Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Alexandre de Vasconcelos Cardoso

Estratégias de Rastreamento Dedicado de Objetos utilizando Inteligência de Enxame

Rio de Janeiro
2019

Alexandre de Vasconcelos Cardoso

Estratégias de Rastreamento Dedicado de Objetos utilizando Inteligência de Enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof.ª Dr.ª Nadia Nedjah
Orientadora: Prof.ª Dr.ª Luiza de Macedo Mourelle

Rio de Janeiro
2019

CATALOGAÇÃO NA FONTE
UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

C268 Cardoso, Alexandre de Vasconcelos
Estratégias de Rastreamento Dedicado de Objetos utilizando Inteligência de Enxame/Alexandre de Vasconcelos Cardoso. – 2019.
120 f.

Orientadora: Nadia Nedjah.
Orientadora: Luiza de Macedo Mourelle.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia Eletrônica - Teses. 2. Sistemas auto-organizáveis - Teses. 3. Sistemas embarcados (Computadores) - Teses. 4. Inteligência coletiva - Teses. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. IV. Título.

CDU 007.52

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta dissertação, desde que citada a fonte.

Assinatura

Data

Alexandre de Vasconcelos Cardoso

Estratégias de Rastreamento Dedicado de Objetos utilizando Inteligência de Enxame

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em: 17 de janeiro de 2019.

Banca Examinadora:

Prof.ª Dr.ª Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof.ª Dr.ª Luiza de Macedo Mourelle (Orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Leandro Augusto Frata Fernandes
Instituto de Computação, UFF

Prof. Dr. Bernardo Sotto-Maior Peralva
Instituto Politécnico, UERJ

Rio de Janeiro
2019

DEDICATÓRIA

Dedico este trabalho à minha filha Alice e à minha esposa Chris, que me alimentam de amor e de esperança todos os dias. Não consigo mensurar a alegria de poder ter vivenciado cada passo do desenvolvimento de minha filhinha de perto durante esses dias de mestrado, vê-la aprender a falar, andar, e ganhar pequenas independências e desenvolver habilidades. Dedico à minha esposa especialmente, que tanto se empenha em cuidar de nossa família e foi minha companheira maravilhosa, conseguimos pois superar juntos todos os momentos difíceis de ser pais de primeira viagem. Também dedico à minha mãe, dona Socorro, que nos ajuda com a Alice sempre que pode e muito nos incentivou; e à minha sogra, dona Amélia (*in memoriam*), que encheu minha casa de alegria enquanto esteve conosco. Não posso deixar de homenagear meu pai, seu Cardoso, e meu sogro, seu Aurélio, ambos *in memoriam*, que muito trabalharam durante suas vidas para que nossas famílias pudessem se fortalecer e pudéssemos alcançar nossos sonhos.

AGRADECIMENTOS

Agradeço primeiramente a Deus pela vida, pela saúde, por todas as oportunidades concedidas a mim, e pela família maravilhosa que o Senhor me deu.

Agradeço às minhas orientadoras, as professoras Nadia Nedjah e Luiza de Macedo Mourelle, pelos ensinamentos, pela disponibilização de infraestrutura e pela convivência durante o curso.

Agradeço aos meus colegas de mestrado Yuri, Luneque, Joelmir, Pedro, Ramon, Reinaldo, Igor, Luigi, Sávio, Noemi e Patrícia pela troca de conhecimentos, pelo companheirismo e pelos momentos de descontração durante o curso. Agradeço especialmente a Yuri e Luneque, que muito me ensinaram sobre o equipamento empregado e às ferramentas de edição de texto, respectivamente.

Agradeço aos professores José Franco Machado do Amaral e Maria Luiza Fernandes Velloso pelos ensinamentos que recebi durante o curso.

Agradeço ao CMASM, em especial ao Capitão de Mar e Guerra (EN) Julio Cesar Simões Pimenta, por ter permitido que eu realizasse este curso. Agradeço à DSAM, em especial ao Capitão de Fragata (EN) Rogério Araújo de Barros, pelo incentivo e a orientação sobre os assuntos de Marinha durante este mestrado.

Agradeço ao PEL-UERJ pelos recursos investidos nesta pesquisa.

Sustentar o fogo que a vitória é nossa.

Segundo sinal de Barroso na Batalha Naval do Riachuelo, 11/06/1865.

RESUMO

CARDOSO, Alexandre de V. *Estratégias de rastreamento dedicado de objetos utilizando inteligência de enxame*. 2019. 120f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2019.

O processamento de imagens é uma importante ferramenta para auxiliar na tomada de decisão. O monitoramento contínuo de qualquer ambiente, tais como áreas públicas e parques industriais, entre outros, permite definir melhores estratégias de ação e decidir o momento correto para agir, dirimindo riscos e potencializando oportunidades. A qualidade da informação resultante do processamento de imagens deve ser boa o suficiente para evitar erros de avaliação de cenário durante o planejamento das ações futuras e a definição das metas a serem atingidas. O tempo para obter essa informação e processá-la é a base do sucesso de qualquer atividade. Assim, a inteligência computacional pode ajudar a acelerar a execução dos procedimentos relacionados às ações planejadas. Em geral, um processo de busca lento acaba sempre atrasando a tomada de decisão de modo que a informação obtida poderá se tornar obsoleta ou insuficiente no momento da tomada da decisão. A técnica *Template Matching* é um dos métodos mais aplicados para localizar padrões em imagens, no qual uma imagem de tamanho reduzido, chamada de alvo, é procurada dentro de outra imagem que representa o ambiente como um todo. Neste trabalho, utiliza-se *Template Matching* via um sistema *co-design* já existente. Faz-se presente um coprocessador para calcular a etapa computacionalmente mais custosa do *Template Matching*, que é o cálculo do coeficiente de Correlação Cruzada Normalizada. O cálculo deste coeficiente permite invariância às alterações globais de brilho nas imagens, porém é computacionalmente mais custosa ao se empregar imagens de dimensões maiores, ou mesmo conjuntos de imagens. Propõe-se investigar seis técnicas diferentes de inteligência de enxame visando acelerar o processo de busca do alvo. Para avaliar o projeto proposto, o tempo de processamento, o número de iterações e a taxa de acerto são comparados. Os resultados mostram que é possível obter abordagens capazes de processar imagens de vídeos com 30 quadros por segundo com uma taxa de acerto média aceitável para detecção do alvo rastreado.

Palavras-chave: *Co-design*; correlação cruzada normalizada; coprocessador; colônia de abelhas; bactérias; pássaro cuco; manada de elefantes; vaga-lumes; fogos de artifício; otimização por enxame de partículas; sistemas embutidos; rastreamento; *template matching*.

ABSTRACT

CARDOSO, Alexandre de V. *Dedicated tracking strategies using swarm intelligence*. 2019. 120f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2019.

Image processing is an important tool to help in decision taking. Continuous observation of some environment, such as public areas and industrial plants, among others, allows defining the best action strategies and deciding the right moment to act, thus reducing risks and magnifying opportunities. The quality of the information resulting from such image processing must be precise enough so that it does not hinder scenario's evaluation for future action planning and goals achievement. The execution time to obtain this information and process it is fundamental for any successful activities. Hence, computational intelligence can help accelerating procedure executions related to planned actions. In general, slow search process always delays decision making in such a way that the recorded data becomes obsolete or insufficient at the decision moment. Template Matching is one of the most used techniques for finding then tracking patterns in images, wherein a small size image, termed the target, is looked for inside another that represents the environment as a whole. In this work, template matching is used via an existing co-design system. A co-processor is used to calculate the most computationally expensive task of template matching, which is the computation of the coefficient of the normalized cross correlation. The computation of this coefficient allows invariance in the case of global brightness changes in image, but it is computationally more expensive when using larger templates and yet more expensive in videos. We propose to investigate six different swarm intelligence based approaches, aiming at accelerating the process of target tracking. To evaluate the proposed project, the metrics regarding the overall processing time, number of iterations and target hit rate are used and compared. The results show that it is possible to obtain search approaches capable of processing videos at a rate of 30 frames per second while achieving an acceptable average hit rate for tracking the target.

Keywords: Co-design; co-processor; bacterial foraging optimization; cuckoo search; elephant herding optimization; fireflies; fireworks; particle swarm optimization; embedded systems; object tracking; template matching; normalized cross correlation.

LISTA DE FIGURAS

1 Objeto com silhueta destacada. 24

2 Macro-componentes da arquitetura do sistema de rastreamento.. 71
3 Macro-arquitetura do coprocessador. 72
4 Micro-arquitetura do Bloco 1 73
5 Micro-arquitetura do Bloco 2 74
6 Micro-arquitetura do Bloco 3 75
7 Micro-arquitetura do componente SINCRO 76
8 Placa Xilinx PicoZed 7Z015 77
9 Ambiente da ferramenta SDK 78

10 Imagens de referência empregadas nos testes 80
11 Função objetivo no espaço de busca para cada imagem de referência 81
12 Resultados para CS sem uso de coprocessador 84
13 Resultados para CS com auxílio de coprocessador operando em modo serial 84
14 Resultados para CS com auxílio de coprocessador operando em modo *pipeline* 85
15 Resultados para ABC sem uso de coprocessador 86
16 Resultados para ABC com auxílio de coprocessador operando em modo serial 86
17 Resultados para ABC com auxílio de coprocessador operando em modo *pipeline* 87
18 Resultados para EHO sem uso de coprocessador 88
19 Resultados para EHO com auxílio de coprocessador operando em modo serial 88
20 Resultados para EHO com auxílio de coprocessador operando em modo *pipeline* 89
21 Resultados para BFOA sem uso de coprocessador 90
22 Resultados para BFOA com auxílio de coprocessador operando em modo serial 90
23 Resultados para BFOA com auxílio de coprocessador operando em modo *pipeline* 91
24 Resultados para FFA sem uso de coprocessador 92
25 Resultados para FFA com auxílio de coprocessador operando em modo serial 92
26 Resultados para FFA com auxílio de coprocessador operando em modo *pipeline* 93
27 Resultados para FWA sem uso de coprocessador 93
28 Resultados para FWA com auxílio de coprocessador operando em modo serial 94
29 Resultados para FWA com auxílio de coprocessador operando em modo *pipeline* 95
30 Comparação de tempo de processamento (ms) sem uso de coprocessador . 96

LISTA DE FIGURAS

31 Comparação de número de iterações sem uso de coprocessador. 96
32 Comparação de taxa de acerto (%) sem uso de coprocessador 97
33 Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo serial 97
34 Comparação de número de iterações com auxílio de coprocessador operando em modo serial 98
35 Comparação de taxa de acerto (%) com auxílio de coprocessador operando em modo serial 98
36 Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline* 99
37 Comparação de número de iterações com auxílio de coprocessador operando em modo *pipeline* 100
38 Comparação de taxa de acerto (%) para coprocessador com auxílio de coprocessador operando em modo *pipeline*. 100
39 Comparação de tempo de processamento (ms) sem uso de coprocessador . 102
40 Comparação de número de iterações sem uso de coprocessador. 103
41 Comparação de taxa de acerto (%) sem uso de coprocessador 103
42 Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo serial 104
43 Comparação de número de iterações com auxílio de coprocessador operando em modo serial 104
44 Comparação de taxa de acerto (%) com auxílio de coprocessador operando em modo serial 105
45 Novos resultados para ABC com auxílio de coprocessador operando em modo *pipeline* 105
46 Novos resultados para FFA com auxílio de coprocessador operando em modo *pipeline* 106

LISTA DE TABELAS

1 Novas configurações dos parâmetros para as técnicas 102

2 Comparação de tempo de processamento (ms) sem uso de coprocessador . 121
3 Comparação de desvio padrão do tempo de processamento (ms) sem uso de coprocessador 121
4 Comparação do número de iterações sem uso de coprocessador 122
5 Comparação de desvio padrão do número de iterações sem uso de coprocessador. 122
6 Comparação da taxa de acerto (%) sem uso de coprocessador 122
7 Comparação de desvio padrão da taxa de acerto (%) sem uso de coprocessador 123
8 Comparação do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial 123
9 Comparação de desvio padrão do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial. 123
10 Comparação do número de iterações com auxílio de coprocessador operando em modo serial 124
11 Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo serial 124
12 Comparação da taxa de acerto (%) com auxílio de coprocessador operando em modo serial 124
13 Comparação de desvio padrão da taxa de acerto (%) com auxílio de coprocessador operando em modo serial 125
14 Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline* 125
15 Comparação de desvio padrão de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline*. 125
16 Comparação de número de iterações com auxílio de coprocessador operando em modo *pipeline* 126
17 Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo *pipeline*. 126
18 Comparação da taxa de acerto (%) com auxílio de coprocessador operando em modo *pipeline* 126
19 Comparação de desvio padrão da taxa de acerto (%) com auxílio de coprocessador operando em modo *pipeline* 127
20 Comparação do tempo de processamento (ms), sem uso de coprocessador 127
21 Comparação de desvio padrão do tempo de processamento (ms), sem uso de coprocessador 127
22 Comparação do número de iterações sem uso de coprocessador 128

LISTA DE TABELAS

23 Comparação de desvio padrão do número de iterações sem uso de coprocessador . . . 128
24 Comparação da taxa de acerto (%) sem uso de coprocessador . . . 128
25 Comparação de desvio padrão da taxa de acerto (%) sem uso de coprocessador . . . 129
26 Comparação do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial . . . 129
27 Comparação de desvio padrão do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial. . . . 129
28 Comparação do número de iterações com auxílio de coprocessador operando em modo serial . . . 130
29 Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo serial . . . 130
30 Comparação da taxa de acerto média (%) com auxílio de coprocessador operando em modo serial . . . 130
31 Comparação de desvio padrão da taxa de acerto média (%) com auxílio de coprocessador operando em modo serial . . . 131

LISTA DE ALGORITMOS

1 Algoritmo do pássaro cuco 43
2 Algoritmo de colônia artificial de abelhas 47
3 Operador de atualização de clã 49
4 Operador de separação 50
5 Algoritmo baseado no comportamento social dos elefantes nas manadas . . 50
6 Algoritmo da colônia de bactérias 54
7 Algoritmo dos vaga-lumes 59
8 Cálculo da posição de uma faísca 62
9 Cálculo da posição de uma faísca específica, com distribuição Gaussiana . . 62
10 Algoritmo dos fogos de artifício 63
11 Algoritmo de otimização por enxame de partículas 67

LISTA DE SIGLAS

ABC	*Artificial Bee Colony*
ACM	*Active Contour Model*
BFOA	*Bacterial Foraging Optimization Algorithm*
CC	Correlação Cruzada
CCN	Correlação Cruzada Normalizada
CS	*Cuckoo Search*
EHO	*Elephant Herding Optimization*
EP	*Evolutionary Programming*
FFA	*Firefly Algorithm*
FPGA	*Field Programmable Gate Array*
FWA	*Fireworks Algorithm for Optimization*
GA	*Genetic Algorithm*
GPS	*Global Positioning System*
MGbSA	*Modified Galaxy based Search meta-heuristic Algorithm*
MSE	*Mean Square Error*
OCR	*Optical Character Recognition*
PDF	*Probability Density Function*
PID	Proporcional-Integral-Derivativo
PSO	*Particle Swarm Optimization*
RMSE	*Root Mean Square Error*
RN	Redes Neurais
SAD	*Sum of Absolute Differences*
SSD	*Sum of Square Differences*
SSDA	*Sequential Similarity Detection Algorithm*
TM	*Template Matching*

SUMÁRIO

INTRODUÇÃO **17**

1 RASTREAMENTO DE OBJETOS **22**
1.1 Metodologias 23
1.2 *Template Matching* 27
1.3 Aplicações da Metodologia *Template Matching* 29
1.4 Considerações Finais 30

2 TRABALHOS RELACIONADOS **31**
2.1 Técnica *Template Matching* 31
2.2 Otimização Aplicada a Rastreamento 35
2.2.1 Técnica Baseada no Comportamento do Pássaro Cuco 35
2.2.2 Otimização por Enxame de Partículas 36
2.2.3 Filtro de Kalman e Filtro de Partículas 37
2.2.4 Outras Técnicas 38
2.3 Implementações em Hardware 39
2.4 Considerações Finais 40

3 TÉCNICAS DE INTELIGÊNCIA DE ENXAME **41**
3.1 Comportamento do Pássaro Cuco 41
3.1.1 Funcionamento da Técnica CS 42
3.1.2 Algoritmo do Pássaro Cuco 43
3.1.3 Algumas Aplicações da Técnica CS 43
3.2 Comportamento de Abelhas 44
3.2.1 Funcionamento da Técnica ABC 45
3.2.2 Algoritmo da Colônia de Abelhas 46
3.2.3 Algumas Aplicações da Técnica ABC 46
3.3 Comportamento de Elefantes 47
3.3.1 Funcionamento da Técnica EHO 48
3.3.1.1 Operador de Atualização de Clã 48
3.3.1.2 Operador de Separação 49
3.3.2 Algoritmo da Manada de Elefantes 50
3.3.3 Exemplo de Aplicação da Técnica EHO 51
3.4 Comportamento de Bactérias 51
3.4.1 Funcionamento da Técnica BFOA 51
3.4.1.1 Quimiotaxia e Comportamento de Enxame 51
3.4.1.2 Reprodução das Bactérias 53
3.4.1.3 Eliminação e Dispersão 53
3.4.2 Algoritmo da Colônia de Bactérias 53

SUMÁRIO

3.4.3 Algumas Aplicações da Técnica BFO . . . 53
3.5 Comportamento de Vaga-Lumes . . . 56
3.5.1 Funcionamento da Técnica FFA . . . 57
3.5.2 Algoritmo dos Vaga-Lumes . . . 58
3.5.3 Algumas Aplicações da Técnica FFA . . . 58
3.6 Comportamento dos Fogos de Artifício . . . 60
3.6.1 Funcionamento da Técnica FWA . . . 60
3.6.2 Algoritmo dos Fogos de Artifício . . . 62
3.6.3 Algumas Aplicações da Técnica FWA . . . 63
3.7 Otimização por Enxame de Partículas . . . 65
3.7.1 Funcionamento da Técnica PSO . . . 65
3.7.2 Algoritmo do PSO . . . 67
3.8 Considerações Finais . . . 68

4 PLATAFORMA DE AVALIAÇÃO . . . **69**
4.1 Motivação para o *Co-Design* . . . 69
4.2 Arquitetura do Coprocessador . . . 71
4.3 Plataformas de Implementação . . . 76
4.3.1 Plataforma de *Hardware* . . . 76
4.3.2 Plataforma de *Software* . . . 77
4.4 Considerações Finais . . . 78

5 ANÁLISE DOS RESULTADOS . . . **79**
5.1 Metodologia de Avaliação . . . 79
5.2 Análise dos Resultados por Técnica . . . 83
5.2.1 Técnica Baseada no Comportamento do Pássaro Cuco . . . 83
5.2.2 Técnica Baseada no Comportamento de Abelhas . . . 85
5.2.3 Técnica Baseada no Comportamento de Elefantes . . . 87
5.2.4 Técnica Baseada no Comportamento de Bactérias . . . 89
5.2.5 Técnica Baseada no Comportamento de Vaga-Lumes . . . 91
5.2.6 Técnica Baseada no Comportamento de Fogos de Artifício . . . 93
5.3 Comparações entre as Técnicas . . . 95
5.3.1 Operação sem Coprocessador . . . 95
5.3.2 Operação com Auxílio de Coprocessador Serial . . . 97
5.3.3 Operação com Auxílio de Coprocessador *Pipeline* . . . 99
5.4 Novas Configurações . . . 101
5.5 Considerações Finais . . . 106

6 CONCLUSÕES E TRABALHOS FUTUROS . . . **108**
6.1 Conclusões . . . 108
6.2 Trabalhos Futuros . . . 110

REFERÊNCIAS . . . **112**

APÊNDICE A – Resultados . . . **121**

INTRODUÇÃO

PROCESSAMENTO de imagens é uma importante ferramenta para a tomada de decisão. O monitoramento contínuo do ambiente permite definir melhores estratégias e decidir o momento correto para agir, dirimindo riscos e potencializando oportunidades. A qualidade da informação deve ser boa o suficiente para evitar erros de avaliação de cenário durante o planejamento das ações futuras e a definição das metas a serem atingidas, como no provisionamento de recurso financeiros em um município, ou mesmo na aplicação de medidas de segurança em um grande evento.

O tempo necessário para obter uma informação e processá-la é diretamente responsável pelo sucesso das ações a serem empreendidas. Por exemplo, a observação do espaço aéreo permite reconhecer a presença de aeronaves e avaliar se há possíveis rotas de colisão. Da mesma forma, assim como o piloto de um avião observa o ambiente para planejar o voo, um artefato móvel autoguiado usa as imagens do ambiente (SALMOND, 2013). Numa situação de incêndio, o rápido apontamento do foco facilita o acionamento do combate ao fogo e dos resgates, reduzindo o impacto negativo (YUAN et al., 2016). Em um centro comercial, a observação do comportamento dos clientes pode trazer informações sobre a propensão dos mesmos em efetuar suas compras (MERAD et al., 2016).

O processamento de imagens e de vídeos proporcionou avanços a diversas áreas importantes de pesquisa. O surgimento de novos sensores e novos equipamentos capazes de capturar, armazenar, editar e transmitir imagens aceleraram o processo de decisão, permitindo definir estratégias com risco menor e maior taxa de acerto. O tempo para obter uma informação verdadeira e processá-la é diretamente responsável pelo sucesso das atividades; assim, a inteligência artificial pode ajudar a acelerar a execução dos procedimentos. Um processo de busca lenta pode atrasar a tomada de decisão de modo que a informação poderá se tornar obsoleta ou insuficiente (JIHANG et al., 2012).

O rastreamento em imagens é o processo que envolve a busca, a identificação e o acompanhamento de um elemento visual previamente escolhido em imagens maiores

(YILMAZ; JAVED; SHAH, 2006). A partir da identificação, a estimativa da posição do alvo em imagens sucessivas permite o acompanhamento da trajetória dele nos quadros de um vídeo, ao longo do deslocamento pelo ambiente. A primeira imagem da sequência de quadros demanda um esforço maior de localização, pois não há necessariamente uma tendência de posicionamento do alvo. Para as imagens seguintes, a localização no quadro anterior sugere a posição inicial de busca, caso o alvo permaneça parado ou o sistema de aquisição de imagens consiga mantê-lo no mesmo enquadramento ao longo do vídeo. Essa condição inicial de busca permite restringir a busca do alvo no entorno de uma vizinhança, o que economiza tempo de busca e tende a aumentar a chance de acerto.

Um alvo, para ser identificado em meio a uma imagem complexa, deve possuir características singulares para que possa se distinguir junto ao ambiente. Em (YILMAZ; JAVED; SHAH, 2006), são analisadas características para identificação do alvo, tais como cor, bordas e texturas. Daí, é possível identificar elementos de referência pertencentes aos objetos sob investigação ou mesmo separar partes da imagem para buscar similaridade com o alvo. O movimento do objeto durante o rastreamento pode causar alterações em sua aparência, como variações no tamanho e rotações. Dessa forma, transformações matemáticas como rotação e escalonamento podem ser aplicadas ao modelo do alvo na tentativa de oferecer mais possibilidades de comparação junto aos objetos suspeitos. Ainda assim, o processo de rastreamento pode ser dificultado por oclusões dos objetos, ruído nas imagens, mudanças na iluminação do ambiente, deformações e movimentos complexos do objeto, o que dificulta a verificação de similaridade. A perda de informação devido à projeção em 2D de um objeto tridimensional também dificulta o reconhecimento do alvo nas imagens. O modelo adotado para alvo possui informações espaciais e de aparência, e constitui o padrão a ser localizado.

A técnica *Template Matching* (TM) é uma das mais aplicadas para encontrar padrões em imagens (AHUJA; TULI, 2013). Trata-se basicamente de contar a ocorrência de uma imagem menor que representa o alvo, dentro de outra imagem maior que representa o ambiente como um todo. Dentre as técnicas de TM, a verificação do coeficiente de Correlação Cruzada Normalizada (CNN) é vastamente empregada devido às suas propriedades de invariância às alterações globais de brilho nas imagens (PERVEEN; KUMAR; BHARDWAJ, 2013), porém é computacionalmente mais custosa ao se empregar imagens de dimensões maiores, ou mesmo conjuntos de imagens. Nesta técnica, é calculado o Coeficiente de

Correlação de Pearson (PCC – *Pearson Correlation Coefficient*), outro nome pelo qual a CNN é conhecida.

O processamento computacional da busca pode exigir em excesso do *hardware* que vai efetuar a procura. Os recursos físicos disponíveis são finitos, como memória, energia elétrica e capacidade de processamento, e dificultam o trabalho sobre grandes volumes de dados (TAVARES, 2016). Torna-se necessário projetar sistemas de busca que suportem a demanda, a partir da otimização dos recursos disponíveis. Uma possibilidade de processamento de imagens com alocação de recursos otimizada para aumentar o desempenho é o uso de dispositivos de *hardware* reconfiguráveis do tipo FPGA (*Field-Programmable Gate Array*) e de circuitos integrados para aplicações específicas (ASIC – *Application Specific Integrated Circuits*).

O uso de *hardware* dedicado ajuda a efetuar mais rapidamente os cálculos pertinentes ao PCC. Contudo, a seleção adequada de qual setor da imagem será comparado com o alvo é fundamental para a localização rápida. Uma busca sequencial por todos os *pixels* da imagem maior poderia levar mais tempo do que o disponível para obter a informação necessária. Assim, o uso de uma técnica de busca otimizada é capaz de acelerar a localização do alvo na imagem maior (TAVARES, 2016). O emprego conjunto de *hardware* dedicado com *software* de busca otimizada torna possível efetuar localização e rastreamento de alvos em imagens referentes a vídeos com 30 quadros por segundo.

Dentre as diversas técnicas de busca otimizada, as de inteligência de enxame são baseadas em comportamentos sociais do mundo real, baseados em interação e organização de agentes computacionais de recursos simples para a realização de tarefas (MORAIS, 2017). Os agentes interagem com o ambiente e cooperam entre si para na tentativa de produzir soluções para problemas complexos, explorando pontos nas vizinhanças de outros anteriormente classificados como bons. Um problema comum a estes algoritmos é a escolha adequada dos parâmetros, o que impacta diretamente no tempo de processamento para atingir o ponto ótimo global, ou mesmo o impacto na susceptibilidade de convergir para ótimos locais, proporcionando falsos positivos e prejudicando a tomada de decisão.

O objetivo da presente dissertação é apresentar um sistema de rastreamento dedicado de objetos em imagens que opere embutido em dispositivo FPGA e utilize inteligência de enxame para a busca. Um subsistema implementado em *software* desenvolvido durante o trabalho efetua a busca de forma otimizada para encontrar o alvo. O sistema permite a

seleção da técnica de busca a ser utilizada pelo FPGA e então executada, o que possibilita a análise comparativa dos desempenhos das técnicas.

Esta dissertação propõe-se a implementar TM em um sistema *co-design* desenvolvido em (TAVARES, 2016), usando seis técnicas diferentes de inteligência de enxame para acelerar o processo de busca, sendo elas baseadas no comportamento: do pássaro cuco, de abelhas, de elefantes, de bactérias, de vaga-lumes e de fogos de artifício. Além disso, faz-se presente o uso de um coprocessador desenvolvido em (TAVARES, 2016) para calcular a etapa computacionalmente mais custosa do TM, que é o PCC. Para cada técnica, foi feita a configuração dos parâmetros para processar imagens pertencentes a vídeos com 30 quadros por segundo. Para avaliar o desempenho de cada técnica, foram considerados o tempo médio de processamento, o número médio de iterações e a taxa média de acerto, e seus respectivos desvios padrões. Todos os experimentos foram repetidos 5000 vezes, divididos em 50 conjuntos de 100 repetições, para permitir o cálculo da média e do desvio padrão.

O sistema recebeu três abordagens diferentes para operação de *hardware*, desenvolvidas e aplicadas em (TAVARES, 2016). A primeira fez uso apenas do processador de uso geral do FPGA para a busca. A segunda empregou um coprocessador operando em modo serial para calcular o PCC, com a finalidade de acelerar o processamento. A terceira empregou um coprocessador operando em modo paralelo, com *pipeline*, para processar as imagens ainda mais rápido.

Cada técnica de busca foi implementada no ambiente de programação da ferramenta *Smart Vídeo Development Kit* (SDK), onde houve acesso ao *hardware* através das três abordagens desenvolvidas em (TAVARES, 2016). Inicialmente, cada técnica teve seus parâmetros configurados empiricamente para obter a taxa de acerto média de pelo menos 60% e tempo médio de processamento de até 33 ms. Depois, os parâmetros de cada técnica de busca foram reconfigurados para se obter uma taxa média de acerto de pelo menos 90%, com novo estudo comparativo.

O restante desta dissertação está estruturado em seis capítulos. Primeiramente, o Capítulo 1 apresenta a definição de rastreamento de alvos em imagens, analisa os elementos e características usados para identificar objetos e para mensurar a similaridade entre imagens. É dada ênfase à TM, técnica adotada neste trabalho para identificar os alvos escolhidos.

Em seguida, o Capítulo 2 faz um levantamento bibliográfico em que as teorias e experimentos associados a busca otimizada são analisadas, com ênfase naqueles com implementação em FPGA. Este capítulo mostra também a análise de conteúdos de periódicos, anais de congresso, livros, dissertações e teses.

O Capítulo 3 apresenta as técnicas de busca otimizada empregadas neste trabalho, mostra seus algoritmos e exemplifica suas aplicações. Em seguida, o Capítulo 4 mostra o estudo do *hardware* desenvolvido em (TAVARES, 2016) para a implementação do cálculo do PCC através de um coprocessador para melhorar o tempo de processamento da imagem durante a busca pelo alvo.

Em sequência, o Capítulo 5 apresenta a metodologia adotada para os experimentos e mostra os resultados da implementação em linguagem C das técnicas de busca otimizada no subsistema de *software*, com o estudo comparativo entre os desempenhos das técnicas. Por fim, o Capítulo 6 apresenta as conclusões sobre quais técnicas mostraram-se mais aptas ao problema de busca e rastreamento de alvos em imagens, em função das metas de desempenho definidas na metodologia. A dissertação termina com sugestão de trabalhos futuros.

Capítulo 1

RASTREAMENTO DE OBJETOS

ESTE capítulo apresenta metodologias para a solução do problema de identificação de um alvo pré-definido em imagens, de maneira a possibilitar seu rastreamento em imagens sucessivas. Diferentes processos podem ser usados para encontrar alvos ao trabalhar sobre características das imagens, como a presença de elementos-chave, diferenças de intensidade, presença de cores específicas, entre outras. A rapidez para se obter a localização do alvo em um quadro pode facilitar encontrá-lo no quadro seguinte, logo permitir o rastreamento.

Uma boa estimativa de como o alvo se move na imagem pode ajudar a rastreá-lo de maneira mais eficiente. A atenta observação do ambiente possibilita identificar objetos de interesse, permite traçar estratégias e definir o momento adequado para agir a partir do acompanhamento do alvo. A qualidade da informação deve ser suficientemente precisa para não induzir ao erro, seja de avaliação do cenário ou de planejamento futuro. Simultaneamente, quanto mais rápido se obtém as informações pertinentes, tão logo será possível traçar metas e planejar os próximos passos, aumentando a certeza e dirimindo riscos.

Entende-se por rastreamento (*tracking*) em imagens o processo de busca, identificação e acompanhamento de um elemento visual previamente definido (alvo) em uma imagem maior ou em um conjunto de imagens (YILMAZ; JAVED; SHAH, 2006). A técnica TM compara o modelo do elemento, o *template*, com recortes da imagem investigada, e quantifica o quão similares eles são de modo a identificar o alvo (AHUJA; TULI, 2013). Em outras palavras, encontra ocorrências de uma subimagem dentro de outra. Para aplicações em tempo real, o estado da arte neste assunto sinaliza para sistemas de *tracking* que empregam meta-heurísticas para otimizar a busca, assim como para implementação em *hardware* de parte do processamento.

Os interesses em efetuar rastreamento envolvem aplicações como: análise do movimento humano, vigilância de área, interações homem/computador (direção do olhar), monitoramento do trânsito, navegação (planejamento do trajeto e desvio de obstáculos), controle de processos em linhas de produção industriais. O processamento realizado nas imagens aproveita características das mesmas relacionadas com a aplicação e que possam acelerar a obtenção da resposta. Por exemplo, o rastreamento de um foco de superaquecimento faz uso das intensidades das cores relativas aos *pixels* de uma imagem térmica, e não necessariamente de um alvo pré-definido (LIU et al., 2017). Em caso de incêndio, um modelo prévio para o alvo poderia não ser eficiente devido aos danos decorrentes do fogo, ou mesmo à dificuldade de se captar imagens que não sejam térmicas por causa da fumaça, por exemplo.

A Seção 1.1 apresenta metodologias para identificação de objetos em imagens. A Seção 1.2 contém a definição da metodologia TM, definida como estratégia de reconhecimento de padrões a ser usada neste trabalho. A Seção 1.3 lista exemplos de aplicações do TM. Por fim, a Seção 1.4 traz as considerações finais do capítulo.

1.1 Metodologias

O problema da identificação para rastreamento de objetos em imagens é enfrentado segundo diferentes abordagens, entre as quais as características das imagens, particularidades dos objetos, movimentos observáveis ao longo de uma sequência de imagens, entre outros. Cada abordagem é adequada e eficiente para a solução de uma gama particular de desafios, de acordo com detalhes no modelo adotado e de necessidades de projeto, como tempo de processamento, verba disponível, consumo de energia elétrica etc. Diversos autores analisaram características de objetos em imagens para facilitar o processamento e a identificação. A seguir, casos pertinentes são expostos, com a ênfase na técnica TM, adotada neste trabalho para solucionar o problema.

