UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO TECNOLÓGICO

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Uma Nova Abordagem em Inteligência de Enxames Aprimorados Aplicada ao Rastreamento de Alvos em Vídeo

Edwards Cerqueira de Castro

Vitória
2021

EDWARDS CERQUEIRA DE CASTRO

Uma Nova Abordagem em Inteligência de Enxames Aprimorados Aplicada ao Rastreamento de Alvos em Vídeo

Tese de Doutorado apresentado ao Programa de Pós-Graduação em Engenharia Elétrica da UFES como parte dos pré-requisitos necessários para a obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Prof. Dr. Evandro Ottoni Teatini Salles (PPGEE) e Prof. Dr. Patrick Marques Ciarelli (PPGEE).

Vitória
2021

Ficha Catalográfica - Sistema Integrado de Bibliotecas - SIBI/UFES

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Castro, Edwards Cerqueira de, 1971-
C355n Uma nova abordagem em inteligência de enxames aprimorados aplicada ao rastreamento de alvos em vídeo / Edwards Cerqueira de Castro. - 2021.
73 f. : il.

Orientador: Evandro Otoni Teatini Salles.
Coorientador: Patrick Ciarelli.
Tese (Doutorado em Engenharia Elétrica) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Programação heurística. 2. Programação dinâmica. 3. Otimização combinatória. 4. Análise de séries temporais. 5. Vídeo digital. 6. Visão por computador. I. Salles, Evandro Otoni Teatini. II. Ciarelli, Patrick. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 621.3

EDWARDS CERQUEIRA DE CASTRO

Uma Nova Abordagem em Inteligência de Enxames Aprimorados Aplicada ao Rastreamento de Alvos em Vídeo

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do título de Doutor em Engenharia Elétrica.

Evandro Ottoni Teatini Salles - UFES
Orientador

Patrick Marques Ciarelli - UFES
Orientador

Aura Conci - UFF
Membro externo

Cornélia Janayna Pereira Passarinho - UESPI
Membro externo

Jacques Facon - UFES
Membro externo

Mariana Rampinelli Fernandes - IFES
Membro externo

Vitória/ES
2021

DEDICATÓRIA

Ao Sr. João Osório de Castro Filho (*in memorian*)
À Sra. Ana Paula Freire de Castro

AGRADECIMENTOS

Eu agradeço os meus orientadores Evandro O. T. Salles e Patrick M. Ciarelli.

Eu também agradeço os membros da banca, o professor Dr. Mauro Cesar Martins Campos (DEST/UFES), os professores e funcionários do Programa de Pós Graduação em Engenharia Elétrica, em especial a secretária Aline, e os meus colegas de doutorado.

Eu agradeço, ainda, os meus familiares, amigos e a todos que me deram força e apoio, principalmente a minha esposa Sra. Ana Paula Freire de Castro. Talvez eu tenha que agradecer também a EDP-ES (ESCELSA) por não ter faltado energia elétrica nos dias de execução dos trabalhos de rastreamento. Muita gratidão a todos, obrigado.

Edwards Cerqueira de Castro
Vitória, 26 de abril de 2021

RESUMO

Este trabalho propõe um balanceamento entre transferência de conhecimento e a diversidade de partículas em algoritmos de inteligência de enxames em problemas de otimização dinâmica. O método proposto foi projetado para ser aplicado em problemas de rastreamento de alvos em vídeos. Também é proposto o uso de uma versão do modelo de Alisamento Exponencial Duplo robusto a valores extremos, utilizada para prever a posição do alvo no *frame* seguinte, auxiliando a delimitação do espaço de busca em uma região promissora. Para averiguar a qualidade da abordagem proposta, um rastreador apropriado para espaço discreto de soluções foi implementado usando a meta-heurística *Shuffled Frog Leaping Algorithm* (SFLA – significa: "Algoritmo de saltos de sapos embaralhados") adaptado para atuar em problemas de otimização dinâmica, chamado de SFLA dinâmico (DSFLA). O DSFLA foi comparado com outros rastreadores, clássicos e atuais, cujos algoritmos são baseados em inteligência de enxames. Os rastreadores foram comparados em termos do tempo médio de processamento por *frame* e da área sob a curva da taxa de sucesso por métrica de Pascal. Para o experimento, uma amostra aleatória de vídeos foi obtida do *benchmark* público *Hanyang visual tracker*. Os resultados experimentais sugerem que o DSFLA tem um tempo médio de processamento por *frame* eficiente e boa qualidade do rastreamento quando comparados aos outros rastreadores analisados. A taxa de sucesso do DSFLA é, em média, cerca de 7 a 76% maior quando comparada com a taxa de sucesso dos rastreadores comparados. A média de tempo de processamento por *frame* é cerca de, pelo menos, 10% mais rápido que os outros rastreadores, exceto um que é cerca de 26% mais rápido que o DSFLA. O erro quadrático médio de previsão mostra que as previsões do modelo de Alisamento Exponencial Duplo Robusto são satisfatórias e delimitam eficientemente o espaço de soluções. Os resultados do DSFLA também foram comparados aos resultados dos 10 primeiros rastreadores posicionados na lista do desafio OPE 100 do *benchmark Hanyang visual tracker*. O resultado do intervalo de 95% de confiança posiciona o DSFLA entre os 6 primeiros da lista.

Palavras chaves: Inteligência de enxames, meta-heurística, problemas de otimização dinâmica, rastreamento de alvos em vídeo, previsão de séries temporais.

ABSTRACT

This work proposes a approach to balance knowledge transfer and particle diversity in swarm intelligence algorithms in dynamic optimization problems. The method was designed to be applied to the problem of video tracking targets. It is also proposed, to use a robust version to outliers of the Double Exponential Smoothing (RDES) model, used to predict the target position in the frame delimiting the solution space in a more promising region for target tracking. To assess the quality of the proposed approach, an appropriate tracker for a discrete solution space was implemented using the meta-heuristic Shuffled Frog Leaping Algorithm (SFLA) adapted to dynamic optimization problems, named the Dynamic SFLA (DSFLA). The DSFLA was compared with other classic and current trackers whose algorithms are based on swarm intelligence. The trackers were compared in terms of the average processing time per frame and the area under curve of the success rate per Pascal metric. For the experiment, we used a random sample of videos obtained from the public Hanyang visual tracker benchmark. The experimental results suggest that the DSFLA has an efficient processing time and higher quality of tracking compared with the other trackers analyzed in this work. The success rate of the DSFLA tracker is about 7.2 to 76.6% higher on average when comparing the success rate of its competitors. The average processing time per frame is about at least 10% faster than competing trackers, except one that was about 26% faster than the DSFLA tracker. The results also show that the predictions of the RDES model are quite accurate. The DSFLA results were also compared to the results of the first 10 trackers placed on the Hanyang visual tracker OPE 100 challenge list. The result of the 95% confidence interval places the DSFLA in the top 6 on the list.

Keywords: swarm intelligence; meta-heuristic; dynamic optimization problems; video target tracking; time series forecasts

LISTA DE TABELAS

Tabela 1 – Os vídeos selecionados de *Hanyang visual tracker benchmark*; ..... Pág. 49

Tabela 2 – Tempo médio de processamento por *frame* em segundos; ..... Pág. 53

Tabela 3 – AUC da taxa de sucesso por métrica de Pascal; ..... Pág. 55

Tabela 4 – O RMSE de previsão de x, y e a distância de previsão (em nº de *pixels*); Pág. 59

Tabela 5 – Média e mediana da AUC e do tempo médio de processamento por *frame* do DSFLA, com e sem delimitação de S, nas Versões 1 e 2, respectivamente; Pág. 60

LISTA DE FIGURAS

Figura 1 – Um exemplo de delimitação do espaço de soluções; ..... Pág. 46

Figura 2 – O *boxplot* da variável tempo médio de processamento por *frame* para todos os rastreadores; ..... Pág. 54

Figura 3 – O *boxplot* da variável AUC para todos os rastreadores; ..... Pág. 56

Figura 4 – A curva da taxa de sucesso por métrica de Pascal do vídeo 4 (*BlurFace*) para os 5 rastreadores; ..... Pág. 57

Figura 5 – A curva da taxa de sucesso por métrica de Pascal do vídeo 7 (*Couple*) para os 5 rastreadores; ..... Pág. 57

Figura 6 – A curva média da taxa de sucesso por métrica de Pascal de uma sequência de 100 vídeos. ..... Pág. 62

LISTA DE SIGLAS

ADSO	*Adaptive Swarm Discrete Optimization* – Otimização Discreta Adaptativa por Enxames (algoritmo SI);
AED	Alisamento Exponencial Duplo (modelo de séries temporais não paramétrico);
AEDR	AED Robusto (para valores discrepantes);
AUC	*Area Under Curve* – Área sobre curva;
BBPSO	*Bare Bones* PSO – Ou PSO Gaussiano (algoritmo SI);
CS	*Cuckoo Search* – Busca Cuco (algoritmo SI);
CV	Coeficiente de Variação (estatística de variação amostral);
DSFLA	SFLA Dinâmico (algoritmo SI aprimorado);
HOG	*Histogram of Oriented Gradients* – Histograma de Gradientes Orientados;
IQR	Intervalo interquartílico (ou intervalo interquartil) (estatística de variação amostral);
KF	*Kalman Filter* – Filtro de Kalman (modelo estocástico);
MS	*Mean Shift* (algoritmo de otimização por gradientes ascendentes);
MSE	*Mean Square Error* – Erro quadrático médio (estatística agregada de variação e vício quadrático);
OPE	*One-Pass Evaluate* – Avaliação a uma passagem (desafio de qualidade para rastreadores);
PF	*Particle Filter* – Filtro de Partículas (modelo estocástico por simulação Monte Carlo);
PSO	*Particle Swarm Optimization* – Otimização por Enxames de Partículas (algoritmo SI);
RMSE	*Root MSE* – Raiz quadrada do MSE;
SD	*Standart Deviation* – Desvio padrão (estatística de variação amostral);
SFLA	*Shuffled Frog Leaping Algorithm* – Algoritmo de Salto de Sapos Embaralhados (algoritmo SI);

SI *Swarm Intelligence* – Inteligência de Enxames (método de otimização);

SRE *Spatial Robust Evaluate* – Avaliação da robustez espacial (desafio de qualidade para rastreadores);

SSA *Salps Search Algotithm* – Algoritmo de Busca Salpas (algoritmo SI);

TLD *Tracking-Learning-Detection* – Rastreia-Aprende-Detecta (rastreador de alvos em vídeo);

TRE *Temporal Robust Evaluate* – Avaliação da robustez temporal (desafio de qualidade para rastreadores);

LISTA DE SÍMBOLOS

$p = (x, y, w, h)$	Uma janela (ou *bounding box*), representando um alvo; ou, uma partícula (ou uma solução factível) de uma meta heurística;
(x, y)	A coordenada 2D do pixel localizado no canto superior esquerdo do *bounding box*;
(w, h)	As dimensões horizontal e vertical (em número de *pixels*) do *bounding box*;
(w_T, h_T)	As respectivas dimensões horizontal e vertical do *template*;
$f: S \to \mathbb{R}$	A função custo (ou de aptidão) de uma meta heurística;
$f: T \times S \to \mathbb{R}$	A função custo (ou de aptidão) de uma meta heurística em problemas de otimização dinâmica;
S	O espaço de solução (ou espaço de busca) de uma meta heurística;
T	O conjunto de índices temporais (em geral: corresponde ao tempo cronológico discreto igualmente espaçados);
t	O índice referente ao tempo (ou ao *frame*) ($t = 1,2,\ldots,n$);
n	O total de tempos (ou *frames*);
$\mathbb{R}$	O conjunto dos números reais;
$\mathbb{Z}_+$	O conjunto dos números inteiros não negativos;
d	A dimensão do espaço de busca;
$P(t)$	É um subconjunto de S no tempo t, $P(t) \subseteq S, \forall t$;
$N(p)$	É um conjunto de vizinhos de p em $P(t)$, ($N(p) \subseteq P(t)$);
R e Q	O total de *pixels* das dimensões horizontal e vertical de um *frame*, respectivamente;
$gBest = (x_g, y_g, w_g, h_g)$	Representa a partícula p de melhor aptidão do enxame de uma meta heurística, isto é, o ótimo global (ou um ótimo local);
$pBest$	Representa a posição de melhor aptidão de uma partícula p de uma meta heurística;
N	O número de partículas do enxame de uma meta heurística;
τ	O índice referente à iteração de um algoritmo SI ($\tau = 1,2,\ldots,n_\tau$);
n_τ	O número total de iterações do algoritmo SI;
p_i^τ	A partícula p_i na iteração τ de uma meta heurística;
$p_{i,j}^\tau$	É a variável j da partícula p_i na iteração τ;
i	O índice referente à partícula do enxame ($i = 1,2,\ldots,N$);
J	O índice referente ao componente (ou variável ou parâmetro) da partícula do enxame de um algoritmo SI (j= $1,2,\ldots,d$);
$f(p_i)$	A função de aptidão da partícula p_i;
$f(p,t)$	A função de aptidão da partícula p_i no tempo t;
p'	Um ótimo local ($p' \in N(p)$);
p^*	Um ótimo global ($p^* \in P(t)$);
μ_i^τ	A média do movimento da partícula p_i na iteração τ do algoritmo BBPSO;
σ_i^τ	A variância do movimento da partícula p_i na iteração τ do algoritmo BBPSO;

$\otimes$	O produto, componente por componente, entre dois vetores de mesma dimensão;
$pBest_i^\tau$	É o $pBest$ da partícula p_i na iteração τ;
$gBest^\tau$	É o $gBest$ na iteração τ;
$\boldsymbol{Z}$	Um vetor aleatório Gaussiano multivariado;
$\boldsymbol{\Sigma_d}$	Uma matriz diagonal de ordem d de variâncias e covariâncias;
U	Um número pseudoaleatório gerado uniformemente no intervalo contínuo unitário;
$\lvert\cdot\rvert$	O valor absoluto de um número ou a cardinalidade de um conjunto;
$\lceil\cdot\rceil$	O menor número inteiro maior ou igual à "·";
H_f	A distribuição de probabilidades adaptativa do algoritmo ADSO;
$H_f^\tau(p_i)$	É a função H_f da partícula p_i na iteração τ (ADSO);
$H_f^0(p_i)$	É a função inicial H_f (ADSO);
th_1, th_2 e th_3	São limiares de oclusão das partículas definidos pelo usuário no algoritmo ADSO;
$novo(p_{i,j})$	A função que substitui a variável j da partícula p_i por um número aleatório dentro dos limites da variável no espaço de solução do algoritmo ADSO;
F^τ	A fonte de alimento F na iteração τ do algoritmo SSA;
L_s e L_i	São, respectivamente, os limites superiores e inferiores das variáveis componentes das partículas no algoritmo SSA;
c_1, c_2 e c_3	São os coeficientes da função que controla o movimento das salpas líderes no algoritmo SSA;
κ_1 e κ_2	São os coeficientes que controlam o balanço entre a exploração local e global do espaço de soluções no algoritmo SSA;
n_l	O número de salpas líderes no algoritmo SSA;
p_w^τ	A posição do sapo de pior aptidão do memeplex na iteração τ do algoritmo SFLA;
p_b^τ	A posição do sapo de melhor aptidão do memeplex na iteração τ do algoritmo SFLA;
p_g^τ	A posição do sapo de melhor aptidão do pântano do algoritmo SFLA;
l_w^τ	O salto realizado pelo sapo de pior aptidão do memeplex na iteração τ do algoritmo SFLA;
l_{Max}	O limite do salto permitido dos sapos no algoritmo SFLA;
n_b	O número de *bins* (ou intervalos ou faixas) de um histograma;
$\hat{n}_b$	O estimador pontual do n_b;
h_b	A amplitude dos *bins* de um histograma;
$\hat{h}_b$	O estimador pontual da h_b;
R_s	A amplitude amostral;
b	O índice referente ao *bin* do histograma ($b = 1,2,\ldots,n_b$);
$\beta(H_T^*, H_P^*)$	A distância de Bhattacharyya entre os histogramas H_T^* e H_P^*;
H_T^* e H_P^*	Os histogramas padronizados (de área unitária) do *template* e da partícula, respectivamente;
$\{Z_t\}_{t\in T}$	Um processo estocástico de parâmetro (ou tempo) discreto;
$\{z_t\}_{t\in T}$	A série temporal observada do processo estocástico $\{Z_t\}_{t\in T}$;
μ_t	O fator de alisamento correspondente ao nível no tempo t dos modelos AED e AEDR;
M_t	O fator de alisamento correspondente à tendência no tempo t dos modelos AED e AEDR;

ε_t	O ruido aleatório dos modelos AED e AEDR, isto é, uma variável aleatória com média zero e variância positiva constante, não correlacionada com $\varepsilon_\iota, \forall \iota \neq t$ e não correlacionada com $Z_t, \forall t$;
$\hat{\mu}_t$ e $\widehat{M}_t$	As estimativas pontuais do nível e da tendência do modelo AEDR, respectivamente;
α_3 e α_4	Os coeficientes (ou constantes) de suavização do modelo AEDR;
LI_t e LS_t	Os respectivos valores dos limites inferior e superior da observação z_t;
$\bar{Z}_t^*$	A média da série observada $\{z_t\}_{t\in T}$ de z_1 até z_t $(t = 1,2,\ldots,n)$;
S_t^*	A variância da série observada de z_1 até z_t $(t = 1,2,\ldots,n)$;
$\hat{Z}_t(k)$	A previsão da variável aleatória Z_{t+k} do processo gerador da série, isto é, a previsão do modelo AEDR da observação Z de horizonte k a partir do instante de tempo t.
$\hat{p}_{t-1}(1)$	A previsão de horizonte 1 de uma partícula estimada a partir do instante de tempo $t-1$ ($\hat{p}_{t-1}(1) = \hat{Z}_{t-1}(1)$);
$\hat{p} = (\hat{x}, \hat{y}, \hat{w}, \hat{h})$	A previsão para partícula de melhor aptidão no *frame* t dada pelo modelo AEDR no *frame* $t-1$ ($\hat{p} = \hat{p}_{t-1}(1) = \widehat{gBest_t}$);
$\alpha_1, \alpha_2, \gamma_1$ e γ_2	As constantes das equações de delimitação do espaço de soluções do modelo DSFLA;
δ_{Min}	A distância mínima permitida entre as partículas (no processo de seleção das partículas para transferência de conhecimento) no modelo SFLA;
α_0	O percentual máximo de transferência de partículas permitido no modelo DSFLA;
S_w^* e S_h^*	As respectivas dimensões horizontal e vertical do espaço de solução delimitado no modelo DSFLA;
S1 e *S2*	As respectivas áreas do *frame* delimitadas correspondentes às respectivas delimitações do espaço de soluções (em função do $gBest$ e do $\hat{p}$, respectivamente) no modelo DSFLA;
$Pa(\xi_{GT}, \xi_C)$	A métrica de pascal entre os *bounding boxes* ξ_{GT} e ξ_C;
ξ_{GT} e ξ_C	Os respectivos *bounding boxes* do Alvo verdadeiro marcada a mão e do alvo candidato;
sd	O desvio padrão amostral;
cv	O coeficiente de variação amostral;
$\bar{p}$	A média amostral de uma variável p observada;
q_1, q_2 e q_3	Os primeiro, segundo e terceiro quartis amostrais, respectivamente.

SUMÁRIO

1 Introdução..... 18

1.1 O Rastreamento de Alvos em Vídeos Digitais..... 18

1.2 As Contribuições do Trabalho..... 21

1.2.1 A Hipótese..... 21

1.2.2 Os Objetivos..... 21

1.2.3 As Contribuições do Trabalho e a Lista de Publicações..... 22

1.3 A Estrutura do Trabalho..... 23

2 Revisão da Literatura..... 24

3 Teorias e Métodos..... 29

3.1 Rastreamento de Alvos em Vídeos..... 29

3.2 Otimização por Inteligência de Enxame..... 29

3.2.1 O Algoritmo PSO..... 31

3.2.2 O Algoritmo ADSO..... 33

3.2.3 O Algoritmo SSA..... 34

3.2.4 O Algoritmo SFLA..... 35

4 A Metodologia da Proposta..... 38

4.1 Definições das Características dos Alvos e da Medida de Similaridade..... 38

4.2 Problemas de Otimização Dinâmica..... 39

4.3 O Modelo Proposto..... 41

4.3.1 O Modelo de Alisamento Exponencial Duplo Robusto..... 41

4.3.2 O Rastreador SFLA Dinâmico aprimorado, o DSFLA..... 44

5 Delineamento Experimental e Análise dos Resultados 48

5.1 O Delineamento Experimental..... 48

5.2 A Configuração dos Rastreadores..... 52

5.3 A Análise dos Resultados..... 53

5.4 O Desempenho do DSFLA no Desafio OPE 100..... 61

6 Conclusões..... 64

7 Referências..... 67

1 Introdução

1.1 O Rastreamento de Alvos em Vídeos Digitais

Nas últimas décadas, tem-se testemunhado um avanço tecnológico dos computadores e das câmeras de vídeos digitais. Além disso, há uma grande oferta desses produtos no mercado mundial promovendo uma crescente popularidade destes produtos e estimulando o interesse por análise automatizada de vídeos. Atualmente, são inúmeras as aplicações de vídeos em problemas do mundo real, tais como: na área de vigilância e segurança (Yokoyama e Poggio, 2005), na interface homem-máquina (Ma et al., 2011) e na robótica (Das Sharma, Chatterjee e Racshit, 2012).

