "vetor-teste" $\vec{U}_{i,G}$ é gerado por meio do operador de DE. Na notação adotada, $i$ é o índice de uma solução da população e $G$ é o índice da geração corrente. O operador de DE mais comumente encontrado na literatura, e também utilizado nesse trabalho, é o "DE/*rand*/1/*bin*", onde "DE" refere-se ao nome do algoritmo, "*rand*" significa que o vetor selecionado para sofrer mutação será escolhido aleatoriamente, "1" indica o número de pares de soluções sorteadas para gerar o diferencial de mutação e "*bin*", por sua vez, assinala que uma recombinação binomial será empregada.

A geração de um vetor-teste pelo operador DE/*rand*/1/*bin* dá-se da seguinte forma [26]:

$r_1, r_2, r_3 \in \{1,2,\dots,NP\}$, (índices aleatórios, mutuamente diferentes e diferentes de $i$, *NP* = tamanho da população);

$j_{rand} \in \{1,2,\dots,n\}$, (índice aleatório de uma variável do problema);

for $(j = 1; j \leq n; j = j + 1)$
{
  if $(rand_j[0,1) < CR \vee j = j_{rand})$
    $u_{j,i,G} = x_{j,r_3,G} + F.(x_{j,r_1,G} - x_{j,r_2,G})$ (4)
  else
    $u_{j,i,G} = x_{j,i,G}$
}

A diferença entre os vetores de índices $r_1$ e $r_2$, ponderada pelo fator $F$, define a magnitude e a direção da mutação. Quando essa composição é adicionada ao vetor de índice $r_3$, passa a representar a mutação dessa solução. Essa característica de busca direcionada é tida como uma das principais vantagens do DE sobre outros métodos evolutivos [27].

Após a geração do vetor-teste $\vec{U}_{i,G}$ por mutação e recombinação, o mesmo é comparado com o vetor pai $\vec{X}_{i,G}$; se a adaptabilidade (ou *fitness*) do vetor-teste for melhor que a do vetor pai, então, este último é substituído na próxima geração do algoritmo. Essa característica do operador de seleção faz do DE um método elistista.

O parâmetro *CR* está relacionado ao operador de recombinação (*crossover*) do algoritmo, representando a probabilidade de uma variável do vetor-teste $\vec{U}_{i,G}$ ser definida pela combinação linear de três soluções aleatórias da população, ao invés de assumir o valor correspondente do vetor pai $\vec{X}_{i,G}$. A condição "$j = j_{rand}$" garante que pelo menos uma variável do vetor-teste seja diferente do vetor pai.

Em função das várias aplicações bem sucedidas do DE em instâncias mono-objetivo, o algoritmo foi estendido e adaptado para atender a outros tipos de problemas, dentre eles, os de otimização multiobjetivo. Uma revisão completa a respeito dos principais algoritmos multiobjetivo baseados em DE pode ser encontrada em [23] [27].

*Generalized Diferential Evolution (GDE)*

A primeira versão do GDE foi concebida com o propósito de estender o DE aos problemas de otimização multiobjetivo, com ou sem restrições, por meio de alterações no operador de seleção do algoritmo original. A segunda versão do GDE incorpora também um mecanismo que avalia a aglomeração de soluções em determinada região do espaço de busca ("*crowdness*"), para decidir qual das soluções deve ser mantida na próxima geração do algoritmo, nos casos em que ambas forem reciprocamente não dominadas.

Um óbice comum identificado nas duas primeiras versões do GDE foi a sensibilidade ao correto ajuste dos parâmetros de controle. A identificação desse óbice levou ao desenvolvimento da terceira versão do GDE, i.e., GDE3 [26], que modifica as propostas anteriores nos aspectos descritos a seguir.

Ao comparar duas soluções, ou seja, um vetor pai com seu correspondente vetor-teste, as seguintes relações são aplicadas:

- se ambas as soluções forem inviáveis, o vetor-teste é selecionado para a próxima geração somente se dominar fracamente o vetor pai no espaço de restrições, caso contrário, o vetor pai é mantido na população;
- entre uma solução viável e outra inviável, a viável é selecionada para a próxima geração;
- se ambas as soluções forem viáveis, o vetor-teste é selecionado para a próxima geração se dominar fracamente o vetor pai no espaço de objetivos. Se o vetor pai dominar o vetor-teste, então, este último é descartado;
- se ambas as soluções forem viáveis e reciprocamente não dominadas, então, ambas são selecionadas para a próxima geração.

O processo descrito acima é repetido até que todas as soluções da população, i.e., pais, tenham "competido" com seus respectivos vetores-testes, gerados a partir do operador de DE. Após o término da comparação da última solução, a população terá tamanho compreendido entre $[NP, 2NP]$. Caso a população tenha crescido, por conta da inclusão das soluções não dominadas, um procedimento de truncagem é empregado, a fim de prepará-la para o início da próxima geração.