Em (YILMAZ; JAVED; SHAH, 2006), os métodos de rastreamento foram estudados, assim como as dificuldades associadas e como as resolver. Segundo eles, são fatores que podem prejudicar o rastreamento: alto tempo para processamento; perda de informação devido à projeção em 2D de um objeto tridimensional; ruído nas imagens; movimentos complexos dos objetos; natureza articulada ou não alinhada dos objetos; oclusão total ou parcial dos objetos; formatos complexos dos objetos; mudanças na iluminação; e neces-

sidades de processamento em tempo real. Também listam algumas maneiras comuns de representar características de aparência dos objetos no contexto de rastreamento, como a Função Densidade de Probabilidade (*Probability Density Function* – PDF), *templates*, modelos de aparência ativa e modelos de aparência multivista. A PDF da aparência de um objeto pode ser Gaussiana, não paramétrica ou ser representada por histogramas. A PDF das características da aparência (cor, textura) pode ser obtida a partir de regiões determinadas por formatos de modelos pré-definidos, como o interior de uma elipse ou de um contorno. O *template* (modelo do alvo) usa formatos simples ou silhuetas, possui informações espacial e de aparência. Uma desvantagem sua é exibir apenas a aparência relativa a um ponto de vista. Os modelos de aparência ativa são conjuntos formados pelas informações de formato e de aparência, com pontos de referência. Já os modelos com aparência multivista possuem informações de diferentes pontos de vista do mesmo objeto. Um figo maduro pode ser identificado pela coloração na Figura 1. Já a definição de um padrão de silhueta, conforme o contorno do fruto assinalado em amarelo na mesma figura, permite efetuar a contagem do número de frutos.

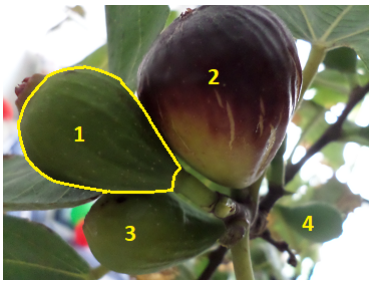


Figura 1: Objeto com silhueta destacada.

Em (YILMAZ; JAVED; SHAH, 2006), os autores apresentam maneiras de utilizar representações de formato e aparência de maneira conjunta para aplicação de rastreamento. Pelo modelo da análise de pontos, o objeto é representado por um ponto (centroide) ou por um conjunto de pontos. Esta representação é adequada para rastrear objetos que ocupem pequenas regiões na imagem. Pelas formas geométricas primitivas, a representação através de retângulo, elipse etc., o movimento do objeto com este tipo de representação é aplicado em casos de translação, rotação, transformação afim e transformação projetiva. São adequadas para objetos simples e rígidos.

Pelo modelo da silhueta e contorno do objeto, o contorno está associado aos limites fronteiriços do objeto (YILMAZ; JAVED; SHAH, 2006). A região no interior do contorno

é denominada silhueta. São adequados a aplicações em objetos não rígidos e de formas complexas. Já os modelos de formato articulado possuem objetos articulados que são compostos por partes do corpo mantidas com junções entre elas. Por exemplo, o corpo humano é um objeto articulado com tórax, pernas, pés, braços, mãos e cabeça conectados por junções. A relação entre os segmentos do corpo é governada pela cinemática de movimento do modelo. Cilindros podem representar braços, pernas e tórax, por exemplo. Os modelos de esqueleto possuem um elemento de referência do objeto que pode ser obtido através do eixo medial de sua silhueta. Este tipo de modelo é empregado como representação de formato, e pode ser utilizado tanto para modelagem e reconhecimento de objetos articulados, quanto para objetos rígidos.

Um objeto a ser rastreado deve ser singular perante os outros e o meio onde está. Segundo (YILMAZ; JAVED; SHAH, 2006), as características para verificar a unicidade do objeto são através de cor, bordas, fluxo óptico e textura. A cor é influenciada pela distribuição espectral da iluminação e pela refletividade da superfície do objeto. Alguns exemplos de representação são: RGB (*Red, Green, Blue* – vermelho, verde e azul, respectivamente) para percepção não-uniforme, Luv (luminância e componentes u e v) e HSV (*Hue, Saturation, Value* – matiz, saturação e valor, respectivamente) para percepções uniformes. As bordas referem-se a mudanças na intensidade das imagens. Este método é menos sensível às variações na iluminação do que a cor o é. O fluxo óptico é um denso campo de vetores de deslocamento que define a translação de cada pixel na região. É empregado em aplicações baseadas em movimento para segmentação e rastreamento. A textura é a medida de variação de intensidade de uma superfície, e quantifica propriedades como suavidade e regularidade.

Diferentes abordagens para detecção de objetos são destacadas, tais como detecção de pontos de referência, segmentação, subtração do fundo da imagem (*backgroung subtraction*) e classificadores supervisionados. Elas podem empregar a informação de um único quadro ou informações temporalmente obtidas de uma sequência de quadros, para reduzir as falsas detecções.

A técnica baseada em pontos de referência apresenta a identificação de pontos de interesse na imagem, obtidos através de variações de intensidade dos *pixels* segundo direções especificadas (horizontal, por exemplo), ou mesmo através de processos matemáticos com filtros Gaussianos em diferentes escalas. Sua vantagem é a invariância a mudanças na

iluminação e no ponto de observação. A técnica *backgroung subtraction* ressalta diferenças entre os fundos dos quadros de uma sequência, de modo que as mudanças significativas entre as imagens representam o movimento dos objetos presentes, e seus respectivos *pixels* são marcados para facilitar o processamento dos quadros seguintes. A técnica da segmentação divide a imagem em partições com características particulares, de modo que duas partições sejam distinguíveis devido a essas peculiaridades. Para a segmentação, é necessário conjugar o critério para uma obter partição considerada boa, juntamente com o método eficiente de particionamento. Já a técnica de classificadores supervisionados aplica o aprendizado automático de diferentes vistas de um objeto a partir da análise de um conjunto previamente fornecido de exemplos do mesmo objeto. O objetivo é obter uma função matemática que faça a classificação da maneira desejada. A seleção de características específicas dos objetos, como a área e o histograma, é importante no processo de classificação.

O conceito de rastreamento trazido por (YILMAZ; JAVED; SHAH, 2006) o define como a geração da trajetória de um objeto ao longo do tempo, localizando suas posições em cada quadro do vídeo. Os eventos associados a encontrar o objeto-alvo e em seguida relacionar sua posição a cada quadro podem ocorrer conjuntamente ou separadamente. Quando ocorre junto, a estimativa de posição é passada para o processamento do quadro seguinte. Se separadamente, possíveis regiões a cada quadro são obtidas pela média das detecções do alvo, e o rastreamento corresponde ao caminho onde houve a detecção ao longo da sequência de quadros. Em ambos os casos, os objetos recebem como representação modelos de formato e/ou de aparência.

As metodologias de rastreamento, segundo (YILMAZ; JAVED; SHAH, 2006) são divididas em: rastreamento de ponto, de *kernel* (núcleo) e de silhueta. No primeiro, objetos detectados em quadros consecutivos são representados por pontos, e a associação entre os mesmos baseia-se no estado prévio do objeto, o que pode incluir posição e movimento. No rastreamento de ponto, destacam-se as aplicações: do filtro de Kalman, que estima o estado de um sistema linear e emprega distribuição Gaussiana para efetuar predição e correção; e do filtro de partículas, em que as variáveis de estado não-Gaussianas. No rastreamento de núcleo, emprega-se o formato e a aparência do objeto para a detecção, como um objeto de formato elíptico e histograma característico, por exemplo. Os objetos são rastreados através do registro do movimento do núcleo através dos quadros consecutivos,

decorrente de translação ou rotação. Nesta metodologia, faz-se presente o TM, em que se busca pelo objeto-alvo em um quadro a partir das proximidades de onde ele ocupava no quadro anterior. Em certas abordagens podem ser aplicadas transformações ao *template*, como rotação e escalamento. Com relação à metodologia de rastreamento através da detecção por silhueta, modela-se formatos complexos (mãos, cabeça, ombros), ou seja, que não podem ser descritos por formas geométricas simples. É necessário obter uma descrição precisa dos formatos dos objetos de referência, na qual se emprega histograma de cores, bordas dos objetos e contorno, por exemplo.

1.2 *Template Matching*

A técnica *Template Matching* (TM) é uma importante estratégia para a identificação e o rastreamento de alvos em imagens, conforme visto na Seção 1.1. Ela pode ser empregada em processamento de sinais, processamento de imagens, reconhecimento de padrões e compressão de vídeo (PERVEEN; KUMAR; BHARDWAJ, 2013).

A definição de TM, segundo (AHUJA; TULI, 2013), como uma técnica para categorizar objetos, cujo objetivo é contar a quantidade de ocorrências de um objeto-alvo pré-definido (também chamado de *template*) no interior de uma imagem maior, através de medições de similaridade entre o *template* e os elementos dessa imagem, usando como referências valores de correlação e de ângulo de fase. Eles definiram a correlação como medida de concordância entre duas variáveis. É realizada uma comparação entre pequenas imagens, em que uma amostra do alvo é a referência para encontrar outros objetos também similares.

Em (AHUJA; TULI, 2013), os autores listam diferentes estratégias matemáticas para o cálculos da similaridade. A Correlação Cruzada Normalizada (CCN) e a raiz quadrada da Soma dos Quadrados das Diferenças (*Sum of Square Differences* – SSD) são usados para medir a similaridade; já a Soma das Diferenças Absolutas (*Sum of Absolute Differences* – SAD) e o Algoritmo de Detecção Sequencial de Similaridade (*Sequential Similarity Detection Algorithm* – SSDA) possuem emprego em reconhecimento de padrões e compressão de vídeo, por exemplo.

Em (PERVEEN; KUMAR; BHARDWAJ, 2013), os autores analisaram diferentes metodologias de TM, como correlação (com ênfase na CCN), casamento baseado em escala de tons de cinza, e casamento de borda. Os autores dividiram as abordagens de TM entre:

as baseadas em características dos objetos e as baseadas em área. São exemplos daquelas baseadas em características as que empregam elementos de referência, como curvas ou pontos, ou mesmo um modelo de superfície. As baseadas em área são conhecidas como métodos correlacionais ou métodos TM propriamente ditos.

Em (PERVEEN; KUMAR; BHARDWAJ, 2013), os autores avaliaram quanto ao desempenho junto às variações de iluminação, ao contraste e a mudanças na posição de observação do alvo. A correlação é obtida medindo-se a similaridade entre duas imagens de mesmas dimensões e devidamente alinhadas: uma imagem pré-definida como referência e um recorte obtido da imagem maior. A verificação de similaridade através do cálculo da Correlação Cruzada (CC) decorre do somatório simples das multiplicações aos pares dos valores dos *pixels* das imagens, conforme apresentado na Equação 1:

$$\rho = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \cdot \sqrt{\sum_{i=1}^{n} y_i^2}}, \tag{1}$$

onde x_i e y_i são as intensidades dos *pixels* em uma imagem de dimensões n x n. A CC perde robustez no cálculo de TM quando as imagens são submetidas a variações de brilho, como no exemplo de um míssil voando no céu ensolarado, que pode resultar em falsos positivos com outros objetos brilhantes.

A solução para este problema é a normalização da CC, ou seja, a aplicação da Correlação Cruzada Normalizada (CCN), em que os valores médios dos *pixels* do alvo e da imagem maior são subtraídos de cada ponto durante o cálculo. Assim, o brilho médio de cada imagem é removido, fazendo com que o clareamento ou o escurecimento de cada imagem não afete o resultado final (PERVEEN; KUMAR; BHARDWAJ, 2013). O resultado passa a ser representado pelo intervalo [-1;1], onde 1 significa a perfeita similaridade entre as imagens, ao passo que -1 implica na similaridade inversa perfeita. Além disso, a CC também sofre falhas em caso de rotações de grande magnitude e também em caso de mudança de escala entre as duas imagens. O cálculo da CCN é efetuado através da Equação 2:

$$\rho = \frac{\sum_{i=1}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}}, \tag{2}$$

onde o resultado é representado por ρ, como também é chamado o PCC, $\bar{x}$ e $\bar{y}$ são os valores médios de x e y.

Segundo (BROWN, 1992), a posição em que ocorre o melhor casamento entre o *template* e a imagem maior ocorre no pico na correlação cruzada. A CC deve ser

normalizada para que seus resultados não sejam influenciados por variações da intensidade de brilho e contraste na imagem.

1.3 Aplicações da Metodologia *Template Matching*

Em (ZITOVA; FLUSSER, 2003), os autores estudaram a medição de similaridade, baseada na sobreposição de duas ou mais imagens da mesma cena captadas em momentos diferentes, por ângulos diferentes e/ou por sensores diferentes. A similaridade entre imagens recebeu emprego em classificação multiespectral, monitoramento de ambiente, detecção de mudanças, integração de imagens etc. Houve a separação do processamento nas seguintes etapas: detecção de característica; casamento de característica (imagem x padrão); estimativa no modelo de transformação (quantificação); reamostragem e transformação da imagem (usa funções de mapeamento).

Em (WEBER; LEFÈVRE, 2012), foi analisada a busca por áreas da imagem onde tanto o alvo quanto o fundo casam com padrões pré-definidos, com aplicação de TM em imagens binárias. Foram definidos: elementos estruturantes para representar os padrões; e um operador acerta-ou-erra (*hit-or-miss*).

Em (RYAN; HANAFIAH, 2015), o método TM foi empregado para a leitura dos campos de cartões de identificação pessoal. Um sistema de reconhecimento óptico de caracteres (*Optical Character Recognition* – OCR) é usado para extrair a informação textual do cartão a partir de uma fotografia. Inicialmente, é aplicado um pré-processamento composto de escalamento, conversão em tons de cinza e binarização. Em seguida, as linhas de texto são extraídas em função das diferenças de aspecto entre o texto e o fundo da imagem. Aplicaram então a segmentação às linhas para obter os trechos de texto a serem aplicados ao OCR. Por fim, os autores aplicaram TM na identificação dos caracteres.

Em (YOO et al., 2014), os autores propuseram um processamento com TM baseado em histograma para obter um modelo tolerante a variações na escala. Introduziram o conceito de gradiente dominante. Calcula-se os gradientes dominantes do *template*, de modo a representá-lo através de composições dos histogramas destes gradientes. Então, os histogramas de regiões arbitrárias da imagem são comparados com o histograma do *template*, através de janelas deslizantes de várias escalas e posições entre ele e o alvo, para determinar a diferença de escala entre eles. Os resultados apresentaram-se mais robustos à variação de escala do que outros métodos de TM.

1.4 Considerações Finais

O presente capítulo apresentou metodologias para a identificação de objetos, decorrentes da presença de alguma característica de referência ou da verificação de similaridade entre um objeto-alvo pré-definido e elementos de uma imagem maior. A metodologia *Template Matching* recebeu ênfase e foi definida para ser empregada neste projeto. O próximo capítulo apresenta um estudo de alguns trabalhos atuais que abordam o problema de localização em imagens, o uso de técnicas de inteligência de enxame, e a implementação em *hardware* destas técnicas.

Capítulo 2

TRABALHOS RELACIONADOS

ESTE capítulo apresenta alguns trabalhos relacionados ao problema de rastreamento, implementado através da técnica de TM. Entende-se por rastreamento (ou *tracking*) a busca, a identificação e o acompanhamento de um elemento visual previamente definido e que está presente em uma imagem maior ou em um conjunto de imagens, de modo que se possa identificar sua trajetória (YILMAZ; JAVED; SHAH, 2006). O elemento é conhecido como *template*. A técnica de TM compara o *template* com recortes da imagem a ser investigada, e quantifica quão similares eles são para efetuar a identificação deste alvo (AHUJA; TULI, 2013). Para melhorar o desempenho de sistemas que efetuam o rastreamento até que possibilitem aplicações em tempo real, o estado da arte neste assunto sinaliza para o emprego de meta-heurísticas visando a otimização de desempenho, assim como para implementação em hardware de parte do processamento.

A Seção 2.1 aborda a definição de TM e cita alguns exemplos de aplicação. A Seção 2.2 lista algumas técnicas de otimização aplicadas a *tracking* presentes em artigos recentes. São abordados os casos particulares das meta-heurísticas do pássaro cuco, filtro de Kalman, filtro de partículas e o *Particle Swarm Optimization* (PSO), além de outras técnicas de otimização. A Seção 2.3 aborda trabalhos nos quais houve implementação de *tracking* em hardware. A Seção 2.4 contém as considerações finais sobre o capítulo apresentado.

2.1 Técnica *Template Matching*

O TM é definido como uma medida de similaridade entre uma imagem-padrão (chamada de *template*) e um recorte alinhado e sobreposto da imagem de entrada, o que equivale a calcular a similaridade entre duas imagens de dimensões iguais (ZITOVA; FLUSSER, 2003;

AHUJA; TULI, 2013; WEBER; LEFÈVRE, 2012). O objetivo é encontrar o *template* (ou uma variante sua) nesta imagem maior.

Em (PERVEEN; KUMAR; BHARDWAJ, 2013), metodologias diferentes de TM são analisadas quanto ao desempenho junto às variações de iluminação, contraste e de posição da observação do alvo em uma imagem. Alguns trabalhos analisados em (PERVEEN; KUMAR; BHARDWAJ, 2013) buscaram aumentar a robustez em relação à capacidade de manter a eficiência na presença destas variações. Os métodos baseados em área e em características também foram definidos. Um sistema com filtro casado é proposto para aumentar a relação sinal ruído para facilitar a identificação (BROWN, 1992). Na presente dissertação, não há aplicação de filtro para remoção de ruído, porém isso poderia melhorar a capacidade de detecção de alvos na sequência de quadros de um vídeo.

As operações matemáticas entre duas imagens obtidas no mesmo cenário ou que contenham partes comuns são abordadas em (ZITOVA; FLUSSER, 2003), com finalidade de identificar as diferenças, encontrar alvos etc. São empregados pontos de vista diferentes e/ou diferentes sensores. As abordagens são classificadas conforme suas naturezas por área ou característica. A etapas empregadas são: detecção de característica, casamento de característica, definição da função de mapeamento e transformação da imagem/reamostragem.

Em (AHUJA; TULI, 2013), os métodos de avaliação para TM são listados, como o Coeficiente de Correlação de Pearson (PCC) e a Correlação Cruzada. O TM é empregado para classificar objetos, em que se pretende encontrar uma pequena imagem (*template*) dentro de outra maior ou para reconhecer imagens similares. O TM e o PCC foram empregados nesta dissertação para encontrar o alvo em cada imagem.

Em (WEBER; LEFÈVRE, 2012), a técnica TM foi modelada a partir de elementos estruturantes previamente definidos para analisar imagens binárias, com um operador *hit or mis*. Eles aplicam o modelamento também ao fundo da imagem.

Em (PERVEEN; KUMAR; BHARDWAJ, 2013), os autores abordam diferentes métodos de medição de similaridade para imagens e seus desempenhos são avaliados quanto às variações de iluminação, de brilho e de posição de observação. O trabalho aborda o método baseado em características, no qual o casamento entre o *template* e a imagem de referência é testado com relação a referenciais e a pontos de controle. Os referenciais são pontos, curvas ou modelos que devem casar com o *template*. O objetivo do método

baseado em características é encontrar o melhor par *template*/referência. Este trabalho também aborda o método baseado em área, também chamado de método correlacional, no qual a combinação entre a detecção de referenciais e o casamento dos pontos de referência durante o rastreamento em movimento e manejo das situações de oclusão.

As definições propostas em (PERVEEN; KUMAR; BHARDWAJ, 2013) para Correlação Cruzada e Correlação Cruzada Normalizada são:

- Correlação Cruzada (CC): é a soma das multiplicações dos *pixels* correspondentes entre o *template* e o recorte. Este método possui problemas com relação à robustez, pois pode ser prejudicado por alterações no brilho global das imagens.

- Correlação Cruzada Normalizada (CCN): para cada pixel do *template* e do recorte, é subtraída a média dos *pixels* da respectiva figura. Esta técnica apresenta duas contribuições para melhora dos resultados: a invariância ao brilho global e coeficiente de correlação normalizado para o intervalo [-1, 1]. A variação de brilho ou escurecimento das imagens não afeta o resultado, já que o brilho médio das imagens é subtraído de seus *pixels*. O valor final do coeficiente de correlação é normalizado, de modo que duas imagens idênticas proporcionam correlação com valor 1,0, enquanto imagens inversamente correlacionadas de modo perfeito produzem o coeficiente -1,0. Assim, a CCN é invariante para alterações lineares de brilho e contraste e, devido à facilidade de implementação em *hardware*, é indicada para aplicações em tempo real. A CC apresenta falhas em caso de rotações de grande magnitude ou mudanças de escala entre as duas imagens. Nesta dissertação, a CCN foi empregada para a detecção dos alvos.

A técnica baseada em escala de cinza é explicada em (PERVEEN; KUMAR; BHARDWAJ, 2013), na qual este algoritmo estende a ideia original de detecção de padrão baseada em correlação, aumentando sua eficiência e permitindo a busca por ocorrências do padrão apesar da orientação dele. São testadas diversas possibilidades de rotação do *template* para se comparar com os recortes. Já o método baseado em borda limita o esforço computacional às bordas dos objetos. O formato destes é fundamentalmente dependente das bordas, daí são analisados apenas os *pixels* nas vizinhanças das mesmas, economizando processamento e obtendo melhoria em termos de desempenho.

A correlação cruzada é uma abordagem clássica para efetuar TM ou reconhecer algum padrão em uma figura, conforme definido em (BROWN, 1992). É uma medida

de similaridade entre uma imagem e um *template* definido. Se o *template* casa com um recorte da imagem definido a partir de um ponto, a função de correlação atinge um pico neste ponto. A correlação cruzada deve ser normalizada para que seus resultados não sejam influenciados por variações da intensidade da imagem no que concerne a brilho e contraste.

Em (BROWN, 1992), é abordado o processamento sobre imagem ruidosa, ou seja, que possui distorções que não podem ser removidas através de transformações, e seu pico de correlação pode não ser claramente detectável. Para certos tipos de ruído, como o ruído branco aditivo, o filtro de correlação cruzada que maximiza a razão entre a potência do sinal e a potência esperada do ruído inserido na imagem é o próprio *template*. Em outros casos, a imagem pode ser filtrada antes de se aplicar a correlação cruzada para que esta propriedade seja mantida. O pré-filtro deve ser usado quando o ruído presente satisfaça certas propriedades estatísticas. Estas técnicas de pré-filtro combinado com filtro de correlação cruzada são chamadas de técnicas de filtros casados, e podem ser computacionalmente intensos. Uma classe de algoritmos mais eficientes do que os de correlação cruzada são os algoritmos de detecção sequencial de similaridade. Estes são baseados nas diferenças de *pixels* entre duas imagens e inclui o valor médio dos *pixels* do *template* e do recorte. A eficiência se dá devido ao fato da correlação empregar muitas multiplicações e normalização. Como os algoritmos de detecção sequencial de similaridade não são normalizados, o mínimo encontrado representa o melhor casamento, com máxima similaridade.

Em (JENKINS et al., 2016), é proposta uma aplicação de TM inserido em rastreamento em tempo real. Eles propõem uma metodologia de *tracking* rápida e comprimida, na qual é empregado um custo computacional dito mínimo, conjugado com alta taxa de quadros por segundo no vídeo. O TM com CCN foi incorporado para aumentar a robustez do algoritmo em via da necessidade em se trabalhar com processamento em tempo real. Os algoritmos de *tracking* utilizam um *template* do alvo para criar um modelo que será usado nos quadros para rastreá-lo. Os autores também listam técnicas a serem aplicadas a rastreamento, tais como classificadores Bayesianos, filtros de correlação, medidas de similaridade e filtro de partículas para determinar a mais provável localização do alvo nos *frames*.

2.2 Otimização Aplicada a Rastreamento

Aplicações em rastreamento em tempo real precisam superar o desafio de diminuir o custo computacional e permitir que todos os cálculos sejam efetuados no tempo necessário. Várias técnicas de otimização de busca foram implementadas em rastreamento na tentativa de reduzir o número de pontos do espaço de busca a serem verificados para encontrar o pico de similaridade entre o *template* e os recortes das imagem captadas. A presente seção lista algumas tentativas recentes para solucionar este problema. Diversas meta-heurísticas bioinspiradas são citadas, como nas técnicas de otimização por enxame de partículas e *Cuckoo Search*. O poder da maioria das modernas meta-heurísticas provem delas imitarem práticas efetivas aplicadas na natureza, especialmente sistemas biológicos que sofreram seleção natural ao longo de milhões de anos (YANG, 2010).

2.2.1 Técnica Baseada no Comportamento do Pássaro Cuco

A técnica do pássaro cuco (YANG; DEB, 2009; YANG, 2010) é baseada no comportamento parasitário aplicado no processo reprodutivo de certas espécies do pássaro cuco, tais como as *Ani* e *Gira*. Esta é uma das técnicas de busca otimizada implementada nesta dissertação. Elas possuem uma estratégia agressiva de reprodução: parasitar ninhos de outras espécies para aumentar a chance de sobrevivência. As fêmeas do cuco põem seus ovos em ninhos que já possuem alguns ovos da espécie do pássaro dono. Em geral, os ovos do cuco chocam antes daqueles que já estavam lá. A primeira ação instintiva do cuco recém-nascido é empurrar os outros ovos para fora do ninho para não ter que dividir a comida com outros filhotes. O dono do ninho pode descobrir a invasão e tomar duas atitudes: eliminar do ninho o ovo invasor ou abandonar o ninho e construir outro em novo local.

Segundo (YANG, 2010), estudos apontam para um padrão de locomoção realizado por vários pássaros e insetos definido pelo chamado voo de Lévy. Em (REYNOLDS; FRYE, 2007), os autores mostraram em seu estudo que a mosca de frutas *Drosophila melanogaster* explora o ambiente empregando uma série de caminhos curtos e retos de voo, encerrados por uma guinada de 90^o, o que leva a um padrão de busca em escala intermitente que caracteriza o voo de Lévy.

Em (WALIA; KAPOOR, 2014), foi aplicada ao problema de *tracking* uma conjunção da técnica de busca otimizada CS com filtro de partículas, e faz a comparação de desempenho com o filtro de partículas genérico. A técnica CS introduz aleatoriedade na busca,

representada pelo abandono de uma fração dos ninhos. Outra contribuição importante do artigo é uma nova maneira para abordar os erros de escalamento e rotação durante o *tracking*. O desempenho desta variante do CS foi avaliado sobre vídeos sintéticos e padronizados e comparados com o filtro de partículas genérico e com filtro de partículas um baseado em PSO. Os resultados obtidos pelo CS foram mais confiáveis e eficientes, baseados em taxa de sucesso, medida F, robustez e métricas de desempenho. Entretanto, o gasto com tempo foi maior do que o realizado pelo filtro de partículas, apesar de operar satisfatoriamente para vídeos em tempo real capturados em 25 quadros por segundo.

Em (LJOUAD; AMINE; RZIZA, 2014), é proposta uma implementação do CS modificado pela inclusão do filtro de Kalman para aplicar no problema de *tracking*. O filtro de Kalman é um processo recursivo de estimação, no qual podem ser estimados os estados passados, o estado presente e estados futuros. Foram aplicados vários conjuntos de teste, como CAVIAR (FUNDED, 2014). O resultado obtido é superior em desempenho ao obtido pelo PSO em *tracking*, especialmente em relação ao tempo de computação.

2.2.2 Otimização por Enxame de Partículas

A técnica PSO é inspirada pelo comportamento coletivo de peixes e de pássaros. Possui vasta aplicação em problemas de otimização em situações nas quais o espaço de busca é muito amplo e a função objetivo é complexa.

Uma variante adaptativa e discreta da técnica PSO foi proposta em (BAE et al., 2016) para aplicação em *tracking* em vídeo. São obtidos histogramas HSV para criar um modelo para o alvo definido. Em caso de oclusão ou desaparecimento do alvo e posterior reaparecimento, a estratégia de busca é atualizada para recapturá-lo. Os resultados superaram o desempenho do PSO tradicional em acurácia e velocidade de busca e reaquisição dos alvos.

Em (SHA et al., 2015),foi aplicada uma variante categórica da técnica PSO em *tracking*, implementada em linguagem C++ e testada em vídeos. As partículas, de acordo com seu *fitness*, são classificadas dentre as seguintes categorias: partículas de alta qualidade, normais ou de baixa qualidade. Cada categoria recebe um peso inercial diferente, de modo que as partículas de alta qualidade vão mover-se mais lentamente e deverão ficar mais próximas entre si, enquanto os de baixa velocidade irão acelerar para ampliar a busca. Partículas normais manterão sua intenção de movimento. Os resultados mostraram desempenho superior em relação ao PSO canônico, ao filtro de Kalman e ao filtro de

partículas canônico. A variante do PSO apresentou-se mais rápida para detectar objetos com movimentos não lineares, bem como esteve mais estável durante os testes.

A implementação de TM com PCC é apresentada em (TAVARES; NEDJAH; MOURELLE, 2015) empregando as técnicas PSO, busca exaustiva e GA. A técnica TM é utilizada para buscar a similaridade entre *template* e os recortes da imagem, através do PCC. Os resultados quantificam desempenho até 196 vezes superior do PSO junto à busca exaustiva.

Em (ABDEL-KADER; ATTA; EL-SHAKHABE, 2014), os autores aplicaram PSO na detecção de olhos em vídeos em que ocorrem movimentos de cabeça, piscada dos olhos e oclusão, situações em que alguma das pupilas pode não estar visível. São aplicados múltiplos *templates* e é feita uma busca adaptativa a cada recorte. Isto aumenta a acurácia da busca e reduz a complexidade computacional. Os resultados apontam para aumento de robustez e acurácia em relação a outros métodos de detecção de olhos aplicados a deformações de *template*, como o algoritmo genético.

2.2.3 Filtro de Kalman e Filtro de Partículas

O filtro de Kalman é um tipo de filtro Bayesiano que estima o estado de um sistema linear cujas variáveis possuem distribuição Gaussiana, sendo capaz de representar o comportamento de maneira simplificada e rápida (YILMAZ; JAVED; SHAH, 2006; STRENS; GREGORY, 2003). Problemas não-lineares recebem modelos linearizados em torno dos pontos críticos, para que este filtro seja aplicado.

O filtro de partículas consegue trabalhar com variáveis de estado que não sejam Gaussianas (YILMAZ; JAVED; SHAH, 2006; SARDARI; MOGHADDAM, 2017) Ele também usa estimativa Bayesiana. Ele possui dois componentes principais: o modelo de transição de estados e o modelo de observação. A cada passo, o filtro gera algumas partículas (hipóteses) para representar os estados, usando os modelos. O modelo de observação serve para dar pesos às partículas.

Um estimador de velocidade foi utilizado em (STRENS; GREGORY, 2003) para resolver o problema de rastreamento na presença de ruído suficientemente forte para ocasionar a perda do alvo. O estimador faz uso de filtro de Kalman e modelos ocultos de Markov (*Hidden Markov Models* – HMM) para calcular a posição do alvo ao longo de sua trajetória. O sistema implementado foi capaz de rastrear o alvo na presença de baixa relação sinal-ruído com poucos *pixels* de tolerância.

Em (SARDARI; MOGHADDAM, 2017), os autores empregaram um sistema com filtro de partículas e uma variante da meta-heurística das galáxias (MGbSA – *Modified Galaxy based Search meta-heuristic Algorithm*) ao problema de rastreamento em condições de mudança de aparência do *template* e também de iluminação. São analisadas mudanças lentas e rápidas na aparência, bem como a constância dela. O sistema foi capaz de detectar a oclusão do alvo sem perdê-lo, e se mostrou robusto em relação às mudanças de aparência. O bom desempenho foi relacionado com a capacidade do filtro de partículas em trabalhar com problemas não-lineares e não-Gaussianos.

Um sistema de navegação autônomo baseado na identificação de pontos de referência em imagens usando o filtro de Kalman foi proposto em (STUBBERUD; KRAMER; STUBBERUD, 2016). Este sistema emprega processos de reconhecimento e rastreamento para tomada de decisão, onde as distâncias para as referências são estimadas, bem como os ângulos entre a trajetória do veículo e uma plataforma de referência fixa. A partir disso, ele consegue modelar a posição e a velocidade.

O filtro de partículas foi aplicado em (GUSTAFSSON et al., 2002) ao problema de navegação com posicionamento e rastreamento. Sua ênfase nessa técnica foi motivada pelo uso de modelos não-lineares e de ruídos não-Gaussianos, que causariam resultados piores se tivesse sido aplicado o filtro de Kalman. Este trabalho, descreve o emprego de um mapa de referências para navegação como base para o deslocamento de automóveis (trajetória horizontal) e de aeronaves (com elevação). O posicionamento é obtido pela estimativa da localização ocupada. O rastreamento refere-se a perseguir um objeto definido e também a evitar colisões durante o percurso. Os testes foram realizados com análise do desempenho em tempo real, e os resultados obtidos foram considerados compatíveis com os de sistema de navegação por satélite (como o GPS – *Global Positioning System*).

2.2.4 Outras Técnicas

Em (GAO et al., 2016), os autores abordam o uso da meta-heurística inspirada em morcegos aplicada a *tracking*. Ele compara o desempenho de seu algoritmo com o de outras técnicas: filtro de partículas, *meanshift* e PSO. A meta-heurística inspirada em morcegos, proposto em (YANG, 2010) baseia-se no sistema de ecolocalização destes mamíferos, em que são emitidos pulsos sonoros que refletem nos objetos dispostos pelo caminho, produzindo eco. Os morcegos ajustam a frequência dos seus pulsos de acordo com barulho decorrente de outros pulsos emitidos por outros morcegos e pelos ecos. Cada morcego gera uma solução

local após um movimento aleatório. O objetivo é encontrar a melhor solução global após um número determinado de iterações. Os resultados obtidos superaram o desempenho das outras meta-heurísticas na maioria das imagens utilizadas nos testes em relação ao tempo para encontrar o alvo.