O rastreamento de alvos em vídeo digitais é a tarefa de estimar a posição e a trajetória de um ou mais alvos em um vídeo digital. Um vídeo digital é uma sequência ordenada de imagens digitais, chamadas de *frames*, e um alvo pode ser definido como qualquer objeto (ou elemento) de interesse numa cena, por exemplo: uma ou mais pessoas andando numa calçada, carros trafegando por uma avenida, um helicóptero voando, animais correndo no campo, etc. Cada *frame*, por ser uma imagem digital[1], é um arranjo retangular bidimensional composto por um número finito de elementos, cada um com sua localização e valor específico, chamados de *pixels*.

Um rastreador de alvos em vídeos é um algoritmo desenvolvido para análise automatizada de vídeos digitais aplicado em uma das três tarefas chaves (Yamaz, Javed e Shah, 2006) que são:

- a detecção de movimentos de interesse do alvo;
- o rastreamento visual dos alvos *frame* a *frame*; e,
- a análise do rastreamento do alvo para o reconhecimento de padrões de comportamento.

Segundo Yamaz, Javed e Shah (2006), para desenvolver um algoritmo capaz de rastrear os alvos em vídeos para qualquer uma das tarefas chaves, o algoritmo deve ser capaz de estimar a posição e o aspecto dos alvos. Assim sendo, é importante definir:

- a forma dos alvos;
- a aparência e as características da aparência dos alvos; e,
- uma medida de similaridade entre os alvos.

Ainda segundo Yamaz, Javed e Shah (2006), um alvo pode ser representado:

[1] Daqui em diante, vídeos e imagens digitais serão chamados apenas de vídeo e imagem, respectivamente.

- pela forma, exemplos:
    - por pontos,
    - por forma geométrica primitiva tais como círculos, elipses e retângulos,
    - por contornos, ou
    - por silhuetas;
- pela aparência, exemplos:
    - por densidades de probabilidades de uma característica selecionada, ou
    - por modelos (*templates*) das características selecionadas e/ou formas das partes de um alvo; e,
- pelas características extraídas de uma região do *frame* pertencente à forma do alvo, exemplos:
    - cor,
    - cantos,
    - bordas,
    - textura,
    - mapas de derivadas orientadas,
    - histogramas locais, ou
    - fluxo óptico.

A escolha dos tipos de forma, aparência e das características dos alvos são dependentes entre si, do tipo de análise e da aplicação.

De acordo com Rathnayake et al. (2020) há dois métodos de estimação da trajetória de um alvo em vídeos:

- *online*: em que é possível usar os *frames* corrente e passados do vídeo para estimar o estado do alvo em cada período de tempo. Este método é adequado para aplicações onde é necessário rastrear o alvo em tempo real;
- *on Batch*: em que é possível usar toda a sequência de *frames* do vídeo para otimizar a estimativa da trajetória do alvo em cada período de tempo utilizando informações passadas, presente e futuras. Entretanto, não é possível empregar este método nas aplicações em que se exige o rastreamento em tempo real.

Sardari e Moghaddan (2017) propõem uma classificação para os modelos de rastreadores em duas categorias:

- a categoria 1: em que os modelos são baseados em algoritmos estocásticos; e,
- a categoria 2: em que os modelos são baseados em algoritmos de detecção do alvo por combinação de padrões (ou *template matching*, em inglês).

Os modelos de rastreadores da categoria 1 usam um algoritmo estocástico para prever a posição e o aspecto do alvo em um *frame* de acordo com o padrão do movimento e das características observáveis dos alvos. Os principais exemplos de modelos desta categoria são: o rastreador baseado no Filtro de Kalman (KF – do inglês: *Kalman Filter*) (Weng, Kuo e Tu, 2006) e o rastreador baseado no Filtro de Partículas (PF – do inglês: *Particle Filter*) (Morelande, Challa e Gordon, 2004).

Já os modelos de rastreadores da categoria 2 selecionam algumas regiões do *frame* corrente (correspondendo aos candidatos a alvo) e extrai delas uma ou mais características observáveis. Cada característica é, portanto, comparada com a respectiva característica de um ou mais modelos (ou *templates*) do alvo a ser rastreado. A região cujas características são mais similares com as respectivas características dos *templates* indica a posição provável do alvo no *frame* corrente. Um rastreador clássico desta categoria é o *Mean Shift* (MS – significa: "mudança média") proposto por Comaniciu, Ramesh e Meer (2003). Todavia, há uma classe de rastreadores nesta categoria em que os modelos são baseados em algoritmos de otimização uma vez que a posição provável do alvo é indicada pela posição de máxima similaridade (ou mínima distância/divergência) entre as características dos *templates* e dos alvos candidatos.

É importante observar que não há uma categoria de rastreadores globalmente mais vantajosa. O rastreamento de alvos em vídeos é um processo estocástico de tempo discreto com espaço de estados discreto, portanto, são problemas combinatórios cuja modelagem é heurística, isto é, a modelagem é específica ao problema.

Há, entretanto, comparações da qualidade do rastreamento entre rastreadores pertencentes às duas categorias (Canalis, Tejera e Nielsen (2006), Tawab, Abdelhalim e Habib (2010), Gao, Yin et al. (2015)) mostrando resultados compatíveis, com pequena vantagem para um método dependendo da aplicação (como esperado).

Há uma classe importante de algoritmos de otimização, aplicada aos rastreadores da categoria 2, que é formada por algoritmos baseados em inteligência de enxames (SI – do inglês: *Swarm Intelligance*) que será empregada e abordada neste trabalho em detalhes no Capítulo 3.

Outra importante observação é que a tarefa de rastrear alvos em vídeos é bastante desafiadora devido à complexidade gerada pela interferência de vários fatores, tais como:

- a oclusão parcial ou total do alvo;
- mudanças bruscas de iluminação do ambiente ou de movimento do alvo;
- movimentos rápidos do alvo ou da câmera;
- rotações, deformações ou mudança de escala do alvo;
- imagem borrada ou ruidosa;
- baixa resolução das imagens;

- fundo com aspecto semelhante ao alvo.

1.2 A Contribuições do Trabalho

1.2.1 A Hipótese

O emprego das técnicas de aprimoramento dos algoritmos SI em problemas de otimização dinâmica, aplicadas ao rastreamento de alvos em vídeos, melhora o desempenho dos rastreadores quando comparados a outros rastreadores baseados em algoritmos SI sem aprimoramento.

1.2.2 Os Objetivos

O objetivo geral deste trabalho é propor, implementar, analisar e discutir uma abordagem de aprimoramento de um algoritmo SI, em problemas de otimização dinâmica, aplicada em rastreamento de alvo em vídeos. A abordagem proposta promove um balanço entre a transferência de conhecimento (partículas de um *frame* para o *frame* seguinte) e a manutenção da diversidade das partículas. Esta proposta de aprimoramento é uma contribuição tanto em rastreamento de alvos em vídeos quanto e em otimização dinâmica.

Este trabalho propõe o emprego de um rastreador da categoria 2 baseado no algoritmo SI *Shuffled Frog Leaping Algorithm* (SFLA – significa: "algoritmo saltos embaralhados de sapos") (Eusuff, Lansey e Pasha, 2006). Este algoritmo é apropriado para explorar espaços discretos de soluções e é mais poderoso para resolver problemas de otimização combinatórias complexas (Tao et al., 2019), como é o caso do rastreamento de alvos em vídeos.

Além disso, em problemas de rastreamento de alvos em vídeos, é comum que o deslocamento do alvo entre dois *frames* consecutivos não seja muito grande na maioria das aplicações. Esta característica pode ser explorada de forma vantajosa limitando, no *frame* corrente, uma região no espaço de busca evitando alguns mínimos locais e acelerando o processo de convergência do algoritmo. Portanto, este trabalho também propõe a delimitação do espaço de solução em uma região promissora da imagem. A delimitação proposta determina uma região, ao redor da previsão da posição do alvo no *frame* corrente, via modelo de Alisamento Exponencial Duplo (AED) (Winters, 1960), numa versão robusta a valores discrepantes (AEDR). Vale mencionar que o uso do modelo de previsão no processo de delimitação do espaço de solução num problema de otimização SI é inédito.

Em síntese, este trabalho propõe, implementa, analisa e discute uma versão dinâmica aprimorada do algoritmo SFLA, chamada de DSFLA, aplicada ao rastreamento de alvos em vídeos, com espaço discreto de busca delimitado numa região promissora próxima à provável localização do alvo no *frame* corrente via modelos AEDR.

Os objetivos específicos são:

- avaliar a qualidade de rastreamento do DSFLA em termos do tempo médio de processamento por *frame* e em termos da precisão da detecção do alvo medida pela área sob a curva da taxa de sucesso por métrica de Pascal (Everingham et al, 2010);
- comparar a qualidade do rastreamento do DSFLA com outros rastreadores clássicos e recentes cujo mecanismo de rastreamento é baseado em algoritmos SI;
- verificar a contribuição da delimitação do espaço de busca em função das previsões da posição do alvo via modelo de AEDR e analisar a qualidade das previsões.

1.2.3 As Contribuições do Trabalho e a Lista de Publicações

Resumidamente, as inovações e contribuições apresentadas neste trabalho são:

- Um novo algoritmo SI aprimorado para problemas de otimização dinâmica;
- Um novo rastreador de alvos em vídeo;
- Um algoritmo adaptado, apropriado e mais eficiente, para problemas de otimização em espaço discreto de busca;
- Um método adaptativo novo de transferência de conhecimento entre dois ambientes de otimização dinâmica aprimorada;
- Um método novo de delimitação do espaço de soluções em função das previsões dada por um modelo eficiente de séries temporais não paramétrico;
- Um caso de estudo na área de rastreamento de alvos em vídeos mostrando resultados compatíveis com modelos estado-de-arte baseados em algoritmo SI;
- Um rastreador de propósito geral, rápido e estável no rastreamento de um único alvo, em especial, robusto às rotações e/ou movimentos rápidos do alvo (ou da câmera) ou em imagens não nítidas ou ruidosas.

Além disso, durante o desenvolvimento desta tese foram geradas três publicações, duas em forma de artigo completo publicado em congresso; e, uma em forma de artigo em revista internacional. As referências são, respectivamente:

- E. C. CASTRO; SALLES, E. O. T.; CIARELLI, P. M. **Rastreamento de Alvos em Vídeo via Algoritmo Shuffled Frog Leaping Optimization.** XXII Congresso Brasileiro de Automática (CBA2018) – João Pessoa/PB, **2018**.
- E. C. CASTRO; SALLES, E. O. T.; CIARELLI, P. M. **Swarm Strategy and Color Probability Density for Visual *Tarcking*.** 4th Brazilian Tecnology Symposium – (BTSym2018) – Campinas/SP, Vol 1, 8910 – 8914, **2018**.
- E. C. CASTRO; SALLES, E. O. T.; CIARELLI, P. M. **A New Approach to Enhanced Swarm Intelligence Applied to Video Target Tracking.** Sensors, **2021**, 21, 1903. https://doi.org/10.3390/s21051903.

1.3 A Estrutura do Trabalho

Este trabalho está estruturado da seguinte forma:

- o Capítulo 2 é dedicado à revisão da literatura;
- o Capítulo 3, de teorias e métodos, trata, na Seção 3.1, de alguns conceitos comuns utilizados no rastreamento de alvos em vídeos; na Seção 3.2, trata da otimização por inteligência de enxames e, na Seção 3.3, trata dos algoritmos SI utilizados neste trabalho: PSO, ADSO, SSA e SFLA, em quatro subseções;
- o Capítulo 4 discorre sobre a metodologia proposta neste trabalho. A Seção 4.1 trata das definições de forma e aparência dos alvos e da medida de similaridade entre os alvos candidatos e o *template*. A Seção 4.2 trata dos problemas de otimização dinâmica; na Seção 4.3 o modelo proposto é explicado em duas subseções: na Subseção 4.3.1 é explicado o modelo AEDR; e, na Subseção 4.3.2, o modelo DSFLA é apresentado em detalhes;
- o Capítulo 5 apresenta, na Seção 5.1, o delineamento do experimento e as métricas de desempenho dos rastreadores usadas neste trabalho; na Seção 5.2, a escolha dos parâmetros dos modelos utilizados; na Seção 5.3, a análise dos resultados experimentais; e, na Seção 5.4, uma comparação dos resultados do DSFLA com outros rastreadores via desafio OPE (*One-Pass Evaluation* – significa: "Avaliação a uma passagem") (Wu, Lim e Yang, 2013); e,
- as conclusões são apresentadas no Capítulo 6.

2 *Revisão da Literatura*

O algoritmo KF (Kalman, 1960) modela os fenômenos dinâmicos estocásticos gerando boas soluções com bastante eficiência em termos de tempo de processamento. No entanto, o KF depende fortemente das hipóteses de linearidade do modelo de transição e dos ruídos aleatórios gaussianos. Porém, estas hipóteses são dificilmente atendidas em problemas de rastreamento de alvos em vídeos.

O algoritmo PF (Gordon, Salmond e Smith, 1993) é um algoritmo alternativo ao KF quando as hipóteses de linearidade e de ruídos aleatórios gaussianos não são plausíveis.

Uma partícula, em PF, é uma unidade amostral simulada de uma distribuição de probabilidade. Nos rastreadores de alvos baseados no algoritmo PF, os alvos candidatos se deslocam de uma posição para a outra de acordo com uma lei probabilística de movimento. Cada alvo candidato é associado a um peso simulado de uma função de densidades de probabilidades chamada de distribuição de importância. Na abordagem Bayesiana, a distribuição *a posteriori* do estado do alvo, obtida pela conjugação da distribuição *a priori* e da função de verossimilhança dada pela distribuição de importância, é estimada pelos pares (candidatos, pesos). A média ponderada *a posteriori* das partículas é um estimador pontual do atual estado do objeto e, no rastreamento de alvos em vídeos, indica a posição provável do alvo no *frame* corrente.

Todavia, o PF apresenta algumas desvantagens:

- há o problema da degeneração das partículas que consiste na perda sucessiva e constante de importância das partículas gerando a necessidade de regenerá-las (ou renová-las). Em geral, isto é feito substituindo as partículas de menor importância por aquelas de maior importância de acordo com seus pesos;
- há o problema do empobrecimento das partículas que consiste na falta de diversidade das partículas quando elas são regeneradas (problema de medidas repetidas); e,
- há o problema de custo computacional: uma vez que as tarefas de regeneração e diversificação das partículas são realizadas com frequência consumindo muita memória e tempo de processamento.

Morelande, Challa e Gordon (2004) foram um dos primeiros a propor um rastreador de alvos em vídeos digitais baseados no algoritmo PF.

O rastreador MS proposto por Comaniciu, Ramesh e Meer (2003) é um modelo clássico da categoria 2. Os alvos modelo e candidatos, em MS, são representados por densidades de probabilidades, no espaço de cor, localizados em uma posição do *frame* e são estimados por

histogramas de cor via kernels. A distância de Bhattacharyya (Kailath, 1967) mede a similaridade entre os histogramas e é representada por uma superfície espacialmente suave, devido ao uso do Kernel. Esta superfície desempenha o papel da função de verossimilhança. A posição de máxima verossimilhança indica, portanto, a provável posição do alvo no *frame* corrente. Comaniciu, Ramesh e Meer (2003) adotaram o procedimento de otimização baseado no método de otimização dos gradientes ascendentes. É importante mencionar que os resultados produzidos em Comaniciu, Ramesh e Meer (2003) não foram comparados com os resultados de outros rastreadores, porém, o rastreador MS se tornou uma referência em rastreamento de alvos em vídeos a partir de sua publicação.

Os modelos de previsão por alisamento exponencial foram originalmente propostos por Holt e Winters na década de 1960 (Winters, 1960) e os detalhes teóricos dessa modelagem foram amplamente discutidos em Brown e Meyer (1961). Em Gardner (1998) e Gardner (2006) estão disponíveis revisões atualizadas sobre várias técnicas e versões distintas desta modelagem.

Snyder, Koehler e Ord (2002) aplicaram o modelo de alisamento exponencial em controle de estoque, Shung e Kim (2013) aplicaram na investigação da compressão de sinais elétricos em dispositivos eletrônicos e Oliveira e Oliveira (2018) aplicaram o método para previsão de consumo de energia elétrica em grandes metrópoles.

Kennedy e Eberhart (1995) propuseram o algoritmo de *Particle Swarm Optimization* (PSO – significa: "otimização por enxames de partículas"), o primeiro algoritmo a empregar a metodologia SI. O PSO imita o comportamento social de um "enxame" de pássaros (as partículas) que trocam informações entre si e com o meio. O mecanismo de troca de informação é chamado de processo de aprendizagem das partículas que se realiza de duas maneiras:

- o aprendizado cognitivo: é o aprendizado baseado na experiência passada da própria partícula;
- o aprendizado social: é o aprendizado baseado na experiência coletiva de todo o enxame.

Uma função objetivo associa um valor de qualidade para cada partícula. A cada iteração do algoritmo, a posição e a velocidade de cada partícula são alteradas e a qualidade delas são atualizadas. A posição de melhor qualidade experimentada por cada partícula do enxame até a iteração corrente é memorizada (aprendizado cognitivo), assim como a posição de melhor qualidade experimentada pelo enxame até a iteração corrente (aprendizado social). O algoritmo continua até alcançar uma condição de parada, situação em que o PSO retorna o resultado da melhor partícula do enxame indicando a solução ótima. Em termos de rastreamento de alvos

em vídeos, a melhor partícula do enxame representa a provável posição do alvo no *frame* corrente.

Canalis, Tejera e Nielsen (2006) foram um dos primeiros a aplicar um algoritmo SI baseado no PSO (Kennedy e Eberhart, 1995) em um problema de rastreamento de alvos em vídeos, obtendo resultados promissores e compatíveis aos tradicionais rastreadores MS e PF.

Tawab, Abdelhalim e Habib (2010) propuseram uma versão melhorada PSO cujos resultados superaram os resultados em Canalis, Tejera e Nielsen (2006). Porém, ambas as abordagens usaram o algoritmo PSO em ambientes de otimização dinâmica não aprimorados e mantiveram o espaço de busca toda a imagem do *frame* sem delimitações.

Em Gao, Yin et al. (2015) foi proposto um rastreador da Categoria 2 baseado no algoritmo SI *Cuckoo Search* (CS – significa: "busca cuco") (Yang e Deb, 2009). O algoritmo CS mimetiza o comportamento predatório do pássaro cuco em relação à postura dos ovos. Gao, Yin et al. (2015) usaram seis vídeos desafiadores[2] e a distância de Bhattacharyya (Kailath, 1967) entre os histogramas de cor para medir a similaridade entre os alvos candidatos e o real. O rastreador CS foi comparado com os rastreadores PF, MS, PSO e mais quatro outros rastreadores baseados em versões do algoritmo PSO. Os resultados mostraram que o rastreador CS superou os demais em termos de tempo de processamento e de qualidade do rastreamento via distância Euclidiana entre os pontos centrais dos alvos estimados e o real. Porém, para o espaço de busca do algoritmo CS, foi considerada toda a imagem do *frame* em um problema de otimização dinâmica não aprimorado.

Bae et al. (2016) apresentaram um rastreador de alvos em vídeos denominado *Adaptive Swarm Discrete Optimization* (ADSO – significa: "otimização discreta adaptativa por enxame").