O procedimento de truncagem utilizado no GDE3 compreende a realização do *non-dominated sorting* [9, p. 40] e a avaliação das soluções pertencentes ao mesmo *front* pela métrica de *crownding distance* [9, p. 248]. O primeiro método citado favorece a preservação das soluções não dominadas da população, e o segundo, por sua vez, promove os desempates dentro de um mesmo *front*, favorecendo as soluções pertencentes às regiões menos "povoadas" do espaço de busca.

O leitor interessado em maiores detalhes a respeito do GDE3 pode se referir também a [28].

## IV. MODELAGEM

*Representação das soluções*

O algoritmo de otimização multiobjetivo utilizado nesse trabalho (GDE3) trabalha com população de soluções e, no presente contexto, cada solução representa uma trajetória em "evolução".

Um indivíduo da população é "codificado" como uma lista de coordenadas geográficas, i.e., $(x_i, y_i, z_i)$, que representam os pontos de controle (*waypoints*) da rota. A trajetória correspondente, gerada com base na lista de pontos, considera que o UAV navega mantendo a reta em direção a um determinado *waypoint*; ao sobrevoar o ponto, uma curva de inclinação constante é iniciada e mantida até que o UAV alcance o rumo ideal para o próximo *waypoint* da trajetória, quando, então, volta a voar em linha reta.

Todos os indivíduos da população possuem a mesma "estrutura base". Esta, por sua vez, compreende um ponto de origem $P_O$, um ponto final $P_F$, e pontos intermediários $P_I$ cujo sobrevoo seja obrigatório na missão (ver Fig. 1). Os trechos da trajetória definidos por esses pontos, denominados fixos, constituem as "pernas de navegação".

Os pontos fixos não são submetidos aos operadores evolutivos, servindo apenas para definir o sequenciamento lógico dos *waypoints* ao longo da trajetória. Portanto, é necessário que um usuário estabeleça o número de pontos variáveis $P_V$ desejados em cada perna da trajetória. Os pontos variáveis serão efetivamente "movimentados" pelo operador de DE visando à otimização das soluções. Qualquer trecho da trajetória definido por dois pontos variáveis, ou por um ponto fixo e outro variável, será tratado como um "segmento".

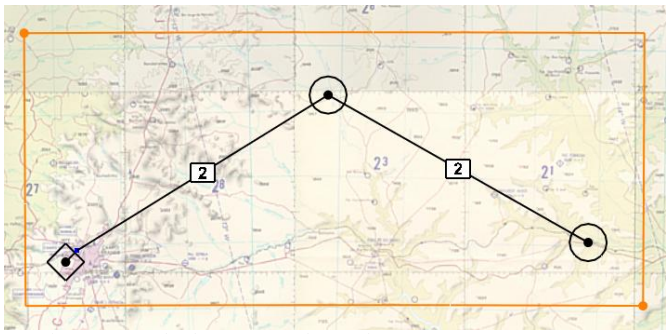


Fig. 1. Exemplo de "estrutura base" de uma solução, composta por um ponto de origem (lado esquerdo), um ponto intermediário (centro), e um ponto final (lado direito). Cada uma das pernas de navegação terá dois pontos variáveis. O retângulo destacado representa os limites do espaço de busca em 2D.

Algumas das funções empregadas na avaliação de um indivíduo requerem a divisão da trajetória gerada em $N$ pontos discretos. Isto é necessário, por exemplo, para verificar se qualquer um dos pontos discretos resulta em colisão com o terreno. O valor de $N$ é fixado *a priori*, e o tamanho do "passo" é ajustado para cada trajetória, em função de seu comprimento.

*Inicialização das soluções*

Na maioria dos algoritmos evolutivos, as soluções são inicializadas de forma completamente aleatória. Entretanto, no problema em estudo, se os valores de latitude, longitude e altitude de cada *waypoint* forem sorteados por todo o espaço de busca, sem restrições, a população inicial conterá percursos não somente inviáveis (o que é esperado para esse tipo de problema), mas também, sem lógica. O sorteio completamente aleatório pode resultar, por exemplo, em curvas que façam a trajetória voltar em direção à origem, ao invés de avançar em direção ao destino.

A fim de minimizar o óbice decorrente do sorteio aleatório das coordenadas, a atribuição de valores aos $P_V$ empregada nessa modelagem se dá conforme a ilustração da Fig. 2.