A meta-heurística inspirada em abelhas foi aplicada em (MAJI et al., 2016) com transformada discreta de cosseno ao problema de segmentação e categorização de imagens. Neste trabalho, durante a busca pelo alimento, as abelhas são divididas em três grupos: operárias, observadoras e desocupadas. Cabe às observadoras buscar as melhores fontes de alimento, representada pelo *fitness* dos pontos ocupados pelas abelhas no espaço de busca. O autor informa ter reduzido o tempo para realizar as tarefas.

Em (KULKARNI; VARGANTWAR; VIRULKAR, 2015), os autores compararam o desempenho do algoritmo de gradiente descendente de *mean-shift* com o desempenho do filtro de Kalman, com o objetivo de modelar o movimento do alvo para ajudar na otimização da posição do alvo através do tempo usando PSO. Os resultados do filtro de Kalman foram mais eficiente em experimento dinâmico, e mais robusto em caso de oclusão e altas necessidades computacionais.

Um sistema desenvolvido em (ERTAŞ et al., 2008) emprega uma rede neural artificial para segmentação de imagens e detecção de lesões em imagens de mamografia. O sistema extrai regiões das imagens usando a rede, produz mapas de intensidade e aplica a técnica TM. A técnica mostrou-se satisfatória, com desempenho próximo às necessidades para aplicação em tempo real.

2.3 Implementações em Hardware

Em (TAVARES; NEDJAH; MOURELLE, 2016), os autores propuseram a aplicação de TM com PSO em um sistema embarcado com plataforma Zynq XC7Z015 e o uso do PCC, com posterior comparação de desempenho junto à busca exaustiva implementada com TM e PCC no mesmo sistema. O PSO com TM e PCC obteve desempenho até 158 vezes superior ao obtido via busca exaustiva. Este sistema embarcado foi empregado na presente dissertação, com a mudança do PSO por outras seis técnicas de busca otimizada usando inteligência de enxame.

Em (JARRAH; JAMALI; HOSSEINI, 2014) abordam a aplicação de filtro de partículas implementado em FPGA (*Field Programmable Gate Array*) para a identificação de objetos

em ambientes não-lineares e não-Gaussianos. Foram usadas implementações de *pipeline* e abordagem paralela para melhorar a eficiência computacional. Esta implementação em FPGA proporcionou *speed up* de até doze vezes, e pode ser aumentado ao se aumentar o número de partículas.

Em (LIU; CHEN; MA, 2015), os autores empregaram a plataforma Xilinx Zynq-7000 FPGA, processador ARM de núcleo duplo e coprocessador vetorial NEON para detecção e rastreamento de objetos móveis. O algoritmo do sistema é baseado na diferença dinâmica do fundo da imagem. Os resultados mostram a aptidão do sistema para trabalhar em tempo real.

Uma solução do problema de *tracking* foi aplicada em (MERAD et al., 2016) através de um projeto conjunto *hardware* e *software* com uso de FPGA e aplicação de filtro de partículas. O objetivo foi acompanhar o movimento de clientes no interior de lojas. As imagens são susceptíveis a variação de iluminação e oclusão do *template*.

2.4 Considerações Finais

Neste capítulo foram apresentados de maneira breve os conceitos de rastreamento e de TM, além de alguns trabalhos que os empregaram, com ênfase naqueles que definem conceitos aplicados neste projeto, como TM e CCN. Foram abordados também exemplos técnicas de otimização que buscam melhorar o desempenho computacional. Além disso, foram citados alguns trabalhos que utilizaram *hardware* auxiliar para acelerar os cálculos e melhorar o desempenho dos sistemas, de modo a satisfazer aplicações em tempo real, inclusive o trabalho que apresentou a plataforma empregada nesta dissertação para o rastreamento de alvos em imagens. No próximo capítulo serão analisadas em detalhes as técnicas de inteligência de enxame empregadas para otimizar o processo de busca nesta dissertação.

Capítulo 3

TÉCNICAS DE INTELIGÊNCIA DE ENXAME

ESTE capítulo apresenta as seis técnicas de busca otimizada aplicadas nesta dissertação ao problema de rastreamento. Estão presentes os respectivos algoritmos canônicos e um breve histórico do emprego dos mesmos em problemas de complexidade elevada, que inviabilizaria a busca exaustiva. Na Seção 3.1, a técnica inspirada no comportamento reprodutivo do pássaro cuco é apresentada. Em seguida, a Seção 3.2 analisa a o método de busca otimizada baseado no comportamento de colônias de abelhas na busca pelo alimento. A Seção 3.3 apresenta a técnica inspirada no comportamento social de elefantes em suas manadas. A Seção 3.4 traz a técnica de otimização que busca repetir o comportamento das bactérias *E. Coli* na busca pelo alimento, no movimento, na reprodução e na própria sobrevivência das mesmas no meio em que habitam. A Seção 3.5 introduz a técnica inspirada no comportamento dos vaga-lumes. Já a Seção 3.6 aborda uma técnica modelada a partir do movimento e da disseminação das partículas incandescentes de fogos de artifícios durante sua detonação. A Seção 3.7 apresenta a técnica de busca utilizada como referência de desempenho. Por fim, as considerações finais a respeito das técnicas de busca otimizada estão presentes na Seção 3.8.

3.1 Comportamento do Pássaro Cuco

O *Cuckoo Search* (CS) é um algoritmo de busca otimizada introduzido em (YANG; DEB, 2009) e (YANG, 2010) baseado na agressiva estratégia de reprodução de algumas espécie do pássaro cuco, como as *Ani* e *Gira*. A fêmea invade um ninho de outra espécie e põe seu ovo nele, agindo de maneira parasitária. O ovo possui características de tamanho e coloração semelhantes aos dos outros ovos lá presentes do pássaro dono do ninho

(hospedeiro). Assim que o ovo do cuco choca, o pássaro recém-nascido instintivamente empurra os outros ovos para fora do ninho, fato que o transforma no único filhote no local e aumenta a sua chance sobrevivência, já que terá toda a atenção da sua mãe adotiva. Entretanto, o hospedeiro pode acabar descobrindo o ovo do cuco invasor, e pode eliminá-lo ou simplesmente abandonar o ninho e fazer outro em um novo lugar.

3.1.1 Funcionamento da Técnica CS

No Algoritmo 1, todos as posições ocupadas pelos ninhos representam soluções no espaço de busca, já que o cuco põe o ovo em um ninho coletivo aleatoriamente escolhido. A implementação da descoberta contribui para o controle da exploração intensiva e diversificada do espaço de busca, bem como evitar convergência prematura ou lenta (JR et al., 2013). O modelo aplicado no CS obedece a três regras:

1. Cada cuco põe um ovo em uma rodada e o deixa em um ninho aleatoriamente escolhido.

2. Os melhores cucos são mantidos na geração seguinte para dar continuidade à informação das posições com melhor ajuste.

3. O número de ninhos disponível para os cucos parasitarem é fixo, e uma fração P_a dos invadidos é descoberta pelo dono e substituída por novos ninhos aleatoriamente posicionados.

A cada geração, um dos cucos é sorteado para efetuar um deslocamento aleatório, implementado através do passo de Lévy, empregado como estratégia de busca global, cuja magnitude é controlada pela distribuição de Lévy de probabilidade (YANG; DEB, 2009), ao invés da distribuição Gaussiana. Essa magnitude pode ser escalada em decorrência de um fator L ajustável, de acordo com a dimensionalidade do problema, conforme mostrado na Equação 3:

$$x_i^{t+1} = x_i^t + L \cdot Levy(\lambda), \tag{3}$$

onde x_i^{t+1} e x_i^t representam a próxima posição do cuco e a atual, respectivamente, L é a escala do passo de Lévy, e $1 \leq \lambda \leq 3$. Segundo (YANG; DEB, 2009), este passo ajuda a criar soluções afastadas de ótimos locais para mitigar a convergência prematura.

3.1.2 Algoritmo do Pássaro Cuco

O Algoritmo 1 apresenta o deslocamento dos cucos (YANG; DEB, 2009). Os parâmetros a serem ajustados são: fator de escala L do passo de Lévy; número de cucos N_c; número máximo de gerações M; probabilidade de descoberta dos ninhos P_a; e critério de parada. Assim que encontra a solução ou atinge o limite de gerações, o Algoritmo 1 retorna o melhor resultado obtido.

Algoritmo 1 Algoritmo do pássaro cuco

Definir a função objetivo $f(x)$
Definir L, N_c, M, P_a
Gerar posição inicial x_i dos ninhos hospedeiros
$t := 1$
Enquanto $(t \leq M)$ e (critério de parada não atingido) **Faça**
  Escolher um cuco j aleatoriamente e deslocá-lo através do passo de Lévy
  Avaliar a qualidade do novo ponto através de $f(x)$
  Ordenar os ninhos conforme $f(x)$
  Abandonar a fração (P_a) dos piores ninhos e criar novos em torno do melhor ninho
  Manter os (1 - P_a) ninhos que tiverem soluções de melhor qualidade
  Ordenar os ninhos e encontrar o melhor
  $t := t + 1$
Fim Enquanto
Retornar a melhor posição

3.1.3 Algumas Aplicações da Técnica CS

Em (WALIA; KAPOOR, 2014), os autores empregam filtro de Partículas conjugado com CS ao problema de TM para melhorar o processo de reconhecimento do alvo, mesmo se ele tiver sofrido rotações ou variação de escala. Eles testaram o algoritmo proposto com um vídeo sintético padrão, o CAVIAR (FUNDED, 2014). Para avaliar o desempenho do seu algoritmo, os autores empregaram FP comum e FP com PSO (KENNEDY; EBERHART, 1995) para comparação. O desempenho temporal foi suficiente para taxas de vídeo de 25 quadros por segundo. O algoritmo proposto mostrou-se mais confiável e robusto do que os outros dois.

Em (LJOUAD; AMINE; RZIZA, 2014), o filtro de Kalman foi aplicado combinado com CS ao problema de TM. O CS usou o filtro para melhorar a qualidade da posição inicial dos cucos. Os testes foram feitos em diversas bases de dados, incluindo CAVIAR (FUNDED, 2014). Os resultados apresentaram mais acurácia e menor tempo de execução

do que o rastreador baseado em *Particle Swarm Optimization* (PSO) para TM com relação ao tempo de processamento.

A técnica CS foi utilizada em (ILUNGA-MBUYAMBA et al., 2016) para efetuar segmentação em imagens usando o modelo de contornos ativos (*Active Contour Model* – ACM) para identificação de tumores cerebrais em imagens médicas. Foram usadas imagens sintéticas e outras reais para avaliar o sistema. Eles experimentaram representar as imagens em duas abordagens: por coordenadas polares e em coordenadas retangulares, com melhores resultados para a primeira. O desempenho do algoritmo proposto foi superior ao do ACM tradicional.

Uma modificação no algoritmo do CS foi proposta em (WALTON et al., 2011) para melhorar o desempenho da técnica. Nela, ocorre a adição de uma informação de troca entre os ovos de cuco de melhor qualidade (melhores soluções até então) para aumentar a taxa de convergência. O novo algoritmo foi testado em funções objetivo clássicas. Os resultados foram comparados aos de CS, PSO e Evolução Diferencial (STORN; PRICE, 1997), com bom desempenho.

3.2 Comportamento de Abelhas

O Algoritmo de Colônia Artificial de Abelhas (*Artificial Bee Colony* – ABC) é uma técnica de busca otimizada proposta em (KARABOGA, 2005; KARABOGA; BASTURK, 2007) inspirada no comportamento forrageiro das abelhas. As posições ocupadas pelas fontes de alimento no espaço de busca representam possíveis soluções para o problema, e a quantidade de néctar disponível na fonte está associado à qualidade da solução, definida através da função objetivo. Esta técnica apresenta a mimetização das atividades de comunicação, alocação de tarefa, escolha do lugar para o enxame, acasalamento e reprodução, busca por néctar e difusão de feromônio. Três tipos de abelhas são definidos: campeiras (*employed bees*), observadoras (*onloocker*) e exploradoras (*scouts*).

A abelhas campeiras levam consigo as rotas para as fontes de alimento e informam outras abelhas sobre a qualidade da fonte. As observadoras aguardam no ninho novas informações vindas das campeiras. As exploradoras procuram novas fontes de alimento pelo ambiente, independentemente de outras fontes já conhecidas. O número de soluções é a quantidade de campeiras e de observadoras juntas.

3.2.1 Funcionamento da Técnica ABC

Inicialmente no algoritmo, as fontes de alimento criadas a partir de estímulos iniciais aleatórios são visitadas pelas abelhas campeiras e avaliadas através da função objetivo. Caso uma fonte de alimento vizinha seja descoberta por uma campeira com valor ainda melhor, a nova fonte é adotada e substitui a anterior. Caso contrário, ela é rejeitada. Abelhas campeiras compartilham a informação da existência da fonte de alimento com as abelhas observadoras em um lugar chamado de área de dança. As observadoras selecionam para qual fonte irão conforme a probabilidade de ter alimento lá, de acordo com as informações fornecidas pelas campeiras. Assim como as campeiras, as observadoras são capazes de encontrar uma nova fonte de alimento vizinha àquela que visitou e a adotar, se for melhor que a outra. Senão, esta é rejeitada.

Uma abelha observadora artificial escolhe uma fonte de alimento i de acordo com a probabilidade associada com a fonte P_i, calculada pela Equação 4:

$$P_i = \frac{f(x_i)}{\sum_1^{SN} f(x_n)}, \tag{4}$$

onde o termo $f(x_i)$ representa a quantidade de néctar da i-ésima fonte. O número de fontes de alimento é SN, que é o total de abelhas N_a. Uma nova fonte vizinha é definida a partir de uma antiga usando-se a Equação 5:

$$v_{ij} = x_{ij} + \phi_{ij} \cdot (x_{ij} - x_{kj}), \tag{5}$$

onde os termos k e j são definidos de modo que $k \in \{1, 2, ..., SN\}$ e $j \in \{1, 2, ..., D\}$, sendo D o número de dimensões do espaço de busca. A posição k é escolhida aleatoriamente, mas é diferente de i. O termo ϕ_{ij} é um número aleatório pertencente ao intervalo $[-1, 1]$, que controla a escolha de fontes vizinhas de x_{ij} e representa a comparação visual que uma abelha faz entre duas fontes, que ficam próximas entre si. Se x_{ij} e x_{kj} são diferentes, x_{ij} irá variar de acordo com a magnitude da diferença. Quando se está perto da solução ótima, o tamanho deste passo tende a ser pequeno.

As fontes de alimento abandonadas pelas abelhas são substituídas por novas fontes através das abelhas exploradoras. Na técnica ABC, uma fonte será abandonada se não tiver sua qualidade melhorada pelas suas vizinhas por um número pré-determinado de ciclos. Para uma fonte x_i e $j \in \{1, 2, ..., D\}$, a abelha exploradora descobre uma nova fonte de alimento conforme mostrado na Equação 6:

$$x_i^j = x_{min}^j + rand(0, 1) \cdot (x_{max}^j + x_{min}^j), \tag{6}$$

onde x^j_{min} e x^j_{max} são, respectivamente, a mínima e a máxima coordenadas ocupadas pelas abelhas no eixo j, enquanto $rand(0,1)$ representa um número aleatório entre 0 e 1, de modo que a fonte de alimento estará localizada na região do espaço de busca já ocupada pelas abelhas.

Para acelerar a convergência, a nova fonte de alimento é descoberta aleatoriamente nas proximidades da melhor fonte até então descoberta (x_{melhor}), dentro de uma vizinhança E_{exp}, tal como mostrado na Equação 7:

$$v_{ij} = x_{melhor} + (-1)^{rand(0,1)} \cdot (rand(0, E_{exp})), \tag{7}$$

onde v_{ij} é uma posição candidata situada aleatoriamente dentro de uma vizinhança E_{exp} da melhor fonte até então descoberta (x_{melhor}).

Após cada posição candidata v_{ij} de fonte ser proposta e visitada por uma abelha, seu desempenho é comparado com o da fonte antiga. Caso seja melhor, ela é utilizada e a outra fica esquecida. Se for pior, a antiga permanece na memória. Isso significa um mecanismo *guloso* para o operador de seleção entre as fontes (KARABOGA; BASTURK, 2007).

3.2.2 Algoritmo da Colônia de Abelhas

O ABC emprega os seguintes parâmetros de controle: o número N_a de abelhas (campeiras ou observadoras), que é igual ao número SN de fontes de alimento; o valor limite do espaço de busca; o número máximo de ciclos MCN; N_{esg} é o número de iterações que define se uma fonte i está esgotada. O pseudo-código para o ABC é apresentado no Algoritmo 2 (KARABOGA, 2005; KARABOGA; BASTURK, 2007).

3.2.3 Algumas Aplicações da Técnica ABC

Uma adaptação técnica ABC foi aplicada em (KARABOGA; BASTURK, 2007) em problemas de otimização com condições restritas. Eles adotaram o método proposto em (GOLDBERG; DEB, 1991) para trabalhar com restrições, ao invés de aplicar o método guloso. O método empregado é baseado em três regras: toda solução factível é preferível em relação a uma solução incapaz de ser implementada; dentre duas possibilidades factíveis, a que tiver a melhor avaliação será a preferida; dentre duas soluções não factíveis, aquela que tiver a menor violação de restrição será preferida. Os resultados mostraram que é necessário avaliar o tipo de restrição a ser implementado pelo método proposto. Também afirmaram

Algoritmo 2 Algoritmo de colônia artificial de abelhas

Iniciar as fontes de alimento (soluções) x_{ij}, $i = 1, ..., SN$ e $j = 1, ..., D$
Definir N_{esg} e E_{exp}
$ciclos := 1$
Enquanto ($ciclos \leq MCN$) e (critério de parada não atingido) **Faça**
  Produzir novas soluções v_{ij} para as abelhas campeiras usando a Equação 5
  Fazer a avaliação das novas soluções através da função objetivo
  Aplicar o processo guloso de seleção
  Calcular os valores da probabilidade P_{ij} para as soluções x_{ij} usando a Equação 4
  Produzir novas soluções v_{ij} para as observadoras a partir das soluções x_{ij} selecionadas dependendo de P_i e as avaliar pela função objetivo
  Aplicar o processo guloso de seleção
  Determinar a fonte a ser abandonada pela abelha exploradora, se atingir N_{esg}
  Substituir a fonte abandonada (se existir) por uma nova fonte x_{ij} aleatoriamente produzida
  Memorizar a melhor solução (fonte) encontrada até então
  $ciclos := ciclos + 1$
Fim Enquanto
Retornar a melhor solução

que o ABC pode ser aplicado para resolver problemas de engenharia com restrições de condições.

As técnicas ABC, CS, Evolução Diferencial (*Differential Evolution* – DE) e PSO foram empregadas a problemas de otimização numérica em (CIVICIOGLU; BESDOK, 2013). Foram escolhidas mais de cinquenta funções de teste. Resultados mostraram que a taxa de sucesso do CS é muito próxima do DE. A complexidade de execução e o número de avaliações do DE foram as menores dentre as técnicas comparadas. Os desempenhos de CS e PSO foram estatisticamente mais próximos ao do DE do que o desempenho medido para o ABC. Os resultados encontrados por CS e DE foram mais robustos e precisos do que os encontrados para PSO e ABC.

3.3 Comportamento de Elefantes

A Otimização Baseada em Manada de Elefantes (*Elephant Herding Optimization* – EHO) (WANG; DEB; COELHO, 2015) é uma técnica baseada em inteligência de enxame e inspirada no comportamento social que ocorre em manadas de elefantes.

Na natureza, os elefantes pertencentes a diferentes clãs vivem juntos sob a liderança de uma matriarca. Os elefantes machos deixam seus grupos familiares quando crescem. Estes dois comportamentos podem ser modelados através de dois operadores: atualização de clã e separação. No EHO, elefantes de cada clã são atualizados levando-se

em consideração a posição atual e a matriarca, durante a ação do operador de atualização de clã. Em seguida, o operador de separação pode aumentar a diversidade da população na fase final de busca.

Os elefantes são animais sociais com estruturas complexas de fêmeas e filhotes. Um grupo de elefantes é composto de muitos clãs sob o comando de uma matriarca, normalmente a mais idosa. O clã consiste de uma fêmea com suas crias ou com algumas outras fêmeas relacionadas. Elefantes fêmeas preferem viver em famílias, enquanto os machos tendem a viver no isolamento, o que os leva a deixar suas famílias ao crescerem. Quando eles se afastam, eles podem manter contato com elefantes de outros clãs através de comunicação por vibrações de baixa frequência que eles provocam batendo com as patas contra o solo.

3.3.1 Funcionamento da Técnica EHO

As seguintes regras são definidas para modelar o comportamento dos elefantes:

1. Cada clã possui um número fixo de elefantes.
2. A cada geração, um número fixo de elefantes machos deixa seu respectivo clã familiar e passa a viver solitariamente e longe do grupo principal de elefantes.
3. Os elefantes de cada clã vivem juntos sob a liderança de uma matriarca.

Os operadores de atualização dos clãs e de separação do elefante macho de cada clã são apresentados a seguir.

3.3.1.1 Operador de Atualização de Clã

Todos os elefantes vivem juntos sob a liderança de uma matriarca em seu respectivo clã c_i. Para cada elemento deste clã, sua próxima geração é influenciada pela sua matriarca. Um elefante j do clã c_i, pode ser representado através da Equação 8:

$$x_{novo,c_i,j} = x_{c_i,j} + \alpha \cdot (x_{melhor,c_i} - x_{c_i,j}) \cdot r, \tag{8}$$

onde os termos $x_{c_i,j}$ e $x_{novo,c_i,j}$ representam as posições de cada elefante j pertencente ao clã c_i: antiga e recentemente atualizada, respectivamente. O fator $\alpha \in [0, 1]$ refere-se à influência que a matriarca exerce sobre $x_{c_i,j}$. A componente x_{melhor,c_i} representa a

matriarca de c_i , que é o elefante melhor ajustado à função objetivo dentro do seu clã. A componente r pertence ao intervalo $[0, 1]$ e possui distribuição uniforme.

O melhor elefante em cada clã não pode ser atualizado pela Equação 8, isto é, $x_{c_i,j} = x_{melhor,c_i}$. O elefante com o melhor ajuste é atualizado através da Equação 9:

$$x_{novo,c_i,j} = \beta \cdot x_{centro,c_i}, \tag{9}$$

onde o fator $\beta \in [0, 1]$ determina a influência de $x_{centro,c_i,d}$ em $x_{novo,c_i,d}$. A posição do novo indivíduo $x_{novo,c_i,d}$ é calculada através da Equação 9, usando as informações de todos os elefantes do clã c_i. O termo x_{centro,c_i} é o centro do clã c_i, que pode ser calculado através da Equação 10 para a d-ésima dimensão:

$$x_{centro,c_i,d} = \frac{1}{n_{c_i}} \cdot \sum_{j=1}^{n_{c_i}} (x_{c_i,j,d}), \tag{10}$$

onde o intervalo $1 \leq d \leq D$ indica a d-ésima dimensão de um total de D. Para o total de clãs N_{cla}, o termo N_{c_i} é o número de elefantes do clã c_i. O termo $x_{c_i,j,d}$ significa o d-ésimo elefante $x_{c_i,j}$. O centro $x_{centro,c_i,d}$ do clã c_i pode ser calculado com D cálculos da Equação 10. O Algoritmo 3 sintetiza a atualização dos clãs (WANG; DEB; COELHO, 2015).

Algoritmo 3 Operador de atualização de clã

Para $c_i := 1, \ldots, N_{cla}$ **Faça**
  Para $j := 1, \ldots, N_{c_i}$ **Faça**
    Atualizar $x_{c_i,j}$
    Gerar $x_{novo,c_i,j}$ através da Equação 8
    Se $x_{c_i,j} = x_{melhor,c_i}$ **Então**
      Atualizar $x_{c_i,j}$ e **gerar** $x_{novo,c_i,j}$
      Gerar $x_{novo,c_i,j}$ através da Equação 9
    Fim Se
  Fim Para
Fim Para

3.3.1.2 Operador de Separação

Em um grupo de elefantes, os machos deixam suas famílias e passam a viver sozinhos quando atingem a puberdade. Este processo pode ser modelado através de um operador de separação para resolver os problemas de otimização.

Para proporcionar melhoria na capacidade de busca do EHO, assume-se que os elefantes com o pior ajuste à função objetivo são submetidos ao operador de separação a

cada geração e são substituídos por outros aleatoriamente gerados, conforme apresentado na Equação 11:

$$x_{pior,c_i} = x_{min} + (x_{max} - x_{min} + 1) \cdot rand, \tag{11}$$

onde os termos x_{max} e x_{min} são, respectivamente, os limites superior e inferior da posição de cada elefante individualmente. O termo x_{pior,c_i} é o pior elefante individualmente do clã c_i. Já $rand \in [0, 1]$ possui distribuição uniforme.

O Algoritmo 4 representa a equação de separação dos elefantes machos apresentada em (WANG; DEB; COELHO, 2015). Neste trabalho, foi empregada a Equação 12:

$$x_{pior,c_i} = x_{melhor,c_i} + rand, \tag{12}$$

onde x_{melhor,c_i} representa a matriarca e $rand$ é um número inteiro aleatório, de maneira que os novos elefantes sejam criados na vizinhança da matriarca, em posição aleatória. O objetivo é acelerar a convergência no processo de busca.

Algoritmo 4 Operador de separação

Para $c_i := 1, \ldots, N_{cla}$ **Faça**
  Substituir o pior elefante x_{pior,c_i} do clã através da Equação 12
Fim Para

3.3.2 Algoritmo da Manada de Elefantes

A técnica EHO é apresentada no Algoritmo 5 (WANG; DEB; COELHO, 2015). Ela utiliza um número N_e de elefantes e um número NG_{max} de gerações e o critério de parada. Os operadores de deslocamento dos elefantes e de substituição do pior elefante são executados sucessivamente até que se atinja o critério de parada.

Algoritmo 5 Algoritmo baseado no comportamento social dos elefantes nas manadas

Definir N_e e NG_{max}
$t := 1$
Enquanto $(t \leq M)$ e (critério de parada não atingido) **Faça**
  Aplicar o operador de atualização de clã
  Aplicar o operador de separação
  $t := t + 1$
Fim Enquanto
Retornar a melhor solução

3.3.3 Exemplo de Aplicação da Técnica EHO

O desempenho da técnica EHO foi comparado em (WANG; DEB; COELHO, 2015) com Algoritmo Genético (*Genetic Algorithm* – GA), Evolução Diferencial e Otimização Biogeográfica (MA; SIMON, 2011), aplicando-a um conjunto de funções de teste como Ackley, Griewank e Rastringin. Os parâmetros empregados para o EHO foram $\alpha = 0,5$, $\beta = 0,1$ e 5 clãs. Todos os clãs tinham $n_{c_i} = 10$ elefantes. Os resultados obtidos pelo EHO foram melhores para a maioria dos testes realizados com relação à precisão e ao tempo de processamento.

3.4 Comportamento de Bactérias

A técnica de busca otimizada baseada no comportamento de colônias da bactéria *E. Coli* (*Bacterial Foraging Optimization Algorithm* – BFOA) foi introduzida em (PASSINO, 2002). As posições das bactérias no espaço de busca representam as soluções, que são avaliadas através da função objetivo previamente definida. Durante o movimento das bactérias, as posições tendem a evoluir até que se atinja o ótimo global da função. Além disso, elas fazem reprodução, agem coletivamente nos movimentos e sofrem as influências facilitadoras ou dificultantes do meio em que estão.

3.4.1 Funcionamento da Técnica BFOA

O comportamento forrageiro das bactérias *E. Coli* é sintetizado através dos seguintes processos (PASSINO, 2002; SHARMA; PATTNAIK; GARG, 2012): quimiotaxia, comportamento de enxame, reprodução, eliminação e dispersão. A combinação deles modela a maneira como as bactérias sobrevivem no meio em que habitam, que pode ser mais favorável (ter alimento em quantidade adequada, por exemplo) ou ser repulsivo (com substâncias tóxicas para as bactérias, por exemplo). As bactérias conseguem perceber este gradiente da situação do meio e tomar a decisão para onde ir.

3.4.1.1 Quimiotaxia e Comportamento de Enxame

A quimiotaxia representa o deslocamento no meio por parte das bactérias conforme a presença de substâncias químicas, que podem ser atrativas ou repulsivas. Serina e aspartato são atrativas para as bactérias, enquanto a presença de leucina, acetato e íons dos metais níquel e cobalto são repulsivas. Mudanças no pH também são repulsivas às bactérias.

A quimiotaxia implica em nados e desvios (*tumbling*) efetuados durante o deslocamento das bactérias. Se a bactéria estiver em um ambiente favorável, ou seja, rico em nutrientes e sem nocividade, ela continua nadando na mesma direção. Considerando-se que a função objetivo deve ser minimizada, a redução no custo representa ambiente progressivamente favorável, enquanto aumento no custo está relacionado com a piora do ambiente, o que a leva a mudar a direção de deslocamento (desvio).

Quando as bactérias estão em um meio considerado neutro (sem alimento ou substâncias nocivas) ou se as bactérias estão no mesmo meio há muito tempo, o deslocamento delas tende a alternar entre corridas e desvios de direção, de modo que elas efetuam um padrão de busca por alimento.

Mudanças de concentração de alimento levam as bactérias a deslocamentos no mesmo sentido, ou seja, com menos desvios. Se o gradiente representar aumento na oferta de alimento, a bactéria seguirá por mais tempo nesta direção; o mesmo fenômeno ocorre em caso de piora no ambiente e consequente fuga. O comportamento das bactérias observado ao microscópio revela movimentos em grupos para procurar alimento ou fugir de ameaças, o que caracteriza comportamento de enxame.

A representação de um desvio possui uma unidade de comprimento cuja direção é aleatoriamente obtida. O ajuste à função objetivo das novas posições é calculado. Se houve melhoria, haverá novos passos de deslocamento; caso contrário, o desvio é desconsiderado e a posição anterior a ele é restabelecida.

Os deslocamentos em sequência estão relacionados ao nado da bactéria na tentativa de chegar a melhores locais no meio para sua sobrevivência. O movimento de cada bactéria com coordenadas m_1, m_2, m_3 é representado pela Equação 13:

$$f_{i,j+1,k,l}(m_1, m_2, m_3) = f_{i,j,k,l}(m_1, m_2, m_3) \; + \; C(i) \cdot \frac{mov(i)}{\sqrt{mov^T(i) \cdot mov(i)}}, \qquad (13)$$

onde o termo $f_{i,j+1,k,l}(m_1, m_2, m_3)$ representa o valor da função objetivo $f(x)$ em cada ponto. O subíndice $j+1$ está relacionado com o movimento de quimiotaxia. O termo $C(i)$ representa a unidade de deslocamento das bactérias. O vetor $mov(i)$ significa as direções do movimento nas D dimensões. O termo T significa matriz transposta. O termo k é o índice do processo de reprodução, enquanto l é o índice do processo de eliminação e dispersão.

3.4.1.2 Reprodução das Bactérias

Após a quimiotaxia, a metade menos saudável das bactérias morre, representada pela parcela com pior ajuste junto à função objetivo. A metade mais saudável, com melhor custo calculado, permanece e cada bactéria dela transforma-se em duas crias próximas de onde a mãe estava. Desse modo, a quantidade de bactérias mantém-se constante.

O processo de reprodução acelera a convergência durante a busca, pois mais bactérias surgem nas regiões melhor avaliadas do espaço de busca. Na Equação 13, o índice k está associado à reprodução.

3.4.1.3 Eliminação e Dispersão

A quimiotaxia e a reprodução podem não ser suficientes para encontrar o ponto ótimo global. As bactérias podem ficar presas a ótimos locais caso estejam nas melhores posições em suas vizinhanças. Para evitar que isto ocorra, os eventos de eliminação e dispersão em posições aleatórias proporcionam uma busca global.

Um percentual das bactérias com pior ajuste à função objetivo é eliminado e a mesma quantidade é recriada no espaço de busca aleatoriamente, mantendo-se constante o número de bactérias. Isto representa a situação na qual o meio onde as bactérias vivem sofre mudanças (na concentração de alimento ou de toxinas, por exemplo) ou parte das bactérias são dispersadas para outras localidades no meio através do movimento de fluidos corporais do hospedeiro da colônia, por exemplo. Na Equação 13, o índice l está associado ao processo de eliminação e dispersão.

3.4.2 Algoritmo da Colônia de Bactérias

A operação do BFOA baseia-se na especificação de seus parâmetros: o número de gerações de bactérias NG_{max}; o número de bactérias N_b; os números de passos de eliminação e dispersão N_{ed}, de reprodução N_{rep}, e de quimiotaxia N_{quim}; a probabilidade P_{ed} de eliminação e dispersão; número N_{sw} de deslocamentos a nado de cada bactéria; e o tamanho K_{step} do nado, conforme é apresentado no Algoritmo 6 (PASSINO, 2002).