O algoritmo PSO foi originalmente projetado para trabalhar em problemas de otimização estacionária com espaço de solução contínuos, portanto, quando o PSO é aplicado em ambiente de otimização com espaço discreto de soluções a chance de o algoritmo convergir prematuramente para um ótimo local é maior. O ADSO é uma versão do algoritmo PSO adaptada para espaço discreto de soluções e um dos pioneiros em empregar um algoritmo SI aprimorado em ambientes dinâmicos de otimização aplicado ao rastreamento de alvos em vídeos.

O ADSO trabalha os ambientes de otimização (os *frames*) aproveitando as partículas (os alvos candidatos) de um ambiente de otimização anterior para o atual de acordo com uma

[2] Vídeos desafiadores, neste trabalho, significa: "os vídeos apresentam ao menos um dos fatores de interferência na maioria dos *frames*, como exemplo: alvos não rígidos, articulados, que se deformam ou que se confunde com o fundo ou com outros objetos semelhante na cena; movimentos rápidos e imagens não nítidas, podendo ou não haver oclusões e/ou rotações diversas do alvo, mudanças bruscas de escala dos alvos e/ou de luz, etc".

função que controla a diversidade das partículas. O algoritmo ADSO atribui um valor a cada dimensão das partículas do enxame, dependendo do grau de oclusão do alvo, em função de três limiares definidos pelo usuário que determinam o valor a ser atribuído. Os valores atribuídos pela função impactam na exploração do espaço de soluções readaptando a estratégia de busca das partículas.

Bae et al. (2016) utilizaram sete vídeos desafiadores da base de dados pública PAMI (http://sites.google.com/site/benchmarkpami) e a distância de Bhattacharyya como medida de similaridade. Os resultados do ADSO superaram os resultados dos rastreadores PSO e outro baseado em algoritmo evolucionário, o Algoritmo Genético (Goldberg e Holland, 1988) em termos de tempo de processamento e de qualidade do rastreamento via distância Euclidiana (em número de *pixels*) dos pontos centrais dos alvos estimados e verdadeiros. Os resultados em Bae et al. (2016) mostraram que o ADSO é um rastreador veloz e eficiente para rastrear alvos que se movimentam rapidamente na cena.

Zang et al. (2020) apresentaram um rastreador baseado no algoritmo SI *Salps Search Algorithm* (SSA – significa: "algoritmo pesquisa por salpas") (Mirjalili et al., 2017). O SSA mimetiza o comportamento de um grupo de salpas nadando e forrageando no oceano profundo. As salpas são um membro da família das *Salpidaes* e possuem um corpo transparente em forma de barril, nadam por propulsão e formam grandes cadeias. A cadeia de salpas (as partículas) é formada por uma salpa líder, que se move em busca de uma fonte de alimento (a solução ótima), e pelas salpas seguidoras, que acompanha o movimento da salpa líder. O movimento da líder é responsável pela exploração global do espaço de soluções e o movimento das seguidoras é responsável pela exploração local. O algoritmo executa L iterações de busca no espaço de soluções e, para cada iteração, o movimento da líder é controlado por uma função que faz um balanço entre a exploração global e local do espaço de soluções.

Zang et al. (2020) usaram 13 vídeos desafiadores de uma base de dados pública (http://www.visual-tracking.net) e o coeficiente de correlação cruzada entre os histogramas de gradientes orientados (HOG – do inglês: *Histogram of Oriented Gradients*) (Dallal e Triggs, 2005) para medir a similaridade entre os alvos candidatos e o real. Os resultados superaram dez outros rastreadores estado-da-arte em termos de velocidade de processamento e de qualidade de desempenho via métrica de Pascal (Everingham et al, 2010). Entretanto, o rastreador SSA opera em ambientes dinâmicos não aprimorados de otimização e não faz delimitação do espaço de soluções.

É importante mencionar que vários autores propuseram o uso de algoritmos de otimização baseados em SI com o objetivo de otimizar o desempenho do algoritmo PF resultando em um rastreador melhor, tanto no tempo de processamento quanto na qualidade do rastreamento (Gao,

Li et al, 2015; Li et al, 2015; Sardari e Moghaddan, 2017). Também, uma grande importância tem sido dada em anos recentes e um grande progresso tem sido alcançado em modelos de rastreadores de alvos em vídeos baseados nos algoritmos de Filtro de Correlação (Li e Zu, 2014; Chen, Hong e Tao, 2015; Zhang et al, 2017), e de *Deep Learning* (significa: "aprendizado profundo") (Held, Thrum e Savarese, 2016; Valmadre et al, 2017; Zhai et al, 2018).

Eusuff, Lansey e Pasha, (2006) propuseram a meta-heurística memética SFLA para resolver problemas de otimização combinatórias sendo, portanto, capaz de explorar espaços discretos de soluções mais eficientemente. Seu mecanismo de exploração do espaço de soluções mimetiza o comportamento de um grupo de sapos (as partículas) em um pântano (espaço de soluções) disputando pelo melhor lugar de alimentação (soluções do problema). Os lugares de disputa são pedras localizadas em pontos (discretos) do pântano que representam os pontos de ótimos locais. O algoritmo gera os sapos virtuais posicionados aleatoriamente em locais discretos do pântano e são agrupados em comunidades de sapos chamados de memeplex. Para cada iteração do algoritmo, os sapos realizam saltos trocando informações com outros sapos dentro e fora dos memeplex em função da qualidade gerada por uma função objetivo. Ao final de cada iteração, os sapos são embaralhados e realocados em novos memeplex. Ao alcançar uma condição de parada, o algoritmo retorna a posição do sapo de maior qualidade do pântano, ou seja, a provável localização do ótimo.

É importante registrar que, apesar do algoritmo SFLA ser utilizado em vários trabalhos aplicados envolvendo otimização de espaços discretos de soluções, tal como Tao et al., (2019) que aplicaram o SFLA em problemas da engenharia civil. Até o momento, não há um rastreador de alvos em vídeos baseado neste algoritmo.

3 Teorias e Métodos

Neste capítulo serão tratados os conceitos de rastreamento de alvos em vídeos e os conceitos de otimização envolvendo aplicações de algoritmos SI. A primeira seção trata do rastreamento de alvos, a segunda seção apresenta a otimização por inteligência de enxames no contexto estacionário seguida de quatro subseções que tratam, respectivamente, dos algoritmos PSO, ADSO, SSA e SFLA.

3.1 Representação dos Alvos em Vídeos

O problema de rastreamento de alvos em vídeos tratado neste trabalho foca na estimação *online* da trajetória de um único alvo *frame* a *frame* de um vídeo.

Neste trabalho, os alvos são representados por um retângulo, sem furos em seu interior, chamados de janelas ou de *bounding boxes* (significa: "caixas delimitadoras") e denotados por um vetor de quatro dimensões $p = (x, y, w, h)$, em que (x, y) é a coordenada 2D do *pixel* localizado no canto superior esquerdo do *bounding box* e (w, h) são as dimensões horizontal e vertical (medidas em número de *pixels*) referentes à largura e à altura do *bounding box*, respectivamente.

Esta representação da forma do alvo é ideal para objetos rígidos, no entanto, ela é flexível o suficiente para representar objetos não rígidos e objetos articulados. É, também, uma representação comumente usada em diversas aplicações de rastreamento de alvos em vídeos, pois, se trata de uma abordagem satisfatoriamente rápida (Yamaz, Javed e Shah, 2006).

3.2 Otimização por Inteligência de Enxame

Em problemas de otimização, uma solução é factível se os valores assumidos para o conjunto de parâmetros (ou variáveis), associado com o problema, satisfizerem às restrições. Uma solução factível é chamada de solução ótima se ela minimizar ou maximizar a função custo (ou função aptidão), $f: S \rightarrow \mathbb{R}$, que associa uma qualidade às soluções. O conjunto de todas as soluções factíveis é chamado de espaço de soluções (ou espaço de busca), $S \subseteq \mathbb{R}^d$, em que d é a dimensão das soluções em S (é igual ao número de parâmetros).

Os algoritmos tradicionais de otimização empregam métodos determinísticos para encontrar a melhor solução. Um exemplo é o algoritmo Força Bruta (ou Busca Exaustiva) que

testa todas as soluções do espaço de busca, portanto, não são adequados para problemas mais complexos em que o tempo de busca cresce exponencialmente com o tamanho do problema.

Algoritmos de busca estocásticos empregam números inicialmente gerados ao acaso para explorar o espaço de soluções em problemas mais complexos, com a vantagem de operar em tempo de busca polinomial em relação ao tamanho do problema. Esses algoritmos trabalham sob a suposição de que boas soluções estão próximas umas das outras. Uma desvantagem desses algoritmos é a possibilidade de falhar em encontrar a solução ótima, mas podem, no entanto, encontrar uma boa solução antes de esgotar o tempo previsto de execução (Lovberg e Krink, 2002). Um exemplo de algoritmo de busca estocástico é o *Simulated Annealing* (significa: "recozimento simulado") (Kirkpatrick e Vecchi, 1983).

Os algoritmos evolucionários são métodos de busca estocásticos, de propósito geral, simulando os processos de seleção natural e da evolução das espécies de Darwin. Estes métodos se diferenciam dos outros por utilizarem uma "população de soluções" que podem trocar informações entre elas e orientar as suas buscas individuais (Engelbrecht, 2002). O exemplo mais conhecido desta classe de algoritmo é o Algoritmo Genético (Goldberg e Holland, 1988).

Os algoritmos SI mimetizam a inteligência originada do comportamento coletivo de um grupo de seres, tais como: colônia de formigas, bando de pássaros, ou de sapos, ou de lobos, etc. São exemplos de algoritmos SI: O PSO (Kennedy e Eberhart, 1995), o SFLA (Eusuff, Lansey e Pasha, 2006), o SSA (Mirjaliliet al., 2017), o Colônia de Formigas (Dorigo, Birattari e Stutzle, 2006), o Colônia de Abelhas Artificiais (Basturk e Karaboga, 2006), o *Firefly Algorithm* (significa: "algoritmo vagalume") (Yang, 2009), o CS (Yang e Deb, 2009), o *Grey Wolf Optimization* (significa: "otimização lobo cinzento") (Mirjalili, Mirjalilie Lewis, 2017). Em Mirjalili, Mirjalili e Lewis (2017) há uma coleção atualizada de algoritmos SI.

Os algoritmos evolucionários e os algoritmos SI representam duas importantes categorias de métodos de otimização inspirados na natureza. Ambos algoritmos são também chamados de meta-heurísticas.

A principal diferença entre as abordagens SI, a parte da fonte de inspiração, é a maneira como o espaço de soluções é explorado local e globalmente pelos agentes (Mavrovouniotis et al, 2017).

Segundo Mirjalili et al. (2017), as vantagens que tornam os algoritmos SI tão populares são:

- a simplicidade;
- a flexibilidade dos algoritmos;
- o emprego de mecanismos livres de derivadas;
- o emprego de mecanismos que evitam ótimos locais; e,

- a proposição de boas soluções para problemas reais.

O interesse na busca por novas meta-heurísticas é explicado pelo teorema *No-Free Lunch* (significa: "sem almoço grátis") de Wolpert e Macready (1998). O teorema estabelece que não há uma meta-heurística capaz de produzir um resultado ótimo para todo problema de otimização, portanto, sempre haverá interesse no desenvolvimento de novos e específicos algoritmos para diferentes campos de aplicação.

Resumidamente, os rastreadores pertencentes à categoria 2 baseados em algoritmos SI trabalham um problema de otimização da seguinte maneira: uma vez dados os *frames* de um vídeo e os *templates*, o algoritmo espalha, inicialmente, N partículas (os agentes) ao acaso dentro do espaço de soluções. Em rastreamento de alvos em vídeos, as partículas são os candidatos a alvo (representados por um *bounding box*) e o espaço de soluções é o *frame*, dado por

$$\boldsymbol{S} = \{\boldsymbol{p} \in \mathbb{Z}_{+}^{\boldsymbol{4}} | \boldsymbol{1} \leq \boldsymbol{x}, \boldsymbol{w} \leq \boldsymbol{R}, \boldsymbol{1} \leq \boldsymbol{y}, \boldsymbol{h} \leq \boldsymbol{Q}\}, \quad (1)$$

em que R e Q são, respectivamente, o total de *pixels* das dimensões horizontal e vertical do *frame* (neste trabalho, parte-se da hipótese de que os *frames* de um mesmo vídeo possuem as mesmas dimensões).

Em seguida, a meta-heurística desloca as partículas, dentro do espaço de soluções, e uma medida de qualidade é atribuída via função custo. Ao atingir uma condição de parada (pode ser, por exemplo, um número máximo de iterações ou um valor mínimo de qualidade da melhor solução) o algoritmo retorna o valor da partícula $p = (x, y, w, h)$ de melhor qualidade do enxame denotado por $gBest = (x_g, y_g, w_g, h_g)$, ou seja, o $gBest$ é o ótimo global (ou um ótimo local). Nos problemas de rastreamento de alvos em vídeo, o $gBest$ indica a provável posição (e dimensão) do alvo no *frame* corrente.

Nas próximas seções serão discutidas com mais detalhes os algoritmos empregados neste trabalho: os algoritmos PSO, ADSO, SSA e SFLA, respectivamente.

3.2.1 O Algoritmo PSO

O algoritmo PSO (Kennedy e Eberhart, 1995) inicia o processo de busca pela solução ótima gerando N partículas ao acaso pertencentes ao espaço de busca (o enxame de pássaros). Em rastreamento de alvos, cada partícula, denotada pelo vetor $p = (x, y, w, h)$, representa um *bounding box* associado a um alvo candidato. Em seguida, uma função objetivo determina uma medida de qualidade da solução para cada uma das partículas.

O algoritmo movimenta as partículas, em cada iteração dentro do espaço de soluções, atribuindo novos valores para a posição e velocidade. Em seguida, o algoritmo atualiza o valor

da aptidão das partículas. A posição de maior aptidão de cada partícula é atualizada, o $pBest$, assim como a posição de maior aptidão do enxame, o $gBest$. O processo se repete até o algoritmo alcançar uma condição de parada.

Quando a condição de parada do algoritmo é alcançada, a posição da partícula de melhor aptidão, $gBest$, é considerada a melhor solução do problema. No caso de rastreamento de alvos, é o melhor alvo candidato e a sua posição indica a posição do alvo no *frame* corrente.

A condição de parada é dependente do problema e pode ser um número pré-fixado de iterações, ou um valor mínimo de qualidade da melhor solução, ou se o $gBest$ não muda de valor após um número de iterações ou se a velocidade das partículas está próxima de zero.

Há na literatura várias versões do PSO. A versão usada neste trabalho foi a *Bare Bones* PSO (BBPSO – significa: "ossos nus" ou "esqueleto") (Kennedy, 2003), também conhecida como PSO Gaussiano. A escolha desta versão se deve ao fato do BBPSO apresentar somente dois parâmetros definidos pelo usuário: o número N de partículas e a topologia de vizinhança que afeta o modo de propagação da informação dentro do enxame.

Há dois tipos de topologia de vizinhança:

- a topologia global em que todas as partículas do enxame se comunicam; e,
- a topologia local em que uma partícula comunica com um grupo de partículas (Kennedy, 1999).

Neste trabalho foi adotada a topologia local, uma vez que, o esquema de vizinhança tem a vantagem de vários grupos de partículas promoverem buscas independentes em diferentes subespaços do espaço de solução ampliando as chances de encontrar o ótimo global.

Outra diferença é que o BBPSO utiliza a função de densidade de probabilidade Gaussiana para atualizar a posição das partículas ao invés de adicionar uma velocidade às partículas (como é feito nas versões clássicas do PSO). A equação de atualização da posição das partículas no BBPSO é dada por

$$\boldsymbol{p_i^\tau = \mu_i^\tau + \sigma_i^\tau \otimes Z} \tag{2}$$

com

$$\boldsymbol{\mu_i^\tau = \frac{pBest_i^{\tau-1} + gBest^{\tau-1}}{2}}, \tag{3}$$

$$\boldsymbol{\sigma_i^\tau = \left| pBest_i^{\tau-1} - gBest^{\tau-1} \right|}, \tag{4}$$

em que p_i^τ é a partícula p_i de dimensão d na iteração τ; μ_i^τ é a média do movimento da partícula p_i na iteração τ; σ_i^τ é o desvio-padrão do movimento da partícula p_i na iteração τ; $\otimes$ é o produto componente por componente entre dois vetores de mesma dimensão; $pBest_i^{\tau-1}$ é o $pBest$ da partícula p_i na iteração $\tau - 1$; $gBest^{\tau-1}$ é o $gBest$ na iteração $\tau - 1$, $\boldsymbol{Z}$ é uma variável aleatória

de dimensão d distribuída como uma Gaussiana multivariada com vetor de média $\mathbf{0}$ e matriz de variâncias e covariâncias igual a matriz diagonal de ordem $d, \boldsymbol{\Sigma_d}; \tau = 1,2,\ldots,n_\tau, i = 1,2,\ldots,N$ em que n_τ é o número de iterações do algoritmo e N é o número de partículas; e, $|\cdot|$ representa o valor absoluto do número.

O pseudocódigo, propriedades de convergência do BBPSO e outros detalhes podem ser consultados em Kennedy (2003).

3.2.2 O Algoritmo ADSO

Bae et al. (2016) propõem o ADSO com o objetivo de contornar a dificuldade do algoritmo PSO em ambientes de otimização com espaço discreto de soluções. O rastreador de alvos em vídeos baseado na meta-heurística ADSO é mais rápido em processar os *frames* e é mais preciso no rastreamento quando comparados aos rastreadores baseados no Algoritmo Genético e na meta-heurística PSO.

O algoritmo ADSO possui uma função de probabilidade adaptativa[3], H_f, para cada partícula em função de sua aptidão. A finalidade de H_f é acomodar três diferentes situações associadas ao grau de oclusão dos alvos candidatos (as partículas). Os graus de oclusão das partículas são: oclusão severa, oclusão parcial e oclusão desprezível. H_f pode ser interpretada como a probabilidade de não renovação da partícula, ou seja, a probabilidade da partícula não receber novos valores ao acaso dentro do espaço de soluções. Neste caso, a partícula recebe valores da partícula de melhor aptidão do grupo.

De acordo com Bae et al. (2016), no caso de uma partícula p_i ter uma aptidão muito baixa é sinal de uma oclusão severa e, portanto, a probabilidade de não renovação da partícula deve ser nula, ou seja, a partícula deve ser renovada. De modo análogo, se uma partícula tem grande aptidão, indica uma oclusão desprezível e, portanto, a probabilidade de não renovação deve ser mantida. Em casos de oclusão parcial, a probabilidade de não renovação deve ser menor aumentando a chances de renovação.

Os graus de oclusão são atribuídos pelo usuário de acordo com três limiares previamente definidos th_1, th_2 e th_3 $(0 < th_1 < th_2 < th_3 < 1)$, os quais são dependentes do problema.

A função H_f assume as seguintes probabilidades para cada partícula

$$\boldsymbol{H_f^{\tau}(p_i)} = \begin{cases} \mathbf{0}, & \boldsymbol{f(p_i) < th_1} \\ \boldsymbol{th_1 . H_f^{\tau-1}(p_i)}, & \boldsymbol{th_1 \leq f(p_i) \leq th_2}, \\ \boldsymbol{H_f^{\tau-1}(p_i)}, & \boldsymbol{th_2 > f(p_i)} \end{cases} \tag{5}$$

[3] É uma função de probabilidade cuja distribuição pode mudar a cada ambiente novo de otimização (para cada partícula).

em que $H_f^{\tau}(p_i)$ é uma função H_f da partícula p_i na iteração τ e $f(p_i)$ é a função de aptidão da partícula p_i ($\tau = 1,2,\ldots,n_\tau$ e $i = 1,2,\ldots,N$). O limiar th_3 sinaliza uma detecção do alvo, isto é, se $f(p_i) > th_3$, então p_i é considerado o alvo estimado e o algoritmo segue para a próxima iteração.

Quando $\tau = 1$, é necessário gerar ao acaso N partículas dentro do espaço de soluções e iniciar a função H_f com uma distribuição inicial $H_f^0(p_i)$. Neste caso, Bae et al. (2016) sugerem $H_f^0(p_i) = 0{,}7, \forall i$.