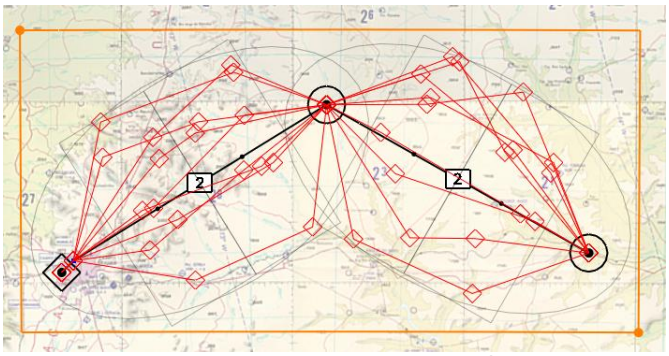


Fig. 2. Exemplo de nove soluções (trajetórias) inicializadas aleatoriamente, com "estrutura base" comum.

De acordo com a Fig. 2, para cada perna contida na estrutura base da trajetória, uma elipse correspondente é gerada. Os focos da elipse são definidos pelo próprio par de pontos fixos da perna. Os limites laterais da elipse são definidos com base em uma relação de comprimento máxima desejada, i.e., a razão entre a soma dos comprimentos dos segmentos que estarão contidos na perna, e o menor comprimento possível (linha reta).

Após a elipse ter sido definida, retângulos são gerados em seu interior. O número de retângulos é igual à quantidade de $P_v$ estabelecida para a perna, e os centros dos retângulos são uniformemente posicionados ao longo da mesma. Por fim, os valores de latitude e longitude de cada $P_v$ são sorteados no interior de um dos retângulos, na ordem em que aparecem, e o valor da altura de voo, posteriormente convertida em altitude, é sorteado com probabilidade uniforme em um intervalo definido pelo usuário.

O sistema de inicialização proposto força o estabelecimento de uma ordenação dos pontos de controle que, caso não fosse "imposta", teria de ser descoberto pelo otimizador ao longo das iterações. Dessa forma, a convergência do algoritmo é acelerada.

*Aeronave*

Um UAV é representado por um modelo genérico de aeronave, que, por sua vez, contempla as seguintes propriedades: direção inicial, velocidade constante de navegação, tabela de consumo (fluxo) de combustível *versus* altitude, ângulo máximo de subida, ângulo mínimo de

descida, máximo fator de carga $g$, e semieixos $(a, b, c)$ de um elipsóide que caracterize a aeronave no cálculo da Seção Reta Radar (*Radar Cross Section* - RCS) da mesma.

*Elementos de Cenário*

O UAV, ao longo de sua trajetória, pode interagir com outros elementos de cenário, os quais, combinados, compõe um cenário de otimização. Os elementos disponíveis para a composição dos cenários são: modelo de elevação do terreno (*Advanced Spaceborne Thermal Emission and Reflection Radiometer* - ASTER) [29], radares, áreas de voo proibidas e unidades de artilharia antiaérea.

*Funções-objetivo*

Os objetivos envolvidos na geração das trajetórias são 3, descritos a seguir:

**$FO_1$ - Minimizar comprimento da trajetória:** nos contextos militares, trajetórias mais curtas são melhores, pois minimizam o tempo de permanência em território inimigo. A função-objetivo é calculada como a razão (divisão) do comprimento real da trajetória, i.e., a soma dos comprimentos dos segmentos que a compõe, pelo menor comprimento possível, i.e., linhas retas que unem os pontos fixos da trajetória;

**$FO_2$ - Minimizar probabilidade de detecção radar:** a redução da probabilidade de detecção do UAV por radares inimigos pode tanto aumentar o efeito surpresa em missões de ataque, quanto prover o sigilo necessário em missões de reconhecimento. Além disso, a maioria das unidades de artilharia antiaérea possui um radar diretor de tiro associado ao sistema de pontaria. Portanto, a redução da detecção radar pode reduzir também a chance do UAV ser destruído por unidades inimigas de defesa antiaérea. O modelo empregado para calcular a probabilidade de detecção acumulada ao longo da trajetória é o proposto em [6] e já utilizado em [7] [16]. Tal modelo considera fatores como a distância entre aeronave e radar, existência de linha de visada, RCS da aeronave (aproximada por um elipsoide), e características operacionais do radar;

**$FO_3$ - Minimizar altura média de voo:** mantendo voo próximo ao solo, o UAV pode se beneficiar do efeito de "mascaramento" proporcionado pelas irregularidades do relevo, reduzindo as áreas de intervisibilidade com as ameaças de solo, aumentando o efeito de furtividade da missão e minimizando a probabilidade de detecção por radares. A função-objetivo acumula as alturas de voo do UAV (diferença entre a altitude de voo e a cota do terreno) em cada ponto discreto da trajetória, dividindo o montante total pela quantidade de pontos discretos utilizados.