3.4.3 Algumas Aplicações da Técnica BFO

A técnica BFOA foi aplicada em (LIU et al., 2011) ao problema de identificação de dispositivo emissor de radiofrequência. Devido ao grande número de aparelhos que se comunicam

Algoritmo 6 Algoritmo da colônia de bactérias

Definir $f(x)$, NG_{max}
Definir N_b, P_{ed}, N_{ed}, N_{rep}, N_{quim}, N_{sw} e K_{step}
Gerar posição inicial x_i das N_b bactérias
$ciclos := 1$
Enquanto ($ciclos \leq NG_{max}$) e (critério de parada não atingido) **Faça**
  Para ($l := 1, \ldots, N_{ed}$) e (critério de parada não atingido) **Faça**
    Para $k := 1, \ldots, N_{rep}$ **Faça**
      Para $j := 1, \ldots, N_{quim}$ **Faça**
        Calcular $f(i, j, k, l)$ e gravar em $f_{anterior}$ com seus respectivos m_1, m_2, m_3
        Gerar vetor $mov(i)$ com valores aleatórios entre -1 e 1
        Mover em desvio usando a Equação 13
        Para $s := 1, \ldots, N_{sw}$ **Faça**
          Mover por N_{sw} passos e gerar f_{desvio}
        Fim Para
        Se $f_{desvio} > f_{anterior}$ **Então**
          Reestabelecer as posições referentes a f_{desvio} para as bactérias
          Para $s := 1, \ldots, N_{sw}$ **Faça**
            Mover − se por N_{sw} passos e gerar novo $f(x)$
          Fim Para
        Fim Se
      Fim Para
      Efetuar a reprodução da melhor metade das N_b bactérias
    Fim Para
    Eliminar a fração P_{ed} das piores bactérias dentre as N_b
    Criar a mesma quantidade eliminada aleatoriamente pelo espaço de busca
  Fim Para
  $ciclos := ciclos + 1$
Fim Enquanto
Retornar a melhor posição

usando radiofrequência, não raras são as falhas de identificação. Uma variante autoadaptativa do BFOA foi desenvolvida para ajustar o tamanho dos movimentos de natação das bactérias dinamicamente durante a busca, de forma a balancear a relação intensificação/diversificação. Se as bactérias descobrem uma região promissora, a magnitude do deslocamento é reduzida, com maior intensidade na exploração. Se o custo da posição ocupada por uma bactéria ficar inalterado por um número definido de passos (exaustão de alimento), o deslocamento é aumentado e a bactéria entra em estado de exploração diversificada. Os resultados obtidos foram melhores do que os de GA, PSO e BFOA.

A técnica BFOA foi aplicada em (GOLLAPUDI et al., 2008) para calcular frequência de ressonância e e ponto de alimentação em antenas de microfitas com finos substratos. A espessura otimizada é empregada para calcular a frequência de ressonância de cada

pedaço da antena. Para obter o ponto de alimentação, a Equação da impedância de entrada do pedaço da antena é utilizada como função de custo a ser otimizada. Os resultados superaram os do GA quanto à precisão.

Uma versão híbrida de BFOA com PSO foi empregada em (GOLLAPUDI et al., 2009) ao cálculo preciso da frequência de ressonância de antena de microfita retangular para quaisquer dimensões e espessuras de substrato, na qual a rápida convergência do PSO é aproveitada para encontrar posições das bactérias no espaço de busca. Desse modo, o espaço de busca é reduzido e ocorre a redução do tempo computacional. A mudança nas frequências de ressonância de diferentes pedaços é calculada primeiro e colocada aleatoriamente no espaço de busca do PSO. Então, é iniciada a busca usando PSO para valores experimentais mais próximos. Em seguida estes valores são fornecidos ao espaço de busca do BFOA. O erro médio quadrático (RMSE – *Root Mean Square Error*) é empregado como função de custo no BFOA. Os resultados apresentam melhoria de acurácia e redução drástica do tempo computacional.

Um conjunto linear de antenas foi otimizado em (DATTA et al., 2008) em fase e amplitude através do uso da técnica BFOA em modo adaptativo (ABFOA) para que o fator de agregação seja máximo para qualquer direção especificada e nulo para outras. A modulação delta é empregada para modular o tamanho do nado do BFOA, sendo usado um conjunto de seis elementos. O ABFOA foi capaz de sintetizar padrões com múltiplos nulos em qualquer direção desejada, com convergência rápida. Os resultados mostraram que os pesos puderam ser otimizados para obter os fatores de agregação desejados.

As redes neurais (RN) optimizadas por BFOA (BFONN) foram aplicadas em (ZHANG; WU; WANG, 2010) ao problema de distribuição de carga elétrica. O dispositivo de busca global do BFOA leva à convergência mais rápida pela rede neural. Para o treinamento da RN, o BFOA foi aplicado no caminho de realimentação. O erro médio quadrático (*Mean Square Error* – MSE) foi empregado como medida da função de custo da BFOA. O algoritmo BFO é utilizado para encontrar os pesos otimizados da RN enquanto minimiza o MSE. Este modelo, aplicado à distribuição de carga da cidade de Nova York, obteve resultados muito precisos. Os resultados das simulações também mostraram que BFONN converge mais rapidamente do que uma RN otimizada por GA (GANN).

Uma abordagem híbrida envolvendo PSO e BFOA foi aplicada em (BISWAS et al., 2007) para otimização de funções multimodais e de altas dimensões. O algoritmo

combina um operador de mutação baseado em PSO com a quimiotaxia para balancear a diversificação e a intensificação da busca e evitar a convergência prematura. O algoritmo é testado junto a cinco funções-padrão (Rosenbrock, Rastrigin, Griewank, Ackley, buracos de raposa de Shekel) e também junto ao espectro largo de um radar de código polifásico. Os resultados encontrados mostram que, no geral, o desempenho do algoritmo híbrido foi superior ao do BFOA, e é ao menos comparável ao PSO e seus variantes.

Uma variante híbrida da técnica BFOA orientada por PSO (BFPSO) foi proposta em (KORANI; DORRAH; EMARA, 2009) para sintonizar os ganhos proporcional, integral e derivativo de um controlador proporcional-integral-derivativo (PID). BFOA convencional depende de direções de busca aleatórias que podem incidir em atraso em alcançar a solução global enquanto o PSO está propenso a ficar preso a um mínimo local. Para obter uma otimização melhor, o novo algoritmo combina vantagens de ambos algoritmos: capacidade do PSO de compartilhar informação social e habilidade do BFOA em encontrar novas soluções pela eliminação e dispersão. Resultados de simulação demonstraram a redução considerável dos *overshoots* do controlador, com rápida convergência. O desempenho superou as versões convencionais do BFOA e do PSO.

3.5 Comportamento de Vaga-Lumes

O Algoritmo de Vaga-Lumes (*Firefly Algorithm* – FFA) é uma técnica inspirada no comportamento de vaga-lumes proposta em (YANG, 2009) para problemas de otimização com restrições. O algoritmo é baseado nos sinais de bioluminescência, ou seja, luzes emitidas pelos insetos, usadas na comunicação entre os vaga-lumes e para afugentar predadores. Os vaga-lumes apresentam características de inteligência de enxame para auto-organização e tomada descentralizada de decisão. Os sinais luminosos não possuem importância apenas no processo de busca por alimento, mas também no ritual de acasalamento destes insetos. Em (KAR, 2016), o autor relatou como vantagem pertencente ao FFA a facilidade de ser usado com outros algoritmos de maneira híbrida para melhorar o desempenho.

O ritmo das emissões, a frequência e a duração delas definem uma comunicação que atrai ambos os sexos. As fêmeas, dependendo da espécie, podem emitir um padrão único de resposta ou podem imitar os padrões de outras espécies para seduzir, atrair e então devorar machos. A percepção da intensidade da luz por parte dos vaga-lumes é inversamente proporcional à distância entre emissor e receptor, de modo que a comuni-

cação entre eles é afetada pela distância. Isto ocorre devido à absorção da luz pelo ar, o que é quantificado através de um coeficiente de absorção. A intensidade do brilho da luz emitida por um inseto também é um indicador de aptidão do vaga-lume masculino. Contudo, a técnica FFA considera os insetos sendo unissex, assim como a atratividade do vaga-lume decorrente da intensidade do brilho. A intensidade, por sinal, refere-se ao valor junto à função objetivo da posição.

Inicialmente, a população de vaga-lumes é criada. Se um dos parâmetros que influenciam na avaliação é alterado, a aptidão dos vaga-lumes é recalculada e eles são ordenados de acordo com ela. Os melhores insetos são mantidos para a próxima rodada de avaliações.

3.5.1 Funcionamento da Técnica FFA

Segundo (YANG, 2009), a técnica FFA baseia-se em três regras idealizadas para funcionar:

1. Todos os vaga-lumes são unissex, de forma que um inseto pode atrair outro independentemente dos sexos.

2. A atratividade do vaga-lume é diretamente proporcional ao brilho. Para duas emissões luminosas, o inseto menos brilhante irá em direção ao mais brilhante. A percepção do brilho (e também a atratividade) é mais forte quão mais perto estiverem os insetos entre si. Se um inseto não emitir brilho mais intenso do que o outro, o movimento será aleatório.

3. A intensidade do brilho é afetada pela função objetivo. Quanto melhor o ajuste, mais forte o brilho.

Por questão de simplificação, o FFA considera que a atratividade refere-se diretamente à intensidade da luz emitida, e está relacionada ao ajuste à função objetivo. O cálculo da intensidade é feito através da Equação 14:

$$I(r) = I_0 \cdot e^{-\gamma \cdot r^2}, \tag{14}$$

onde $I(r)$ representa o decaimento da intensidade com o quadrado da distância r entre dois pontos, o termo I_0 representa o brilho na própria fonte e γ, o coeficiente de absorção empregado. Foi adotada a forma Gaussiana para evitar divisão por zero, em caso da

distância ser nula (YANG, 2009). Segundo (FISTER; YANG; BREST, 2013), o parâmetro γ possui impacto crucial na velocidade de convergência, e tipicamente varia entre 0,1 e 10.

O uso da Equação 15 é sugerido em (YANG, 2009) caso se queira um decaimento monotonicamente decrescente para a intensidade:

$$I(r) = \frac{I_0}{1 + \gamma \cdot r^2}. \tag{15}$$

Isso porque a expansão via série de Taylor para ambas as equações é idêntica na primeira ordem (YANG, 2009).

O cálculo da distância entre dois pontos no espaço n-dimensional é efetuado através da norma Euclidiana para funções objetivo com dimensões de mesma natureza (todas de números reais, por exemplo). Porém, há situações em que isso não ocorre, como no problema de agendamento, em que uma distância r pode ser definida como o intervalo entre dois eventos. Em (YANG, 2009), r pode ser definido como uma combinação da proximidade dos vértices dos agrupamentos do modelo empregado no problema com graus dos agrupamentos.

3.5.2 Algoritmo dos Vaga-Lumes

Para iniciar o FFA, é necessário definir: o número de vaga-lumes N_v, a função objetivo $f(x)$, o coeficiente de absorção luminosa do ar γ, o número de dimensões D, o número máximo de gerações N, o critério de parada. O Algoritmo 7 apresenta o funcionamento do FFA (YANG, 2009).

3.5.3 Algumas Aplicações da Técnica FFA

Um levantamento de avaliações e aplicações do FFA foi efetuado em (FISTER; YANG; BREST, 2013), além de versões modificadas e híbridas da técnica, comparando os desempenhos com o de outras técnicas de referência na solução de problemas complexos, como GA e PSO. Foram analisadas aplicações em diversos tipos de problema de otimização contínua, multiobjetivo, com restrições, dinâmica e ruidosa. Fister apontou como motivos para o bom desempenho do FFA: a técnica automaticamente subdivide a população em subgrupos (a atração local é mais forte do que a de longa distância), o que se mostrou adequado a aplicações não lineares e multimodais; o não uso de um melhor global, fato que melhoraria a questão da convergência prematura; o uso do componente γ como fator de escala para ajustar a busca ao ambiente.

Algoritmo 7 Algoritmo dos vaga-lumes

Definir: N_v, D, $f(x)$, γ, N
Gerar os vaga-lumes nas posições iniciais x_i para $i = (1, ..., N_v)$
Calcular a intensidade luminosa I_i das emissões através de $f(x_i)$
$t := 1$
Enquanto $(t \leq N)$ e (critério de parada não atingido) **Faça**
  Para $i := 1, \ldots, N_v$ **Faça**
    Para $j := 1, \ldots, N_v$ **Faça**
      Se $(I_j > I_i)$ **Então**
        Mover o vaga-lume i em direção ao j
      Fim Se
      Calcular a atratividade percebida pelos vaga-lumes
      Avaliar as soluções com $f(x)$ e atualizar a intensidade das luzes
    Fim Para
  Fim Para
  Ordenar os vaga-lumes
  $t := t + 1$
Fim Enquanto
Retornar a melhor solução

Em (WANG et al., 2017), foi realizado um estudo sobre o possível aumento do tempo de processamento e as oscilações no processo de busca devido ao fato de cada vaga-lume poder ser atraído por muitos outros que sejam mais brilhantes do que ele. Assim, propuseram uma variante do FFA baseada na atratividade da vizinhança, na qual cada vaga-lume é atraído apenas pelos pertencentes a uma lista pré-definida de vizinhos, ao invés de todos do enxame. Houve melhoria na precisão e redução no tempo de processamento.

Uma solução usando FFA foi proposta em (KARTHIKEYAN et al., 2015) junto a problemas de otimização discreta multiobjetivo, e o aplicaram ao problema de agendamento flexível de tarefas em um conjunto de máquinas. O FFA é aplicado junto com um dispositivo de busca local. São definidos três objetivos para minimização: tempo de processamento, carga de trabalho da máquina crítica e carga de trabalho total das máquinas. Os resultados obtidos mostraram que o algoritmo proposto é adequado ao problema.

A técnica FFA foi aplicada em (GAO et al., 2013) ao problema de rastreamento, no qual estudaram o comportamento dos parâmetros de sensibilidade e adequação do FFA ao problema. O desempenho do algoritmo proposto foi comparado com o de filtro de partículas, PSO e *mean shift* (COMANICIU; RAMESH; MEER, 2003) aplicados ao mesmo problema. Os resultados do FFA superaram os obtidos pelas outras técnicas com rela-

ção à velocidade e à precisão, e mostraram a possibilidade do FFA ser empregado para rastreamento em diferentes condições desafiadoras.

Em (GAO et al., 2015), a técnica FFA foi conjugada com filtro de partículas e aplicaram ao problema de rastreamento de imagem em vídeo na presença de falhas na amostragem dos quadros. O FFA é empregado para ajustar o número de partículas do filtro. Os resultados apontam melhoria de desempenho em relação ao filtro de partículas convencional, além da versão do trabalho ter rastreado de maneira mais robusta.

3.6 Comportamento dos Fogos de Artifício

O Algoritmo de Otimização baseado em Fogos de Artifício (*Fireworks Algorithm for Optimization* – FWA), proposto em (TAN; ZHU, 2010), modela as partículas incandescentes (faíscas) da explosão dos fogos e é empregada em problemas complexos de otimização. Na técnica FWA, há dois processos de busca empregados que estão relacionados a dois tipos específicos de explosão. Mecanismos para manter a diversidade das faíscas são aplicados.

3.6.1 Funcionamento da Técnica FWA

Quando fogos de artifício são acionados, um jato de faíscas preenche o espaço em volta. Em (TAN; ZHU, 2010), a explosão pode ser interpretada como um processo de busca ao redor de um ponto específico por parte das faíscas produzidas. Na tentativa de encontrar um ponto específico x_j que satisfaça $f(x_j) = y$, fogos de artifício são continuamente acionados no espaço de busca até que uma faísca acerte o ponto procurado ou uma região considerada suficientemente próxima dele.

Para cada geração de explosões, são selecionados N_f locais onde os fogos são detonados. Após cada explosão, as faíscas atingem novos pontos, que são avaliados. Se o local ótimo é encontrado, o algoritmo para. Por outro lado, N_f outros lugares são selecionados dentre as faíscas atuais para dar origem à próxima geração.

Através da observação das explosões de fogos de verdade, identificam-se dois tipos específicos para serem modelados. Se as detonações são corretas, um grande número de faíscas é produzido no entorno do centro da explosão. Em caso de detonação defeituosa, poucas faíscas são liberadas e de forma espalhada pelo ambiente.

Uma boa detonação de fogos no FWA significa que as faíscas encontraram uma área promissora, ou seja, que pode estar mais próxima do ótimo procurado; logo é adequado empregar mais faíscas para efetuar a busca ao redor do local de detonação. Ao mesmo

tempo, uma detonação de má qualidade está relacionada ao fato do ponto da explosão estar longe do ótimo desejado, o que necessitaria de um raio de busca maior. No algoritmo FWA, um bom cenário para a busca recebe muitas faíscas com raio pequeno de amplitude. O número de faíscas é definido através da Equação 16:

$$s_i = m \cdot \frac{\xi + y_{pior} - f(x_i)}{\xi + \sum_{i=1}^{n}(y_{pior} - f(x_i))}, \tag{16}$$

onde a determinação do número s_i de faíscas produzidas por cada um dos N_f fogos de artifício detonados leva em consideração o pior caso y_{pior} da função a ser otimizada, bem como um fator m de escala e uma constante ξ de fins algébricos. O termo ξ evita que ocorra divisão por zero no cálculo de s_i. A função objetivo $f(x)$ vai avaliar os pontos x_i.

Para evitar efeitos danosos de fogos de artifício, ocorre a definição de limites para s_i. Neste caso, são usadas duas constantes a e b, conforme mostrado pela Equação 17,

$$\hat{s}_i = \begin{cases} \lceil a \cdot m \rceil, & \text{se } s_i < a \cdot m, \\ \lfloor b \cdot m \rfloor, & \text{se } s_i > b \cdot m,\ a < b < 1, \\ \lfloor s_i \rfloor, & \text{caso contrário,} \end{cases} \tag{17}$$

onde o valor de a controla a quantidade de faíscas geradas em caso de má explosão, de modo a garantir um número mínimo de faíscas produzidas. Já o valor de b afeta a quantidade gerada em caso de boa explosão, para que a quantidade de faíscas produzidas não supere um número máximo.

Em oposição ao número de faíscas, a amplitude percorrida pela faísca em um bom cenário é menor em relação ao que ocorre se o cenário é ruim. A amplitude da explosão é dada pela Equação 18:

$$A_i = \hat{A} \cdot \frac{\xi + f(x_i) - y_{melhor}}{\xi + \sum_{i=1}^{n}(f(x_i) - y_{melhor})}, \tag{18}$$

onde o termo $\hat{A}$ simboliza a máxima amplitude. O melhor valor que $f(x)$ pode assumir é y_{melhor} através dos N_f fogos que explodem.

A localização de uma faísca de um aparato de fogos de artifício x_i é calculada através do Algoritmo 8 (TAN; ZHU, 2010). Durante as explosões, faíscas podem deslocar-se em todas as z direções, conforme a Equação 19:

$$z = \lceil d \cdot r \rceil, \tag{19}$$

onde r é um número uniformemente distribuído entre 0 e 1.

Algoritmo 8 Cálculo da posição de uma faísca

Iniciar a posição de uma faísca $\hat{x}_j = x_i$
Calcular $z = \lceil d \cdot r \rceil$
Escolher aleatoriamente z dimensões de $\hat{x}_j$
Calcular o deslocamento $h = A_i \cdot r$
Para cada dimensão $\hat{x}_k^j \in z$ **Faça**
  $\hat{x}_k^j = \hat{x}_k^j + h$
  Se $(\hat{x}_k^j < \hat{x}_k^{melhor})$ ou $(\hat{x}_k^j > \hat{x}_k^{pior})$ **Então**
    Mapear $\hat{x}_k^j = \hat{x}_k^{melhor} + |\hat{x}_k^j| \ \% \ (\hat{x}_k^{pior} - \hat{x}_k^{melhor})$
  Fim Se
Fim Para

O autor de (TAN; ZHU, 2010) propôs um dispositivo para melhorar a diversidade das faíscas, decorrente de um processo de geração baseado em distribuição Gaussiana (média unitária, desvio padrão unitário) para o cálculo do deslocamento das faíscas, conforme mostrado no Algoritmo 9 (TAN; ZHU, 2010). A cada geração de explosões, $\hat{m}$ faíscas deste tipo são produzidas.

Algoritmo 9 Cálculo da posição de uma faísca específica, com distribuição Gaussiana

Iniciar a posição de uma faísca $\hat{x}_j = x_i$
Calcular $z = \lceil d \cdot r \rceil$
Escolher aleatoriamente z dimensões de $\hat{x}_j$
Calcular o coeficiente da explosão Gaussiana g
Para cada dimensão $\hat{x}_k^j \in z$ **Faça**
  $\hat{x}_k^j = \hat{x}_k^j \cdot g$
  Se $(\hat{x}_k^j < \hat{x}_k^{melhor})$ ou $(\hat{x}_k^j > \hat{x}_k^{pior})$ **Então**
    Mapear $\hat{x}_k^j = \hat{x}_k^{melhor} + |\hat{x}_k^j| \ \% \ (\hat{x}_k^{pior} - \hat{x}_k^{melhor})$
  Fim Se
Fim Para

No início da operação do FWA, n lugares poderiam ser selecionados durante a explosão dos fogos. A informação sobre o melhor lugar obtido até então (x^*) em relação à função objetivo é mantida para a próxima geração de explosões. Assim, $n-1$ locais são selecionados com base na distância uns dos outros, para manter a diversidade das faíscas.

3.6.2 Algoritmo dos Fogos de Artifício

O Algoritmo 10 apresenta o FWA (TAN; ZHU, 2010). Durante cada geração de explosões, dois tipos de faíscas são produzidas baseadas nos Algoritmos 8 e 9. Para o primeiro tipo, o número de faíscas e a amplitude delas dependem da qualidade de cada cenário. Já para o segundo, as faíscas são produzidas através do processo Gaussiano de explosão, o qual

executa a busca ao redor do local da explosão. Após obter os locais dos dois tipos de explosões, N_f locais são selecionados para a próxima geração de explosões. Durante a execução do FWA, uma quantidade de $n + m + \hat{m}$ cálculos é efetuada a cada geração. Dessa forma, se o ponto ótimo é encontrado após T gerações, implica em uma ordem de complexidade $O(T \cdot (n + m + \hat{m}))$.

Algoritmo 10 Algoritmo dos fogos de artifício

Definir: N_f, a, b, m, ξ, $f(x)$, T
Selecionar aleatoriamente os N_f locais de detonação dos fogos
$t := 1$
Enquanto $(t \leq T)$ e (critério de parada não atingido) **Faça**
  Detonar os N_f fogos nos locais selecionados
  Para $i := 1, \ldots, N_f$ **Faça**
    Calcular o número $\hat{s}_i$ de faíscas, de acordo com a Equação 17.
    Obter as localizações de $\hat{s}_i$ faíscas dos fogos x_i através do Algoritmo 8.
  Fim Para
  Para $k := 1, \ldots, \hat{m}$ **Faça**
    Selecionar aleatoriamente fogos x_j.
    Gerar uma faísca específica para os fogos usando o Algoritmo 9.
  Fim Para
  Selecionar o melhor local encontrado até então e mantê-lo para a próxima geração
  Selecionar aleatoriamente $N_f - 1$ locais para os dois tipos de faíscas e os atuais fogos.
  $t := t + 1$
Fim Enquanto
Retornar a melhor posição

3.6.3 Algumas Aplicações da Técnica FWA

Em (TAN; ZHU, 2010), a técnica FWA foi aplicada a funções de teste padrão e compararam seu desempenho com duas versões do PSO: a canônica e a clonal. O FWA superou o PSO em relação à velocidade de convergência e à acurácia global da solução.

Uma variante do FWA foi proposta em (ZHANG et al., 2015) para melhorar o desempenho da técnica. Três aspectos foram trabalhados: um novo operador Gaussiano de mutação, no qual as faíscas recebem informação daquela melhor localizada até então, bem como seus cálculos empregam a distribuição Gaussiana de probabilidade; integração entre o operador regular de explosão do FWA com um otimizador biogeográfico; adoção de uma nova estratégia de seleção para a população, para melhorar a qualidade das soluções da próxima geração sem aumentar o custo computacional. Estes três aspectos combinados

melhoraram as interações no algoritmo, o que proporcionou ganho na diversidade das soluções e diminuiu a convergência prematura.

Adaptações do FWA foram aplicadas a problemas dinâmicos em (PEKDEMIR; TOPCUOGLU, 2016), e ocorreram em: verificação de amplitude mínima ao explodir; operador de geração de faíscas; operador de mapeamento, para limitar o espaço de busca e evitar sobreposição; operador de seleção. A validação foi feita usando o banco de dados com uma função objetivo de picos móveis (*Moving Peaks Benchmark* – MPB) (BRANKE, 1999; ALBA; NAKIB; SIARRY, 2013). Os resultados foram bem-sucedidos em buscar o ótimo global.

A técnica FWA recebeu emprego na otimização do ganho em um conjunto de antenas do tipo Yagi-Uda em (BOUDAHER; HOORFAR, 2016). Os resultados obtidos foram considerados promissores, ao comparar o desempenho junto às técnicas PSO, GA, Programação Evolucionária (*Evolutionary Programming* – EP) e técnica de matriz de covariâncias com estratégia evolucionária (*Covariance Matrix Evolution Strategy* – CMAS-ES), o que motivou os autores a aplicar variantes do FWA para outros tipos de antena.

Uma adaptação da técnica FWA foi proposta em (CHEN et al., 2015) para usar informações do espaço de busca para balancear as propriedades de busca intensiva (*exploitation*) e diversificada (*exploration*). A intensificação em excesso pode levar à convergência prematura para ótimos locais, e a diversificação excessiva melhora a acurácia enquanto consome mais tempo. A informação do ambiente de busca é obtida através dos pontos candidatos (faíscas) a soluções e é descrita através dos seguintes vetores: núcleo, área de cobertura e dispersão das faíscas. As informações armazenadas nestes vetores são empregadas na criação das faíscas da próxima geração, para proporcionar um ajuste mais preciso quanto à exploração do espaço. Assim, foi possível encontrar mais rapidamente regiões promissoras do espaço de busca em relação ao FWA canônico.

Uma variante da técnica FWA que usa um operador de mutação baseado em covariância foi proposta em (YU; TAN, 2015). Ele emprega a informação das faíscas com melhor ajuste à função objetivo para gerar novas faíscas em potenciais locais de alta qualidade. Os resultados apresentam melhoria de desempenho em relação ao FWA.

Uma aplicação da técnica FWA em vídeos de imagens de microscopia de fluorescência (SHI et al., 2015), com o objetivo de identificar células. As imagens são ruidosas e possuem o número de células variante. Há situações de oclusão entre células e problemas

de contraste na imagem. As faíscas do FWA efetuam a detecção e são ajustadas conforme a informação relativa às posições ocupadas. O sistema proposto foi capaz de se manter identificando as células, mesmo com outras entrando e saindo da imagem.

Em (MISRA; SI, 2017), os autores empregaram a técnica FWA com um filtro adaptativo ao problema de segmentação de imagem baseado em agrupamento. O filtro calcula o número de faíscas e suas amplitudes, e sua função de transferência decorre da avaliação das posições das faíscas pela função objetivo. Os resultados encontrados para o agrupamento apresentaram-se melhores do que os obtidos pelos métodos *K-means*, PSO e pelo FWA canônico.

3.7 Otimização por Enxame de Partículas

A técnica de busca otimizada Otimização por Enxame de Partículas (PSO), proposta por (KENNEDY; EBERHART, 1995), foi inspirada pelo comportamento coletivo de pássaros e de peixes. No PSO, o espaço de busca é explorado por um número definido de partículas que se move para encontrar o ponto ótimo da função objetivo estipulada.

3.7.1 Funcionamento da Técnica PSO

As N_p partículas possuem características de posição ocupada em d dimensões e de velocidade individual que são constantemente atualizadas em função do movimento coletivo e de suas próprias experiências. A melhor posição até então ocupada por cada partícula é armazenada ($P_{i_{best}}$, ou *particle best*) e a melhor posição obtida pelas partículas do enxame até o momento também é armazenada (em S_{best}, a componente social). A aptidão da posição ocupada pelas partículas no espaço de busca é calculada através da função objetivo $f(x)$ que modela o problema, e cada ponto representa uma solução em potencial, sendo ser avaliado.

A posição de cada partícula i do enxame é calculada através da Equação 20:

$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{20}$$

onde o termo $x_i(t+1)$ significa a próxima posição da partícula i, $x_i(t)$ é a posição atual e $v_i(t+1)$ representa a velocidade.

A velocidade de deslocamento de cada partícula é uma composição ponderada da sua velocidade atual (chamada de componente inercial), da melhor posição anteriormente

atingida por ela (componente cognitiva) e da melhor posição coletivamente encontrada (componente social) (ENGELBRECHT, 2006).

A componente inercial representa a memória do movimento prévio da partícula, e impede que ocorram mudanças drásticas na direção, também sendo chamado de componente de inércia. A componente cognitiva modela a memória individual da partícula e a tendência dos indivíduos de retornar a situações e a lugares que melhor os satisfizeram no passado. Por fim, a componente social quantifica o desempenho de uma partícula i em relação a um grupo de partículas ou mesmo de vizinhos, particularmente. A componente social está associada a uma norma ou padrão do enxame que cada partícula busca atender, na qual elas são atraídas pela melhor posição até então encontrada pelo seu grupo. O modelo do cálculo da velocidade de cada partícula é apresentado pela Equação 21:

$$v_i(t+1) = \omega \cdot v_i(t) + \phi_1 \cdot r_1 \cdot (P_{i_{best}} - x_i(t)) + \phi_2 \cdot r_2 \cdot (S_{i_{best}} - x_i(t)), \tag{21}$$

onde o termo ω significa a próxima posição da partícula i, phi_1 é o peso da componente cognitiva, phi_2 é o peso da componente social, $r1$ e $r2$ são números aleatórios entre 0 e 1, e $P_{i_{best}}$ representa a melhor posição até então ocupada por cada partícula até então. O termo $S_{i_{best}}$ representa a melhor posição obtida pelas partículas do enxame até o momento sob o ponto de vista da partícula i, para as quais podem ser consideradas todas elas ou apenas uma parcela.

Em (ENGELBRECHT, 2006), duas abordagens do PSO foram propostas com características particulares: o PSO *Global Best* (PSO-G) e o PSO *Local Best* (PSO-L). O PSO-G considera todas as partículas do enxame para o cálculo da componente social da velocidade, de modo que $S_{i_{best}}$ assume a forma G_{best} (*global best*) na Equação 22:

$$v_i(t+1) = \omega \cdot v_i(t) + \phi_1 \cdot r_1 \cdot (P_{i_{best}} - x_i(t)) + \phi_2 \cdot r_2 \cdot (G_{best} - x_i(t)), \tag{22}$$

onde G_{best} é empregado para a cálculo da componente social do PSO-G. Da mesma forma, o PSO-L emprega, durante o cálculo da mesma componente, a informação de duas partículas vizinhas, refletindo em uma topologia em anel. Nesta situação, a componente $S_{i_{best}}$ passa a ser representado por $L_{i_{best}}$ (*local best*) na Equação 23:

$$v_i(t+1) = \omega \cdot v_i(t) + \phi_1 \cdot r_1 \cdot (P_{i_{best}} - x_i(t)) + \phi_2 \cdot r_2 \cdot (L_{i_{best}} - x_i(t)), \tag{23}$$

onde L_{ibest} é empregado para a cálculo da componente social do PSO-L.

O crescimento sem controle da velocidade pode causar o deslocamento da partícula por grandes distâncias, saindo do espaço de busca ou divergindo, principalmente se ela estiver longe de $P_{i_{best}}$ e de $S_{i_{best}}$. Para mitigar este problema, emprega-se um limite de velocidade para cada dimensão (ENGELBRECHT, 2006).

A cada ciclo de iterações, usa-se a função objetivo para testar cada posição de partícula e atualizar $P_{i_{best}}$ e S_{best}. Os ciclos são executados até atingir o número máximo estipulado M ou o critério de parada.

O trabalho desenvolvido em (TAVARES, 2016), empregado como referência para comparação de desempenho desta dissertação, utilizou o PSO *Global Best* para a busca de alvos em imagens.

3.7.2 Algoritmo do PSO

Para iniciar o PSO, é necessário definir: o número de partículas N_p, a função objetivo $f(x)$, o coeficientes ω, ϕ_1, ϕ_2, c_1 e c_2, o número máximo de gerações M, o critério de parada. Os números aleatórios r_1 e r_2 são calculados no início do algoritmo. O Algoritmo 11 apresenta o modelo de comportamento do enxame de partículas.

Algoritmo 11 Algoritmo de otimização por enxame de partículas

Definir: ω, ϕ_1, ϕ_2, $f(x)$, M, o limitador de velocidade
Inicializar aleatoriamente as N_p partículas no espaço de busca
Inicializar aleatoriamente r_1 e r_2
Inicializar $P_{i_{best}} = f(x_i)$
Inicializar S_{best}
$t := 1$
Enquanto $(t \leq N)$ e (critério de parada não atingido) **Faça**
  Para cada uma das partículas x_i **Faça**
    Calcular a velocidade das N_p partículas através da Equação 21
    Aplicar o limitador de velocidade
    Calcular a posição das N_p partículas através da Equação 20
    Atualizar $P_{i_{best}}$ e S_{best}
  Fim Para
  Selecionar o melhor local encontrado até então e mantê-lo para a próxima geração
  $t := t + 1$
Fim Enquanto
Retornar a posição da melhor partícula

3.8 Considerações Finais

Neste capítulo foram abordadas as seis técnicas de busca otimizada a serem empregadas nesta dissertação, inspiradas nos comportamentos do pássaro cuco, de abelhas, de elefantes, de vaga-lumes e de faíscas de fogos de artifício. Estudos demonstraram a capacidade destas técnicas em resolver problemas complexos em engenharia com desempenho semelhante à técnica PSO, largamente utilizada como referência.

O objetivo desta dissertação é implementar cada uma das seis técnicas descritas no subsistema de *software*, escolhendo seus parâmetros para satisfazer as metas de desempenho para rastrear alvos em imagens pertencentes a vídeos de 30 quadros por segundo. O próximo capítulo apresenta uma análise da plataforma de avaliação empregada.

Capítulo 4

PLATAFORMA DE AVALIAÇÃO

ESTE capítulo apresenta o equipamento físico (*hardware*) empregado na consecução do projeto. O objetivo é utilizar uma plataforma portátil que conjugue *hardware* e *software* e que seja adequada para efetuar rastreamento de alvos através de análise de imagens com desempenho que permita o processamento de vídeos de 30 quadros por segundo. As técnicas de busca otimizada são implementadas em *software*, enquanto a etapa mais custosa computacionalmente do processo de identificação do alvo, o cálculo da correlação cruzada normalizada, ocorre em coprocessador dedicado. Assim, a Seção 4.1 apresenta a motivação para o emprego de *hardware* e *software*; a Seção 4.2 aborda a arquitetura do coprocessador; a Seção 4.3 descreve as plataformas de *hardware* e *software* empregadas no projeto; e a Seção 4.4 apresenta as considerações finais sobre o Capítulo.