O algoritmo movimenta as N partículas, dentro do espaço de soluções, atribuindo um valor para a variável j ($j = 1,2,\ldots,d$) de cada partícula p_i na iteração τ da seguinte maneira:

$$\boldsymbol{p_{i,j}^{\tau}} = \begin{cases} \boldsymbol{gBest_j^{\tau}}, & \boldsymbol{U < H_f^{\tau}(p_i)} \\ \boldsymbol{novo(p_{i,j})}, & \boldsymbol{U \geq H_f^{\tau}(p_i)} \end{cases}, \tag{6}$$

em que $p_{i,j}^{\tau}$ é a variável j da partícula p_i na iteração τ; $gBest_j^{\tau}$ é a variável j da melhor partícula do enxame na iteração τ; U é um número pseudoaleatório gerado uniformemente no intervalo contínuo unitário e $novo(p_{i,j})$ é a ação de substituir a variável j da partícula p_i por um valor aleatório dentro dos limites da variável j ($j = 1,2,\ldots,d$, $i = 1,2,\ldots,N$ e $\tau = 1,2,\ldots,n_\tau$) no espaço de solução.

O algoritmo prossegue atualizando os valores da função aptidão de cada partícula e atualiza, no final de cada iteração, o valor da partícula de melhor aptidão, $gBest^{\tau}$. O algoritmo termina quando encontrar uma partícula com aptidão superior ou igual a th_3 ou alcançar a última iteração $\tau = n_\tau$.

Para mais detalhes do algoritmo ADSO e o seu pseudocódigo, consulte Bae et al. (2016).

3.2.3 O Algoritmo SSA

O algoritmo SSA simula o comportamento de um grupo da Salpas (as partículas), formando uma grande cadeia, nadando nas profundezas do oceano (o espaço de soluções) e que se movimenta em busca de uma fonte de alimento (a solução ótima). As salpas são classificadas em uma líder e várias seguidoras. Zhang et al. (2020) consideram a possibilidade de haver mais de uma salpa líder (assim foi considerado neste trabalho).

O algoritmo espalha ao acaso N partículas de dimensão d dentro do espaço de soluções. O algoritmo atualiza as posições das salpas líderes pela equação

$$\boldsymbol{p_i^{\tau}} = \begin{cases} \boldsymbol{F^{\tau-1} + c_1(c_2(L_s - L_i) + L_s)}, & \boldsymbol{c_3 \leq 0,5} \\ \boldsymbol{F^{\tau-1} - c_1(c_2(L_s - L_i) + L_s)}, & \boldsymbol{c_3 > 0,5} \end{cases}, \tag{7}$$

em que p_i^τ é a salpa líder i na iteração τ; $F^{\tau-1}$ é a fonte de alimento na última iteração; L_s e L_i são, respectivamente, os limites superiores e inferiores das d variáveis componentes do vetor p_i^τ; c_1, c_2 e c_3 são coeficientes, onde c_2 e c_3 são números pseudoaleatórios uniformemente gerados no intervalo contínuo unitário, definindo uma direção e um tamanho do deslocamento da partícula dentro do espaço de soluções, respectivamente, e c_1 controla o balanço entre a exploração local e global do espaço de soluções ($i = 1,2,\ldots,n_l$, em que n_l é o número de salpas líderes com $1 \leq n_l < N$ e $\tau = 1,2,\ldots,n_\tau$ em que n_τ é o número de iterações do algoritmo).

Para $n_l < i \leq N$ a posição das salpas seguidoras é alterada pela equação

$$\boldsymbol{p_i^\tau = \frac{p_i^{\tau-1} + p_{i-1}^{\tau-1}}{2}}. \tag{8}$$

Segundo Zhang et al. (2020), o coeficiente c_1 varia a cada iteração de acordo com a função

$$\boldsymbol{c_1 = \kappa_1\, exp\left\{-\left(\frac{\kappa_2 \tau}{n_\tau}\right)^2\right\}}, \tag{9}$$

em que os coeficientes κ_1 e κ_2 são constantes predefinidas e dependente do problema que alteram a forma da curva c_1, possibilitando maneiras distintas de explorar localmente e globalmente o espaço de soluções.

Em cada iteração, é calculada a aptidão das partículas e a posição da melhor salpa é atualizada como a posição da fonte de alimento. O processo se repete n_τ vezes quando o algoritmo devolve a posição da salpa de maior aptidão da cadeia. Para outros detalhes e o pseudocódigo do algoritmo SSA, consulte Mirjalili et al. (2017) e Zhang et al. (2020).

3.2.4 O Algoritmo SFLA

A meta-heurística memética SFLA foi proposta por Eusuff, Linsey e Pasha (2006) para resolver problemas de otimização combinatória sendo, portanto, mais eficiente em explorar espaços discretos de soluções.

O algoritmo inicia espalhando N sapos aleatoriamente no pântano (os sapos são partículas de dimensão d dentro do espaço discreto de soluções e o pântano representa o espaço de soluções) e agrupando-os nos memeplex. Os sapos realizam saltos modificando suas posições dentro do espaço de soluções e são influenciados pelo sapo de maior aptidão dentro do memeplex e pelo sapo de maior aptidão do pântano. A posição do sapo de pior aptidão do memeplex é alterada de acordo com

$$\boldsymbol{p_w^\tau = p_w^{\tau-1} + l_w^\tau}, \tag{10}$$

em que $p_w^{\tau-1}$ é a posição do sapo de pior aptidão do memeplex na iteração anterior, p_w^τ é a posição do sapo de pior aptidão do (mesmo) memeplex na iteração atual e l_w^τ é o salto executado

pelo sapo de pior aptidão do memeplex na iteração atual τ ($\tau = 1,2,\ldots,n_\tau$, sendo n_τ o número máximo de iterações da meta-heurística). Os saltos são limitados por uma constante positiva predefinida e dependente do problema l_{Max} ($-l_{Max} \leq l_w^\tau \leq l_{Max}, \forall\tau$).

O cálculo do salto do sapo de pior aptidão do memeplex é feito provisoriamente adicionando um valor aleatório na direção do sapo de melhor aptidão do memeplex da seguinte maneira:

$$\boldsymbol{l_w^\tau = U\left(p_b^{\tau-1} - p_w^{\tau-1}\right)}, \tag{11}$$

em que $p_b^{\tau-1}$ refere-se à posição do sapo de melhor aptidão do memeplex na iteração $\tau - 1$ e U é um número pseudoaleatório uniformemente distribuído no intervalo unitário contínuo.

O valor calculado provisoriamente pela Equação (11) será aceito se e somente se estiver dentro dos limites estabelecidos para o tamanho do salto e se a nova posição p_w^τ melhorar a aptidão do sapo de pior aptidão, senão, l_w^τ será descartada e receberá um outro valor provisório calculado da seguinte maneira

$$\boldsymbol{l_w^\tau = U\left(p_g^{\tau-1} - p_w^{\tau-1}\right)}, \tag{12}$$

em que $p_g^{\tau-1}$ refere-se à posição do sapo de melhor aptidão do pântano na iteração $\tau - 1$.

O valor calculado provisoriamente pela Equação (12) será aceito se e somente se estiver dentro dos limites estabelecidos para o tamanho do salto e se a nova posição p_w^τ melhorar a aptidão do sapo de pior aptidão, senão, l_w^τ será descartada e receberá um valor aleatório definitivo dentro dos limites de cada variável, ou seja, o sapo de pior aptidão será substituído por um novo sapo virtual gerado ao acaso dentro do espaço de soluções.

Após realizar os saltos para os sapos de pior aptidão de todos os memeplex, os sapos são redistribuídos aleatoriamente nos memeplex e o procedimento é repetido n_τ vezes (n_τ é predefinido e dependente do problema).

O algoritmo realiza simultaneamente uma exploração local independente, dentro do espaço de soluções, por cada memeplex. A exploração global do espaço de busca é garantida através do embaralhamento dos sapos e pela reorganização dos memeplex. O algoritmo também garante a geração aleatória de sapos virtuais aumentando a oportunidade de gerar novas informações.

As principais vantagens do SFLA são que ele é um algoritmo mais poderoso em resolver problemas de otimização combinatória complexa, possui uma capacidade de busca mais rápida e é mais robusto em determinar o ótimo global por causa da evolução dos vários memeplex (estruturas responsáveis pela exploração local do espaço de soluções) e por causa do processo de embaralhamento (responsável pela exploração global do espaço de soluções) que acrescentam qualidade aos indivíduos (Tao, Li et al., 2019).

Para mais detalhes sobre a meta-heurística SFLA e o seu pseudocódigo consulte (Eusuff, Linsey e Pasha, 2006).

4 A Metodologia da Proposta

Neste capítulo, os métodos e modelos propostos serão apresentados e analisados em três seções: a Seção 4.1 que apresenta os detalhes metodológicos empregados nas definições dos alvos e na medida de similaridade; a Seção 4.2 que apresenta a técnica de otimização em ambientes dinâmicos; a Seção 4.3 que trata do modelo proposto em duas subseções: a Subseção 4.3.1 que fala dos modelos não-paramétrico de séries temporais para previsão utilizando a técnica de Alisamento Exponencial Duplo Robusto aos valores discrepantes (o AEDR); e, a Subseção 4.3.2 que apresenta o modelo DSFLA em detalhes.

4.1 As Definições das Características dos Alvos e da Medida de Similaridade

A representação da aparência e a característica extraída dos alvos adotadas neste trabalho são os histogramas padronizados do primeiro canal do modelo de cor YCbCr. O espaço de cor YCbCr possui um componente de luminância (Y) e dois componentes de crominância (crominância azul, Cb, e crominância vermelha, Cr) e é amplamente usado em imagens digitais e processamento de vídeos (Equasys, 2013). A vantagem deste modelo de cor sobre o modelo RGB é que o YCbCr leva em consideração a percepção visual humana em que os olhos são mais sensíveis à intensidade do que a cor permitindo reduzir as informações de crominância sem perdas significativas de qualidade da imagem colorida digital (Equasys, 2013).

É possível utilizar outras características para representar os alvos, tais como textura ou formato (veja outras alternativas em Yamaz, Javed e Shah, (2006)). Entretanto, o foco deste trabalho não é a representação dos alvos em si, mas é a avaliação, sob as mesmas condições, da capacidade dos algoritmos baseados em SI conseguirem rastrear os alvos.

Pela experiência adquirida em etapas anteriores desta pesquisa, o ganho em qualidade do rastreamento com a inclusão do segundo e do terceiro canais do modelo de cor YCbCr são quase nulos (em outros termos, não acrescenta poder de discriminação dos alvos) ao custo de um gasto maior de tempo de processamento. Este efeito pode ser explicado, ao menos em parte, pela pequena variação dos níveis de crominância dos canais Cb e Cr e pelas altas correlações entre os canais Y e Cb e entre os canais Y e Cr (Equasys, 2013).

Um histograma padronizado é o histograma de área unitária e é um estimador assintoticamente não enviesado e consistente da função densidade de probabilidade (Martinez e Martinez, 2002). A escolha do histograma padronizado de cor é devida à sua capacidade de

invariância a rotações e a mudanças de escala (Yang et al., 2011); às suas propriedades estatísticas assintóticas; e, à sua rapidez no processamento de extração das características dos alvos.

A alternativa adotada neste trabalho para estimar a densidade via histograma foi pela regra de Struges (Martinez e Martinez, 2002), devido à sua simplicidade de cálculo. Esta regra consiste em estimar o número de *bins*[4], n_b, de acordo com o estimador

$$\widehat{\boldsymbol{n}}_{\boldsymbol{b}} = \boldsymbol{Max}(\mathbf{4}, \lceil \mathbf{1} + \mathbf{3.32}\, \boldsymbol{log}(\boldsymbol{w_T h_T}) \rceil), \tag{13}$$

na qual é usado o logaritmo na base 10; (w_T, h_T) são, respectivamente, as dimensões horizontal e vertical do *template*, sendo o produto $w_T h_T$, o total de *pixels* do *template* (a dimensão amostral); e, a função $\lceil \cdot \rceil$ retorna o menor número inteiro maior ou igual ao número real "·". Portanto, a amplitude dos *bins*, h_b, é estimada por:

$$\widehat{\boldsymbol{h}}_{\boldsymbol{b}} = \frac{\boldsymbol{R_s}}{\widehat{\boldsymbol{n}}_{\boldsymbol{b}}}, \tag{14}$$

em que R_s é a amplitude amostral (a diferença entre o maior e o menor valores observados na amostra).

A medida de similaridade usada neste trabalho é a distância de Bhattacharyya (Kailath, 1967). A distância de Bhattacharyya, denotada por $\beta(H_T^*, H_P^*)$, é uma medida relativa de distância entre dois histogramas padronizados limitada pelo intervalo unitário contínuo. Quanto mais próximo de zero é a distância de Bhattacharyya maior é a similaridade entre os histogramas (indicando maior similaridade entre o alvo candidato e o *template*). A distância de Bhattacharyya é definida por

$$\boldsymbol{\beta}(\boldsymbol{H_T^*}, \boldsymbol{H_P^*}) = \sqrt{\mathbf{1} - \sum_{\boldsymbol{b}=\mathbf{1}}^{\boldsymbol{n_b}} \sqrt{\boldsymbol{H_T^*(b) H_P^*(b)}}}, \tag{15}$$

na qual H_P^* e H_T^* são os histogramas padronizados do alvo candidato e do *template*, respectivamente, e n_b é o número de *bins* dos histogramas.

Para outras propriedades da distância de Bhattacharyya consulte Kailath, (1967) e Comaniciu, Ramesh e Meer (2003).

4.2 Problemas de Otimização Dinâmica

Um problema de otimização em que os parâmetros, o espaço de soluções, as restrições e a função objetivo não mudam durante o processo é chamado de otimização estacionária. Entretanto, em várias situações do mundo real, em particular no rastreamento de alvos em

[4] *bins* são os intervalos de um histograma.

vídeos, os problemas de otimização estão sujeitos à ambientes dinâmicos em que a solução ótima pode sofrer mudanças de posição dentro do espaço de soluções durante o processo de otimização, estes casos são chamados de problemas de otimização dinâmica.

Originalmente, as técnicas de SI foram projetadas para problemas de otimização estacionária. A principal característica dos algoritmos de otimização baseados em SI é a convergência rápida para a solução ótima (ou de uma solução próxima do ótimo). Entretanto, essa característica é responsável por limitações quando aplicada em problemas de otimização dinâmica, em particular no rastreamento de alvos em vídeo, em que a posição do alvo muda a cada *frame* devido à interferência dos fatores citados na Seção 1.1. Uma forma de adaptar os algoritmos SI para trabalhar em problemas de otimização dinâmica é considerar cada mudança de cenário do processo de otimização em um novo ambiente de otimização estacionária.

Segundo Mavrovouniotis et al. (2017), um problema de otimização dinâmica pode ser definido e formalmente descrito da seguinte maneira:

A otimização dinâmica consiste em otimizar $f(p,t)$ sujeito a

$$\boldsymbol{P(t) \subseteq S, t \in T}, \tag{16}$$

no qual $P(t)$ é um subconjunto de S indexado por t, S é o espaço de soluções, t é o tempo, e

$$\boldsymbol{f: T \times S \rightarrow \mathbb{R}} \tag{17}$$

é a função objetivo, que associa um número real a cada solução factível $\boldsymbol{p \in P(t) \subseteq S, \forall t}$. Uma solução factível $\boldsymbol{p}$ é um vetor de $\boldsymbol{d}$ dimensões, $\boldsymbol{p = (p_1, p_2, \ldots, p_d)}$, em que cada componente corresponde a uma variável (ou parâmetro) do problema de otimização.

Para cada solução factível em $\boldsymbol{P(t)}$ é associado um conjunto de vizinhos $\boldsymbol{N(p) \subseteq P(t)}$.

A solução factível $p' \in N(p)$ é um ótimo local se, e somente se,

$$\boldsymbol{f(p', t) \leq f(p, t), \forall p \in N(p)}, \tag{18}$$

se é um problema de minimização ou

$$\boldsymbol{f(p', t) \geq f(p, t), \forall p \in N(p)}, \tag{19}$$

se é um problema de maximização.

Similarmente, a solução factível $p * \in N(p)$ é um ótimo global se, e somente se,

$$\boldsymbol{f(p *, t) \leq f(p, t), \forall p \in P(t)}, \tag{20}$$

se é um problema de minimização ou

$$\boldsymbol{f(p*,t) \geq f(p,t), \forall p \in P(t),} \quad (21)$$

se é um problema de maximização.

Entretanto, devido à habilidade de convergência dos algoritmos SI, a capacidade deles se adaptarem a um novo cenário de otimização é limitada uma vez que a diversidade das partículas é perdida na convergência. Por outro lado, é possível aproveitar boas soluções, obtidas em ambientes de otimização anteriores, para acelerar a busca do ótimo no ambiente atual de otimização, isto é, promover a transferência de conhecimento. Porém, se muito conhecimento é transferido, o processo de otimização, no ambiente de otimização atual, pode ser iniciado próximo à um local desfavorável e convergir prematuramente para um ótimo local (Mavrovouniotis et al., 2017).

Os algoritmos SI que promovem um balanço entre transferência de conhecimento e diversidade de partículas são chamados de algoritmos SI aprimorados (Mavrovouniotis et al., 2017).

Segundo Mavrovouniotis et al. (2017), há algumas maneiras de promover esse aprimoramento, por exemplo, manter um esquema de memória das melhores partículas de ambientes de otimização anteriores e usá-las no ambiente de otimização atual ou manter múltiplas populações de partículas alocadas em diferentes regiões do espaço de soluções.

Neste trabalho, a técnica de aprimoramento usada foi a seleção de um número máximo predefinido de partículas, iniciando com o ótimo obtido no último ambiente de otimização, mantendo uma distância mínima entre elas (os detalhes serão discutidos na Seção 4.3.2).

4.3 O Modelo Proposto

4.3.1 O Modelo de Alisamento Exponencial Duplo Robusto

Os modelos de Alisamento Exponencial, também chamados de modelos de Holt-Winters (Winters, 1960), trabalham uma série temporal decompondo-a em quatro fatores: nível, tendência (linear), fator sazonal e um fator residual não previsível chamado de ruído aleatório.

O processo de estimação desses fatores é baseado na suavização exponencial, isto é, o processo de suavização consiste na eliminação das variações bruscas presentes na série temporal observada, deixando-a mais suave após descrevê-la em termos dos seus componentes estruturais (os quatro fatores descritos no parágrafo anterior). A estimação desses fatores envolve o cálculo de médias aritméticas ponderadas cujos pesos decaem exponencialmente com o tempo à medida em que se avança no tempo passado da série.

O método de Holt-Winters é bastante utilizado atualmente para obter previsões de curto prazo devido à sua simplicidade de codificação, baixo custo de operação, capacidade de ajustamento automático com atuação rápida em face das mudanças da série produzindo boas previsões em aplicações do mundo real.

Uma série temporal $\{z_t\}_{t\in T}$ é a observação de um processo estocástico de parâmetro discreto $\{Z_t\}_{t\in T}$ em que $\{Z_t\}_{t\in T}$ é constituído por uma sequência de variáveis aleatórias definidas no mesmo espaço amostral e indexadas por t (chamado de parâmetro do processo) e T é o espaço paramétrico (em séries temporais, o espaço paramétrico é discreto e representa o tempo cronológico igualmente espaçado). Os métodos estatísticos de previsão de séries temporais baseiam-se na hipótese do padrão de comportamento futuro da série seja o mesmo do passado, portanto, os dados observados da série, $\{z_t\}_{t\in T}$, possuem informações úteis para prever o seu futuro (Harvey, 1993).

O modelo de Alisamento Exponencial Duplo decompõe o processo $\{Z_t\}_{t\in T}$ em nível, tendência e um termo de erro aleatório de acordo com

$$\boldsymbol{Z_t = \mu_t + M_t + \varepsilon_t, \forall t \in T}, \tag{22}$$

em que Z_t é a variável aleatória do processo $\{Z_t\}_{t\in T}$ no tempo t, μ_t é o fator de alisamento correspondente ao nível no tempo t, M_t é o fator de alisamento correspondente à tendência no tempo t e ε_t é uma variável aleatória com média zero e variância positiva constante, não correlacionada com $\varepsilon_\iota, \forall \iota \neq t$ e não correlacionada com $Z_t, \forall t$.