*Restrições*

Uma trajetória deve satisfazer a 9 restrições para ser considerada viável. As funções que representam as restrições do problema foram concebidas para atuar da seguinte forma: se o indivíduo for viável, todas as funções resultarão zero; se o indivíduo for inviável, o valor retornado por pelo menos uma das funções será maior que zero, e deverá refletir o "tamanho" da violação à restrição. Essa forma de proceder permite distinguir uma solução viável de outra inviável, e também possibilita identificar, entre duas soluções inviáveis, aquela com menor nível de violação às restrições.

As restrições consideradas na modelagem do problema são as seguintes:

**$R_1$ - Variação máxima de proa durante as curvas:** essa restrição previne a geração de trajetórias com curvas cujas variações de proa, em graus, sejam maiores do que um limite definido pelo usuário. A função penalidade é calculada como o somatório das violações cometidas por cada curva da trajetória individualmente. Por exemplo, supondo o limite estabelecido de 90°, então, se todas as curvas tiverem variações de proa menores que este valor, a função retornará 0; se duas curvas da trajetória resultarem em 110° e 120°, a função retornará (110 - 90) + (120 - 90) = 50, penalizando o indivíduo inviável;

**$R_2$ - Razão máxima de comprimento da trajetória:** essa restrição previne a geração de trajetórias cujas razões de comprimento (divisão do comprimento total pelo menor comprimento possível) excedam um limite definido pelo usuário. O valor dessa função é calculado da seguinte maneira: supondo a razão máxima de 1.5, se a razão de comprimento da trajetória resultar em 1.4, a função retornará 0, no entanto, se a razão resultante for 1.6, a função retornará (1.6 - 1.5) = 0.1, de forma a penalizar a violação;

**$R_3$ - Comprimento mínimo dos segmentos:** em percursos longos, um número grande de curvas, causado pela excessiva proximidade entre pontos consecutivos da trajetória, pode aumentar os erros de navegação. Esta restrição, portanto, visa garantir que exista uma distância mínima (definida pelo usuário) entre quaisquer dois pontos da trajetória que definam um segmento. A função é calculada como o somatório das violações cometidas por cada segmento da trajetória individualmente, de maneira análoga à procedida em $R_1$ e $R_2$;

**$R_4$ - Raio mínimo de curva:** para ser plenamente executável, uma trajetória não pode conter curvas que excedam os limites aerodinâmicos da aeronave. Com base na velocidade de navegação e no máximo fator de carga $g$ suportado pelo UAV, ambos especificados no modelo da aeronave, é possível calcular o raio mínimo de curva tolerado. O planejador calcula o raio de curva requerido em cada uma das mudanças de direção previstas na trajetória, e a presente restrição penaliza as curvas que tiverem um raio menor do que o mínimo suportado pela aeronave, de maneira similar ao já descrito nas restrições anteriores;

**$R_5$ - Ângulos de arfagem:** a atitude de voo da aeronave em todos os pontos discretos da trajetória deve ser tal que o ângulo máximo de subida e o ângulo mínimo de descida previstos no modelo da aeronave não sejam violados. O planejador verifica o ângulo de subida ou descida da aeronave entre cada par de pontos discretos, incrementando o valor da função em uma unidade a cada violação detectada;

**$R_6$ – Combustível disponível:** o UAV carrega uma quantidade finita de combustível a bordo, e a trajetória gerada deve consumir um valor menor do que esse limite para ser considerada viável. Para calcular o total de combustível

consumido, o algoritmo considera a velocidade de navegação e a tabela de 'consumo *versus* altitude', ambas definidas no modelo da aeronave. Se a quantidade de combustível requerida for menor que a disponível, essa função retorna 0, do contrário, a função retorna a quantidade extra de combustível necessária;

**$R_7$ - Altura mínima de voo:** o voo à baixa altura pode favorecer a furtividade e o sigilo da missão, entretanto, uma trajetória que contenha ao menos um de seus pontos discretos resultando em (i) colisão com o terreno, ou (ii) em altura de voo menor do que o valor mínimo estabelecido pelo usuário, deverá ser considerada inviável. O valor dessa função é incrementado toda vez que a altura de voo do UAV em um ponto discreto da trajetória estiver abaixo do limite mínimo estabelecido, de maneira análoga ao empregado nas restrições $R_1$ – $R_4$, i.e., somatório das violações identificadas em cada ponto discreto individualmente;

**$R_8$ - Regiões de voo proibidas (RVP):** nos contextos de operações aéreas militares é comum existirem volumes tridimensionais onde as aeronaves são proibidas de adentrar, quer seja por se tratarem de regiões de alto risco, quer seja por conta da incerteza associada às ameaças nelas contidas. Uma trajetória de voo viável, portanto, não deve passar pelo volume de uma região de voo proibida. O algoritmo penaliza cada ponto discreto da trajetória localizado no interior de uma região de voo proibida;