4.1 Motivação para o *Co-Design*

A tomada de decisão passa por obter as informações no momento correto para embasar a estratégia definida; porém, isso tem um preço. Na verdade, uma composição de preços, traduzida pelo gasto financeiro com a tecnologia, pelo espaço físico ocupado pelo equipamento e pela energia consumida. Um sistema de apoio à tomada de decisão que seja portátil e dedicado a soluções de problemas complexos (*NP-hard*, por exemplo) sofre ainda mais com estas restrições (NEDJAH; MOURELLE, 2007).

O presente trabalho emprega uma solução para unir o desempenho necessário para efetuar rastreamento de alvos em vídeos de 30 quadros por segundo, executado em um equipamento portátil e de custo limitado. Este custo implica em equipamentos com capacidade de processamento limitada, o que provoca o desafio de otimizar o projeto para atingir as metas de desempenho de tempo gasto e taxa de acerto compatível.

Em (SHI et al., 2017), os autores pesquisaram aplicações *co-design* relacionadas ao guiamento autônomo de veículos, com ênfase na detecção de outros veículos e de pedestres, identificação da faixa de rolamento e detecção de caminho adequado à direção. Os autores analisaram: Unidade Central de Processamento (CPU – *Central Processing Unit*), Unidade de Processamento Gráfico (GPU – *Graphics Processing Unit*), FPGA e Circuitos Integrados de Aplicação Específica (ASIC – *Application-Specific Integrated Circuit*), como componentes principais de plataformas de *hardware* eficientes para operação em tempo real. A percepção visual para um veículo autônomo é crítica, e todas as decisões importantes baseiam-se na percepção visual do ambiente no entorno do local; sem isso, as decisões para controlar o veículo não são seguras.

Segundo (SHI et al., 2017), um FPGA é um circuito integrado que pode ser configurado para implementar diferentes funções lógicas, e é convencionalmente empregado para emular plataformas para validação em estágio inicial de ASIC. Como vantagens para aplicação em computação de alto desempenho, destacam-se os fatos de ser reconfigurável e de poder ser programado para implementar diferentes funções lógicas. Em relação a um projeto ASIC clássico, o uso de FPGA reduz custos não recorrentes de engenharia ao diminuir os tempos de projeto e validação. Além disso, proporciona eficiência computacional e economia de energia em comparação a outras ferramentas como CPU e GPU, de propósito geral. Eles concluíram que: algoritmos robustos e precisos são necessários para trabalhar em ambientes como ruas e cruzamentos com segurança; o *hardware* deve ser poderoso para dar suporte aos algoritmos para trabalhar em tempo real; acurácia e eficiência na validação de sistemas autônomos complexos não é trivial, e afirmam que qualquer percepção visual neste contexto baseada em aprendizado de máquina não é 100% certa.

Em (TAVARES, 2016), o autor verificou qual seria a parte mais custosa em tempo de processamento de um sistema *co-design* para localização de alvos em imagens. O PSO foi a técnica empregada para acelerar busca e localização do alvo, foi implementado em *software* e executado pelo processador de emprego geral do FPGA utilizado, trabalhando em conjunto com um coprocessador desenvolvido para calcular o PCC. O *hardware* mostrou-se apto a aplicar o rastreamento em vídeos de 30 quadros por segundo, quando apoiado pela excelência do PSO no processo de busca.

4.2 Arquitetura do Coprocessador

O *hardware* desenvolvido em (TAVARES, 2016) foi definido para a implementação da metodologia *co-design* neste projeto. Estes componentes foram modelados através da linguagem de descrição de *hardware* VHDL (*Very high speed integrated circuits Hardware Description Language*), e posteriormente sintetizados com o uso da ferramente Vivado, também da Xilinx.

Após a síntese, os arquivos produzidos passaram a ser referência para um ambiente de programação em *software*, o SDK. Lá, os códigos produzidos fizeram uso deles para implementar a estratégia de reconhecimento do alvo, com acesso ao XC7Z015 e ao coprocessador sintetizado.

A placa SVDK foi empregada para que as técnicas de otimização utilizassem o processador de uso geral do XC7Z015, um coprocessador dedicado para o cálculo do PCC, blocos de memória que armazenassem a imagem principal e o alvo (*template*) e o controle de acesso às memórias supracitadas, conforme mostrado na Figura 2.

Os blocos de memória dedicados são `BRAM_IMG` e `BRAM_TMP`. Os controladores `GET_TMP` e `GET_IMG` proporcionam acesso às memórias, de modo a disponibilizar ao coprocessador os conteúdos das imagens em tempo correto, sendo o primeiro referente ao *template* e o segundo, à imagem principal.

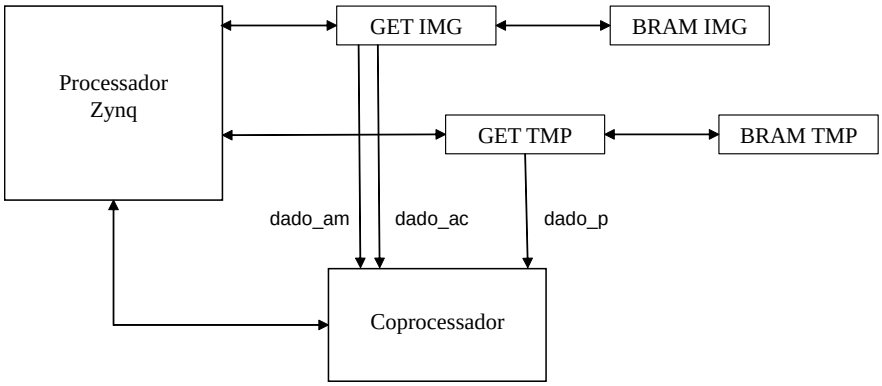


Figura 2: Macro-componentes da arquitetura do sistema de rastreamento.

O coprocessador proposto recebe os pontos do *template* e da amostra da imagem, com o objetivo de efetuar o cálculo do Coeficiente de Correlação de Pearson (PCC). Este é o valor da correlação cruzada normalizada, em complemento a 2 (C2) e é codificado em

32 bits. Tanto o *template* quanto amostra possuem tamanho de 64 x 64 *pixels* cada um, o que incorre em 4096 *pixels* correspondentes a 4 kBytes.

A arquitetura é disposta em três blocos interconectados, na forma de *pipeline*, que representam a equação do PCC e permitem o cálculo de suas etapas separadamente entre si, dando margem a paralelização de tarefas. Os blocos também recebem como elementos comuns os sinais de `CLK` (*clock*), `CLR` (*clear*), `EN` (*enable*) e de sincronismo (`SINCRO`), sendo este último decorrente do processamento de `CLK`. O componente `SINCRO` envia um pulso de sincronismo a cada 4013 transições de subida dos pulsos de *clock*.

Três sinais alimentam o processador:

- `dado_p`: um *pixel* da imagem alvo (*template*), representado por 8 bits;
- `dado_ac`: um *pixel* da imagem a comparar, representado por 8 bits;
- `dado_am`: um *pixel* da imagem seguinte que será comparada, representado por 8 bits;

A Figura 3 exibe os três blocos constituintes da arquitetura interna do coprocessador, com este demarcado através da linha tracejada. O primeiro bloco recebe as entradas do componente coprocessador propriamente dito, enquanto o terceiro bloco fornece os sinais de saída que vão para processador de uso geral do XC7Z015.

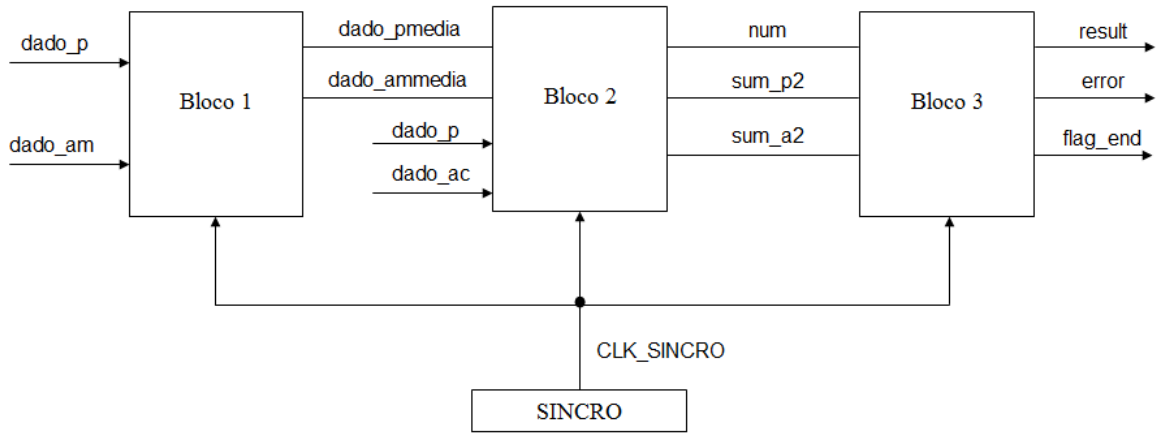


Figura 3: Macro-arquitetura do coprocessador

O componente Bloco 1, mostrado na Figura 4, recebe a magnitude dos *pixels* da figura e do *template*, calcula os valores médios através de duas componentes `avg`, devolve cada resultado através de `out_avg` e informa o final da operação através de `end_avg`. Os

resultados são armazenados em registradores de N bits para o bloco seguinte, e fornecidos pelo uso de `dado_pmedia` e de `dado_amedia`, onde o primeiro refere-se ao *template* e o segundo, à amostra. Ao final do cálculo de média, as duas componentes `avg` são reinicializadas pela ação de `SINCRO`.

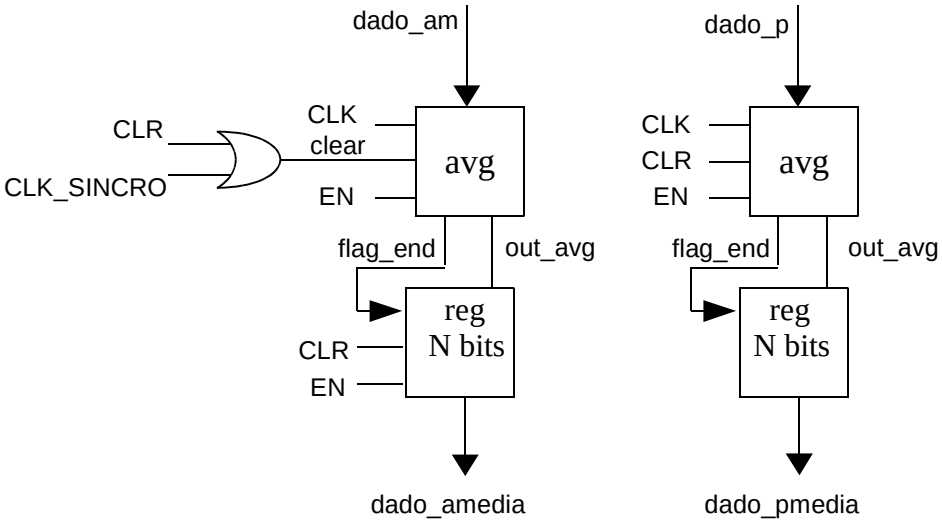


Figura 4: Micro-arquitetura do Bloco 1

O componente Bloco 2, presente na Figura 5, é responsável por calcular o numerador da equação do PCC, bem como os somatórios pertencentes ao denominador da equação. Ele recebe do primeiro bloco os sinais de `dado_pmedia` e de `dado_amedia`, através de dois registradores de N bits. Dois blocos subtratores `subt_C2` calculam a diferença em C2 entre estes sinais e `dado_p` e `dado_ac`, respectivamente e os resultados são encaminhados a três multiplicadores `mult_CLK`.

Os sinais produzidos seguem para três somadores `sum_C2` em C2, que direcionam seus resultados `out_sum` a registradores de N bits, e produzem sinal `flag_end` de final de operação para ativar a carga nos registradores. O multiplicador que recebe a diferença entre `dado_avg` e `dado_ac` tem sua entrada `CLR` controlada pela combinação entre `CLR` e `CLR_SINCRO` via porta OU. O pulso de sincronismo faz com que os elementos `subt_C2`, `sum_C2` e `mult_CLK` sejam reiniciados.

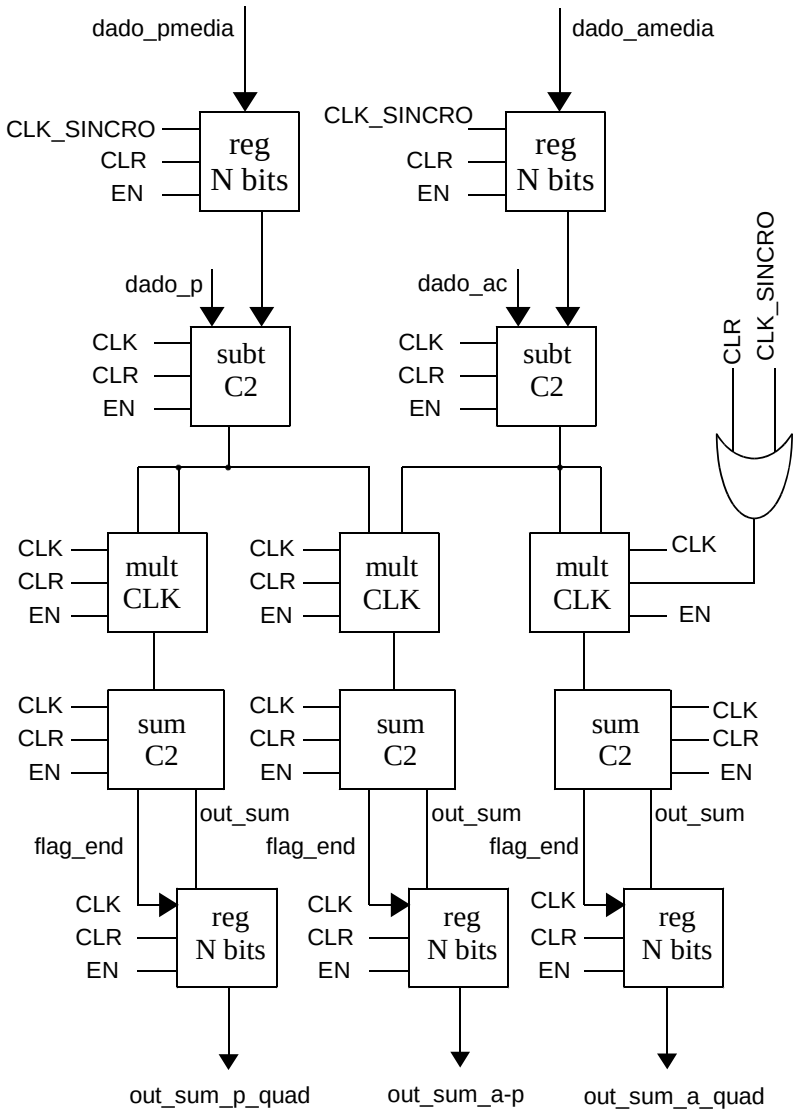


Figura 5: Micro-arquitetura do Bloco 2

O componente Bloco 3, mostrado na Figura 6, é o bloco do *pipeline* responsável por fornecer o resultado do cálculo final do PCC com precisão de 24 bits, bem como prover o produto do somatórios do denominador da equação do PCC e a operação de raiz quadrada deste produto. O bloco faz uso da máquina de estado `FSM` para garantir a sequência correta de ações.

Três registradores de N bits recebem os resultados dos somatórios do Bloco 2. O elemento multiplicador `mult_CLK` efetua o produto ponto-a-ponto entre diferenças entre as médias e os valores dos *pixels* e submete ao elemento `SQRT` que efetua a raiz quadrada do produto.

O elemento `div_frac_C2` aplica a divisão em C2 entre a saída de `SQRT` e o sinal de um dos registradores de N bits, produzindo o resultado `result`, o sinal de erro `error` e o indicador de final de operação `flag_end`. Os dois primeiros são guardados em registradores de N bits para que possam ser disponibilizados para o processador de propósito geral da placa SVDK.

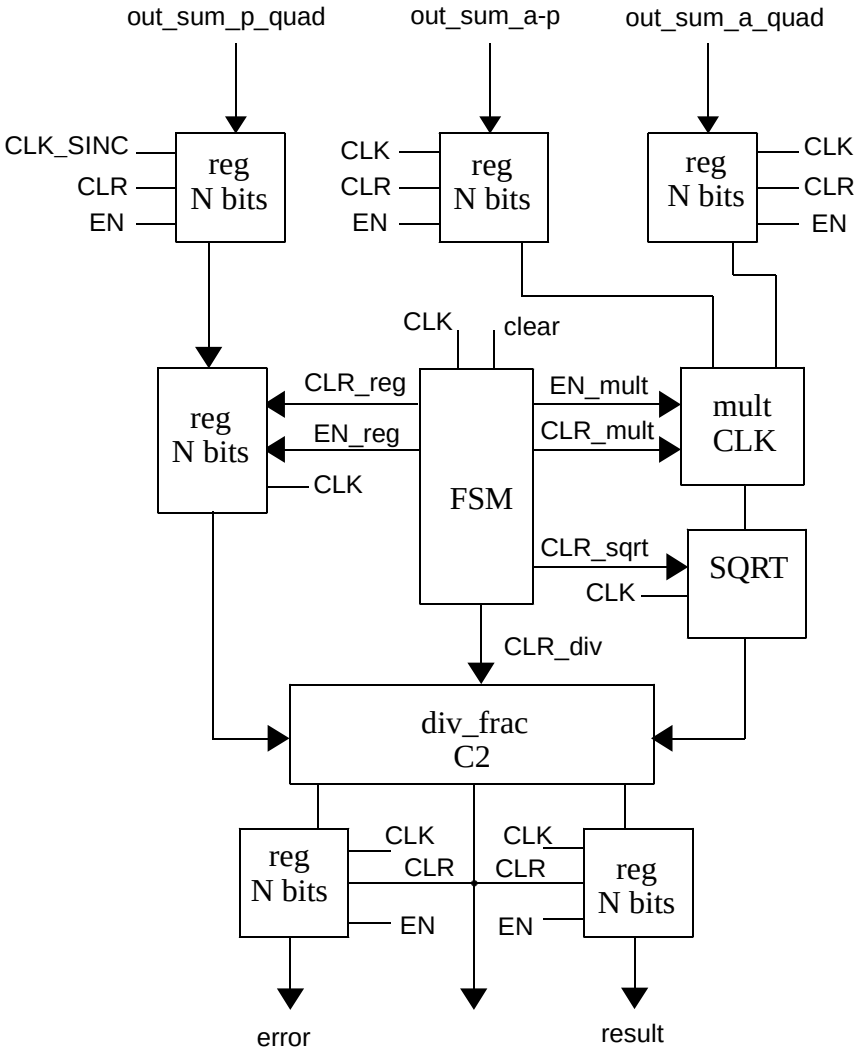
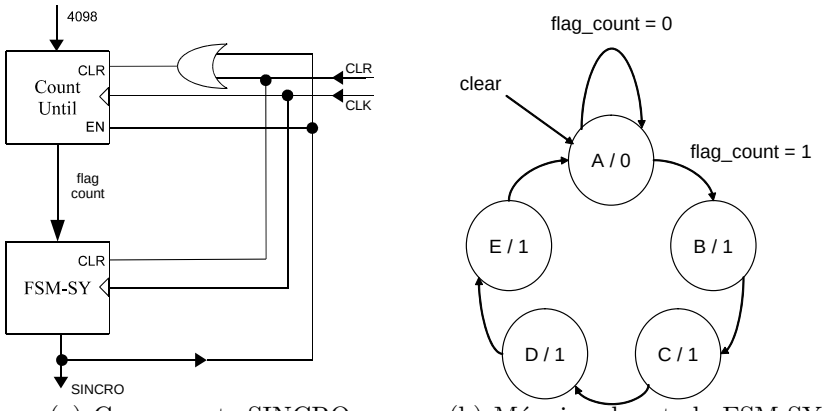


Figura 6: Micro-arquitetura do Bloco 3

O componente `SINCRO`, cuja micro-arquitetura é apresentada na Figura 7 possui como função produzir o pulso de sincronismo para os estágios do coprocessador. Ele possui um contador e uma máquina de estados finitos do tipo Moore. A saída da máquina de estados é o sinal `CLK_SINCRO`, que permanece em zero durante 4098 pulsos de *clock*, quando o *flag* do contador é setado. Esta máquina mantém o sinal `CLK_SINCRO` durante quatro pulsos de *clock*, que etão associados aos estados B, C, D e E da máquina.

A quantidade de 4098 pulsos de *clock* refere-se ao tempo em que o componente Bloco 2 necessita para efetuar as suas operações, conforme descrito em (TAVARES, 2016). Ela decorre do processamento de 4096 *pixels* de cada imagem, com um *pixel* recebido por pulso, com outros dois pulsos necessários para a conclusão do estágio.



(a) Componente SINCRO (b) Máquina de estado FSM-SY

Figura 7: Micro-arquitetura do componente SINCRO

4.3 Plataformas de Implementação

O *hardware* desenvolvido em (TAVARES, 2016) foi definido como modelo para o presente projeto, que tem como estratégia empregar diferentes técnicas de busca otimizada baseadas em inteligência de enxame em abordagem *co-design*. O processador de uso geral da placa executará via *software* cada técnica de busca, com toda a configuração de parâmetros necessária para se encontrar o alvo com tempo de processamento e taxa de acerto satisfatórios.

4.3.1 Plataforma de *Hardware*

O equipamento físico empregado neste projeto, tal como em (TAVARES, 2016), consistiu de uma placa *Smart Vision Development Kit* (SVDK) rev 1.2, com módulo Xilinx PicoZed 7Z015 System-On-Module (com um *chip* Zynq XC7Z015), 1 GB de memória DDR3 e oscilador de 33,333 MHz. O XC7Z015 possui sistema de processamento de propósito geral baseado no processador *dual-core* ARM Cortex-A9 e lógica programável em um único circuito integrado. A placa também possui recursos como interface UART, *encoder* de vídeo HDMI, Ethernet PHY *tri-mode*, I/O de propósito geral, interfaces USB3, GigE

Vision e CoaXPress. Seu barramento PCI Express permite a instalação de sensores. A placa pode ser vista na Figura 8.

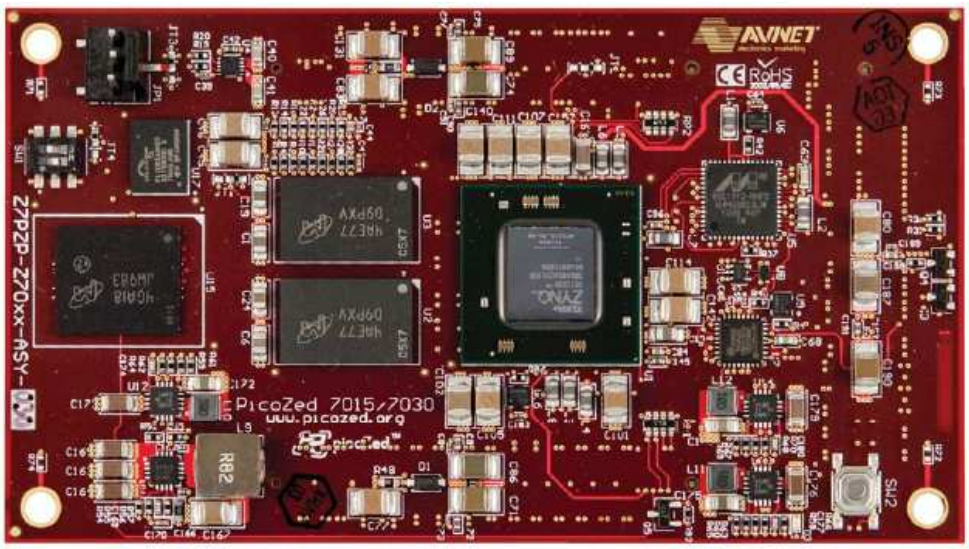


Figura 8: Placa Xilinx PicoZed 7Z015

A arquitetura proposta possui uma técnica de busca otimizada (CS, ABC, BFOA, EHO, FFA ou FWA) em *software* sendo executado pelo processador do XC7Z015, trabalhando em conjunto com um coprocessador dedicado especificamente desenvolvido em (TAVARES, 2016) para calcular o PCC. Este cálculo é a parte mais onerosa do processamento durante a localização de alvos, logo o emprego do coprocessador melhoraria significativamente o desempenho, fato comprovado pelos resultados.

4.3.2 Plataforma de *Software*

Em seguida, empregou-se a ferramenta SDK para programação usando o conteúdo criado em (TAVARES, 2016) de modo a configurar a lógica programável do XC7Z015, conforme apresentado na Figura 9. Através do SDK, constrói-se o programa em linguagem C que faz acesso ao XC7Z015 e ao coprocessador para implementar a detecção do alvo, com o auxílio de inteligência de enxame para otimizar a busca.

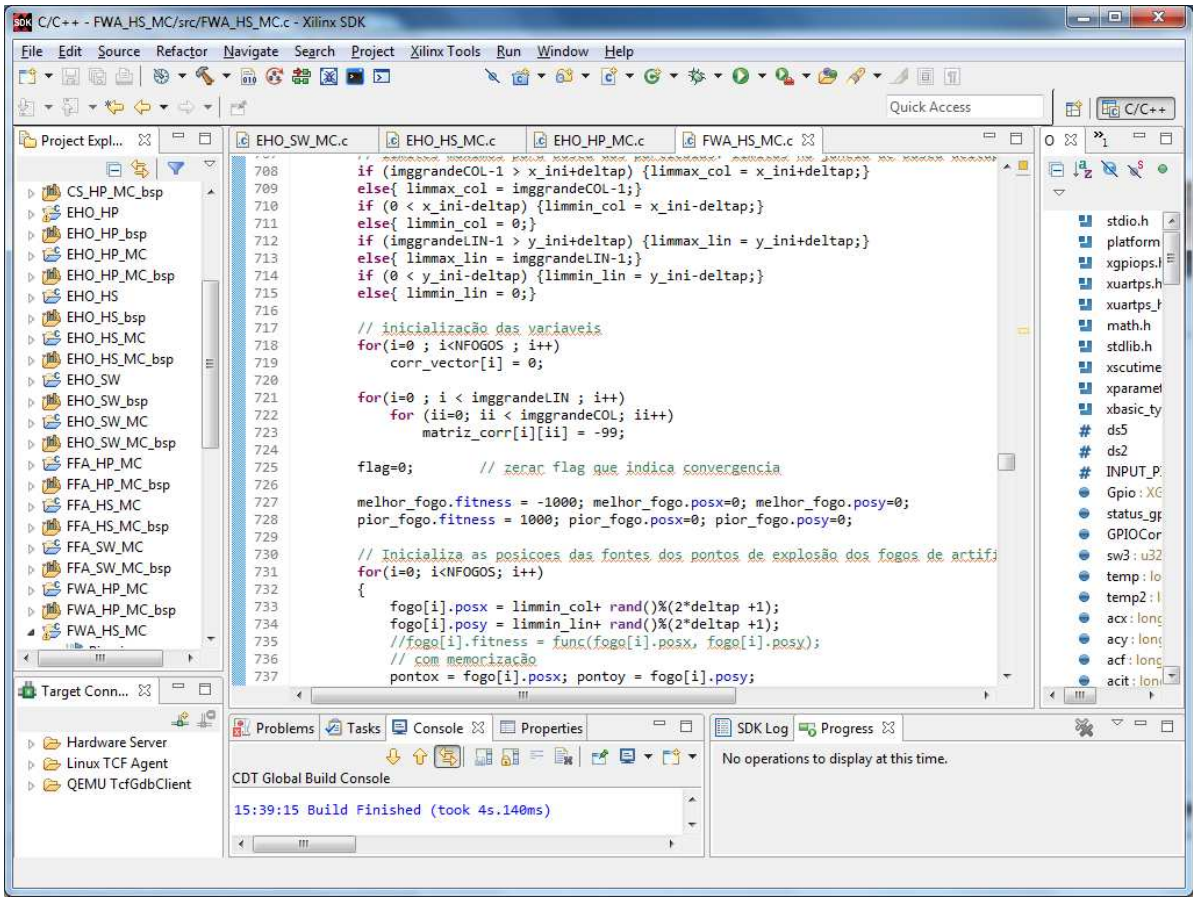


Figura 9: Ambiente da ferramenta SDK

4.4 Considerações Finais

O presente capítulo apresenta o equipamento físico utilizado no projeto. A placa SVDK possui as condições necessárias para implementar o rastreamento de um *template* pré-definido, através de um sistema conjunto de *hardware* e *software* com a presença de um coprocessador dedicado para melhorar o desempenho. As técnicas de inteligência de enxame têm sua execução efetuada por *software* via o processador de uso geral da placa; já a parte mais custosa em tempo de processamento, o cálculo do PCC, é providenciado pelo coprocessador dedicado. Os resultados obtidos pelo sistema serão mostrados e analisados no próximo capítulo.

Capítulo 5

ANÁLISE DOS RESULTADOS

ESTE capítulo apresenta a metodologia de avaliação do sistema de rastreamento presente, utilizando seis técnicas de inteligência de enxame propostas, bem como a análise dos resultados alcançados. A avaliação decorre da execução do sistema *co-design* para encontrar alvos pré-definidos em imagens. A métrica para avaliação dos resultados recebe sua definição na Seção 5.1. A execução do sistema produziu os resultados apresentados e analisados na Seção 5.2, que são organizados por técnica de busca inteligente empregada. A Seção 5.3 apresenta a comparação dos resultados obtidos pelas técnicas. A Seção 5.4 apresenta os resultados de novas configurações dos parâmetros de cada técnica de forma a se obter taxa de acerto de pelo menos 90%. Por fim, a Seção 5.5 apresenta as considerações finais obtidas a partir da análise dos resultados.

5.1 Metodologia de Avaliação

O sistema de rastreamento proposto neste projeto emprega o equipamento descrito na Seção 4.2 do Capítulo 4 para encontrar alvos definidos nas imagens apresentadas na Figura 10, considerados para avaliação de processamento computacional (COLLINS; ZHOU; TEH, 2005; TAVARES, 2016), bem como apresenta os tamanhos originais das imagens empregadas nesta dissertação. Os alvos escolhidos nas imagens estão destacados e possuem 64x64 *pixels*. A janela de busca empregada é de 101x101 *pixels*. O sistema é avaliado em três configurações:

1. a técnica de busca inteligente é implementada em *software* e é executada pelo processador de uso geral do XC7Z015 para o cálculo do PCC;

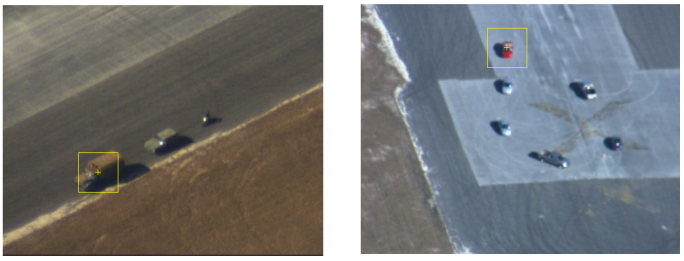
2. a técnica de busca é implementada em *software*, e o cálculo do PCC é realizado pelo coprocessador, configurado em modo serial de operação; e

3. a técnica de busca é implementada em *software*, e cálculo do PCC é realizado pelo coprocessador, configurado em modo paralelo de operação, com *pipeline*.

(a) Cars 521x411 – I_1 (b) Pickup 522x410 – I_2

(c) Sedan 524x414 – I_3 (d) IR1 328x266 – I_4

(e) IR2 335x272 – I_5 (f) IR3 329x265 – I_6

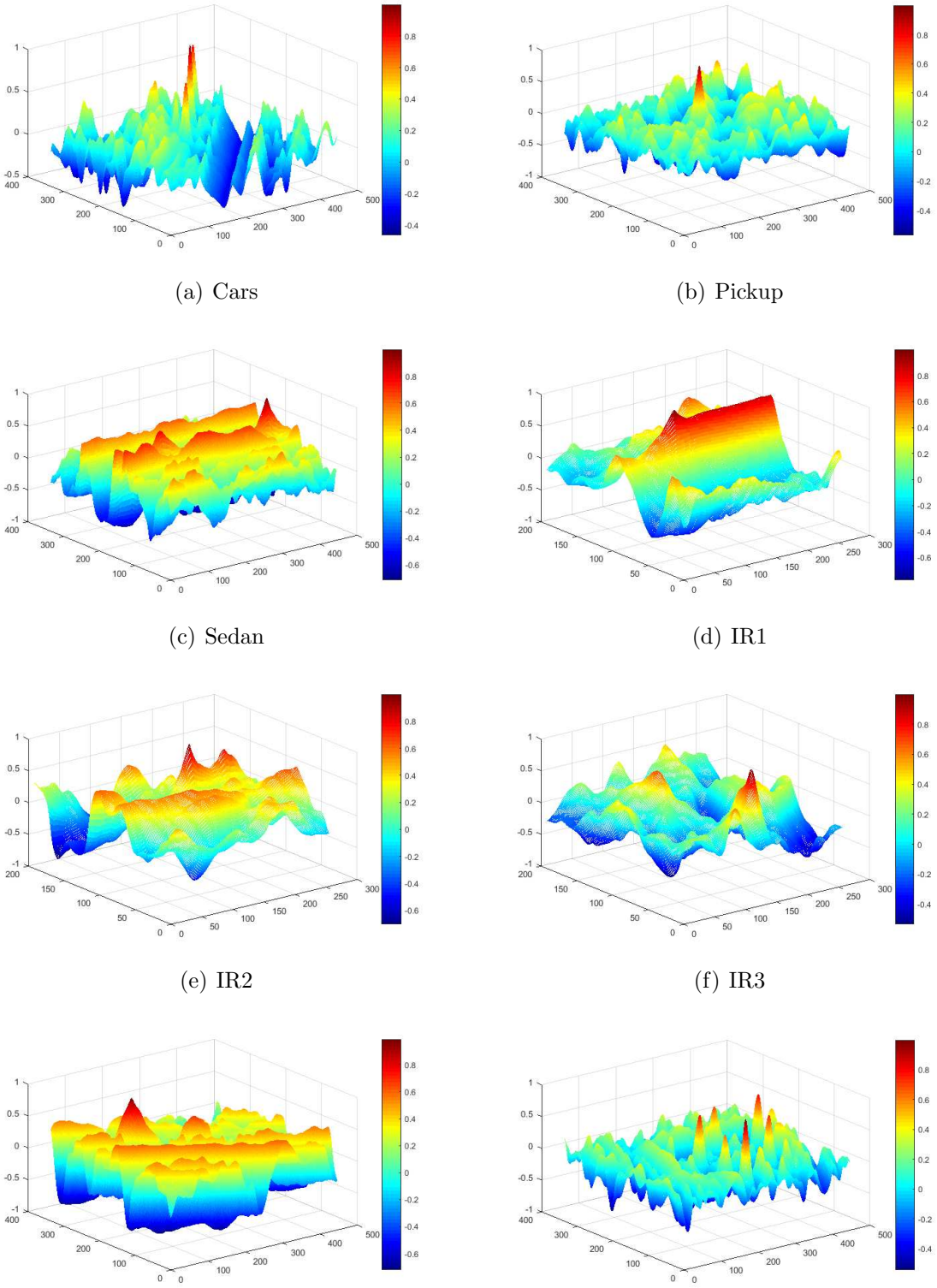


(g) Truck 522x410 – I_7 (h) Rcar 524x413 – I_8

Figura 10: Imagens de referência empregadas nos testes

A Figura 11 apresenta o comportamento da função objetivo, que é o cálculo do PCC, ao longo dos ponto x e y para cada imagem de referência. Os gráficos foram gerados através do programa MATLAB versão R2016B.É possível verificar nas imagens a presença de máximos locais que podem atrair os enxames durante a busca, prejudicando

a convergência. Para evitar a detecção de falsos positivos, foi escolhido o limiar mínimo de 0,95 para a função objetivo, o PCC, para que o alvo seja considerado como detectado.