As estimativas do nível, $\hat{\mu}_t$, e da tendência, $\widehat{M}_t$, são dadas, respectivamente, por

$$\boldsymbol{\hat{\mu}_t = \alpha_3 z_t + (1-\alpha_3)\left[\hat{\mu}_{t-1} + \widehat{M}_{t-1}\right], 0 < \alpha_3 < 1}, \tag{23}$$

$$\boldsymbol{\widehat{M}_t = \alpha_4(\hat{\mu}_t - \hat{\mu}_{t-1}) + (1-\alpha_4)\widehat{M}_{t-1}, 0 < \alpha_4 < 1}, \tag{24}$$

em que os coeficientes α_3 e α_4 são chamados de constantes de suavização (quanto maior é o valor dos coeficientes menor é o peso dado aos valores passados de cada fator); z_t é o valor atual da série observada; $\hat{\mu}_t$ é o valor atual de suavização temporal usado como estimativa do nível; e, $\widehat{M}_t$ é a estimativa da tendência no presente. Quando $t = 1$ é necessário iniciar os valores de $\hat{\mu}_0$ e $\widehat{M}_0$, em geral (mas não necessariamente), $\hat{\mu}_0 = z_1$ e $\widehat{M}_0 \neq 0$ (ou $\widehat{M}_0 = z_2 - z_1$ caso as previsões comecem a partir de $t = 2$), a determinação dos valores iniciais é dependente do problema.

As previsões de horizonte de tempo k a partir do instante t são dadas por

$$\widehat{\boldsymbol{Z}}_{\boldsymbol{t}}(\boldsymbol{k}) = \widehat{\boldsymbol{\mu}}_{\boldsymbol{t}} + \boldsymbol{k}\widehat{\boldsymbol{M}}_{\boldsymbol{t}}, \tag{25}$$

em que $\hat{Z}_t(k)$ é a estimativa (o valor previsto) da variável aleatória Z_{t+k} do processo gerador da série.

A versão do modelo AED usada neste trabalho é uma versão robusta para valores discrepantes, o AEDR. Quando um valor discrepante é observado na série, ele provoca um viés persistente no processo de previsão. Nesta versão, a observação no tempo t, z_t, considerada discrepante é substituída, na Equação (22), pelo valor do limite inferior, LI_t, se $z_t < LI_t$ ou pelo valor do limite superior, LS_t se $z_t > LS_t$. Os valores limites são calculados e atualizados a cada instante no tempo de acordo com as respectivas equações recursivas

$$\boldsymbol{LI_t} = \bar{\boldsymbol{Z}}_{\boldsymbol{t}}^{*} - \mathbf{3}\sqrt{\boldsymbol{S_t^*}}, \tag{26}$$

$$\boldsymbol{LS_t} = \bar{\boldsymbol{Z}}_{\boldsymbol{t}}^{*} + \mathbf{3}\sqrt{\boldsymbol{S_t^*}}, \tag{27}$$

em que $\bar{Z}_t^*$ é a média da série observada de z_1 até z_t dada por

$$\bar{\boldsymbol{Z}}_{\boldsymbol{t}}^{*} = \frac{\boldsymbol{t-1}}{\boldsymbol{t}}\bar{\boldsymbol{Z}}_{\boldsymbol{t-1}}^{*} + \frac{\mathbf{1}}{\boldsymbol{t}}\boldsymbol{z_t}, \boldsymbol{t} \geq \mathbf{1}, \tag{28}$$

e S_t^* é a variância da série observada de z_1 até z_t dada por

$$\boldsymbol{S_t^*} = \frac{\boldsymbol{t-1}}{\boldsymbol{t}}\boldsymbol{S_{t-1}^*} + \frac{\boldsymbol{t-1}}{\boldsymbol{t}}\bar{\boldsymbol{Z}}_{\boldsymbol{t-1}}^{*\mathbf{2}} + \frac{\mathbf{1}}{\boldsymbol{t}}\boldsymbol{z_t^2} - \bar{\boldsymbol{Z}}_{\boldsymbol{t}}^{*\mathbf{2}}, \boldsymbol{t} \geq \mathbf{1}, \tag{29}$$

e, para $t = 1$, $\bar{Z}_0^* = 0$ e $S_0^* = 0$.

A escolha do modelo AEDR, para previsão de um local promissor na realização da delimitação do espaço de busca, se deve ao fato do AEDR ser um modelo não paramétrico, portanto, não é necessário estabelecer hipóteses sobre o modelo probabilístico do processo gerador da série, como é o caso do modelo KF. Também, como no AEDR é necessário estimar somente duas estruturas do modelo dependendo apenas de cálculos de médias aritméticas ponderadas de forma recursiva, o seu projeto é simples e consome pouca memória e o esforço computacional é insignificante quando comparada com os custos associados ao PF.

Assim sendo, este trabalho propõe a integração da técnica de previsão via modelo AEDR em abordagens SI para problemas de rastreamento de alvo em vídeos. O índice temporal t corresponde ao *frame* corrente e o valor observado da série, z_t, corresponde ao $gBest_t$ fornecida pela meta-heurística. A previsão de horizonte 1, $\hat{Z}_t(1)$, denotada por $\hat{p}_{t-1}(1)$, é a previsão dada pelo AEDR, no *frame* anterior, $t-1$, para o $gBest_t$, ou seja, $\hat{p}_{t-1}(1)$ é a

previsão de $gBest_t$. Para simplificar a notação, $\hat{p}_{t-1}(1)$, será doravante denotado simplesmente por $\hat{p} = (\hat{x}, \hat{y}, \hat{w}, \hat{h})$.

4.3.2 O Rastreador SFLA Dinâmico, o DSFLA

O modelo de rastreamento de alvos em vídeos proposto neste trabalho, chamado de rastreador DSFLA (ou rastreador SFLA dinâmico) pertence a categoria 2. O DSFLA é uma versão aprimorada da meta-heurística SFLA para trabalhar em problemas de otimização dinâmica promovendo o balanço entre a transferência de conhecimento e a diversidade das partículas. Nesta proposta, também é apresentada um esquema de delimitação do espaço de soluções de acordo com a posição do alvo estimado e do alvo previsto no *frame* anterior. A previsão da posição do alvo é fornecida pelo modelo AEDR.

A função objetivo adotada neste trabalho, para todos os algoritmos, é a distância de Bhattacharyya entre os histogramas padronizados do *Template* e do alvo candidato (neste caso, a máxima aptidão equivale à mínima distância).

O algoritmo DSFLA, no primeiro *frame*, gera N partículas ao acaso dentro do espaço de soluções S dado pela Equação (1). Em seguida, o valor da distância de Bhattacharyya é calculada para todas as partículas. A lista das melhores partículas é atualizada e o SFLA é executado até que a distância de Bhattacharyya da partícula de melhor aptidão do enxame seja menor que 0,005 ou se o número de iterações da meta-heurística tenha sido alcançado (neste trabalho, o número de iterações é igual a dez). Quando o algoritmo alcança a condição de parada, a meta-heurística fornece o $gBest_t$ e inicia o processo de previsão da posição (e tamanho) do $gBest$ para o próximo *frame* via AEDR. Logo após a previsão, o processo de delimitação do espaço de soluções é executado.

A delimitação proposta do espaço de soluções considera a menor região retangular contendo a união de duas outras regiões retangulares; uma, centrada no canto superior esquerdo do alvo estimado pela meta-heurística, $gBest = (x_g, y_g, w_g, h_g)$, e, uma outra, centrada no canto superior esquerdo do alvo previsto pelo modelo AEDR no *frame* anterior, $\hat{p} = (\hat{x}, \hat{y}, \hat{w}, \hat{h})$.

A primeira região, centrada no $gBest$, simula o resultado do modelo de passeio aleatório simples, assumindo a hipótese do alvo não se deslocar por uma grande distância entre dois

frames consecutivos, A segunda região, centrada no $\hat{p}$, determina um subespaço no espaço de soluções centrado no provável local do alvo no *frame* seguinte.

A delimitação do espaço de soluções proposta é, portanto, sugerida da seguinte maneira:

I. os limites das coordenadas (x, y) do espaço de solução são obtidos por:

$$\boldsymbol{Min}\left(\boldsymbol{x_g - \alpha_1 w_g}, \hat{\boldsymbol{x}} - \boldsymbol{\alpha_1}\hat{\boldsymbol{w}}\right) \leq \boldsymbol{x} \leq \boldsymbol{Max}\left(\boldsymbol{x_g + \alpha_1 w_g}, \hat{\boldsymbol{x}} + \boldsymbol{\alpha_1}\hat{\boldsymbol{w}}\right), \quad (30)$$

$$\boldsymbol{Min}\left(\boldsymbol{y_g - \alpha_2 h_g}, \hat{\boldsymbol{y}} - \boldsymbol{\alpha_2}\hat{\boldsymbol{h}}\right) \leq \boldsymbol{y} \leq \boldsymbol{Max}\left(\boldsymbol{y_g + \alpha_2 h_g}, \hat{\boldsymbol{y}} + \boldsymbol{\alpha_2}\hat{\boldsymbol{h}}\right), \quad (31)$$

II. e os limites das coordenadas (w, h) do espaço de solução são obtidos por

$$(\boldsymbol{1 - \gamma_1})\boldsymbol{w_g} \leq \boldsymbol{w} \leq (\boldsymbol{1 + \gamma_1})\boldsymbol{w_g}, \quad (32)$$

$$(\boldsymbol{1 - \gamma_2})\boldsymbol{h_g} \leq \boldsymbol{h} \leq (\boldsymbol{1 + \gamma_2})\boldsymbol{h_g}, \quad (33)$$

em que as constantes $0 < \alpha_1 \leq 2$, $0 < \alpha_2 \leq 2$, $0 < \gamma_1 < 1$ e $0 < \gamma_2 < 1$ são predefinidas e dependentes do problema.

A Figura 1 mostra um exemplo de delimitação do espaço de soluções proposta neste trabalho utilizando o décimo *frame* do vídeo *BlurBody* selecionado da base pública de dados *Hanyang visual tracker benchmark* (Wu, Lim e Yang, 2013). O *bounding box* azul corresponde ao alvo verdadeiro marcado a mão e o *bounding box* magenta corresponde ao alvo estimado pela meta-heurística $gBest_9$. A região preenchida de verde corresponde à delimitação de $\boldsymbol{S}$ centrada na coordenada (x_g, y_g) do $gBest_9$ (o canto superior esquerdo do *bounding box* magenta) e a região preenchida de amarelo corresponde à delimitação de S centrada na coordenada $(\hat{x}, \hat{y})$ do $\hat{p}$ estimado pelo AEDR (corresponde ao canto superior esquerdo do *bounding box* gerado pela previsão $\hat{p} = \left(\hat{x}, \hat{y}, \hat{w}, \hat{h}\right)\Big)$. A área delimitada pela caixa retangular vermelha é a região de S delimitada pelas Equações (30) e (31).

No exemplo dado pela Figura 1, somente é possível visualizar a delimitação do espaço de soluções correspondente ao subespaço 2D dada pelas coordenadas (x, y). Para as coordenadas (w, h), associadas à dimensão do *bounding box*, a atribuição dos valores é dada pelas Equações (32) e (33) de forma independente, uma vez que, pela experiência passada, as dimensões das janelas possuem uma variação pequena de um *frame* para o outro.

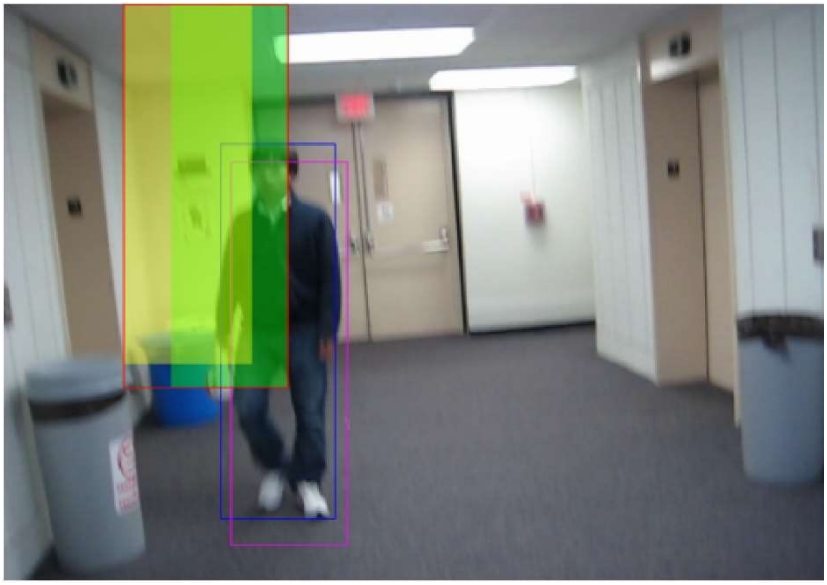

Figura 1. Um exemplo de delimitação do espaço de soluções.
Fonte: O décimo *frame* do vídeo "*BlurBody*" obtido da base de dados pública (*open access*): *visual tracking Hanyang benchmark* (Wu, Lim e Yang, 2013).

A partir do segundo *frame*, após a delimitação do espaço de soluções determinada no *frame* anterior pelas Equações (30)-(33), o processo de seleção das partículas é iniciado seguido pelo processo adaptativo de substituição das partículas selecionadas.

O processo proposto de seleção de partículas as escolhe de tal modo que elas estejam distantes entre si por uma distância mínima, δ_{Min} (o objetivo é garantir a diversidade das partículas).

δ_{Min} é calculada da seguinte maneira:

$$\boldsymbol{\delta_{Min}} = \frac{\boldsymbol{Max}(S_w^*, S_h^*)}{\boldsymbol{\alpha_0 N}}, \tag{34}$$

em que S_w^* e S_h^* são, respectivamente, as dimensões horizontal e vertical do espaço de solução delimitado (correspondem à largura e à altura do retângulo vermelho da Figura 1, respectivamente), N é o número de partículas e α_0 é um percentual máximo de transferência predefinido pelo usuário (α_0 é dependente do problema).

A escolha das partículas é feita por ordem crescente de aptidão, começando pelo $gBest$ e respeitando a distância mínima δ_{Min} entre elas.

O processo adaptativo de transferência de partículas proposto gera N partículas ao acaso no espaço de soluções delimitado e as ordena em ordem decrescente de aptidão. Após a aptidão

das partículas selecionadas serem atualizadas e ordenadas em ordem crescente, elas são comparadas com as piores partículas geradas e, aquela com a pior aptidão, é descartada.

O processo de transferência de partícula nem sempre aceita todas as $\alpha_0 N$ partículas selecionadas (até mesmo nenhuma delas), uma vez que o processo de seleção continua enquanto houver partículas a serem selecionadas ou enquanto não selecionar as $\alpha_0 N$ partículas, sempre mantendo a distância mínima entre elas. Também é possível haver partículas transferidas posicionadas fora do espaço de soluções delimitado, porém, dentro do espaço de soluções S dado pela Equação (1).

Os passos do algoritmo DSFLA discutidos neste capítulo estão sintetizados no pseudocódigo em Algoritmo 1 e no fluxograma do pseudocódigo em Figura 1.

Algoritmo 1. Pseudocódigo do rastreador DSFLA.

Entrada:	n *frames*, *template*, *ground truth* e as variáveis iniciais: $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \gamma_1, \gamma_2$.
Saída:	Os n alvos estimados e suas medidas de qualidade: tempo de processamento e AUC.
1:	***para*** f = 1:n % para todos os n *frames*
2:	***para*** i = 1:N % para todas as *N* partículas
3:	***se*** f = = 1
4:	Gerar ao acaso a partícula p_i em *S*; % Eq. (1)
5:	***senão***
6:	Gerar ao acaso a partícula p_i em *S* delimitado; % Eqs. (30) - (33)
7:	***fim***
8:	Extrair o histograma e calcular a aptidão da partícula p_i; % Eq (15)
9:	***fim***
10:	***se*** f > 1
11:	Processo de transferência de partículas;
12:	***fim***
13:	Atualizar o *gBest*;
14:	***enquanto*** <condição de parada = = falso>
15:	Executar a meta-heurística SFLA;
16:	***fim***
17:	Armazenar os dados: *gBest* e as medidas de desempenho do DSFLA;
18:	Processo de seleção de partículas;
19:	Delimitar *S* em função do *gBest*: *S1*;
20:	Executar o AEDR; % Eqs. (16) - (21)
21:	Delimitar *S* em função da previsão de *gBest* dado pelo AEDR: *S2*;
22:	Obter *S* delimitado em função da união de *S1* e *S2*; % Eqs. (30) - (33)
23:	***fim***

5 *Delineamento Experimental e Análise dos Resultados*

Neste capítulo, a Seção 5.1 trata do delineamento experimental e das métricas usadas para avaliar a qualidade do desempenho dos rastreadores. A Seção 5.2 mostra a configuração final dos parâmetros de cada rastreador considerado neste trabalho. E na Seção 5.3, os resultados experimentais são analisados e discutidos.

5.1 O Delineamento Experimental

Para investigar a eficiência do método proposto neste trabalho, uma amostra aleatória de 15 vídeos foi selecionada da base de dados pública[5] *Hanyang visual tracker benchmark* (Wu, Lim e Yang, 2013). Esta base de dados (doravante chamada somente de *benchmark*), disponibiliza 100 vídeos os quais exibem os mais variados cenários em que variados fatores desafiadores atuam na cena. O *benchmark* também disponibiliza um arquivo de dados contendo a marcação manual do(s) alvo(s), para cada *frame* dos vídeos disponíveis, chamados de *ground truth* (significa: "verdade fundamental").

O *benchmark* é uma coleção dos vídeos mais comumente usados em rastreamento de alvos contendo vídeos de várias bases de dados públicas conhecidas, tais como a VIVID (Collins, Zhou e The, 2005), a CAVIAR[6] e a PAMI[7]. Também, é possível ter acesso livre à maioria dos códigos dos rastreadores avaliados além da divulgação de uma análise do desempenho dos algoritmos testados.

O processo de amostragem foi executado da seguinte maneira: os 100 vídeos disponíveis no *benchmark* estão dispostos em um arranjo matricial de 20 linhas e 5 colunas. Cada entrada deste arranjo corresponde a um vídeo. Os vídeos foram enumerados de 1 a 100 iniciando a enumeração na linha 1 e coluna 1 (vídeo número 1) e, seguindo da esquerda para a direita e de cima para baixo, os vídeos foram sequencialmente enumerados de 1 a 100. Em seguida, foram gerados 15 números inteiros positivos pseudoaleatórios no intervalo $[1; 100]$ no *MatLab* com semente aleatória. Os vídeos selecionados estão de acordo com a numeração gerada pelos números pseudoaleatórios. Portanto, trata-se de um plano probabilístico de amostragem aleatória simples de tamanho 15.

A Tabela 1 mostra os vídeos selecionados com as seguintes informações: número de identificação do vídeo (a numeração está em ordem alfabética dos nomes e, portanto, não

[5] O endereço eletrônico do *benchmark*: http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html.
[6] O endereço eletrônico do *benchmark*: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1
[7] O endereço eletrônico do *benchmark*: http://sites.google.com/site/benchmarkpami

corresponde aos números dos vídeos no momento da amostragem), o nome original do vídeo, o tamanho do vídeo (em número de *frames*), a resolução horizontal e vertical da imagem (em número de *pixels*) e os principais desafios presentes na cena. Os desafios presentes nos vídeos selecionados são: rotação do alvo no plano da imagem (*In-Plane Rotation* – IPR), rotação do alvo fora do plano da imagem (*Out-of-Plane Rotation* – OPR), movimentos rápidos do alvo (*Fast Motion* – FM), movimento borrado (*Motion Blur* – MB), baixa resolução da imagem (*Low Resolution* – LR), variação de escala do alvo (*Scale Variation* – SV), deformação do alvo (*Deformation* – DEF) e oclusão (*Occlusion* – OCC) (Wu, Lim e Yang, 2013). Vale a pena comentar que o *benchmark* disponibiliza as imagens sequenciais dos *frames* dos vídeos, portanto, não é disponibilizado informações sobre a taxa de *frames* por segundo.

Tabela 1. Os vídeos selecionados de *Hanyang visual tracker benchmark.*

Vídeo	Nome	Tamanho	Resolução	Desafios
1	BlurBody	334	(640,480)	IPR, FM, MB, SV, DEF
2	BlurCar2	585	(640,480)	FM, MB, SV
3	BlurCar4	380	(640,480)	FM, MB
4	BlurFace	493	(640,480)	IPR, FM, MB
5	BlurOwl	631	(640,480)	FM, MB, SV
6	Boy	602	(640,480)	IPR, OPR, FM, MB, SV
7	Couple	140	(320,240)	OPR, FM, SV, DEF, BC
8	David2	537	(320,240)	IPR, POR
9	Deer	71	(704,400)	IPR, FM, MB, LR, BC
10	Dog	127	(352,240)	OPR, SV, DEF
11	Dog1	1350	(320,240)	IPR, OPR, SV
12	Jumping	313	(352,288)	FM, MB
13	MountainBike	228	(640,360)	IPR, OPR, BC
14	Twinnings	471	(320,240)	OPR, SV
15	Walking2	500	(384,288)	LR, SV, OCC
Total	-	6762	-	-

As métricas adotadas para medir a qualidade do desempenho em Wu, Lim e Yang (2013) são:

- a métrica *success rate* (significa: “taxa de sucesso”): mede a área sob a curva (AUC – do inglês: *Area Under Curve*) da taxa de sucesso por métrica de Pascal, e,
- a métrica *accuracy* (significa: “acurácia”): mede a distância Euclidiana (em número de *pixels*) entre os pontos centrais dos *bounding boxes* dos alvos estimado e real.