**$R_9$ - Artilharia Antiaérea (AAe):** uma trajetória viável deve evitar as ameaças provenientes das unidades de defesa antiaérea inimigas, preservando a integridade física do UAV. O modelo de unidade de artilharia antiaérea utilizado contempla as seguintes propriedades: raio de alcance efetivo do armamento, ângulo de tiro mínimo e ângulo de tiro máximo. Além disso, uma antiaérea só é capaz de empregar seu armamento contra uma aeronave se possuir linha de visada com a mesma, isto é, se não houver obstruções provenientes do relevo. O algoritmo penaliza cada ponto discreto da trajetória considerado "sob ameaça" de uma antiaérea, isto é, se estiver ao alcance do armamento, dentro dos limites de ângulo mínimo e máximo de tiro, e com linha de visada estabelecida.

## V. EXPERIMENTOS

A fim de verificar a validade da abordagem e das ferramentas empregadas na modelagem do problema, aplicamos nosso planejador (conjunto de modelos, funções matemáticas e otimizador) a alguns cenários de otimização. Tais cenários são baseados em algumas instâncias propostas em [16].

O planejador foi implementado em um Sistema de Informações Geográficas (SIG) chamado Plataforma AEROGRAF, desenvolvido pelo Instituto de Estudos Avançados (IEAv). Esse SIG baseia-se no conceito de componentes do tipo *plug-ins*, que são adicionados a uma estrutura de desenvolvimento denominada *framework*. Este *framework* fornece as principais funcionalidades requeridas nas tarefas de visualização de um cenário georreferenciado em duas e três dimensões.

Em todos os experimentos realizados, utilizou-se o mesmo ajuste de parâmetros do GDE3, NP = 50, CR = 1.0, e F = 0.5. Estes parâmetros foram escolhidos após alguns testes-piloto, e nenhum esforço no sentido de realizar ajuste fino dos mesmos foi realizado.

*Cenário 1*

Este cenário, ilustrado na Fig. 3, é composto por duas RVP (retângulos azuis), três unidades de AAe, cujas áreas de atuação são representadas pelos círculos vermelhos, e três radares associados às AAe, cujos alcances são representados pelos círculos tracejados em laranja. A distância em linha reta que separa o ponto inicial do destino é de, aproximadamente, 230 km, representando, portanto, características realistas para um cenário militar. A única perna de navegação da trajetória foi configurada com 2 pontos variáveis.

As características desse cenário testam a capacidade do algoritmo de encontrar caminhos viáveis (i.e., livres das ameaças das AAe), passando por corredores estreitos (cerca de 5 Km) [16].

As soluções apresentadas na Fig. 3 foram obtidas após 100 iterações apenas, e indicam que o algoritmo foi capaz de alcançar o resultado esperado. As trajetórias em cor verde identificam as soluções dominadas da população, e as azuis, as não dominadas. Em [16], foi reportado que se gastaram 200 iterações para a obtenção de uma única solução.

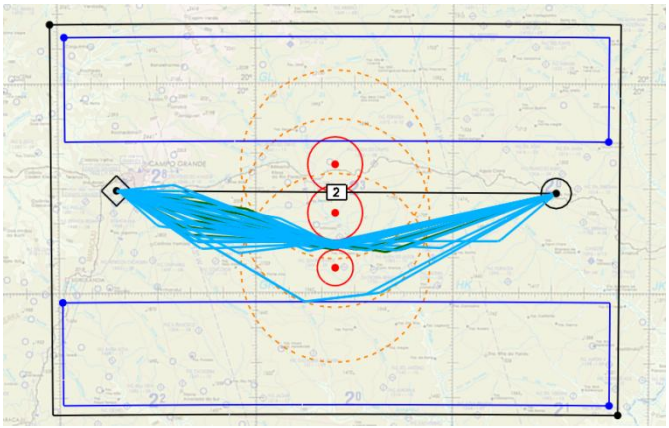

Fig. 3. Características do Cenário 1 e trajetórias viáveis encontradas pelo planejador ao final de 100 iterações.

*Cenário 2*

Neste cenário, 16 unidades de AAe e 16 radares associados a elas são dispostos em forma de espiral, conforme ilustrado na Fig. 4. O ponto final da trajetória é posicionado no interior da espiral, em uma área onde não há risco de destruição por parte das AAe. O algoritmo, então, é solicitado a encontrar trajetórias viáveis, partindo de um ponto externo à espiral de ameaças. O resultado esperado é que o planejador encontre trajetórias que não adentrem os volumes de atuação das AAe.

Segundo [16], as características deste cenário testam a capacidade do planejador de encontrar trajetórias viáveis em ambientes com muitas AAe, onde as áreas livres de ameaça assumem formatos complexos.