(a) Cars (b) Pickup

(c) Sedan (d) IR1

(e) IR2 (f) IR3

(g) Truck (h) Rcar

Figura 11: Função objetivo no espaço de busca para cada imagem de referência

O objetivo do trabalho é propor um sistema capaz de trabalhar com vídeos de 30 quadros por segundo, o que demanda encontrar o alvo em até 33 ms. Ao mesmo tempo, para a detecção ser considerada verdadeira, há a necessidade de definir um limiar mínimo do PCC. A eficácia do sistema em encontrar o alvo requer a definição de uma taxa mínima de acerto, medida ao longo de conjuntos de 100 buscas. As metas de avaliação definidos para o presente projeto foram:

- tempo máximo para identificação do alvo: 33 ms;
- taxa mínima de acerto: 60%.

Cada uma das três configurações do sistema foi aplicada às imagens da Figura 10 e recebeu avaliação quanto a tempo de processamento, número de iterações necessário e taxa de acerto obtida. Este procedimento foi repetido por 50 vezes, para obter média e desvio padrão do tempo, das iterações e da taxa de acerto para cada conjunto de 100 buscas. Isto resultou no total de 5000 repetições, para calcular média e desvio padrão.

O critério de parada adotado para a execução dos códigos é encontrar o alvo com $PCC \geq 0,95$ ou atingir o máximo de 10 iterações. Foram verificadas as metas de desempenho para as 5000 repetições da execução do sistema para cada uma das seis técnicas de busca, para as três configurações de *hardware*.

A janela de busca do alvo dentro da imagem principal especifica a região onde a técnica de enxame procurará pelo alvo. Considerando-se um vídeo, se o alvo mudar sua posição lentamente de quadro para quadro, a posição em que foi detectado em um quadro será próxima daquela onde ele será achado no quadro seguinte. Dessa maneira, não é necessário efetuar a busca por toda a imagem, mas sim na vizinhança da localização anterior. Neste trabalho, foi definida uma janela de busca de tamanho 101x101 *pixels*, empregada por (TAVARES, 2016) e empregada em todas as configurações de *hardware*. Os resultados obtidos foram comparados com o desempenho do PSO implementado em (TAVARES, 2016).

Por fim, os parâmetros de cada técnica foram reconfigurados empiricamente para obter taxa média de acerto de pelo menos 90%. Assim, é possível avaliar a relação entre a acurácia alta e o tempo médio de processamento necessário para detectar o alvo em cada imagem.

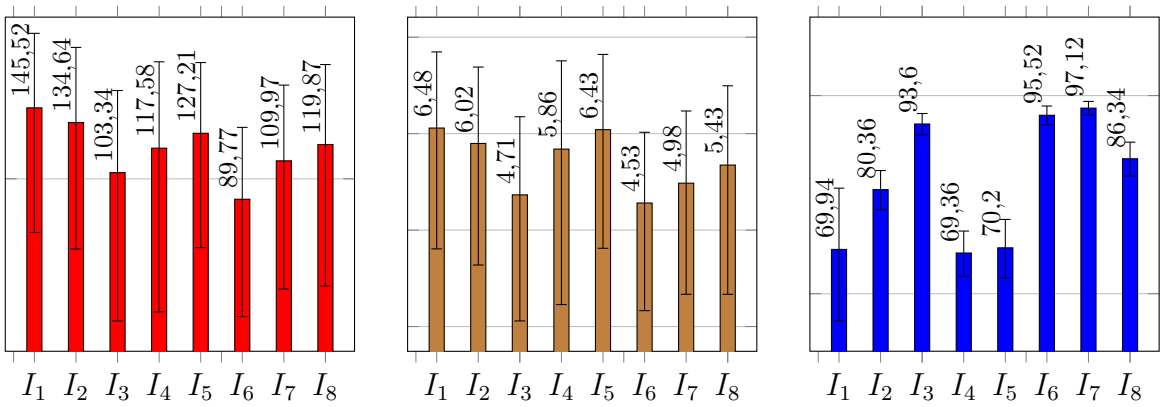
5.2 Análise dos Resultados por Técnica

A presente seção apresenta as configurações empregadas em cada uma das seis técnicas de busca inteligente e os respectivos resultados obtidos, de acordo com a métrica de avaliação definida na Seção 5.1. As técnicas são, respectivamente baseadas no comportamento de: pássaro cuco, abelhas, elefantes, bactérias, vaga-lumes e fogos de artifício. Os parâmetros de cada técnica foram ajustados de maneira empírica de modo a maximizar a taxa de acerto e reduzir o tempo médio de processamento. Os resultados refletem o desempenho do sistema referente ao tempo de processamento (em ms), ao número de iterações e à taxa de acerto, em valores médios e seus respectivos desvios padrões. As Tabelas 2 a 19, presentes no Apêndice, apresentam os resultados para as configurações das técnicas feitas para as metas de tempo de processamento de até 33 ms e taxa de acerto mínima de 60%.

5.2.1 Técnica Baseada no Comportamento do Pássaro Cuco

Os parâmetros utilizados na configuração do CS são: 50 cucos; probabilidade de descoberta: 25%; e fator de escala do voo de Lévy: 15. As Figuras 12 a 14 destacam o desempenho do sistema nas três configurações avaliadas, considerando as imagens de referência. As três figuras estão em escala logarítmica. Os vaores numéricos da média e do desvio padrão encontram-se nas Tabelas

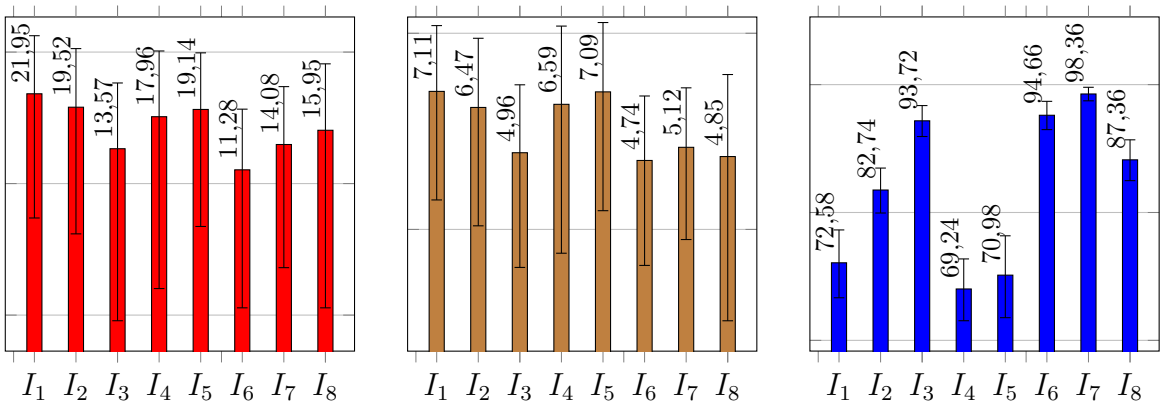
A Figura 12 apresenta os resultados do sistema sem o uso de coprocessador. Os tempos de processamento obtidos são acima do necessário (33 ms), o que faz necessária a melhoria do sistema. A taxa de acerto foi superior a 60% para todas as imagens. O melhor desempenho ocorreu para a imagem $IR3 - I_6$, com a maior taxa de acerto média (97,12%), menor tempo médio para encontrar o alvo (89,77 ms) e o menor número médio de iterações (4,53). O pior desempenho em tempo médio ocorreu na busca do alvo na imagem $Cars - I_1$ (145,52 ms), para o qual foi necessário o maior número médio de iterações (6,48). As piores taxas de acerto ocorreram para as imagens $IR1 - I_4$, $IR2 - I_5$ e $Cars - I_1$, contudo todas de pelo menos 69%. As maiores dispersões dos valores ocorreram para as imagens $Cars - I_1$ e $IR1 - I_4$, representadas pelos seus desvios padrões mais altos em relação às outras imagens para tempo, iterações e taxa de acerto.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 12: Resultados para CS sem uso de coprocessador

A Figura 13 apresenta os resultados para a configuração do sistema com coprocessador operando em modo serial. A meta de tempo de processamento foi atingida para todas as imagens e as taxas de acerto obtidas foram superiores a 69%. O melhor desempenho ocorreu para a busca dos alvos situados nas imagens $IR3 - I_6$, $Sedan - I_3$ e $Truck - I_7$, todos com tempos médios abaixo de 15 ms e com taxas de acerto médias superiores a 93%. O pior desempenho foi obtido através da busca dos alvos encontrados nas imagens $Cars - I_1$, $IR1 - I_4$ e $IR2 - I_5$, cujas taxas de acerto médias ficaram aquém de 72,58%. A maior dispersão de valores de tempo ocorreu para $Cars - I_1$, enquanto a maior para taxa de acerto foi $IR2 - I_5$.



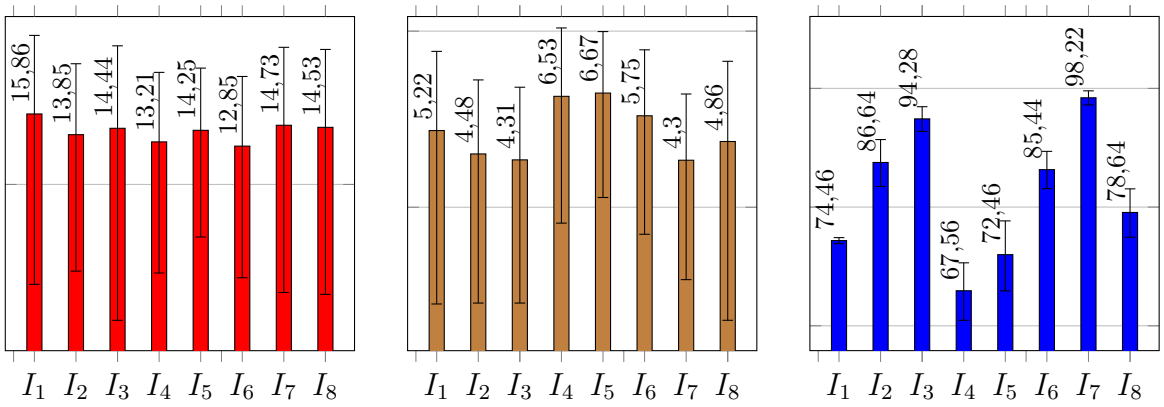
(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 13: Resultados para CS com auxílio de coprocessador operando em modo serial

A Figura 14 exibe o comportamento do sistema quando em uso do coprocessador operando em modo *pipeline*. Os tempos ficaram todos abaixo de 16 ms, com o melhor

desempenho do sistema ocorrido para a imagem $IR3 - I_6$, com tempo médio de 12,85 ms e menor desvio padrão de tempo dentre as imagens. A melhor taxa de acerto média foi obtida junto à imagem $Truck - I_7$, com 98,22% e desvio padrão de 1,34%.

O pior desempenho de taxa média de acerto do alvo aconteceu junto às imagens $IR1 - I_4$ e $IR2 - I_5$, para as quais as taxas de acerto médias foram as menores (67,56% e 72,46%, respectivamente), assim como as quantidades médias de iterações foram as maiores (7,78 e 5,85, respectivamente). Alguns resultados obtidos com a técnica CS foram publicados em (CARDOSO et al., 2018b; CARDOSO; NEDJAH; MOURELLE, 2018).



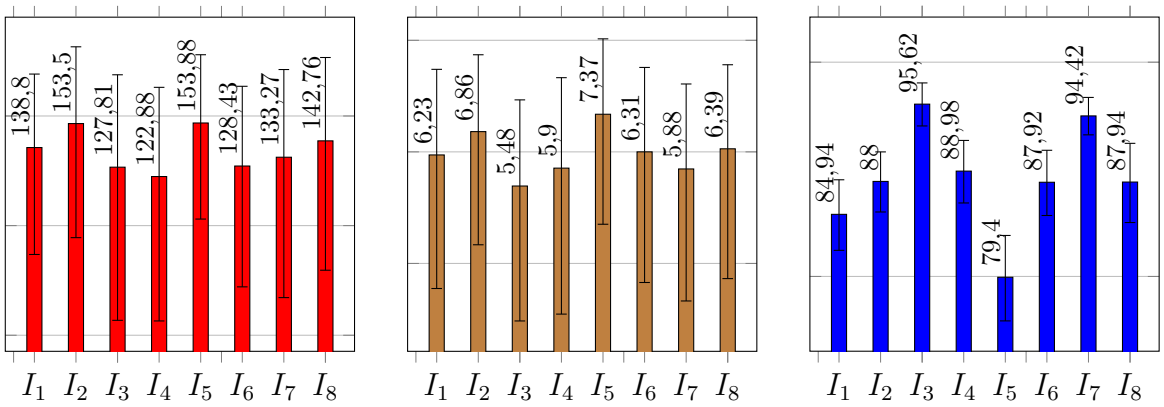
(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 14: Resultados para CS com auxílio de coprocessador operando em modo *pipeline*

5.2.2 Técnica Baseada no Comportamento de Abelhas

O código ABC foi configurado com 8 abelhas, que é o mesmo número SN de fontes de alimento. O número de iterações N_{esg} que define o esgotamento de uma fonte é 1. As Figuras 15 a 17 exibem o desempenho do sistema nas três configurações de *hardware* para as imagens de referência. As três figuras estão em escala logarítmica.

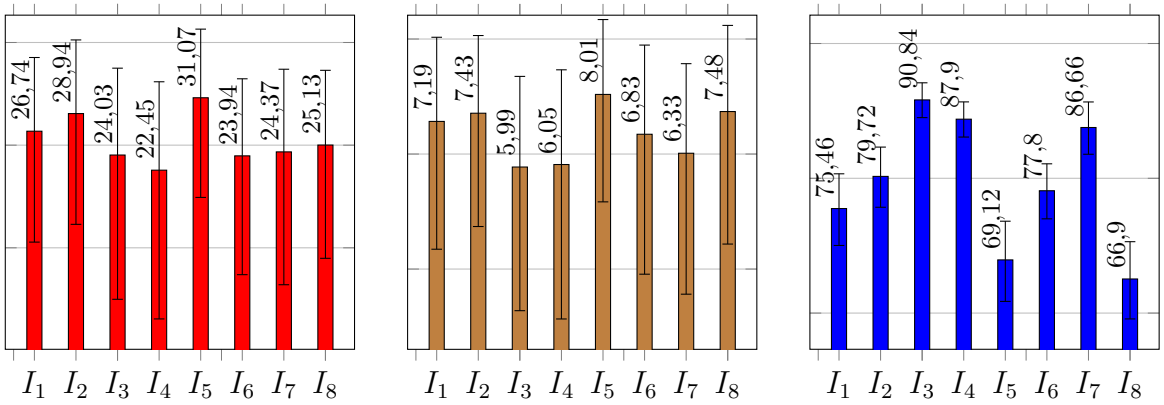
A Figura 15 apresenta os resultados do sistema sem uso de coprocessador. Nenhum dos tempos médios obtidos é menor do que 33 ms. O melhor tempo de processamento médio foi obtido junto à imagem $IR1 - I_4$ (122,88 ms). As taxas médias de acerto são superiores a 70%, sendo que apenas a busca na imagem $IR2 - I_5$ não foi maior do que 80%. As melhores taxas de acerto médias ocorreram através das imagens $Sedan - I_3$ e $Truck - I_7$, superiores a 94%. O pior desempenho foi percebido na busca do alvo em $IR2 - I_5$, para a qual foram necessárias as maiores quantidades médias de tempo e de iterações. A mesma imagem proporcionou a pior taxa média de acerto (79,40%).



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 15: Resultados para ABC sem uso de coprocessador

A Figura 16 apresenta os resultados para a configuração do sistema com coprocessador operando em modo serial. Houve sensível melhora no tempo necessário para achar o alvo, de modo que para todas as imagens a meta temporal foi cumprida, onde o menor tempo médio obtido ocorreu com a imagem $IR1 - I_4$, com 22,45 ms, com desvio padrão de 10,93 ms. A taxa média de acerto obtida é superior a 65%. A busca pelo alvo em $IR2 - I_5$ demandou tempo médio maior (31,07 ms) e mais iterações, bem como obteve uma taxa média de acerto inferior às buscas pelos alvos nas outras imagens.



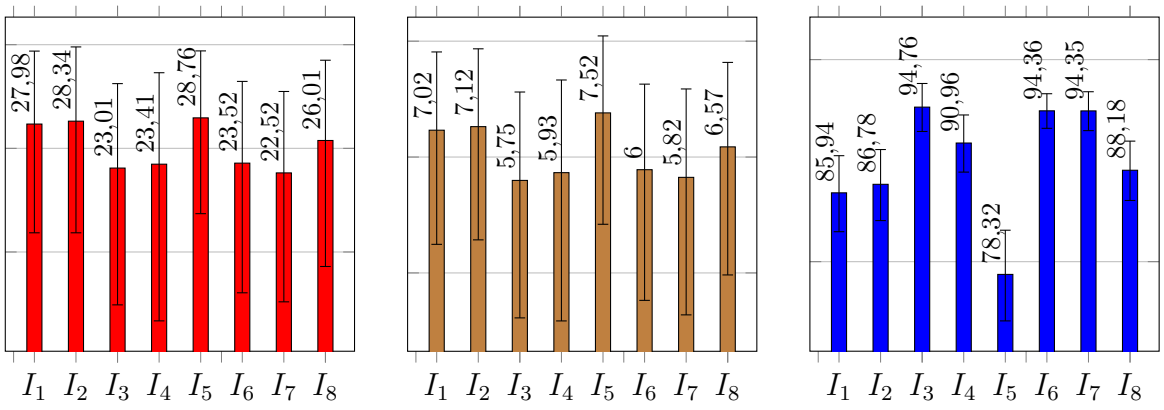
(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 16: Resultados para ABC com auxílio de coprocessador operando em modo serial

A Figura 17 exibe o comportamento do sistema quando com auxílio do coprocessador em modo *pipeline*. Houve ligeira redução no tempo médio e no número de iterações, com a meta temporal de 33 ms sendo atingida para todas as imagens, com as taxas médias

de acerto superiores a 78,32%. O menor tempo médio obtido ocorreu através da imagem $Sedan - I_3$, com 23,01 ms e desvio padrão de 10,48 ms.

O pior desempenho ocorreu para a busca em $IR2 - I_5$, cujo tempo médio de processamento atingiu 28,76 ms. O número de iterações para esta imagem também foi o maior.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

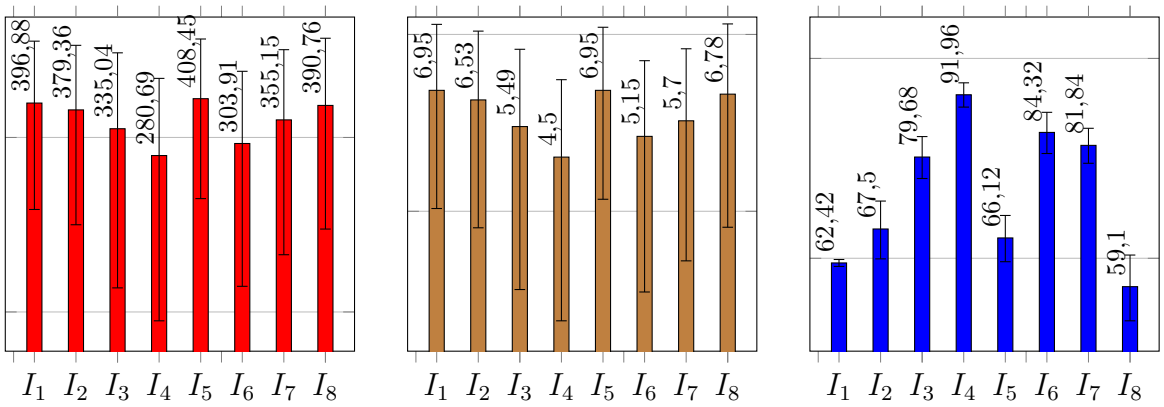
Figura 17: Resultados para ABC com auxílio de coprocessador operando em modo *pipeline*

5.2.3 Técnica Baseada no Comportamento de Elefantes

Os parâmetros utilizados na configuração do EHO são: 32 elefantes; 4 clãs; $\alpha = 0{,}75$; e $\beta = 0$. As Figuras 18 a 20 apresentam os resultados de desempenho para as três configurações empregadas para análise. As três figuras estão em escala logarítmica.

A Figura 18 apresenta os resultados do sistema sem uso de coprocessador. Tal como ocorreu com as técnicas anteriores, os tempos de processamento obtidos são muito acima da meta estabelecida (33 ms), o que torna necessário melhorar o desempenho do sistema. O menor tempo médio foi obrtido para a imagem $IR1 - I_4$, e o pior para $IR2 - I_5$, imagem esta que apresentou o maior número médio de iterações.

A taxa de acerto foi superior a 60%, exceto para a busca pelo alvo na imagem *Rcar*, que obteve 59,10%, com 4,47% de desvio padrão. Os melhores desempenhos ocorreram para as imagens $IR1 - I_4$ e $IR3 - I_6$, com os menores tempos médios e maiores taxas médias de acerto.

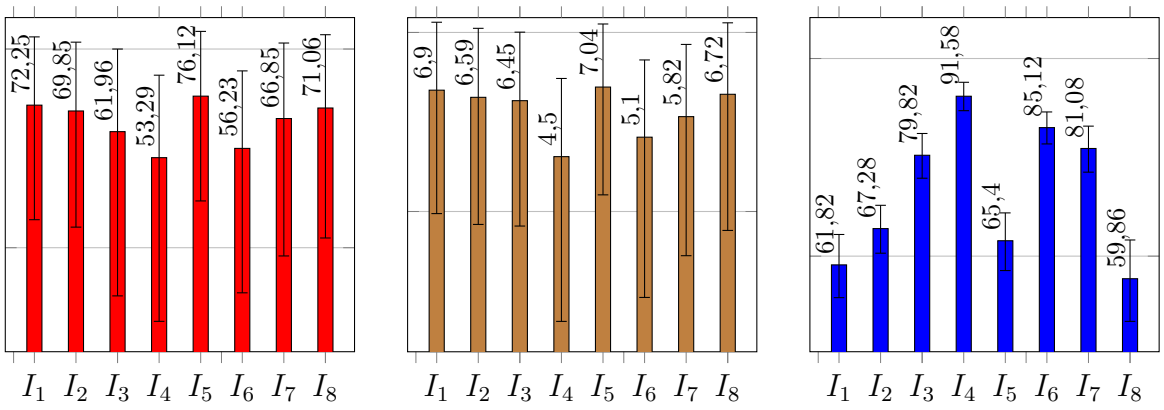


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 18: Resultados para EHO sem uso de coprocessador

A Figura 19 apresenta os resultados do sistema com o uso do coprocessador com operação em modo serial. Houve melhora no tempo de processamento, mas não o necessário para se ter tempos inferiores a 33 ms.

O melhor desempenho ocorreu em tempo médio e taxa média de acerto durante a busca ao alvo em $IR1 - I_4$. O pior, novamente, foi com $Rcar - I_8$, imagem para a qual a taxa média de acerto foi inferior a 60% (59,86% com desvio padrão de 5,66%), além do tempo médio de processamento mais alto para encontrar o alvo (71,06 ms).

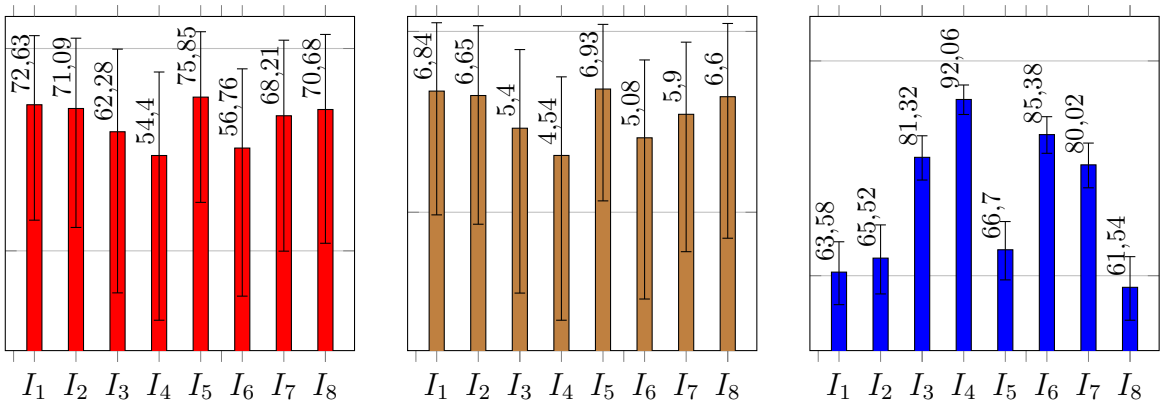


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 19: Resultados para EHO com auxílio de coprocessador operando em modo serial

A Figura 20 exibe o comportamento do sistema na configuração com emprego do coprocessador em operação em modo *pipeline*. A meta de tempo de 33 ms não foi atingida em nenhuma das imagens. Novamente, a busca ao alvo pertencente à imagem $Rcar - I_8$ apresentou taxa de acerto pior do que 60%, logo não atingiu a meta mínima

de desempenho. Alguns resultados parciais da técnica EHO com coprocessador em modo *pipeline*, mas com busca em janela 51 x 51 *pixels*, foram publicados em (CARDOSO et al., 2018a).



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

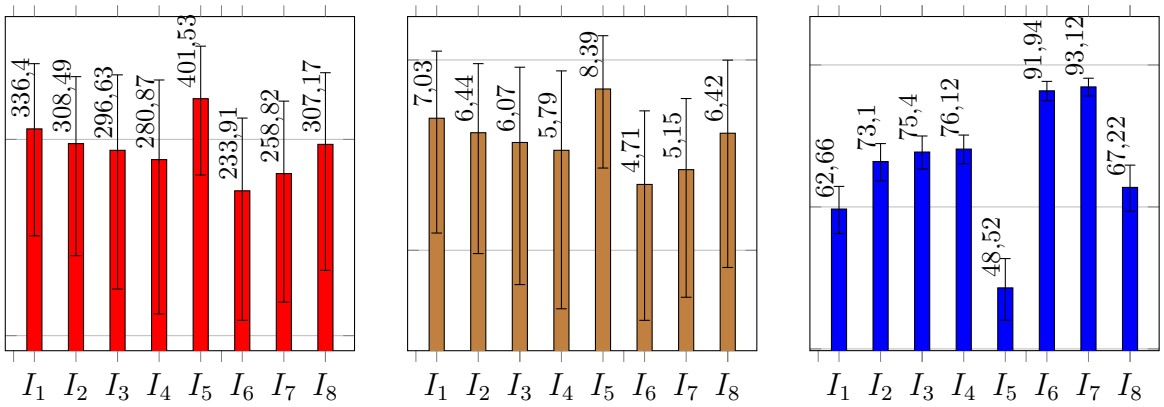
Figura 20: Resultados para EHO com auxílio de coprocessador operando em modo *pipeline*

5.2.4 Técnica Baseada no Comportamento de Bactérias

Os parâmetros utilizados na configuração do BFOA são: 15 bactérias; 1 ciclo de eliminação e dispersão; probabilidade de eliminação e dispersão igual a 30%; 1 ciclo de reprodução; 1 ciclo de quimiotaxia; e 1 ciclo de deslocamento por nado.

As Figuras 21 a 23 mostram os resultados do BFOA com estes parâmetros para as três configurações avaliadas. As três figuras estão em escala logarítmica.

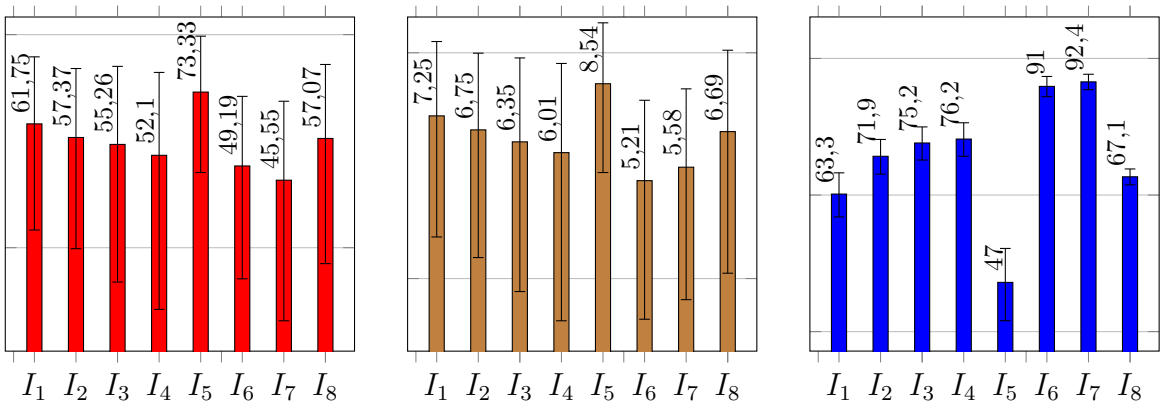
A Figura 21 apresenta o os resultados do sistema sem emprego de coprocessador. Os tempos de processamento obtidos são muito acima do necessário (33 ms), o que novamente solicita a melhoria do sistema. A taxa média de acerto foi superior a 60%, exceto para a busca do alvo na imagem $IR2 - I_5$, que foi de 48.5% e desvio padrão de 3,52%, devido convergência para ótimos locais. O sistema executou melhor a busca dos alvos das imagens $IR3 - I_6$ e $Truck - I_7$, respectivamente.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 21: Resultados para BFOA sem uso de coprocessador

A Figura 22 mostra os resultados do sistema com o uso do coprocessador operando em modo serial. Houve melhora no tempo de processamento, porém ainda com valores de tempo médio superiores a 33 ms. Novamente, os melhores desempenhos referem-se às imagens $IR3 - I_6$ e $Truck - I_7$, enquanto o pior ocorreu na busca do alvo em $IR2 - I_5$, na qual a taxa média de acerto obtida foi de apenas 47%.

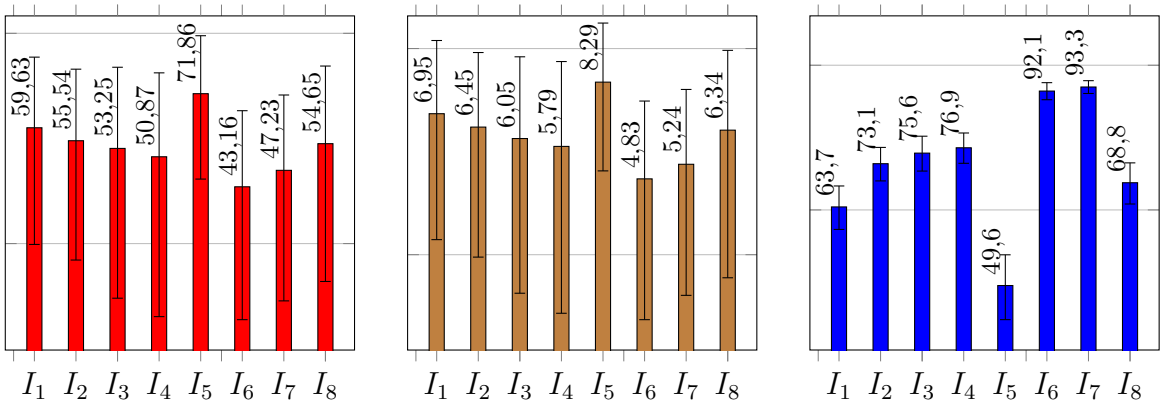


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 22: Resultados para BFOA com auxílio de coprocessador operando em modo serial

A Figura 23 exibe o comportamento do sistema na configuração com emprego do coprocessador com operação em modo *pipeline*. Houve melhora no tempo de resposta em relação ao uso do coprocessador com operação em modo serial. Os melhores desempenhos ocorreram para a busca dos alvos contidos nas imagens $IR3 - I_6$ e $Truck - I_7$, com os menores tempos médios e maiores taxas de acerto. O pior desempenho ocorreu durante a

execução do sistema na busca na imagem $IR2 - I_5$, com taxa de acerto média de 49,6% e desvio padrão de 5,10%, inferior a 60%.