A métrica de Pascal (Everingham et al, 2010) é definida de acordo com

$$\boldsymbol{Pa}(\xi_{GT}, \xi_C) = \frac{|\xi_{GT} \cap \xi_C|}{|\xi_{GT} \cup \xi_C|}, \quad (35)$$

em que ξ_{GT} é o *bounding box* correspondente ao *ground truth*, ξ_C é o *bounding box* correspondente ao alvo candidato e $|\cdot|$ representa a cardinalidade do conjunto.

A métrica de Pascal mede a qualidade do rastreador em detectar o alvo quantificando o percentual de *pixels* compartilhados entre os dois *bounding boxes*, ou seja, mede o percentual de sobreposição das janelas. A métrica de Pascal varia de 0,0, quando não há sobreposição, até 1,0, quando os dois *bounding boxes* são coincidentes. Geralmente, os alvos são considerados detectados quando a métrica de Pascal é igual ou superior a um certo limiar, por exemplo, 0,5.

A taxa de sucesso por métrica de Pascal é o percentual de *frames* de um vídeo em que o alvo foi detectado para um dado limiar da métrica de Pascal. Portanto, a curva de taxa de sucesso por métrica de Pascal é uma curva obtida pelas taxas de sucesso variando o limiar da métrica de Pascal de 0,0 a 1,0. A vantagem de observar a curva está em visualizar o desempenho do rastreador para todos os limiares da métrica, logo, o cálculo da AUC constitui numa variável mais robusta e completa na avaliação do desempenho de um rastreador ao invés de comparar a taxa de sucesso de um único limiar da métrica de Pascal. A AUC varia de 0,0 a 1,0 e quanto mais próximo de 1,0 melhor é o desempenho do rastreador para todos os limiares da métrica.

Já a métrica *accuracy*, mede a qualidade do rastreamento quantificando a precisão do rastreador em detectar a posição do alvo no *frame*, em termos da distância entre os centros dos *bounding boxes* sem, contudo, medir o ajuste das janelas.

Neste trabalho optou-se pela métrica *success rate* via variável AUC, uma vez que ela mede a qualidade da detecção do alvo tanto em termos de precisão quanto do ajuste das janelas.

O *benchmark* também propõe três desafios (ou testes) para avaliar a qualidade e robustez dos rastreadores, utilizando ambas as métricas *success rate* e *accuracy*:

- o desafio *One-Pass Evaluation* (OPE – significa: "avaliação em uma passagem") que avalia o desempenho do rastreador considerando todos os *frames* de um vídeo numa sequência que vai, sucessivamente, do primeiro ao último;
- o desafio *Temporal Robust Evaluation* (TRE – significa: "avaliação da robustez temporal") que avalia o rastreador usando uma sequência de *frames* começando de algum *frame* aleatório e indo sequencialmente até o último *frame* do vídeo; e,
- o desafio *Spatial Robust Evaluation* (SRE – significa: "avaliação da robustez espacial") o qual o *template* é modificado por 0,8 e 1,2 da sua escala original iniciando no primeiro *frame*, partindo de 12 diferentes locais próximos ao *template*, e indo sequencialmente até o último *frame*.

Em todos os testes, o *template* é o *ground truth* do primeiro *frame*.

Para comparar a eficiência do DSFLA, quatro outros rastreadores cuja meta-heurística de busca é baseada em SI foram considerados: o rastreador cuja meta-heurística é o BBPSO (kennedy, 2003), o rastreador ADSO proposto por Bae et al. (2016), o rastreador cuja meta-heurística de busca é o SFLA (Eussuf, et al, 2006), o rastreador SSA proposto por (Zhang et al., 2020).

Neste trabalho, a eficiência dos rastreadores considerados será mensurada avaliando o resultado dado pelo desafio OPE através da métrica *success rate*. O tempo médio de processamento por *frame* (em segundos) para cada vídeo também será avaliado e comparado.

Os valores obtidos serão exibidos em tabelas. Cada valor corresponde à média aritmética simples de seis replicações de cada vídeo para cada rastreador, denominados por: PSO, ADSO, SFLA, DSFLA e SSA (sempre nesta ordem). Vale a pena observar que, em experiências passadas, foram analisados os resultados obtidos com 3, 6 e 12 replicações, entretanto, os resultados observados para 6 e 12 replicações apresentaram, em média, uma diferença muito pequena. O coeficiente de variação não supera 10% da escala da variável AUC. Portanto, por uma questão de economia de tempo de execução do experimento, optou-se por 6 replicações.

É importante observar também, que há outras medidas de qualidade do rastreamento e desafios propostos na literatura, veja uma lista em Yamaz, Javed e Shah (2006). Porém, cabe lembrar, o objetivo deste trabalho é medir a eficiência do método proposto e compará-lo com o desempenho de outros rastreadores da categoria 2 que adotam abordagem SI, logo, o foco não é comparar técnicas de identificação dos alvos, medidas de similaridades ou métricas. O importante é garantir que todos os rastreadores avaliados sejam submetidos às mesmas condições, medidas e métricas.

Para avaliar e comparar os valores do tempo médio de processamento por *frame* e assegurar que as condições do experimento sejam as mesmas para todos os rastreadores e vídeos, o experimento foi executado no mesmo computador (processador *Intel Pentium Dual-Core* de 1,86 *GHz* e 2 *GBytes* DDR2) e todos os algoritmos foram programados e executados no MatLab.

As linhas das tabelas exibem o resultado de cada vídeo para cada rastreador e, nas três últimas linhas, as estatísticas: média, mediana e o coeficiente de variação (cv) global dos 15 vídeos.

O cv é a razão entre o desvio padrão amostral (sd – do inglês: *standart deviation*) e a média amostral, $\bar{p}$, de uma variável p observada, ou seja,

$$\boldsymbol{cv} = \frac{\boldsymbol{sd}}{\bar{\boldsymbol{p}}}. \tag{36}$$

O cv é uma medida relativa da dispersão dos dados amostrais independente da escala da variável e pode ser expressa em percentual de variação (quanto menor é o percentual de variação medida pelo cv, mais homogênea é a amostra observada e mais representativa é a estimativa pontual da média e da mediana).

5.2 Configuração dos rastreadores

A escolha adequada dos parâmetros dependentes do problema deve ser criteriosa para que cada rastreador alcance o seu melhor desempenho. A escolha de uma boa configuração para os parâmetros foi obtida com base em um experimento prévio à parte do experimento principal (descrita na seção 5.1) cujas etapas foram:

- a primeira etapa: quatro vídeos foram selecionados ao acaso do *benchmark Hanyang* (Wu, Lim e Yang, 2013);
- a segunda etapa: O tempo médio de processamento por *frame* e AUC da taxa de sucesso por métrica de Pascal foram calculadas em diferentes cenários de configuração dos parâmetros de cada rastreador (um total de 34 cenários) e os valores anotados correspondem à média aritmética simples de três replicações de cada vídeo; e,
- a terceira etapa: os valores obtidos para o tempo médio e a AUC foram comparados para se chegar na configuração de melhor desempenho de cada rastreador.

A escolha de utilizar quatro vídeos neste estágio se justifica, pois, foram rodados cerca de quatro a doze cenários diferentes por rastreador num total de quarenta e quatro cenários distintos. A escolha de rodar três replicações de cada vídeo deve-se ao fato dos resultados serem homogêneos, isto é, a variação dos resultados para cada replicação do vídeo era baixa com cv entre 30 a 60% da escala da variável, dependendo do rastreador.

Os vídeos sorteados nesta etapa do experimento foram: "*Couple*" e "*Deer*" (corresponde aos vídeos 7 e 9 da Tabela 1, respectivamente), "*Bolt2*" (com 293 *frames*; resolução: 480x270;

desafios: IPR, DEF, BC) e "*Football1*" (com 74 *frames*; resolução: 352x288; desafios: IPR, OPR, BC), todos da mesma base pública de dados.

Os resultados finais da configuração dos parâmetros de cada rastreador foram os seguintes:

- PSO: 150 partículas com 15 grupos locais de 10 partículas.
- ADSO: 40 partículas, os limiares $th_1 = 0{,}45$; $th_2 = 0{,}1$; $th_3 = 0{,}005$; e, a distribuição de probabilidade inicial $H_f^0 = 0{,}7$.
- SFLA: 50 partículas; 10 memeplexes com 5 partículas cada; número máximo de iterações igual a 10; e, o tamanho máximo do salto igual a 10 *pixels*.
- DSFLA: Os mesmos parâmetros do SFLA mais: $\alpha_0 = 0{,}3$; $\alpha_1 = \alpha_2 = 1{,}0$; $\gamma_1 = \gamma_2 = 0{,}1$; $\alpha_3 = [0{,}9; 0{,}9; 0{,}8; 0{,}8]$; e, $\alpha_4 = [0{,}4; 0{,}4; 0{,}5; 0{,}5]$.
- SSA: 80 partículas; o número máximo de iterações igual a 100; e, 20 salpas líderes.

5.3 Análise dos Resultados

A Tabela 2 mostra o desempenho dos rastreadores de acordo com o tempo médio de processamento por *frame* de cada vídeo.

Tabela 2. Tempo médio de processamento por *frame* em segundos (os melhores resultados estão em negrito).

Vídeo	PSO	ADSO	SFLA	DSFLA	SSA
1	3,0914	**0,4776**	1,1673	1,0911	6,8387
2	2,1559	**0,5382**	0,9801	0,9436	4,7558
3	2,9967	**0,2774**	1,1992	1,0951	6,5406
4	2,0773	**0,4318**	0,9154	0,8588	4,6733
5	1,7022	**0,2508**	0,8447	0,7370	3,6917
6	1,4303	**0,2082**	0,8014	0,6653	3,0493
7	1,4158	**0,1959**	0,7841	0,6641	3,0406
8	0,5485	**0,1933**	0,6779	0,5570	2,9147
9	1,7645	**0,3321**	0,9555	0,8803	3,8661
10	1,4918	**0,1947**	0,6860	0,6147	3,2001
11	1,4474	**0,2138**	0,7166	0,6182	3,0866
12	1,4036	**0,1939**	0,7864	0,6572	2,9656
13	1,6324	**0,2263**	0,9009	0,7829	3,5079
14	1,6444	**0,2147**	0,6960	0,6470	3,5379
15	1,6005	**0,1991**	0,8188	0,6511	3,4203
Média	1,7602	**0,2765**	0,8620	0,7642	3,9393
Mediana	1,6324	**0,2147**	0,8188	0,6653	3,5079
cv (%)	35,96	41,53	**18,74**	22,27	31,77

Pela Tabela 2, o rastreador SSA toma mais tempo de execução e o rastreador ADSO é o mais rápido, entretanto, o DSFLA é, em média, o segundo mais rápido. Todos os valores são representativos, pois, o cv apresenta um baixo percentual, exceto para o ADSO que apresenta maior instabilidade em termos de tempo de processamento.

A Figura 2 mostra o *boxplot* de Whiskers (resultado gerado pela função interna do MatLab: "*boxplot*") para os dados observados do tempo médio de processamento por *frame* para cada rastreador. Cada um dos quatro segmentos do *boxplot* corresponde a 25% dos valores observados na amostra, os círculos pequenos correspondem aos valores discrepantes. A parte central mais escura representa o intervalo interquartílico (IQR – do inglês: *Interquartile Range*), $q_3 - q_1$, em que q_3 é o terceiro quartil e q_1 é o primeiro quartil. O ponto central representa a mediana (q_2) e os triângulos representam os extremos do intervalo de 95% de confiança centrado na mediana e são calculados de acordo com

$$\boldsymbol{q_2 \pm \frac{1{,}57(q_3 - q_1)}{\sqrt{N}}}, \tag{37}$$

onde N é o tamanho da amostra observada (Nelson, 1989).

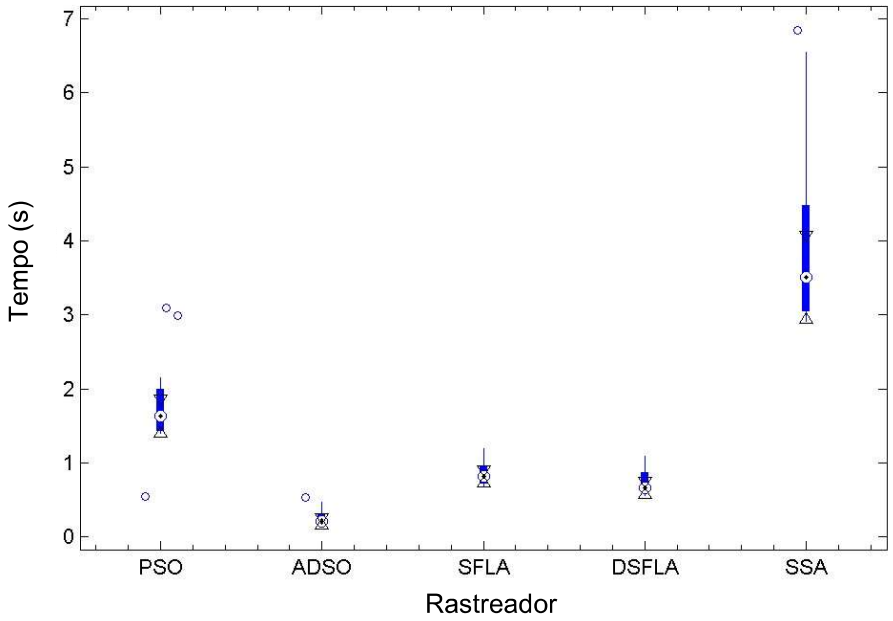

Figura 2. O *boxplot* da variável tempo médio de processamento por *frame* para todos os rastreadores.

Se os intervalos de confiança não se interceptam (em relação à projeção ortogonal dos triângulos sobre o eixo das ordenadas), então pode-se concluir com 95% de confiança que há uma diferença significativa entre as medianas. Esta conclusão equivale a um teste estatístico

não paramétrico de hipóteses em que a hipótese nula (H_0: a diferença entre as medianas é igual a zero) é rejeitada a 5% de significância.

Do gráfico na Figura 2, conclui-se que todos os tempos médios de processamento por *frame* são significativamente diferentes, exceto para os tempos médios entre os rastreadores SFLA e DSFLA. Entretanto, empiricamente, o rastreador DSFLA é sistematicamente cerca de 10% mais rápido que o SFLA (consulte a Tabela 2).

A Tabela 3 mostra o desempenho dos rastreadores em relação a qualidade do rastreamento de acordo com a variável AUC da taxa de sucesso por métrica de Pascal. É possível observar que o cv é pequeno para SFLA, DSFLA e SSA (cerca de 23% de variação), apresentando uma boa estabilidade e uma estabilidade de moderada para alta para os rastreadores PSO e ADSO (de 38 a 50%). Logo, se pode afirmar, dos resultados observados, que os rastreadores são satisfatoriamente estáveis em termos de qualidade do rastreamento. Também pode-se notar que as médias e medianas de todos os rastreadores apresentam valores muito próximos sugerindo uma assimetria pequena na distribuição dos resultados. Portanto, tanto a média quanto a mediana são estimativas representativas da média teórica de desempenho.

Tabela 3. AUC da taxa de sucesso por métrica de Pascal (os melhores resultados estão em negrito).

Vídeo	PSO	ADSO	SFLA	DSFLA	SSA
1	0,4698	0,3654	0,5746	**0,5851**	0,5795
2	0,2518	0,2025	0,3234	**0,3633**	0,3632
3	0,2683	0,0803	0,4033	**0,4091**	0,3791
4	0,4927	0,4558	0,5906	**0,6896**	0,6428
5	0,2347	0,2257	**0,4489**	0,4114	0,4191
6	0,1139	0,3176	0,2882	0,4271	**0,4985**
7	0,3254	0,3198	0,4923	**0,5264**	0,4226
8	0,0206	0,1582	0,2949	0,2436	**0,3244**
9	0,4213	0,4084	0,4261	**0,4874**	0,3623
10	0,1838	0,3494	0,3215	**0,4047**	0,3337
11	0,2435	0,4649	0,4877	**0,5419**	0,4493
12	0,3022	0,1543	0,4668	**0,5300**	0,4838
13	0,2572	0,3229	0,4794	0,4457	**0,4986**
14	0,1905	0,3063	0,4011	**0,4824**	0,3964
15	0,1384	0,2711	0,2973	**0,3423**	0,2708
Média	0,2609	0,2935	0,4284	**0,4593**	0,4285
Mediana	0,2518	0,3176	0,4261	**0,4457**	0,4191
cv (%)	49,69	38,20	23,82	23,63	**23,22**

A Figura 3 mostra o gráfico dos *boxplots* para a AUC da taxa de sucesso por métrica de Pascal para todos os rastreadores considerados neste trabalho. Embora o desempenho do rastreador DSFLA seja significativamente melhor que os rastreadores PSO e ADSO, não é

possível rejeitar a hipótese de qualidade compatível entre os rastreadores DSFLA, SFLA e SSA. Entretanto, dentre os quinze vídeos selecionados, o DSFLA superou o desempenho dos demais rastreadores em onze deles (73% dos vídeos) além de apresentar o melhor resultado na média e na mediana global. Também, como mostra a Tabela 3, os vídeos 2, 7, 8, 10, 14 e 15 (40% dos vídeos selecionados) são os mais desafiadores para todos os rastreadores considerados neste trabalho e o DSFLA foi o rastreador que apresentou os melhores resultados em cinco desses seis vídeos (83% dos vídeos mais desafiadores). Além disso, a taxa de sucesso do DSFLA é, em média, cerca de 7,2 a 76,6% maior quando comparada com a taxa de sucesso dos demais rastreadores considerados (consulte a Tabela 3). Portanto, empiricamente, os resultados apontam para o DSFLA como o melhor desempenho dentre os rastreadores.

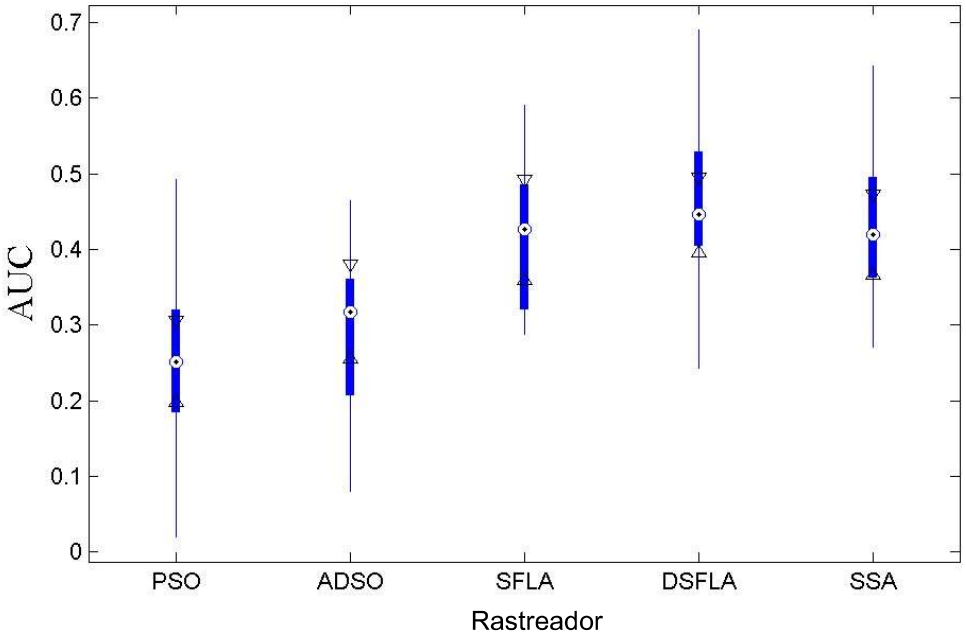

Figura 3. O *boxplot* da variável AUC para todos os rastreadores.