Neste cenário, a estrutura base da trajetória foi configurada com 4 pontos variáveis. A distância em linha reta entre ponto de início e destino é de, aproximadamente, 230 km.

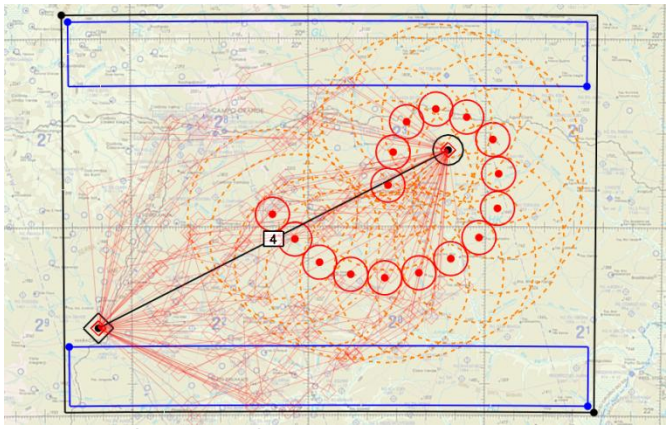


Fig. 4. Características do Cenário 2 e população inicial de soluções.

Os resultados do experimento, obtidos ao final de 200 iterações, são apresentados na Fig. 5, e indicam que o problema foi resolvido com sucesso pelo algoritmo. Em [16], reportou-se que a solução ótima do problema foi obtida após 800 gerações.

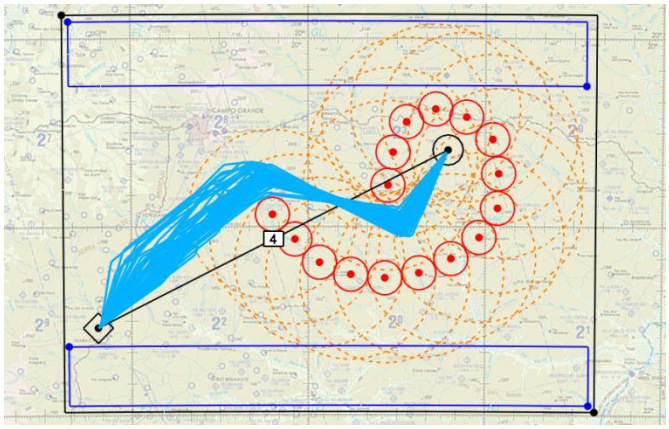


Fig. 5. Soluções não dominadas encontradas pelo planejador no Cenário 2, ao final de apenas 200 iterações.

*Cenário 3*

Esta instância, ilustrada na Fig. 6, é composta por uma RVP, 4 "duplas" AAe + radar, além de outra AAe isolada, posicionada nas proximidades do ponto final. Há uma "passagem" viável com cerca de 8 km de largura existente entre duas das quatro unidades de artilharia posicionadas na parte central do cenário. A porção inferior esquerda do cenário é caracterizada por uma região de serra, e a parte superior direita, por uma região litorânea.

Por empregar ferramentas de otimização multiobjetivo, espera-se que o planejador seja capaz de encontrar desde as trajetórias mais longas e menos expostas aos radares, até as rotas mais curtas, porém com maior suscetibilidade à detecção.

Neste experimento, a perna de navegação foi configurada com 4 pontos variáveis, e a distância que separa o ponto de início do ponto de término é de 270 km.

Os resultados do experimento são apresentados nas Fig. 7 e 8.

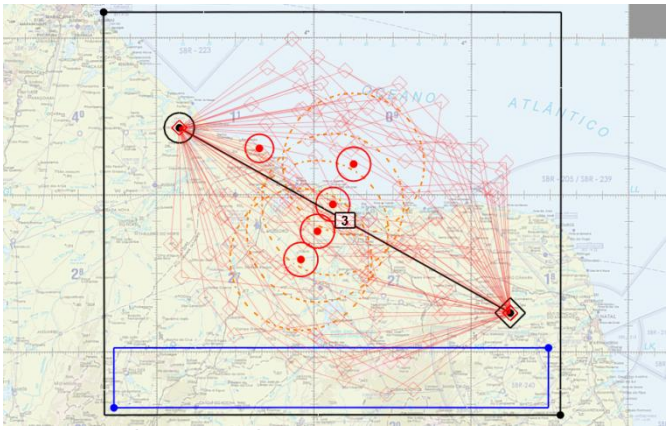


Fig. 6. Características do Cenário 3 e população inicial de soluções.

Conforme esperado, as soluções retornadas ao final de 200 iterações indicam que o algoritmo foi capaz de encontrar trajetórias viáveis, e que refletem diferentes relações de compromisso à disposição do decisor.