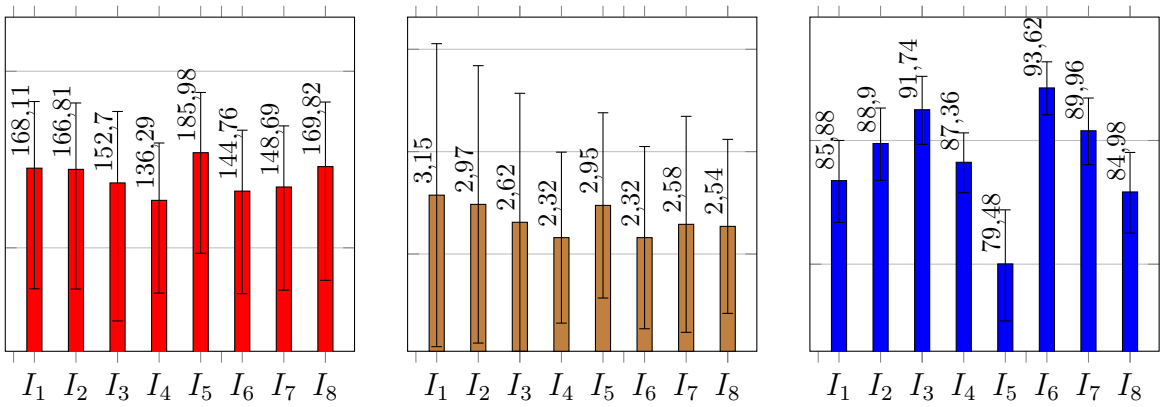
(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 23: Resultados para BFOA com auxílio de coprocessador operando em modo *pipeline*

5.2.5 Técnica Baseada no Comportamento de Vaga-Lumes

Os parâmetros utilizados na configuração do FFA são: 11 vaga-lumes; $\alpha = 3$; $\beta = 1{,}6$; e $\gamma = 0{,}0005$. As Figuras 24 a 26 apresentam os resultados do FFA com estes parâmetros para as três configurações avaliadas. As figuras de tempo de processamento e taxa de acerto estão em escala logarítmica, e a de iterações está em escala linear. Como o número médio de iterações para a técnica FFA para achar o alvo com esta configuração de parâmetros é próximo de 3, foi imposta uma limitação de 7 para o número máximo de iterações, ao invés de 10, de forma que o desvio padrão não fosse superior à média.

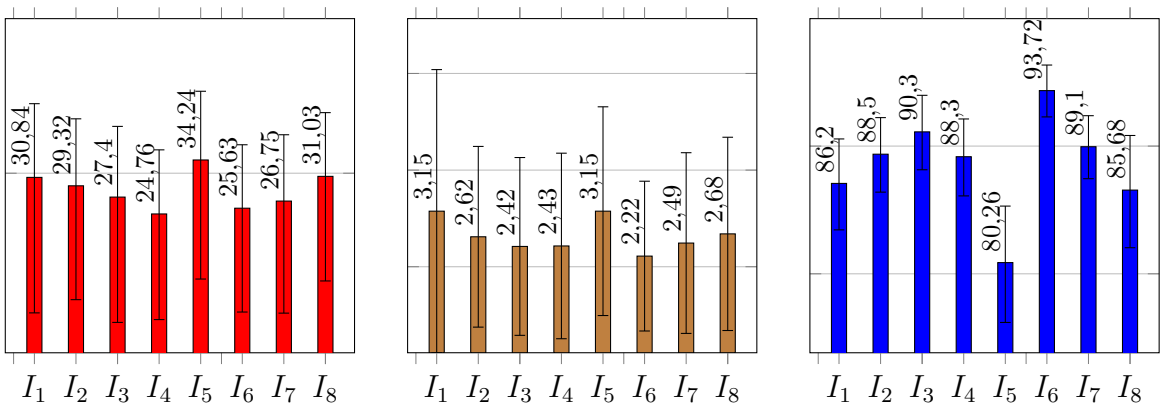
A Figura 24 destaca os resultados do sistema sem uso de coprocessador. Os tempos de processamento decorrentes são acima da meta de 33 ms, fato que mostra ser necessária a melhoria do sistema. A taxa de acerto foi superior a 79% nas buscas dos alvos em todas as imagens, e os melhores desempenhos do sistema ocorreram para $IR1 - I_4$, $IR3 - I_6$ e $Truck - I_7$, com menores tempos médios de processamento, menores números médios de iteração e maiores taxas médias de acerto. Já o pior tempo médio de busca ocorreu em relação à imagem $IR2 - I_5$, mas com taxa de acerto semelhante a das outras imagens, devido à complexidade da função objetivo com relação a esta imagem.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 24: Resultados para FFA sem uso de coprocessador

A Figura 25 mostra os resultados do sistema com o uso do coprocessador de arquitetura serial. Houve melhora no tempo de processamento, na qual a meta de tempo de processamento foi atingida para sete imagens, apenas não o foi para $IR2 - I_5$ (34,24 ms). O alvo foi encontrado mais rapidamente em $IR1 - I_4$, com 24,76 ms de média. Todas as taxas médias de acerto foram superiores a 80%.

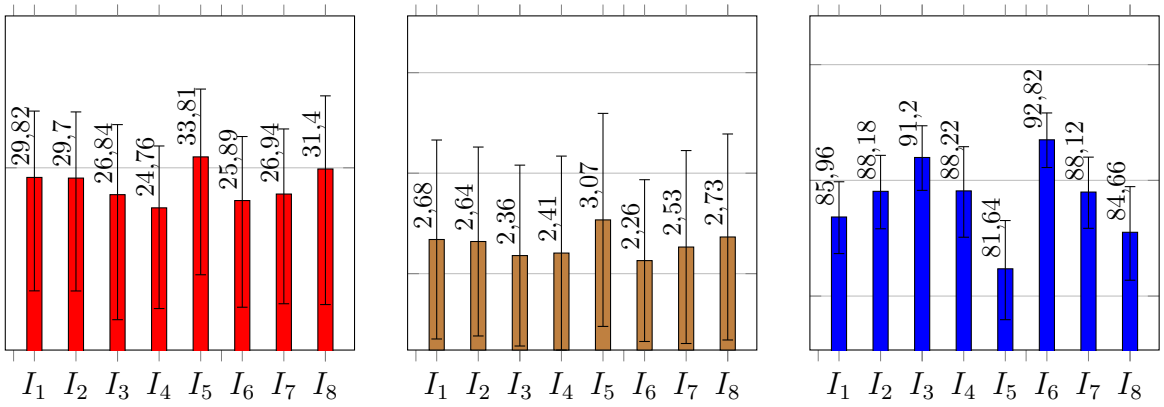


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 25: Resultados para FFA com auxílio de coprocessador operando em modo serial

A Figura 26 exibe o comportamento do sistema na configuração com emprego do coprocessador operando em modo *pipeline*. Em relação ao tempo médio, houve melhora em relação ao modo serial, atingindo a meta novamente para sete imagens. Apenas $IR2 - I_5$ ficou acima da meta, com 33,8 ms Todavia, a taxa média de acerto foi superior a 80%, cumprindo a meta. Os melhores desempenhos ocorreram para as imagens $IR1 - I_4$

e $IR3 - I_6$, com os menores tempos médios de busca. As maiores taxas médias de acerto ocorreram para $IR3 - I_6$ e $Sedan - I_3$.

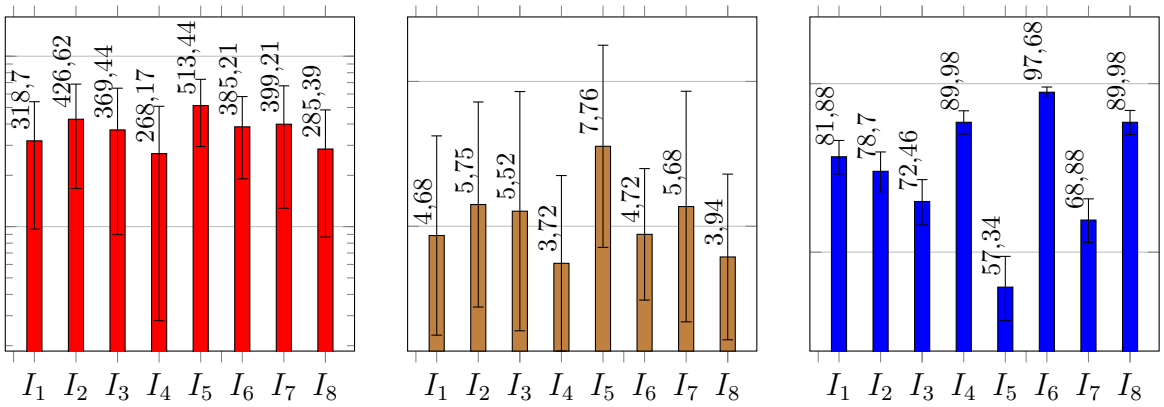


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 26: Resultados para FFA com auxílio de coprocessador operando em modo *pipeline*

5.2.6 Técnica Baseada no Comportamento de Fogos de Artifício

Os parâmetros utilizados na configuração do FWA são: número de fogos de artifício $N_F = 25$; $M = 50$; $\alpha = 0{,}02$; $\beta = 1{,}00$; e $\xi = 0{,}0001$. As Figuras 27 a 29 exibem os resultados do FWA com estes parâmetros em relação às três configurações avaliadas. As figuras de tempo de processamento e taxa de acerto estão em escala logarítmica, e a de iterações está em escala linear.



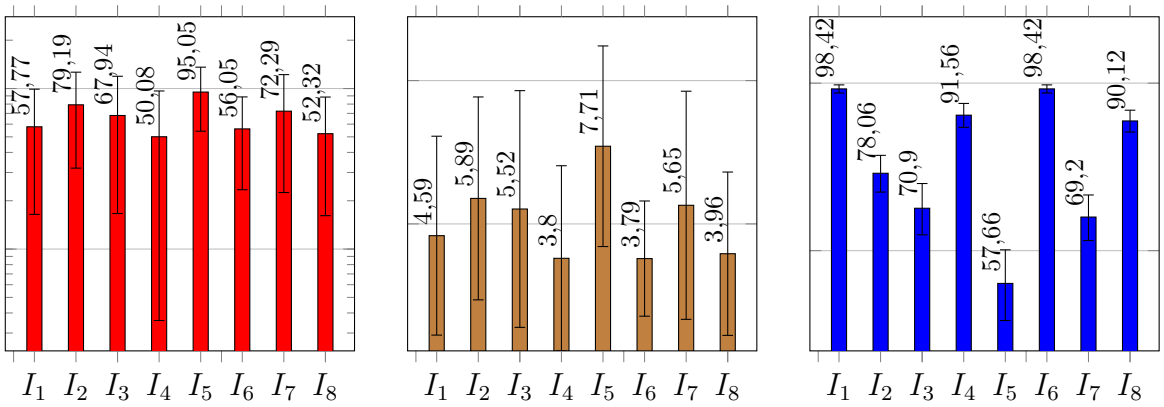
(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 27: Resultados para FWA sem uso de coprocessador

A Figura 27 apresenta os resultados do sistema sem uso de coprocessador. Os tempos médios obtidos são acima da meta de 33 ms, e se obriga a implementar a melhoria

sistema. As taxas de acerto médias foram superiores a 60%, exceto para a busca ao alvo na imagem $IR2-I_5$, que atingiu apenas 57,34% com desvio padrão de 3,52%. Os melhores desempenhos referem-se as buscas nas imagens $IR3-I_6$, $IR1-I_4$ e $Rcar-I_8$, com taxas médias de acerto de pelo menos 90%.

A Figura 28 mostra os resultados do sistema com o uso do coprocessador operando em modo serial. Houve melhora no tempo médio de processamento, mas os valores foram acima de 33 ms. A taxa média de acerto obtida cumpre a meta de 60%, exceto novamente para a busca do alvo na imagem $IR2-I_5$, que atinge apenas 57,66%.

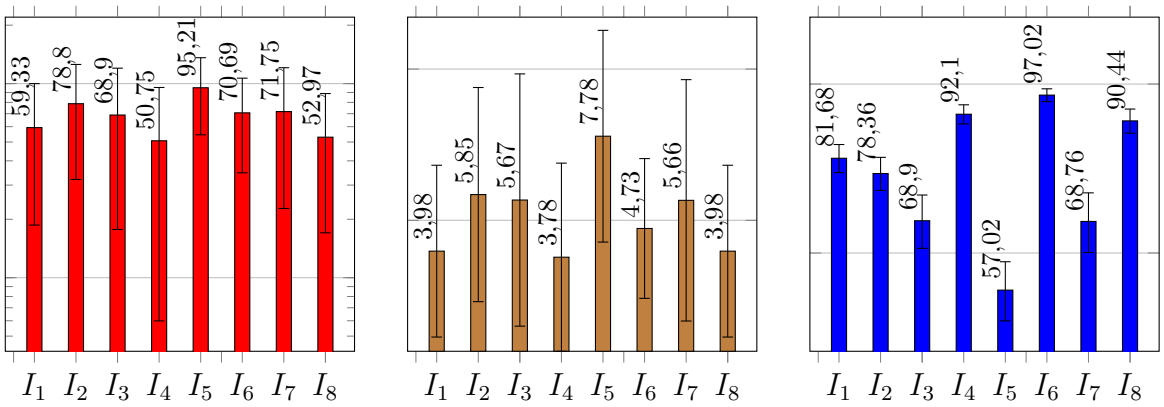


(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 28: Resultados para FWA com auxílio de coprocessador operando em modo serial

A Figura 20 exibe o comportamento do sistema na configuração com emprego do coprocessador com operação em modo *pipeline*. Os resultados foram próximos dos obtidos com coprocessador com operação em modo serial.

O melhor desempenho em tempo médio ocorreu para a imagem $IR1-I_4$ (50,75 ms, superior aos 33 ms de meta), e a melhor taxa média de acerto foi de 97,02% para $IR3-I_6$. O sistema comportou-se pior para a imagem $IR2-I_5$, para a qual a busca não atingiu as metas de tempo e nem de taxa de acerto, com 95,21 ms e 57,02% para os valores médios, respectivamente.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 29: Resultados para FWA com auxílio de coprocessador operando em modo *pipeline*

5.3 Comparações entre as Técnicas

A presente seção destaca a comparação dos resultados obtidos pelas técnicas, conforme as três configurações listadas na Seção 5.1. Para que seja adequado a trabalhar com vídeos de 30 quadros por segundo, o sistema deve encontrar o alvo nas imagens em até 33 ms. Além disso, a taxa de acerto do sistema precisa ser igual ou maior do que 60% para que a informação seja considerada verdadeira. Os valores obtidos estão listados nas Tabelas 2 a 19, presentes no Apêndice desta dissertação.

5.3.1 Operação sem Coprocessador

Esta subseção apresenta a comparação dos resultados obtidos na configuração do sistema de rastreamento em que as técnicas de busca inteligente são implementadas em *software* e são executadas pelo processador de uso geral do XC7Z015 para o cálculo do PCC. Não há uso de coprocessador para acelerar a busca.

A Figura 30 mostra que nenhuma técnica atingiu a meta de tempo de processamento sem usar o coprocessador. As técnicas CS e ABC necessitam de tempos menores para encontrar o alvo, inclusive em relação ao PSO. As técnicas EHO e FWA gastam mais tempo em comparação com as outras. De modo geral, a busca pelo alvo na imagem $IR2 - I_5$ foi a que precisou de mais tempo para que seu alvo fosse encontrado, enquanto $IR1 - I_4$ foi a imagem que demandou menos tempo.

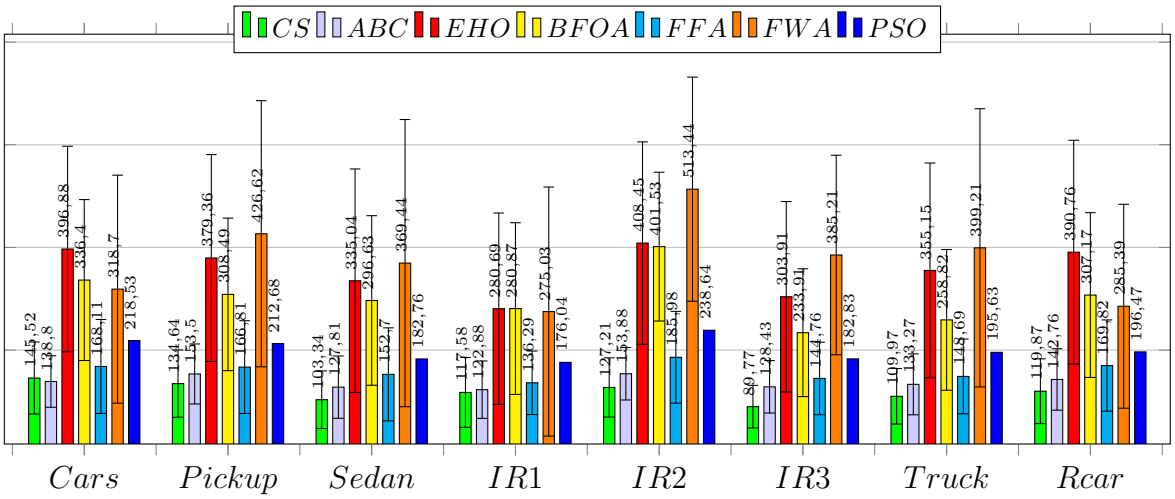


Figura 30: Comparação de tempo de processamento (ms) sem uso de coprocessador

A Figura 31 aponta que as técnicas necessitam de aproximadamente a mesma quantidade de iterações para encontrar o alvo, exceto FFA. A imagem $IR2 - I_5$ demandou o maior número de iterações para que seu alvo fosse encontrado.

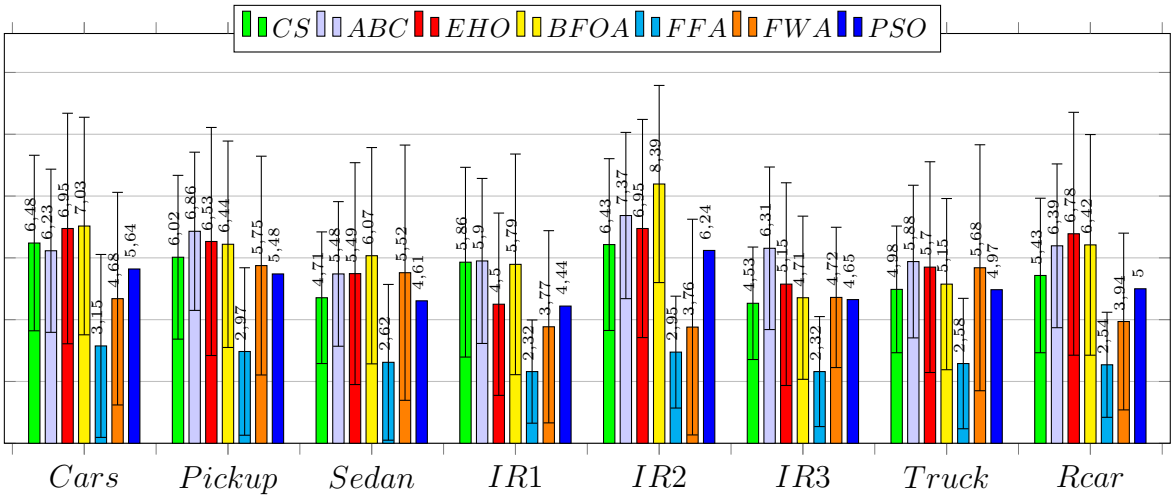


Figura 31: Comparação de número de iterações sem uso de coprocessador

A Figura 32 mostra a comparação de desempenho das técnicas implementadas com relação à taxa média de acerto, usando como meta a taxa de 60%. As implementações do sistema com as técnicas de busca atingiram a meta para as imagens, exceto para $IR2 - I_5$ (BFOA e FWA) e $Rcar - I_8$ (EHO). As técnicas CS e ABC proporcionaram as maiores taxas médias de acerto, ainda que inferiores à obtidas pela técnica PSO para os mesmos critérios de parada.

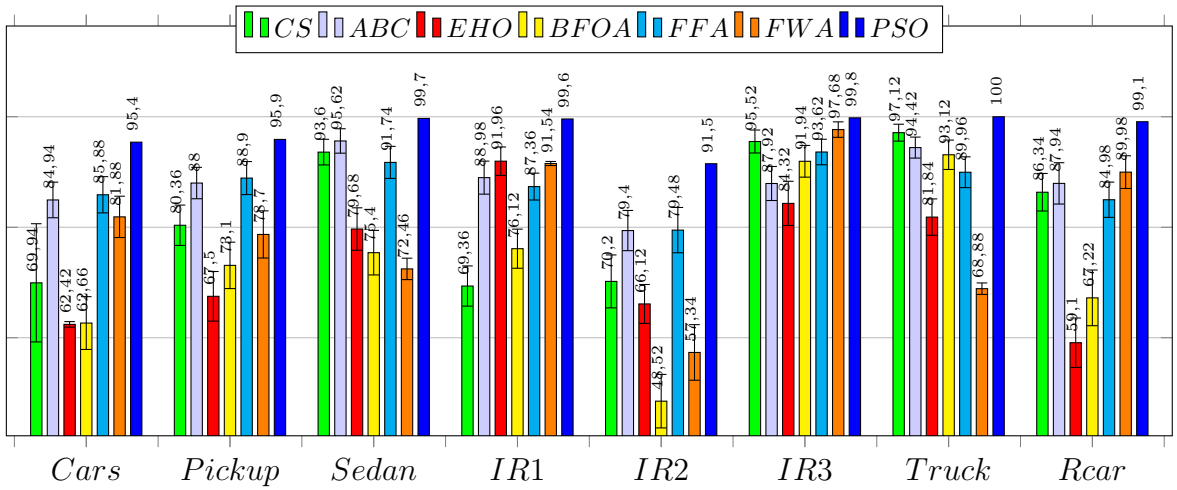


Figura 32: Comparação de taxa de acerto (%) sem uso de coprocessador

5.3.2 Operação com Auxílio de Coprocessador Serial

Esta subseção apresenta a comparação dos resultados obtidos na configuração do sistema de rastreamento em que as técnicas de busca inteligente são implementadas em *software* e o cálculo do PCC, neste caso, é realizado pelo coprocessador configurado em modo serial de operação. A Figura 33 mostra que houve redução no tempo médio de processamento para encontrar o alvo. Somente as técnicas CS e ABC atingiram a meta de 33 ms para todas as imagens de referência. A técnica FFA atingiu para sete imagens. A técnica de referência, PSO, atingiu a meta para apenas duas imagens ($Sedan - I_3$ e $IR1 - I_4$).

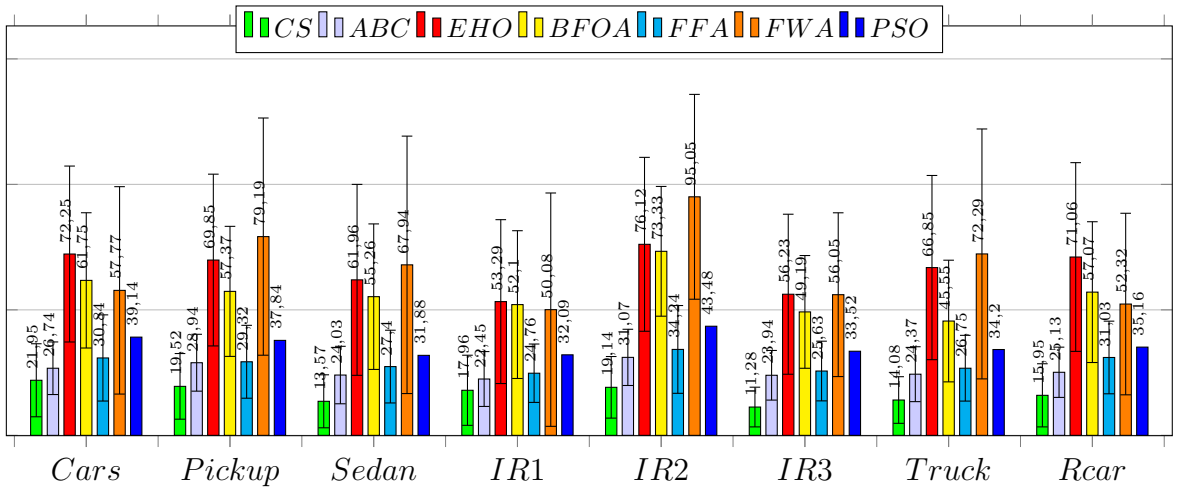


Figura 33: Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

A Figura 34 mostra que as técnicas novamente empregaram uma quantidade semelhante de iterações para encontrar o alvo, exceto a técnica FFA, que precisou em média

de menos de 3 iterações para todas as imagens. A imagem $IR2 - I_5$ exigiu o maior número de iterações para todas as seis técnicas implementadas e para o PSO, devido a sua complexidade perante o cálculo do PCC.

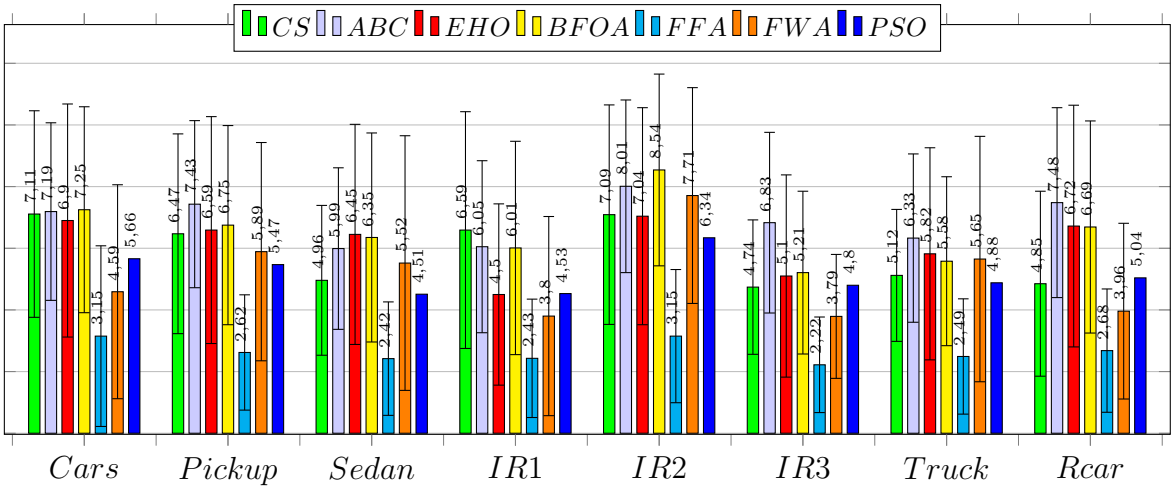


Figura 34: Comparação de número de iterações com auxílio de coprocessador operando em modo serial

A Figura 35 mostra a comparação de desempenho das técnicas implementadas com relação à taxa média de acerto para o sistema usando coprocessador operando em modo serial. Para a meta de 60%, as implementações das técnicas de busca atingiram a meta para as imagens, exceto para $IR2 - I_5$ (BFOA e FWA) e $Rcar - I_8$ (EHO), tal como ocorreu sem o coprocessador.

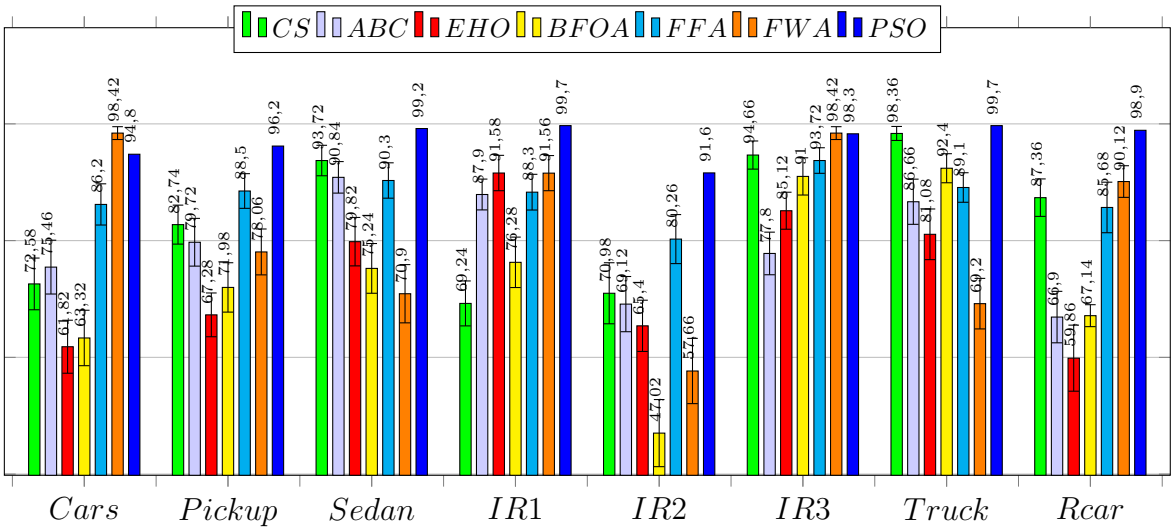


Figura 35: Comparação de taxa de acerto (%) com auxílio de coprocessador operando em modo serial

As técnicas CS e ABC novamente proporcionaram as maiores taxas médias de acerto, porém inferiores à obtidas pela técnica PSO em (TAVARES, 2016). A técnica CS conseguiu superar 95% para $Sedan - I_3$ e $Truck - I_7$, e a FWA superou para a imagem $IR3 - I_3$.

5.3.3 Operação com Auxílio de Coprocessador *Pipeline*

Esta subseção apresenta os resultados obtidos quando busca inteligente foi implementada em *software* e o cálculo do PCC foi realizado pelo coprocessador trabalhando em *pipeline*.

A Figura 36 mostra os resultados para a busca dos alvos nas imagens de referência, com emprego do sistema usando o coprocessador operando em modo *pipeline*. As técnicas CS e ABC atingiram a meta de tempo médio de 33 ms para todas as imagens de referência. A técnica FFA atingiu a meta de tempo médio em sete imagens.

Já as outras técnicas não obtiveram tempos inferiores à meta, sendo os piores desempenhos obtidos por FWA e EHO. A imagem que precisou de maior tempo médio para ter seu alvo encontrado foi $IR2 - I_5$, enquanto a que precisou de menos tempo médio foi $IR1 - I_4$.

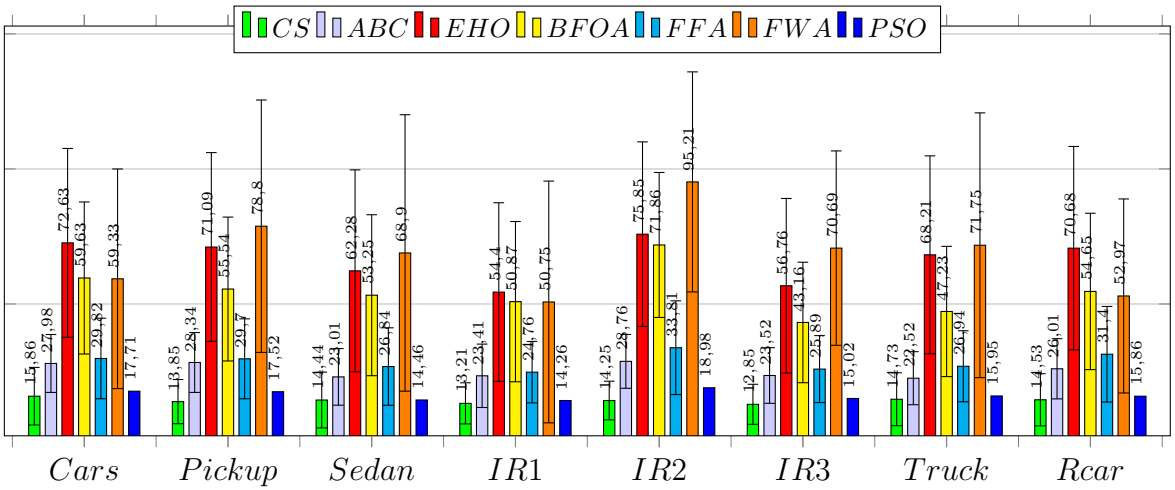


Figura 36: Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline*

A Figura 37 mostra os resultados relativos à execução do sistema usando um coprocessador operando em modo *pipeline*, em relação ao número de iterações. A técnica FFA necessitou sempre de menos iterações para achar o alvo do que as outras. A imagem $IR2 - I_5$ demandou o maior número de iterações para as técnicas implementadas, e está

relacionado à menor taxa média de acerto para esta imagem em comparação com as outras.