As Figuras de 4 e 5 mostram dois exemplos, escolhidos ao acaso, do desempenho de rastreamento dada pela curva da taxa de sucesso por métrica de Pascal de todos os rastreadores para os vídeos 4 e 7, respectivamente. Em ambos os gráficos das Figuras 4 e 5, a curva que representa o desempenho do rastreador DSFLA (a curva verde pontilhada) está acima das outras curvas, na maioria dos limiares da métrica de Pascal, indicando uma taxa maior de detecção do alvo.

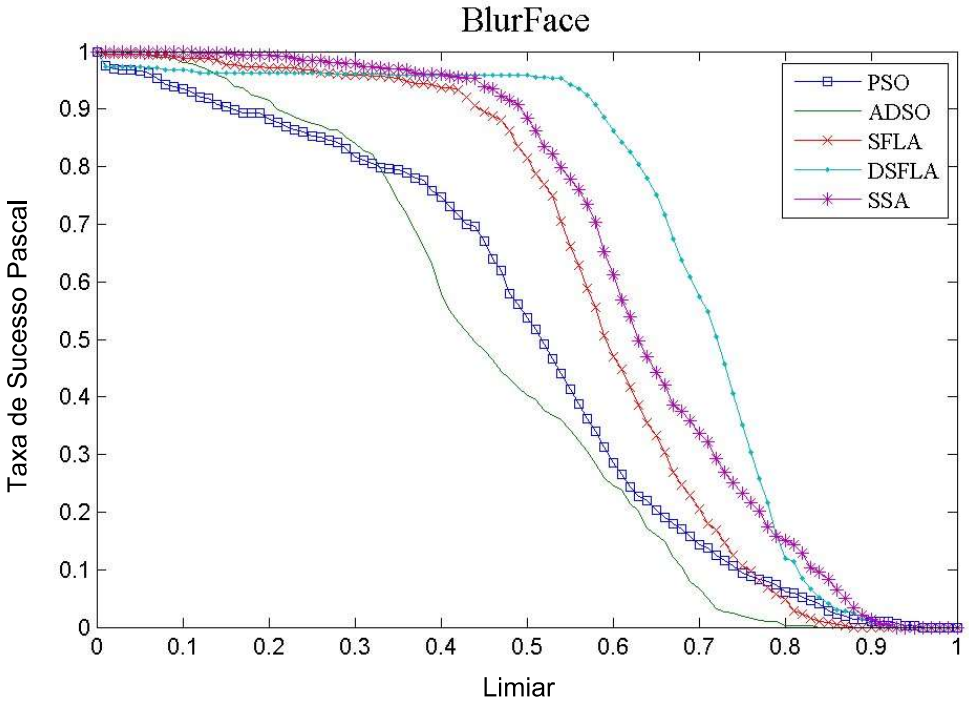

Figura 4. A curva da taxa de sucesso por métrica de Pascal do vídeo 4 (*BlurFace*) para os 5 rastreadores.

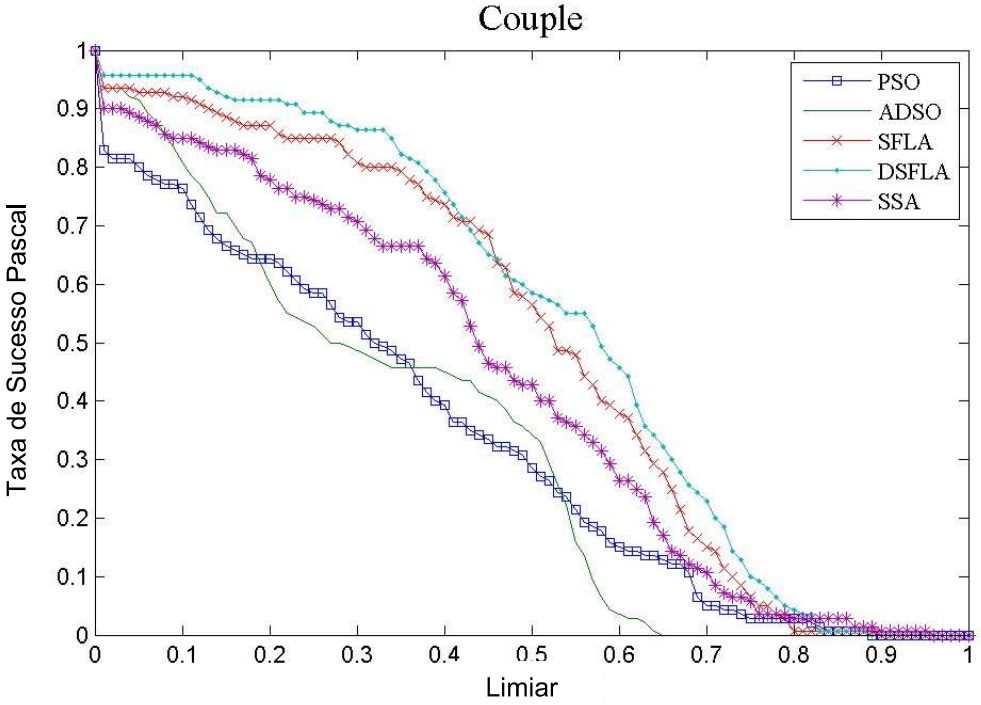

Figura 5. A curva da taxa de sucesso por métrica de Pascal do vídeo 7 (*Couple*) para os 5 rastreadores.

Os gráficos das curvas de taxa de sucesso por métrica de Pascal da maioria dos outros vídeos mostram um desempenho semelhante aos exibidos nos gráficos das Figuras 4 e 5.

Os vídeos em que o DSFLA exibiu os melhores resultados de AUC em relação aos demais rastreadores foram: 1, 2, 3, 4, 7, 9, 10, 11, 12, 14 e 15. As características em comum entre estes

vídeos são: uma variação mínima de luz ambiente, oclusões eventuais e mínimas dos alvos, poucas características em comum entre os alvos e o fundo ou outros objetos e a ausência de mudanças bruscas na escala dos alvos. Também em comum, a presença constante de movimentos rápidos dos alvos ou da câmera, rotações em diversos planos em relação à imagem dos alvos e a constante presença de alvos com imagens não nítidas.

Os vídeos em que todos os rastreadores, considerados neste trabalho, exibiram o pior desempenho foram o vídeo 10 (*Dog*) e o vídeo 15 (*Walking2*). O vídeo 10 apresenta uma mudança de escala do alvo muito ampla e rápida (os rastreadores até conseguem acompanhar os alvos em termos de distância Euclidiana, em *pixels*, entre os alvos estimados e os alvos verdadeiros, porém, eles falham nas sobreposições das janelas). Já o vídeo 15, em um dado *frame* do vídeo, o alvo sofre uma oclusão superior à 50%. Neste ponto, os rastreadores passaram a seguir um outro objeto semelhante ao alvo (esta perseguição persistiu por quase ¼ dos *frames*) quando, após a oclusão completa do objeto perseguido, os rastreadores conseguem recuperar e rastrear o alvo correto até o último *frame*. Mesmo nestes casos, o DSFLA apresentou os melhores resultados entre os rastreadores considerados.

Uma fraqueza comum de todos os rastreadores analisados neste trabalho está relacionada à variação de luz ambiente. Esta fraqueza, provavelmente, é causada pelo histograma padronizado de cor usado na representação das características dos alvos. Um histograma de cor é sensível a qualquer mudança da luz ambiente, pois, pode mudar consideravelmente a cena assim como as frequências dos histogramas de forma não proporcional, uma vez que a mudança na intensidade dos *pixels* não é linear.

Uma outra fraqueza em comum entre os rastreadores é quando o alvo e o fundo da cena apresentam características semelhantes. Novamente, a causa provável é a escolha do histograma de cor usado para representar as características do alvo. Neste caso, os *bounding boxes* de diferentes dimensões podem conter proporções similares de *pixels* de mesma intensidade representando histogramas similares em aparência, ou seja, vários alvos candidatos distintos e com áreas distintas dos *bounding boxes* podem apresentar aparências similares. Isso reduz a qualidade do rastreamento, pois, aumenta as chances de estimar um alvo com pouca sobreposição das janelas.

Uma possível estratégia para superar estas dificuldades é incluir uma característica do alvo baseada no aspecto da forma do objeto, tal como a característica HOG (Dallal e Triggs, 2005), o que implicará em maior custo computacional.

Para medir a qualidade das previsões do modelo AEDR foi calculada a raiz quadrada do erro quadrático médio de previsão (RMSE – do inglês: *Root of the Mean Square Error*) para cada uma das coordenadas (x, y, w, h) do $gBest$ (em número de *pixels*) da seguinte maneira

$$\boldsymbol{RMSE} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{t=1}}^{\boldsymbol{n}}\left(\boldsymbol{gBest_t} - \boldsymbol{\hat{p}_{t-1}(1)}\right)^{\mathbf{2}}}, \tag{38}$$

em que n é o número total de *frames*, $\hat{p}_{t-1}(1)$ é a previsão dada pelo AEDR no *frame* anterior, $t-1$, para o $gBest_t$, isto é, $\hat{p}_{t-1}(1)$ é a previsão de $gBest_t$.

O erro quadrático médio (MSE – do inglês: *Mean Square Error*) mede o valor agregado da variância e do quadrado do viés de previsão, ou seja, quanto maior é o valor do MSE (e do RMSE) mais heterogêneas e enviesadas são as previsões.

Também foi calculada a distância média entre todas as previsões, isto é, a média das distâncias Euclidianas entre o $\hat{p}_{t-1}(1)$ e $gBest_t, \forall t$, em número de *pixels*. Quanto mais próximo $\hat{p}_{t-1}(1)$ estiver do $gBest_t$ melhor é a previsão no instante t. Portanto, quanto mais próxima a média estiver de zero melhor será o desempenho do modelo AEDR.

A Tabela 4 mostra a qualidade das previsões do modelo AEDR através do RMSE de previsão para cada uma das coordenadas x e y, e a distância Euclidiana média entre $\hat{p}$ e o $gBest$ (em número de *pixels*). As coordenadas (x, y) correspondem ao ponto do canto superior esquerdo dos *bounding boxes*.

Tabela 4. O RMSE de previsão de x, y e a distância de previsão (em nº de *pixels*).

Vídeo	x	y	Distância
1	48,5473	31,7972	17,0507
2	78,8097	33,0285	15,3551
3	99,6405	41,2005	22,3768
4	43,2171	24,7496	20,4736
5	58,8349	81,6870	17,2760
6	30,4878	23,5310	9,6919
7	25,6458	15,6825	6,2391
8	7,2707	7,8175	1,2050
9	67,5223	43,8717	17,0988
10	11,1590	4,8302	2,7205
11	12,1733	9,6148	3,3553
12	41,3441	39,5144	8,7537
13	19,8933	12,0985	4,7146
14	15,1080	5,3767	2,8219
15	3,7789	9,0745	2,4323
Média	37,5621	25,7911	10,1044
Mediana	30,4878	23,5310	8,7537
cv (%)	76,24	80,82	73,47

Usando os dados da Tabela 4 e a Equação (37), o intervalo de 95% de confiança para o RMSE das coordenadas $\boldsymbol{x}$ e $\boldsymbol{y}$ é, respectivamente, $[\mathbf{11,57; 49,44}]$ e $[\mathbf{11,19; 35,87}]$. Portanto, o RMSE de previsão não excede 50 *pixels*. Para se ter uma ideia do que significa 50 *pixels* de RSME de previsão, em uma interpretação geométrica, dado que o MSE é a soma da variância e do quadrado do vício de previsão, o valor do MSE corresponde ao valor de um quadrado de lado igual a 50 *pixels* (ou de área igual a 2500 *pixels* ao quadrado). Sendo a área do menor *frame* dentre os vídeos amostrados igual a 76800 *pixels*, com 95% de confiança, o RMSE de previsão não alcança 5% de variação e vício ao quadrado de previsão em relação a qualquer *frame* do experimento. Pode-se concluir que as previsões do modelo AEDR são satisfatoriamente homogêneas e com viés não significante.

De modo análogo, o intervalo de 95% de confiança para a distância Euclidiana entre o $\hat{p}$ e o $gBest$ é $[2{,}96; 14{,}48]$. Ou seja, a distância máxima da previsão em relação ao alvo estimado não excede 15 *pixels*. Considerando que a maior diagonal dentre os vídeos amostrados é de 800 *pixels*, as distâncias das previsões do modelo AEDR, com 95% de confiança, não alcança 4% de erro. Pode-se concluir com 95% de confiança de as previsões do modelo AEDR são bastante precisas.

Dado que as previsões geradas pelo modelo AEDR apresentam boas qualidades (as previsões são precisas e possuem pouca variância e pouco viés), segue uma investigação para verificar o quanto a delimitação do espaço de soluções, sugerida neste trabalho, proporciona um aumento da qualidade de rastreamento do DSFLA. Para esta verificação, foi usado o mesmo esquema experimental realizado na Seção 4.2 para calibração dos parâmetros.

A Tabela 5 mostra a média e a mediana global das três replicações dos quatro vídeos para as variáveis AUC e tempo médio de processamento por *frame*. Duas versões do rastreador DSFLA foram consideradas: a Versão 1 que corresponde ao rastreador com a delimitação do espaço de soluções sugerida neste trabalho (o DSFLA original) e a Versão 2 que não delimita o espaço de soluções (o DSFLA sem o procedimento de delimitação do espaço de busca).

Tabela 5. Média e mediana da AUC e do tempo médio de processamento por *frame* do DSFLA, com e sem delimitação de *S*, nas Versões 1 e 2, respectivamente (os melhores resultados estão em negrito).

	Versão 1	**Versão 2**
AUC médio	**0,3579**	0,3303
AUC mediano	**0,3932**	0,3408
Tempo médio (segundos)	**0,7193**	0,8092
Tempo mediano (segundos)	**0,6719**	0,7649

A Tabela 5 mostra que a área mediana sob a curva da taxa de sucesso por métrica de Pascal da Versão 2 é cerca de 87% da área da Versão 1 e o tempo mediano de processamento da Versão 2 é cerca de 14% maior que o tempo mediano de processamento da Versão 1. Portanto, evidências empíricas sistemáticas sugerem que DSFLA se beneficia do processo de delimitação do espaço de soluções tanto na qualidade do rastreamento quanto no tempo médio de processamento por *frame*.

5.4 O Desempenho do DSFLA no Desafio OPE 100

Embora o objetivo deste trabalho seja comparar o desempenho do rastreador DSFLA com outros rastreadores que empregam a metodologia SI, um esforço extra foi feito para comparar os resultados de qualidade de rastreamento com outros rastreadores que estão disponíveis publicamente no *benchmark Hanyang* (Wu, Lim e Yang, 2013) e que empregam diferentes abordagens (nenhum deles empregam a abordagem SI).

Os resultados de qualidade dos rastreadores disponíveis no *benchmark* foram obtidos por meio dos desafios OPE 50 e OPE 100, em que os rastreadores são submetidos ao teste OPE via resultados médios da AUC após o rastreamento dos 50 primeiros e dos 100 primeiros vídeos do *benchmark*, respectivamente.

O desafio OPE para o DSFLA só pode ser estimado por meio de um intervalo de confiança para OPE 100 uma vez que o DSFLA operou 15 vídeos escolhidos ao acaso de um universo dos 100 vídeos disponíveis no *benchmark* (não é possível fazer uma comparação com os resultados do desafio OPE 50, pois, trata-se de universos de amostragem diferentes).

O *benchmark* possui 39 rastreadores avaliados e disponibiliza a classificação dos 10 rastreadores de melhor desempenho em relação aos testes OPE 50 e OPE 100.

Os resultados da variável AUC dos rastreadores submetidos ao desafio OPE 100 disponibilizados no *benchmark* vão de 0,364 a 0,462. Os dez primeiros rastreadores classificados com os melhores resultados estão listados na legenda da Figura 6. São eles: o rastreador STRUCK de Hare, Saffari e Torr (2011) com AUC igual a 0,462 (o primeiro da lista), o SCM de Zhong, Lu e Yang (2012) com AUC igual a 0,448 (o segundo da lista), o CXT de Dinh, Vo e Medioni (2011) com AUC igual a 0,413 (o terceiro da lista), o ASLA de Jia, Lu e Yang (2012) (o quarto da lista), o TLD de Kalal, Matas e Mikolajczyk (2010) (o quinto da lista), o CSK de Henriques et al. (2012) (o sexto da lista), o LSK de Liu, Huang et al. (2011) (o sétimo da lista), o VTD de Kwon e Lee (2010) (o oitavo da lista), o OAB de Grabner, Grabner

e Bischof (2006) (o nono da lista) e o VTS de Kwon e Lee (2011) com AUC igual a 0,364 (o décimo da lista).

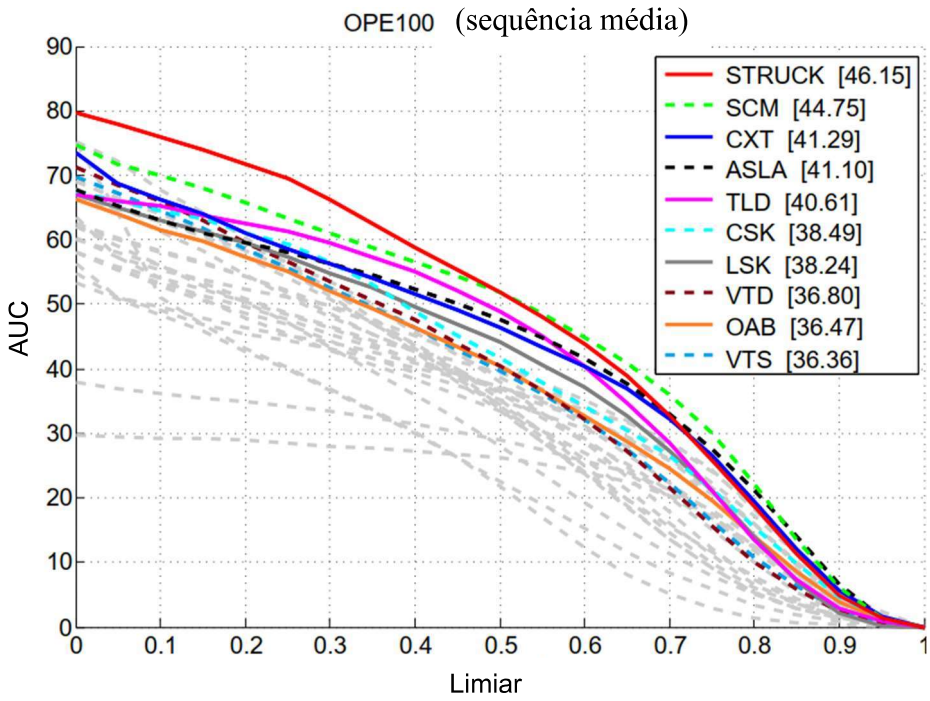

Figura 6. A curva média da taxa de sucesso por métrica de Pascal de uma sequência de 100 vídeos.

Fonte: Obtido da base de dados pública (*open access*) *visual tracking Hanyang benchmark* (Wu, Lim e Yang, 2013) diretamente de http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html, acessado em 04/04/2021.

Usando os dados da Tabela 3 e a Equação (37), o intervalo de 95% de confiança da AUC é dado por $[0{,}396; 0{,}496]$. Portanto, pode-se concluir com 95% de confiança, que o DSFLA se posiciona entre a sexta e a primeira posição na classificação dada pelo *benchmark* ao desafio OPE 100.

Um importante resultado que merece ser destacado, é que 90% dos valores compatíveis da AUC, estimado pelo intervalo de 95% de confiança do DSFLA, superam o resultado do rastreador TLD (*Tracking-Learning-Detection* – significa: "Rastreamento-Aprendizado-Detecção"). Este resultado é importante, uma vez que, o rastreador TLD continua sendo uma referência de rastreamento de alvos em vídeos e é constantemente considerado um rastreador competidor em vários outros trabalhos, por exemplo, em Kalal, Mikolajczyk e Matas, (2012).

O TLD, segundo Kalal, Matas e Mikolajczyk (2010), é baseado no conceito de aprendizado em tempo real em que o alvo é aprendido à medida que ele assume novas aparências. O algoritmo é composto de três módulos: o *tracking* (T), o aprendizado (L) e o detecção (D).

Resumidamente, o TLD rastreia um alvo da seguinte maneira: o módulo de *tracking* estima a posição do alvo por fluxo óptico entre dois *frames* consecutivos; o módulo de detecção utiliza classificadores baseados em árvores e correlação normalizada em várias regiões do *frame* corrente. Na sequência, uma fusão dos resultados, independentemente gerados pelos módulos T e D, é realizada para gerar um exemplo mais refinado de alvo e da posição. O módulo L é responsável por treinar os classificadores com os exemplos gerados na etapa anterior de fusão.