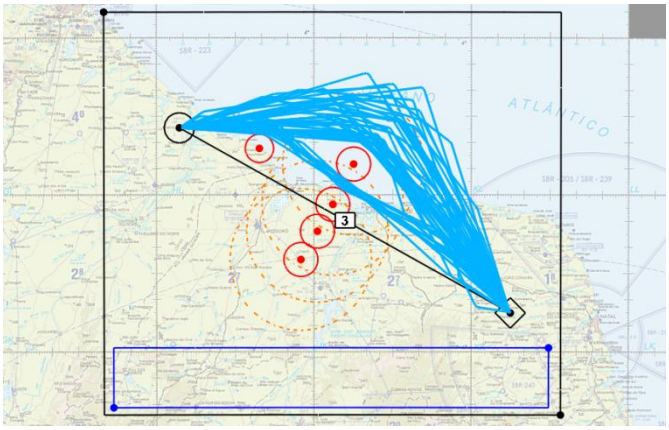


Fig. 7. Soluções não dominadas encontradas pelo planejador no Cenário 3.

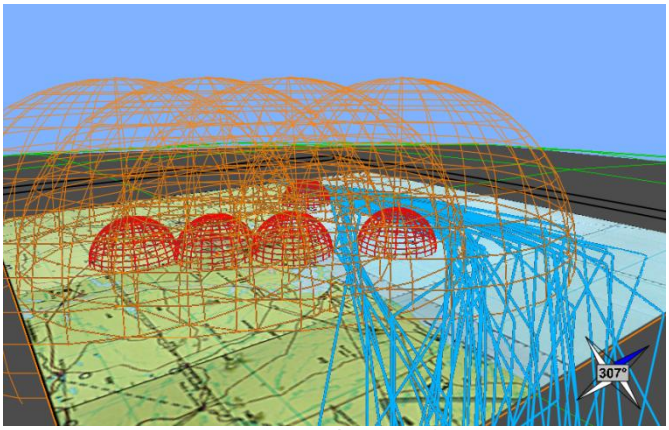


Fig. 8. Visão tridimensional do Cenário 3 ao final de 200 iterações. O planejador é capaz de encontrar diversas soluções, que refletem diferentes relações de compromisso entre os objetivos, em uma única execução.

## V. CONCLUSÕES

Neste trabalho, modelamos o problema de geração de trajetórias destinadas a UAVs como um problema de otimização multiobjetivo, buscando minimizar simultaneamente a distância percorrida e a exposição às ameaças de solo.

Usamos ferramentas de EMOO na abordagem, que permitiram a obtenção de diversas soluções "Pareto-ótimas" em uma única execução código. No presente contexto, essa característica permite que o decisor tenha conhecimento das diversas possibilidades antes de decidir pela trajetória mais adequada ao cenário. Alguns estudos empíricos encontrados na literatura sugeriram a superioridade dos algoritmos baseados em DE sobre outros baseados em AG. Em consequência, optamos por empregar o algoritmo GDE3 como otimizador do planejador proposto.

Por fim, testamos nosso planejador em alguns cenários elaborados a partir de instâncias propostas na literatura, com diferentes características e níveis de dificuldade. O planejador mostrou-se capaz de encontrar trajetórias viáveis em todos os cenários testados, indicando que a modelagem e as ferramentas foram efetivas.

A fase atual de nossa pesquisa envolve o planejamento de trajetórias para múltiplos UAVs operando no mesmo cenário.

## REFERÊNCIAS


[1] G. Sanders and T. Ray, "Optimal Offline Path Planning of a Fixed Wing Unmanned Aerial Vehicle (UAV) using an Evolutionary Algorithm," in *IEEE Congress on Evolutionary Computation CEC'07*, Singapore, 2007.

[2] A. Tsourdos, B. White and M. Shanmugavel, Cooperative Path Planning of Unmanned Aerial Vehicles, 1st ed., Chichester, UK: Wiley, 2011.

[3] R. J. Szczerba, P. Galkowski, I. S. Glickstein and N. Ternullo, "Robust Algorithm for Real-Time Route Planning," *IEEE Transactions on Aerospace and Eletronic Systems*, vol. 36, no. 3, pp. 869-878, July 2000.

[4] R. J. Szczerba, "Threat Netting for Real-Time, Intelligent Route Planners," in *IEEE Symposium on Information, Decision and Control*, Adelaide, 1999.

[5] J. J. Ruz, O. Arévalo, J. M. De la Cruz and G. Pajares, "Using MILP for UAVs trajectory optimization under radar detection risk," in *IEEE Conference on Emerging Technologies and Factory Automation*, Prague, 2006.

[6] P. T. Kabamba, S. M. Meerkov and F. H. Zeitz, "Optimal path planning for Unmanned Combat Aerial Vehicles to Defeat Radar Tracking," *Journal Of Guidance, Control, and Dynamics*, vol. 29, no. 2, pp. 279-288, March-April 2006.

[7] E. Besada-Portas, L. De la Torre, J. M. De la Cruz and B. Andres-Toro, "Evolutionary Trajectory Planner for Multiple UAVs in Realistic Scenarios," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 619-634, August 2010.

[8] C. Zheng, L. Li, F. Xu, F. Sun and M. Ding, "Evolutionary Route Planner for Unmanned Air Vehicles," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 609-620, August 2005.

[9] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, 1st ed., Chichester, U.K.: Wiley, 2001.

[10] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 33, no. 6, p. 898–912, December 2003.

[11] R. Zhang, C. Zheng and P. Yan, "Route Planning for Unmanned Air Vehicles with Multiple Missions Using an Evolutionary Algorithm," in *Third International Conference on Natural Computation ICNC 2007*, Haikou, Hainan, China, 2007.

[12] I. Hasircioglu, H. R. Topcuoglu and M. Ermis, "3-D Path Planning for the Navigation of Unmanned Aerial Vehicles by using Evolutionary Algorithms," in *Genetic and Evolutionary Computation Conference GECCO'08*, Atlanta, USA, 2008.

[13] I. K. Nikolos and A. N. Brintaki, "Coordinated UAV Path Planning Using Differential Evolution," in *13th Mediterranean Conference on Control and Automation*, Limassol, Cyprus, 2005.

[14] X. Zhang, J. Chen, B. Xin and H. Fang, "Online Path Planning for UAV using an Improved Differential Evolution Algorithm," in *18th International Federation Automatic Control (IFAC) World Congress*, Milano, Italy, 2011.

[15] S. Mittal and K. Deb, "Three-Dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms," in *IEEE Congress on Evolutionary Computation*, Singapore, 2007.

[16] J. M. De la Cruz, E. Besada-Portas, L. Torre-Cubillo, B. Andres-Toro and J. A. Lopez-Orozco, "Evolutionary Path Planner for UAVs in Realistic Environments," in *Genetic and Evolutionay Computation Conference GECCO'08*, Atlanta, USA, 2008.

[17] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182-197, April 2002.

[18] K. Deb, A. Sinha and S. Kukkonen, "Multi-Objective Test Problems, Linkages, and Evolutionary Methodologies," in *Genetic and Evolutionary Computation Conference GECCO'06*, Seattle, USA, 2006.

[19] H. Li and Q. Zhang, "A Multiobjective Differential Evolution Based on Decomposition for Multiobjective Optimization with Variable Linkages," in *Parallel Problem Solving from Nature - PPSN IX*, vol. 4193, Springer Berlin / Heidelberg, 2006, pp. 583-592.

[20] T. Tušar and B. Filipič, "Differential Evolution versus Genetic Algorithms in Multiobjective Optimization," in *Evolutionary Multi-Criterion Optimization*, vol. 4403, Springer Berlin / Heidelberg, 2007, pp. 257-271.

[21] C. A. Coello Coello, "A Short Tutorial on Evolutionary Multiobjective Optimization," in *Evolutionary Multi-Criterion Optimization*, vol. 1993, Berlin, Heidelberg, Springer, 2001, pp. 21-40.

[22] E. Zitzler, M. Laumanns and S. Bleuler, "A Tutorial on Evolutionary Multiobjective," in *Metaheuristics for Multiobjective Optimisation*, vol. 535, Berlin, Springer, 2004, pp. 3-38.

[23] C. A. Coello Coello, G. B. Lamont and D. A. Van Veldhuizen, Evolutionary Algorithms for solving Multi-Objective Problems, 2nd ed., New York, USA: Springer, 2007.

[24] A. Konak, D. W. Coit and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering and System Safety*, vol. 91, p. 992–1007, January 2006.

[25] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuos Spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.

[26] S. Kukkonen and J. Lampinen, "GDE3: The Third Evolution Step of Generalized Differential Evolution," in *IEEE Congress on Evolutionary Computation*, Edinburgh, UK, 2005.

[27] E. Mezura-Montes, M. Reyes-Sierra and C. Coello Coello, "Multi-objective Optimization Using Differential Evolution: A Survey of the State-of-the-Art," in *Advances in Differential Evolution*, vol. 143, Springer Berlin / Heidelberg, 2008, pp. 173-196.

[28] S. Kukkonen and J. Lampinen, "Performance Assessment of Generalized Differential Evolution 3 with a Given Set of Constrained Multi-Objective Test Problems," in *IEEE Congress on Evolutionary Computation CEC'09*, Trondheim, Norway, 2009.