Atualmente, o TLD conta com 2110 citações em outros artigos e 14 patentes internacionais o citam, só de janeiro de 2020 a março de 2021 houve 1585 citações (ieeexplore.ieee.org/6104061/metrics, acessado em 26/04/2021).

6 *Conclusões*

Este trabalho propôs uma nova abordagem para o aprimoramento de algoritmos SI adaptados para problemas de otimização dinâmica, promovendo um balanço entre transferência de conhecimento e a diversidade de partículas. A proposta foi direcionada para aplicação em problemas de rastreamento *online* de um único alvo em vídeos digitais, portanto, aplicados em problemas de otimização dinâmica com espaço de soluções discreto. Esta abordagem também propôs uma delimitação do espaço de soluções ao redor de uma região promissora, baseada na previsão da posição do alvo obtida pelo modelo de Alisamento Exponencial Duplo Robusto a valores extremos, o AEDR. A meta-heurística escolhida para implementação da proposta foi baseado no algoritmo SFLA (Eusuff, Lansey e Pasha, 2006). A versão aprimorada do SFLA proposta nesta tese é chamada de SFLA dinâmico ou DSFLA.

A qualidade do DSFLA foi comparada com outros rastreadores cuja abordagem é baseada em SI: PSO (baseado na meta-heurística BBPSO proposto por Kennedy, (2003)), ADSO (proposto por Bae et al. (2016)), SFLA (também proposto nesta tese), SSA (proposto por Zhang et al., (2020)). As variáveis de qualidade do desempenho observadas foram: o tempo médio de processamento por *frame* e a AUC da taxa de sucesso por métrica de Pascal. Os rastreadores foram submetidos ao desafio OPE com base numa amostra aleatória de 15 vídeos, num universo de 100, disponíveis publicamente no *benchmark Hanyang* (Wu, Lim e Yang, 2013).

Os resultados finais apresentados neste trabalho são estáveis e homogêneos. Em termos do tempo médio de processamento, os resultados mostram que o DSFLA é o segundo rastreador mais rápido. Os tempos médios de processamento por *frame* são significativamente diferentes entre os rastreadores, exceto entre SFLA e DSFLA. Entretanto, empiricamente, o rastreador DSFLA é sistematicamente cerca de 10% mais rápido que o SFLA. Isso demonstra que o DSFLA é mais vantajoso ao delimitar o espaço de soluções e aproveitar as boas soluções do *frame* anterior em relação ao SFLA.

Em termos de AUC, o rastreador DSFLA, apresenta resultados significativamente melhores que os resultados dos rastreadores PSO e ADSO. Entretanto, empiricamente, o DSFLA supera os resultados dos rastreadores SFLA e SSA em mais de 70% dos vídeos (inclusive, mais de 80% dos mais desafiadores dentre os vídeos selecionados). A taxa de

sucesso do DSFLA é cerca de 7,2 a 76,6% maior quando comparada com a taxa de sucesso dos outros rastreadores.

Enquanto à qualidade da previsão do modelo AEDR, a variância e o quadrado do viés do erro de previsão, dados pelo RMSE, não alcançam 5% da área do *frame*. E a distância Euclidiana média entre as posições dos alvos previstas e as estimadas pela meta-heurística, não alcança 4% de erro. Ambas as afirmações são declaradas com 95% de confiança. Portanto, pode-se concluir que as previsões são significativamente homogêneas, não enviesadas e precisas.

Também é possível concluir que a área mediana sob a curva da taxa de sucesso por métrica de Pascal do rastreador proposto, sem a delimitação do espaço de soluções (versão 2 da Tabela 5), é cerca de 87% da área do DSFLA. Também, o tempo mediano de processamento do rastreador proposto, sem a delimitação do espaço de soluções, é cerca de 14% maior que o tempo mediano de processamento do DSFLA. Portanto, pode-se concluir que as previsões do AEDR fornecem boas estimativas na busca de regiões mais promissoras, auxiliando o processo de delimitação do espaço de soluções. Ou seja, a delimitação do espaço de soluções proposta constitui numa importante técnica para melhorar o desempenho do DSFLA.

Quando o desempenho do DSFLA, em relação ao desafio OPE 100, é comparado com o desempenho de outros 39 rastreadores disponíveis publicamente no *benchmark*, o DSFLA fica entre a sexta e a primeira posição na classificação geral (com 95% de confiança). No pior cenário, o DSFLA possui resultados compatíveis com o rastreador TLD (Kalal, Matas e Mikolajczyk, 2010).

Portanto, o rastreador proposto é vantajoso e eficiente tanto em termos de tempo médio de processamento por *frame* quanto em qualidade do rastreamento dada pela AUC da taxa de sucesso por métrica de Pascal se comparados aos outros rastreadores considerados neste trabalho.

Os resultados apresentados pelo DSFLA, o classificam como um rastreador de propósito geral, rápido e estável no rastreamento de qualquer alvo em situações de luz ambiente controlada, com oclusão mínima do alvo e/ou alvos com características distintas às características do fundo. Também, O DSFLA apresenta estabilidade e robustez nos resultados na presença de rotações e/ou movimentos rápidos do alvo (ou da câmera) e/ou de imagens não nítidas ou ruidosas.

Trabalhos futuros poderão ser conduzidos com o objetivo de melhorar o desempenho do DSFLA em situações mais desafiadoras e torná-lo mais robusto às variações de luz ambiente, às variações de escala do alvo e às oclusões mais severa dos alvos. Para superar esses desafios,

a estratégia de pesquisa poderá considerar as seguintes linhas: a inserção de múltiplas populações de partículas alocadas em diferentes sub-regiões do espaço de soluções no processo de seleção de partículas; a elaboração de um esquema adaptativo da quantificação do número de partículas no processo de transferência de conhecimento em função das similaridades entre as sub-regiões do espaço de soluções de dois *frames* consecutivos; e, a inclusão, no modelo de aparência do alvo, da característica HOG (Dallal e Triggs, 2005).

7 *Referências*

L. ANTON-CANALIS; HERNANDEZ-TEJERA, M.; SANCHES-NIELSEN, E. **Particle Swarm as Video Sequence Inhabitants for Object Tracking in Computer Vision.** IEEE 6th Inter. Conf. on Intelligent Systems and Applications, Vol. 2: 604 – 609, **2006**.

M. ARULAMPALAM; MASKELL, S.; GORDON, N; CLAPP, T. **A Tutorial on Particle Filter for Online Nonlinear/non-Gaussian Bayesian Tracking.** IEEE Trans. on Signal Processing, 50: 174 – 188, **2002**.

C. BAE; KANG, K.; LIU, G.; CHUNG, Y. Y. **A Novel Real Time Video Tracking Framework Using Adaptive Discrete Swarm Optimization.** Expert Systems with Applications, 64: 385 – 399, **2016**.

B. BASTURK, KARABOGA; D. **An Artificial Bee Colony (ABC) Algorithm for Numeric Function Optimization.** IEEE Swarm Intelligence Symposium, 12 – 14, **2006**.

R. G. BROWN; MEYER, R. F. **The Fundamental Theorem of Exponential Smoothing.** Oper. Res., Vol. 9, No 5: 673 – 685, **1961**.

A. Z. CHEN; HONG, Z. TAO, D. **An Experiment Survey on Correlation Filter-based Tracking.** *arXiv2015*: 1509,05520v1(cs), **2015**.

R. T. COLLINS; ZHOU, X.; THE, S. K. **An Open Access Tracking Testbed and Avaluation Web Site.** In Proceedings of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2005). Beijing, China, 15–16 October **2005**.

D. COMANICIU; RAMESH, V.; MEER, P. **Kernel-Based Object Tracking.** IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 25, No 5, **2003**.

N. DALAL; TRIGGS, B. **Histograms of Oriented Gradients for Human Detection.** In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), Piscataway, NJ, USA; Institute of Electrical and Electronics Engineers (IEEE); pp. 886–893, **2005**.

K. DAS SHARMA; CHATTERJEE, A.; RACSHIT, A. **A PSO-Lyapunov Hybrid Stable Adaptative Fuzzy Tracking Control Approach for Vision-Based Robot Navigation.** IEEE Trans. Instr. Meas, 61(7): 1908 – 1914, **2012**.

T. B. DINH; VO, N.; MEDIONI, G. **Context Tracker: Exploring Supporters and Distracters in Unconstrained Environment.** In CVPR, 1177-1184. **2011**.

M. DORIGO; BIRATTARI, M.; STUTZLE, T. **Ant Colony Optimization.** IEEE Trans. Comp. Intell, Magazine,1: 28 – 39, **2006**.

A. ENGELBRECHT. **Computational Intelligence: An Introduction.** John Wiley & Sons, **2002**.

EQUASYS. **Color formats**. **2013**. http://www.equasys.de/colorformat.html (acessado em 12/11/2020).

M. EVERINGHAM; VAN GOOL, L; WILLIANS, C. K.; WINN, J.; ZISSERMAN, A. **The Pascal Visual Object Classes (VOC) Challenge.** Int. J. Comp. Vision, 88: 303 – 338, **2010**.

M. M. EUSUFF; LANSEY, K.; PASHA, F. **Shuffled Frog Leaping Algorithm: A Memetic Meta-Heuristic for Discrete Optimization.** Engineering Optimization, Vol. 38, No 2: 129 – 154, **2006**.

M. L. GAO; LI, L. L.; SUM, X. M.; YIN, L. J.; LI, H. T. **Firefly Algorithm (FA) Based Particle Filter Method for Visual Tracking.** Optik, 126: 1705 – 1711, **2015**.

M. L. GAO; YIN, L. J.; ZEO, G. F.; LI, H. T.; LIU, W. **Visual Tracking Method Based on Cuckoo Search Algorithm.** Optical Engineering, 54(7): 093 – 105, **2015**.

E. S. GARDNER JR. **Exponential Smoothing: State of Art.** J. Forecast, 4(1): 1 – 8, **1998**.

E. S. GARDNER JR. **Exponential Smoothing: The State of Art – Part II.** I. J. Forecast, 22(4): 637 – 666, **2006**.

D. E. GOLDBERG; HOLLAND, J.H. **Genetic Algorithm and Machine Learn.** Machine Learn, 3:39 – 59, **1988**.

N. GORDON; SALMOND, D.; SMITH, A. E. M. **Novel Approach Non-Linear/Non-Gaussian Bayesian State Estimation.** IEEE Proc. On Radar Signal Process, 140: 107 – 113, **1993**.

H. GRABNER; GRABNER, M; BISCHOF. **Real-Time Tracking via On-line Boosting.** In BMVC, Vol. 1, 47–56. **2006**.

S. HARE; SAFFARI, A.; TORR, P. H. S. **Struck: Structured Output Tracking with Kernels.** in ICCV, 263–270. **2011**.

A. C. HARVEY. **Time Series Models.** Prentice Hall/Harvester Weatsheaf, 2nd Edition, **1993**.

F. HENRIQUES; CASERO, R.; MARTINS, P.; BATISTA, J. **Exploiting the Circulant Structure of tracking-by-Detection with Kernels.** in ECCV, 702–715. **2012**.

D. HELD; THRUN, S.; SAVARESE, T. **Learn to Track at 100 fps with Deep Regression Networks.** In *Computer Vision—ECCV 2016. ECCV 2016*; Springer: Cham, Switzerland, Volume 9905, pp. 749–765, **2016**.

X. JIA; LU, H.; YANG, M. -H. **Visual Tracking via Adaptive Structural Local Sparse Apparence Model.** In *CVPR*, 1822–1829. **2012**.

T. KAILATH. **The Divergence and Bhattacharyya Distance Measures in Signal Selection.** IEEE Trans. Comm. Technology, Vol 15: 52 – 60, **1967**.

Z. KALAL; MATAS, J.; MIKOLAJCZYK, K. **P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints.** *In CVPR*, 49–56. **2010**.

Z. KALAL; MIKOLAJCZYK, K.; MATAS, J. **Tracking-Learning-Detection.** *In IEEE Trans. on Patter Analysis and Machine Intelligence, Vol 34, no. 7, pp. 1409-1422,* July **2012**. doi: 10.1109/TPAMI.2011.239.

R. E. KALMAN. **A New Approach to Linear Filtering and Prediction Problems**. *Journal of Basic Engineering*. 82 (1): 35–45, **1960**.

J. KENNEDY. **Small world and Mega-Mind: Effects of Neighborhood Topology on Particle Swarm Performance.** Proc. of the congress on Evolutionary Computation, Vol. 3, 1931–1938. **1999**.

J. KENNEDY. **Bare Bones Particles Swarm.** IEEE Proc. on Swarm Intelligence Symposium, 80–87. **2003**.

J. KENNEDY; EBERHART, R. **Particle Swarm Optimization.** IEEE Proc. Inter. Conf. on Neural Networks, Vol. 4: 1942 – 1948, **1995**.

J. KWON; LEE, K. M. **Visual Tracking Decomposition.** *In CVPR*, 1269–1276. **2010**.

J. KWON; LEE, K. M. **Tracking by Sampling Trackers.** *In ICCV*, 1195–1202. **2011**.

M. LI; PANG, B.; HE, Y.; NIAN, F. **Particle Filter Improved by Genetic Algorithm and Particle Swarm Optimization Algorithms.** *J. Software. Vol. 8*, 666–672, doi:10.4304/jsw.8.3.666.672, **2013**.

Y. LI; ZU, J. **A Scale Adaptive Kernel Correlation Filter Tracker.** *European Conference on Computer Vision*; Springer: Cham, Switzerland; pp. 254–265, **2014**.

B. LIU; HUANG, J.; YANG, L.; KULIKOWSK. **Robust Tracking Using Local Sparse Appearance Model and K-Selection.** *In CVPR*, 1313–1320. **2011**.

M. LOVBERG; KRINK, T. **Extending Particle Swarm Optimization with Self-Organized Criticality.** Proc. of 4th Congress on Evolutionary Computation. Vol. 2, 1588 – 1593, **2002**.

Y. H. MA; MAO, Z. H.; JIA, W. Y.; LI, C. L.; YANG, J. W.; SUM, M. G. **Magnetic Hand Tracking for Human-Computing Interface.** IEEE Trans. Magn, 47(5): 970 – 973, **2011**.

W. L. MARTINEZ; MARTINEZ, A. R. **Computational Statistic Handbook with MatLab.** Chapman & Hall/CRC, 584 pp, **2002**.

M. MAVRAVOUNIOTIS; LI, C.; MAO, C. **A Survey of Swarm Intelligence for Dynamic optimization: Algorithm and Application.** Swarm Evol. Comp. 33: 1–17, **2017**.

S. MIRJALILI; MIRJALILI, S. M.; LEWIS, A. **Grey Wolf Optimizer.** Advances in Engineering Software, 69, 46 – 61, **2014**.

S. MIRJALILI; GANDOMI, A. H.; MIRJALILI, S. Z.; SAREMI, S.; FARIS, H.; MIRJALILI, S. M. **Salp Swarm Algorithm: A Bio-Inspired optimizer for Engineering Design Problems.** Advances in Engineering Software, 114, 163 – 191, **2017**.

M. R. MORELANDE; CHALLA, S.; GORDON, N. J. **Application of Particle Filters to Single-Target Tracking Problems.** Proc. SPIE 5204, Signal and Data Pracessing of Small Targets, (5 January 2004); **2003.** https://doi.org/10.1117/12.503553 (acessado em 11/02/2021).

L. S. NELSON. **Evaluating Overlapping Confidence interval.** J. Qual. Technol. 21: 140–141, **1989**.

E. M. de OLIVEIRA; OLIVEIRA, F. L. C. **Forecasting Mid-Long Term Electric Energy Consumption Through Bagging ARIMA and Exponential Smoothing Methods.** Energy, 144: 776 – 788, **2018**.

R. RARDIN. **Optimization in Operations Research.** Pretice Hall, N. Jersey, **1998**.

T. RATHNAYAKE; GOSTAR, A.K.; HOSEINNEZHAD, R.; TENNAKOON, R.; HADIASHAR, A. B. **On-Line Visual tracking with Occlusion Handling.** *Sensors.*, ***20*, 929**; doi:10.3390/s20030929, **2020**.

F. SARDARI; MOGHADDAN, M. E. **A Hybrid Occlusion Free Object Tracking Method Using Particle Filter and Modified Galaxy Based Search Meta-Heuristic Algorithm.** Applied Soft Computing, 50: 280 – 299, **2017**.

M. G. SHUNG; KIM, S-K. **Efficient jitters compensation Using Double Exponential Smoothing.** Information Sciences ,227: 83 – 89, **2013**.

R. D. SNYDER; KOEHLER, A. B.; ORD, J. K. **Forecasting for Inventory Control with Exponential Smoothing.** Inter. Journal of Forecasting, 18(1): 5 – 18, **2002**.

X. TAO; LI, H.; MAO, C.; WANG, C.; YAP, J. B. H.; SEPASGOZAR, S.; SHIROWZHAN, S; TIMOTHY, R. **Developing Shuffled Frog-Leaping Algorithm (SFLA) Method to Solve Power Load-Constrained TCRTO problems in Civil Engineering.** Adv. Civ. Eng. 1 – 16, **2019**.

A. M. A. TAWAB; ABDELHALIM, M. B.; HABIB, M. B. **Efficient Multi-Feature PSO for Fast Gray Level Object-Tracking.** Applied Soft Computing, 14: 317 – 337, **2014**.

J. VALMADRE; BERTINETTO, L.; HENRIQUES, J. F.; VEDALDI, A.; TORR, P.H. **End-to-End Representation Learning for Correlation Filter Based Tracking.** In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26; pp. 2805–2813, July **2017**.

S. WENG; KUO, C.; TU, S. **Video Object Tracking Using Adaptive Kalman Filter.** J. Visual Commun. Image Represent.,17(6): 1190 – 1208, **2006**.

P. R. WINTERS. **Forecasting Sales by Exponentially weighted moving averages.** Maneg. Sci., 6: 324 – 342, **1960**.

D. H. WOLPERT; MACREADY, W, G. **No Free Lunch for Optimization.** IEEE Trans. on Evolutionary Comp., 1:67 – 82, **1997**.

Y. WU; LIM, J.; YANG, M. H. **Online Object Tracking: A Benchmark.** IEEE Conference. On Computer Vision and Patter Recognition., 2411 – 2418, **2013**. http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html (acessado em 05/06/2020).

A. YAMAZ; JAVEL, O; SHAH, M. **Object Tracking: A Survey.** ACM Comput.Surv.,38, 4, Article 13, 45 pages. **2006**.

H. YANG; SHAO, L.; ZHENG, F.; WANG, L.; SONG, Z. **Recent Advances and Trend in Visual Tracking: A Review.** J. Neurocomputing, 16: 190 – 208, **2011**.

X. YANG. **Firefly Algorithm for Multimodal Optimization.** Lect. Notes Comput. Sci, 169 – 178, **2009**.

X. YANG; DEB, S. **Cuckoo Search via Lèvy Flight.** IEEE Proc. Word Congress on Nature and Biologically Inspired Computing,210 – 214, **2009**.

M. YOKOYAMA; POGGIO, T. **A Contour Based Moving Object Detection and Tracking.** IEEE Inter. Workshop on Visual Survillance and Performance Evaluation of Tracking and Surveillance, 271 – 276, **2005**.

M. ZHAI; CHEN, L.; MORI, G.; ROSHTKHARI, M. J. **Deep Learning of Apparence Models for Online Object Tracking.** In *Computer Vision—ECCV 2018 Workshops. ECCV 2018*; Springer: Cham, Switzerland, **2018**; Available online: http://link.springer.com/conference/eccv (acessado em 23/12/2020).

H. ZHANG; LIU, J; NIE, Z; ZHANG, J; ZHANG, J. **A New Visual Tracking Approach Based on Salp Swarm Algorithm for Abrupt Motion Tracking.** KSII Trans. on Internet and Information Systems, Vol. 1, No 3: 1142 – 1166, **2020**.

H. ZHANG; ZHANG, J.; WU, Q.; QIAN, X.; ZHOU, T.; HENGCHEG, F. U. **Extended Kernel Correlation Filter for Abrupt Motion Tracking.** *KSII Transect. Internet Inf. Syst. 11*, 4438–4460. **2017**.

W. ZHONG; LU, H.; YANG, M. -H. **Robust Object Tracking via Sparsity-based Collaborative Model.** *In CVPR*. 1838–1845. doi:10.1109/CVPR.2012.6247882. **2012**.