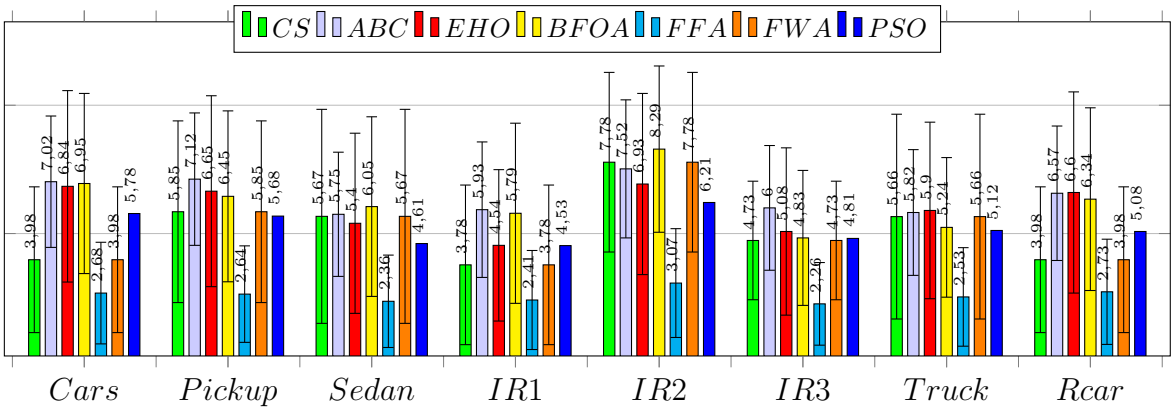


Figura 37: Comparação de número de iterações com auxílio de coprocessador operando em modo *pipeline*

A Figura 38 apresenta os resultados para a taxa média de acerto. A implementação da técnica CS atingiu a meta de 60% para todas as imagens de referência, com taxas superiores a 90% para $Truck - I_7$ e $Sedan - I_3$. A técnica de busca ABC também atingiu a meta para todas as imagens, com taxas médias superiores a 85% para seis delas. A técnica EHO superou a meta de 60% para busca dos alvos em todas as imagens de referência. A técnica BFOA só atingiu a meta mínima de taxa de acerto para sete imagens, para a configuração de parâmetros escolhida, ficando abaixo de 60% na busca em $IR2 - I_5$, com apenas 49,62%. A técnica FFA proporcionou resultados acima de 80% para todas as imagens, sendo superior a 91% para $Sedan - I_3$ e $IR3 - I_6$. A técnica FWA superou a meta para sete imagens, com os melhores resultados sendo obtidos para $IR3 - I_6$ e $IR1 - I_4$, com 97,02% e 92,10%, respectivamente.

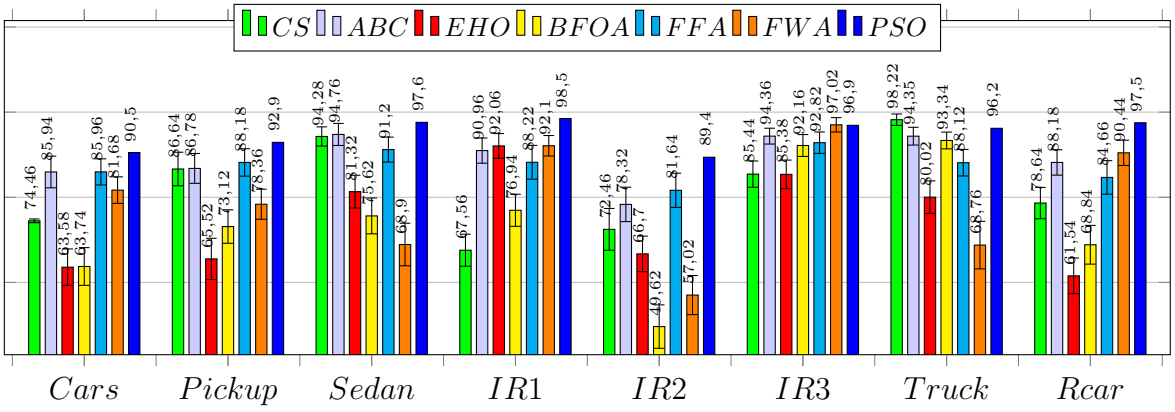


Figura 38: Comparação de taxa de acerto (%) para coprocessador com auxílio de coprocessador operando em modo *pipeline*

As técnicas que apresentaram acurácia abaixo de 60% ainda podem ser aproveitadas com as configurações de parâmetros empregadas para localização do alvo, desde que uma outra ferramenta seja empregada conjuntamente para reforçar o processo decisório. O sistema empregado com esta técnica precisaria processar uma imagem do ambiente com um alvo pré-definido para obter a taxa de acerto que seria usada como peso para tomada de decisão. Se a taxa média for alta, o peso será alto. Dessa maneira, pode-se aplicar um processo de validação cruzada. Para que uma técnica apresente taxas de acerto maiores, os parâmetros precisariam ser alterados tendo como referência a imagem na qual apresentou o pior desempenho em taxa de acerto para uso de coprocessador com operação em modo paralelo; todavia, estas mudanças de configuração podem implicar em possíveis tempos de processamento piores.

5.4 Novas Configurações

As técnicas tiveram seus parâmetros empiricamente reconfigurados para obter taxa média de acerto de pelo menos 90% para todas as oito imagens definidas para referência. O objetivo é mostrar o impacto no tempo médio de processamento de cada técnica em caso de necessidade de se aplicar o sistema em aplicações com grau de certeza mais próximo de 100%. Assim, obtiveram-se novos valores médios de tempo de processamento e de iterações. Os parâmetros de cada técnica, alterados ou mantidos, estão presentes na Tabela 1.

Um número maior de indivíduos do enxame leva a mais cálculos da função objetivo a cada ciclo de iterações, o que tende a aumentar o tempo de processamento. Contudo, também tende a posicionar indivíduos perto do ótimo global, o que levaria a aumentar a taxa média de acerto. Cabe ressaltar a limitação do número de 50 indivíduos para a configuração do coprocessador operando em modo *pipeline* desenvolvido em (TAVARES, 2016). Assim, nova análise de desempenho baseia-se nas abordagens sem uso de coprocessador e com emprego do sistema com coprocessador operando em modo serial. Os valores obtidos estão listados nas Tabelas 20 a 31, presentes no Apêndice desta dissertação.

Tabela 1: Novas configurações dos parâmetros para as técnicas

Técnica	Parâmetros ajustados		Parâmetros mantidos	
CS	N_c	250	P_a	25%
			λ	15
ABC	N_a	21	N_{esg}	1
			E_{exp}	5
EHO	N_e	105	α	2,75
	N_{cla}	15	β	0,001
BFO	N_b	65	N_{ed}	1
			N_{re}	1
	P_{ed}	30%	N_{quim}	1
FFA	N_v	17	β	1,60
			γ	0,0005
FWA	N_f	200	B	1
	A	0,01		
	K	800	ξ	0,001

A Figura 39 exibe os resultados referentes ao tempo médio de processamento obtido sem uso do coprocessador, para 5000 repetições. Em todos os casos, os tempos obtidos são muito acima da meta estabelecida de 33 ms, o que inviabiliza a utilização em aplicações em vídeos de 30 quadros por segundo.

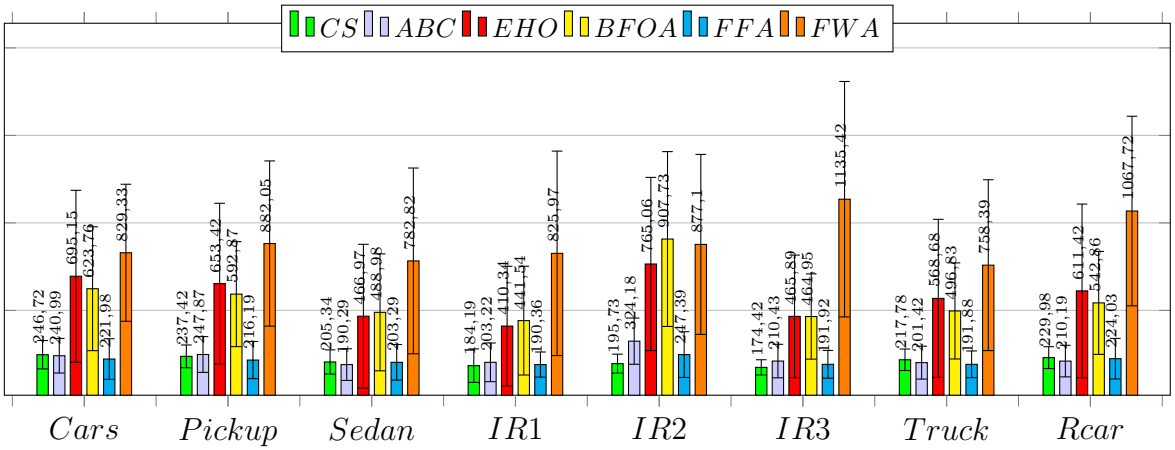


Figura 39: Comparação de tempo de processamento (ms) sem uso de coprocessador

A Figura 40 mostram os resultados referentes ao número médio de iterações que cada técnica necessitou, sem fazer uso do coprocessador. Para as novas configurações, as quantidades médias de iterações foram menores daquelas produzidas para as configurações das técnicas apresentadas nas Seções 5.2 e 5.3. Isso se deve ao fato do número maior de

elementos empregado em cada enxame, o que tende a posicionar um indivíduo perto do ótimo global.

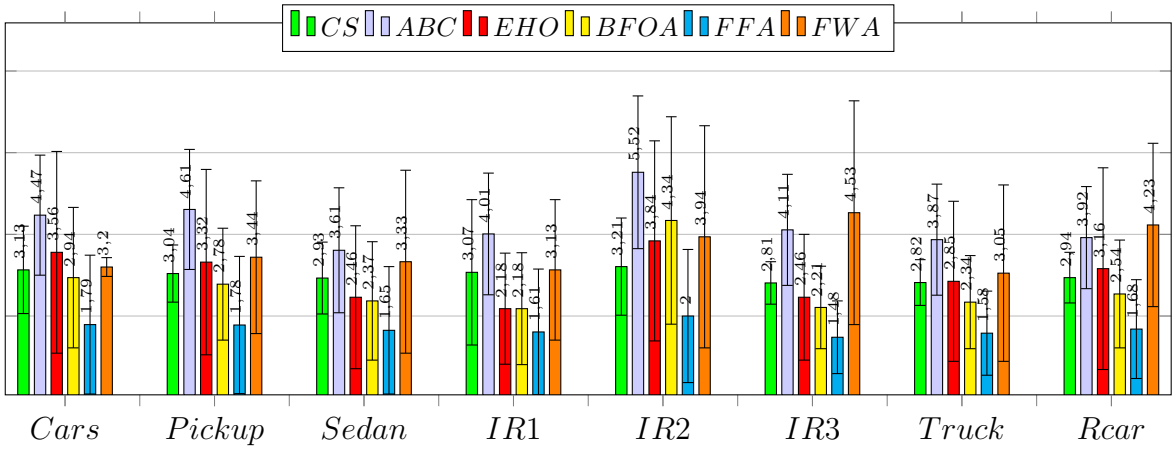


Figura 40: Comparação de número de iterações sem uso de coprocessador

A Figura 41 apresenta os resultados referentes à taxa média de acerto obtida sem uso do coprocessador. Todas as taxas foram próximas ou superiores a 90%.

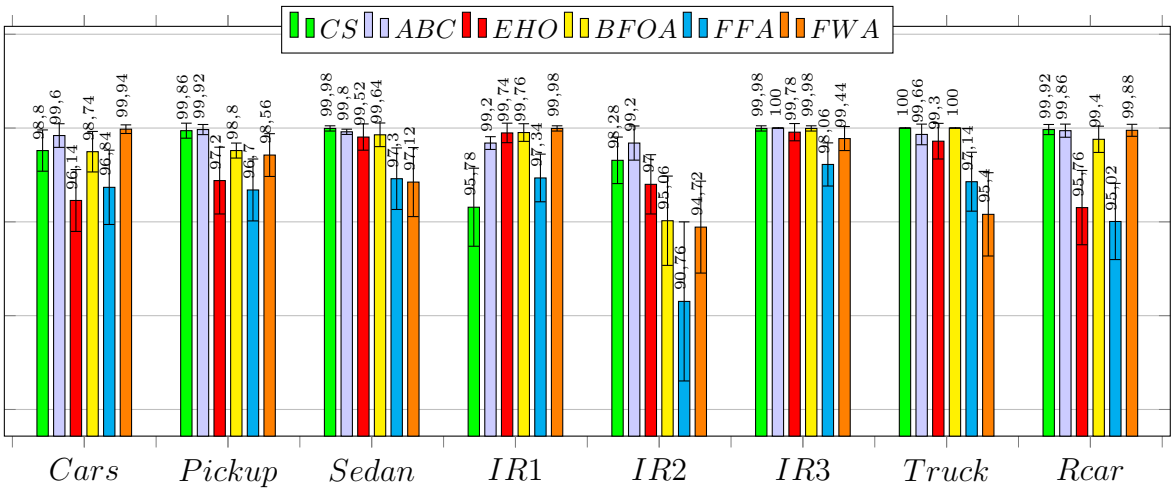


Figura 41: Comparação de taxa de acerto (%) sem uso de coprocessador

A Figura 42 apresenta os resultados relativos ao tempo médio de processamento obtido com uso do coprocessador de arquitetura serial. Os tempos foram piores do que os obtidos com a configuração anterior de parâmetros para as técnicas. Apenas a técnica CS obteve tempos inferiores à meta de 33 ms para todas as imagens. As outras todas ficaram acima da meta, sendo que o pior desempenho coube à técnica FWA. A técnica FFA proporcionou tempos inferiores a 40 ms para seis imagens.

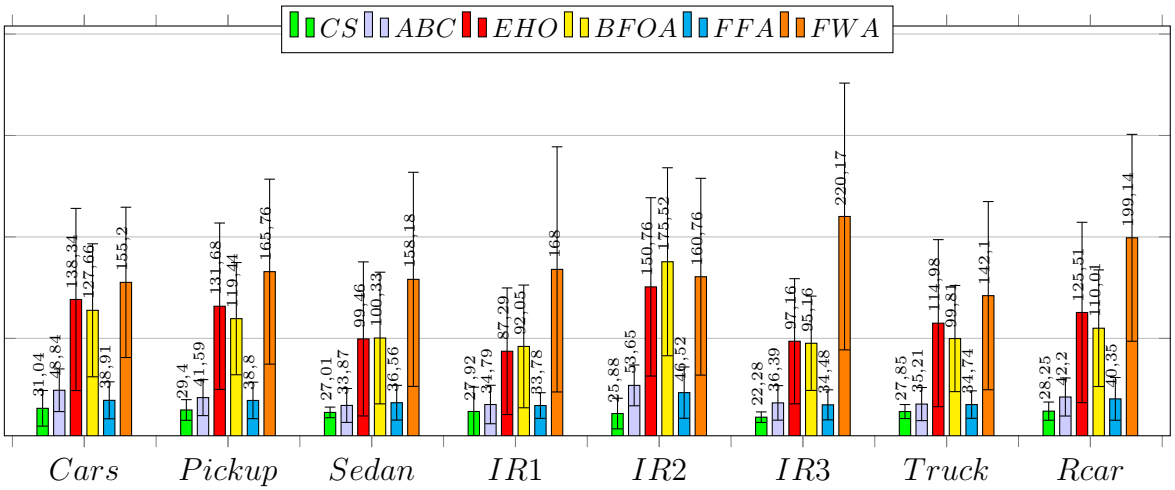


Figura 42: Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

A Figura 43 exibe os resultados da quantidade média de iterações obtida com uso do coprocessador de arquitetura serial. O número é menor do que o necessário na configuração apresentada nas Seções 5.2 e 5.3, pelo fato do maior número de elementos no enxame distribuídos pelo espaço de busca, o que tende a posicionar indivíduos próximos do ótimo global. A técnica de busca FFA apresentou quantidades próximas de dois, o que significa que na maioria das vezes achou o alvo até o segundo ciclo de iterações. A imagem que demandou o maior número médio de iterações para encontrar o alvo foi $IR2 - I_5$, enquanto a imagem $IR1 - I_4$ precisou do menor número de iterações.

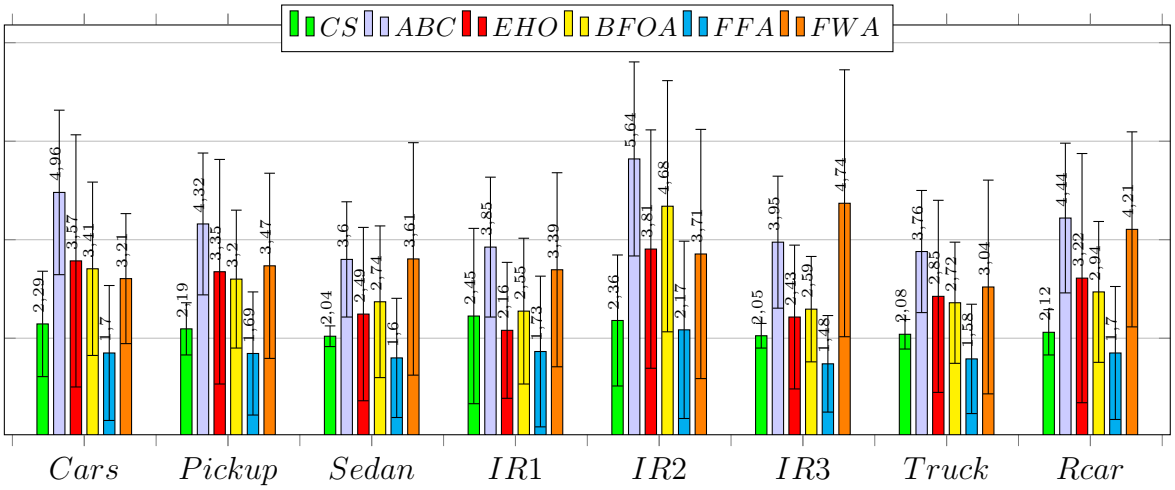


Figura 43: Comparação de número de iterações com auxílio de coprocessador operando em modo serial

A Figura 44 apresenta os resultados referentes à taxa de acerto média obtida com uso do coprocessador operando em modo serial. Todas as taxas médias obtidas são

superiores a 90%, exceto para a técnica FFA, que apresentou 89,94% para a imagem $IR2 - I_5$, mas com desvio padrão de 3,29%. A imagem cuja busca pelo alvo proporcionou as maiores taxas de acerto foi $IR3 - I_6$, enquanto $IR2 - I_5$ proporcionou as taxas médias mais baixas.

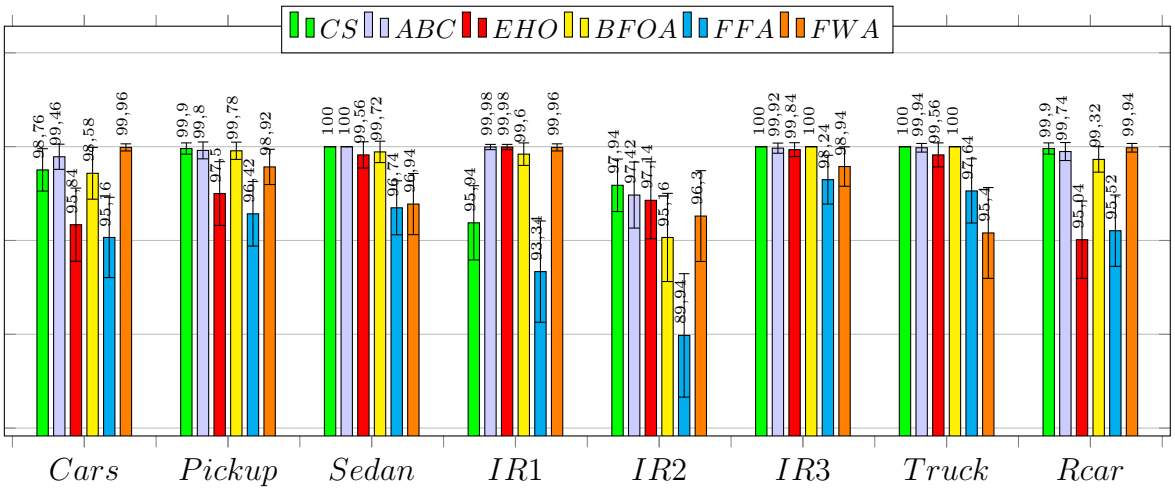
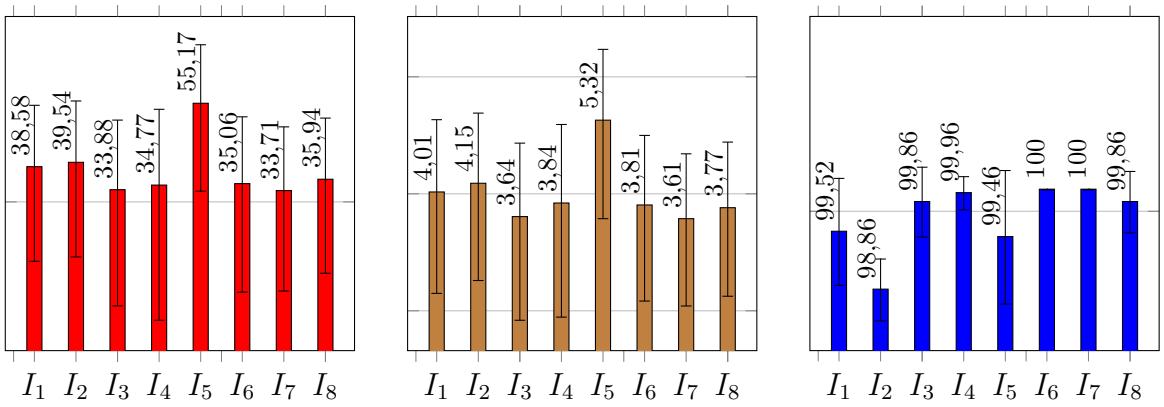


Figura 44: Comparação de taxa de acerto (%) com auxílio de coprocessador operando em modo serial

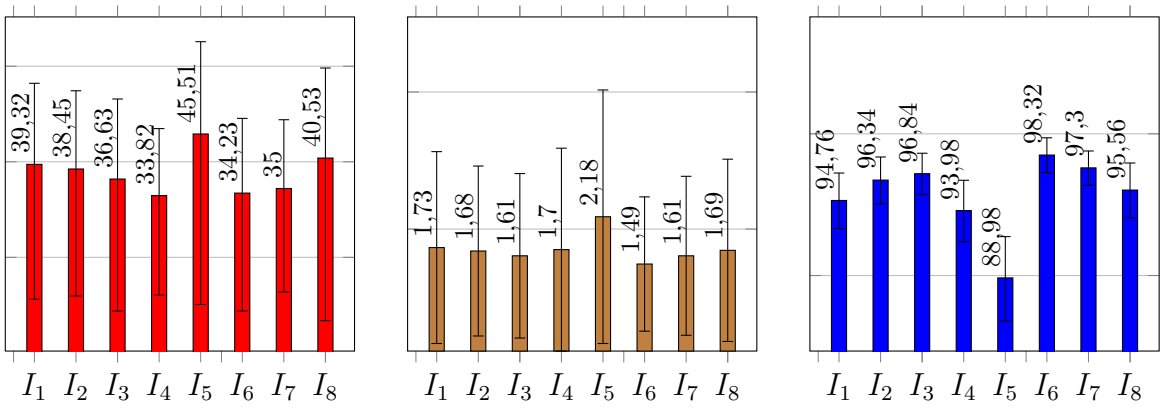
A nova configuração de parâmetros da técnica ABC permitiu o uso com coprocessador em operação no modo *pipeline*. Isto decorre da quantidade de abelhas inferior a 50. As taxas de acerto são todas superiores a 90%, porém a meta de tempo médio de 33 ms não é atingida, conforme mostrado na Figura 45. Os piores desempenhos em tempo médio para encontrar o alvo ocorreram para as imagens $Pickup - I_2$, $IR2 - I_5$ e $Cars - I_1$.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 45: Novos resultados para ABC com auxílio de coprocessador operando em modo *pipeline*

Tal como ocorreu em relação à técnica ABC, a nova configuração de parâmetros da técnica FFA permitiu o uso com coprocessador em operação no modo *pipeline*,devido ao fato da quantidade de vaga-lumes ser inferior a 50. Todas as taxas de acerto são superiores a 90%; contudo, o tempo médio é superior a 33 ms para todas as imagens, conforme mostrado na Figura 46. Os piores desempenhos em tempo médio para encontrar o alvo ocorreram para as imagens $Pickup - I_2$, $IR2 - I_5$ e $Cars - I_1$. Para a imagem $IR2 - I_5$, a taxa de acerto média obtida foi de 88,98%, mas com desvio padrão de 3,04%.



(a) Tempo de processamento (ms) (b) Número de iterações (c) Taxa de acerto (%)

Figura 46: Novos resultados para FFA com auxílio de coprocessador operando em modo *pipeline*

5.5 Considerações Finais

Os resultados indicam que o uso da abordagem *co-design* com emprego do FPGA e uma técnica de busca inteligente é uma maneira tangível de efetuar detecção e rastreamento de alvos em imagens. Dependendo da técnica escolhida e das condições de configuração da mesma, é possível aplicar o sistema ao problema em tempo real para vídeos de 30 quadros por segundo. Para todas as técnicas implementadas, o melhor resultado obtido em tempo de processamento ocorreu com uso de coprocessador operando em modo *pipeline*, o que justifica o uso desta abordagem na solução do problema.

As técnicas CS e ABC atingiram as metas de tempo médio de processamento e taxa de acerto para todas as imagens, quando se fez uso do coprocessador. CS apresentou melhor desempenho com relação ao tempo médio de processamento, enquanto ABC obteve taxas de acerto médias mais altas. A técnica FFA proporcionou tempos abaixo de 33 ms

para sete imagens, e com taxas de acerto médias acima de 80%. Também necessitou de poucas iterações (menos que 3, na maioria das vezes).

A técnica EHO apresentou, para a configuração adotada, taxas de acerto média próximas a 60%, e mesmo assim os tempos foram superiores a 33 ms. Para a imagem $Rcar - I_8$, a taxa de acerto foi inferior à meta mínima, ao se considerar o desvio padrão.

A técnica BFOA chegou a obter taxas de acerto médias de mais de 90% para $IR3 - I_6$ e $Truck - I_7$, entretanto seu desempenho ficou acima de 33 ms para todas as imagens, inclusive com uma ($IR2 - I_5$) proporcionando taxa de acerto inferior a 60%.

A técnica FWA apresentou tempos superiores a 33 ms, mesmo com a taxa de acerto média para a imagem $IR2 - I_5$ inferiores a 60%. Ao mesmo tempo, apresentou taxa de acerto média acima de 95% para $IR3 - I_6$.

A imagem $IR2 - I_5$ foi a que mais demandou tempo de processamento e iterações para encontrar o alvo. Isto ocorreu por causa da complexidade da imagem perante a função objetivo adotada, e à presença de ótimos locais para onde os elementos dos enxames convergiam.

De um modo geral, o PSO foi mais rápido e mais preciso que as técnicas de busca otimizada investigadas. Dentre as seis técnicas implementadas, aquela que obteve os melhores resultados foi a baseada no pássaro cuco. Seu processamento foi mais rápido do que o PSO; entretanto, sua taxa de acerto média foi pior.

A nova configuração de parâmetros para as técnicas mostrou que, para atingir taxa de acerto de pelo menos 90%, o tempo de processamento fica inadequado para o emprego em vídeos de 30 quadros por segundo. Porém, estas configurações podem ser empregadas em outras situações onde a taxa de acerto é crítica, como reconhecimento de padrões em imagens médicas ou aplicações em vídeos com menos quadros por segundo. O próximo capítulo apresenta a conclusão sobre o projeto implementado nesta dissertação.

Capítulo 6

CONCLUSÕES E TRABALHOS FUTUROS

ESTE capítulo apresenta as principais conclusões sobre o sistema dedicado para rastreamento empregando diferentes técnicas de inteligência de enxame, proposto nesta dissertação e descritas na Seção 6.1. Por fim, a Seção 6.2 elenca possíveis melhorias no sistema, com vista em gerar resultados mais adequados ao trabalho em tempo real em vídeos de 30 quadros por segundo, mantendo-se uma taxa de acerto alta para garantir a qualidade da solução, e apresenta possibilidades de trabalhos futuros.

6.1 Conclusões

Esta dissertação abordou o problema de detecção de objetos-alvo em imagens. A necessidade do projeto exigia que o sistema fosse possível de ser embutido em um equipamento, e demostrasse desempenho adequado ao trabalho em tempo real em vídeos de 30 quadros por segundo. A taxa de acerto deveria ser alta, para garantir a qualidade da resposta, evitando-se falsos positivos, de modo a permitir a correta tomada de decisão por parte dos operadores do sistema.

A identificação do alvo na imagem baseia-se em alguma característica singular dele e na metodologia de identificação empregada. A técnica TM foi adotada para analisar recortes da imagem e compará-los com o alvo. O cálculo do coeficiente de CCN, também conhecido como PCC, foi empregado para calcular a similaridade entre o alvo e os recortes, devido as suas vantagens em caso de variação de brilho na imagem. O Capítulo 1 apresentou o conceito de rastreamento e trouxe análise de técnicas de identificação de objetos, com ênfase em TM. O Capítulo 2 exibiu uma análise bibliográfica sobre rastreamento e

identificação de objetos em imagens, especialmente as aplicações em implementação em *hardware*.

O sistema empregado consistiu de uma abordagem *co-design* na qual foi utilizada uma placa SVDK com módulo Xilinx PicoZed 7Z015 System-On-Module e *chip* Zynq XC7Z015. Neste equipamento, foram empregadas separadamente seis técnicas de busca otimizada implementadas em *software*, baseadas em inteligência de enxame, executadas pelo processador do XC7Z015. Em conjunto, foi utilizado um coprocessador especificamente desenvolvido em (TAVARES, 2016) para calcular a etapa computacional mais custosa, o PCC. As técnicas de busca otimizada foram apresentadas no Capítulo 3.

A execução do sistema foi realizada com três diferentes configurações do equipamento desenvolvidas em (TAVARES, 2016). A primeira, com o processador de uso geral Zynq XC7Z015 executando a busca e o cálculo do PCC em *software* e trabalhando em conjunto com a técnica de busca implementada. A segunda, com o processador Zynq executando a busca em conjunto com um coprocessador dedicado para o cálculo do PCC, operando em modo serial. A terceira, com o processador Zynq executando a busca e o coprocessador dedicado operando em modo *pipeline* para o cálculo do PCC. O Capítulo 4 apresentou um estudo sobre o *hardware* empregado.

Cada técnica recebeu avaliação separadamente quanto a tempo médio de processamento, número médio de iterações e taxa média de acerto. Os resultados foram comparados com aqueles obtidos com o uso de PSO em (TAVARES, 2016). Os parâmetros de cada técnica foram configurados empiricamente para atingir as metas de desempenho especificadas na Seção 5.1 do Capítulo 5. Em seguida, os parâmetros de cada técnica foram reconfigurados empiricamente para cumprir a meta de taxa média de acerto de 90%.

Na comparação entre os modos de operação do coprocessador, o modo *pipeline* apresentou vantagem no desempenho quanto à taxa de processamento em relação ao modo serial para as técnicas de busca implementadas. O modo serial indicou vantagem em relação ao sistema sem coprocessador. Contudo, o coprocessador utilizado possui a limitação de 50 indivíduos para a configuração em modo *pipeline*, o que demandaria alteração no projeto de (TAVARES, 2016). Cabe ressaltar que nenhuma das técnicas cumpriu a meta de tempo máximo de processamento de 33 ms sem uso de coprocessador.

As técnicas baseadas no pássaro cuco e em abelhas cumpriram a meta de tempo médio de processamento de 33 ms para todas as imagens de referência com sistema em-

pregando o coprocessador, tanto em modo serial quanto em modo *pipeline*. A técnica CS apresentou tempos menores do que o PSO. Como as taxas médias de acerto para estas técnicas estiveram adequadas para a meta de 60%, as técnicas CS e ABC mostraram-se, portanto, adequadas ao processamento de vídeos de 30 quadros por segundo.

A técnica FFA apresentou tempos abaixo de 33 ms para sete imagens e apresentou taxas de acerto médias acima de 80%. As técnicas BFOA, EHO e FWA deixaram de cumprir a meta mínima de 60% de taxa média de acerto para uma das imagens, mesmo com seus tempos acima de 33 ms.

De um modo geral, o PSO foi mais rápido e mais preciso que as outras técnicas de busca otimizada analisadas. Dentre as seis técnicas implementadas, aquela que obteve os melhores resultados foi a baseada no pássaro cuco. Seu processamento foi mais rápido do que o PSO; entretanto, sua taxa de acerto média foi pior.

Com as novas configurações para garantir o cumprimento da meta de taxa média de acerto de 90%, todas as técnicas permitiram que sistema atingisse a meta, porém com tempos de processamento superiores ao limite de 33 ms. Portanto, estas configurações são inadequadas ao processamento em tempo real de vídeos de 30 quadros por segundo, porém ainda podem ser aplicadas em situações em que o tempo não seja crítico.

Os resultados indicam que o uso da abordagem *co-design* com emprego do FPGA e uma técnica de busca inteligente é uma maneira tangível de efetuar detecção e rastreamento de alvos em imagens. Dependendo da técnica escolhida e das condições de configuração da mesma, é possível aplicar o sistema ao problema em tempo real para vídeos de 30 quadros por segundo. Para todas as técnicas implementadas, o melhor resultado obtido em tempo de processamento ocorreu com uso de coprocessador operando em modo *pipeline*, o que justifica o uso desta abordagem na solução do problema.

6.2 Trabalhos Futuros

A fim de contribuir com a melhoria do método proposto, sugere-se a alteração do projeto do coprocessador para que possa empregar mais elementos no enxame. Isso permitiria a configuração mais adequada das técnicas de busca, de forma a obter tempo de processamento menor e taxa de acerto maior.

Da mesma forma, sugere-se o emprego de versões adaptadas destes algoritmos para a solução do problema de localização e rastreamento de objetos-alvo em imagens.

No presente trabalho, buscou-se usar as versões canônicas das técnicas (exceto para ABC e EHO, que receberam pequenas modificações). Poderia-se empregar versões híbridas, inclusive que permitissem estimar a posição do alvo nos quadros futuros de um vídeo.

Como trabalho futuro, sugere-se aplicar alterações no *template*, como rotações e variações de escala, seguido do reprojeto do coprocessador para que se empregue mais de 50 elementos no enxame, além de poder receber diversos quadros ao invés de apenas um. Dessa forma, é possível melhorar o desempenho da busca, ao mesmo tempo em que se conseguiria atualizar o *template* a cada quadro, facilitando o rastreamento. Sugere-se também uma abordagem onde um coprocessador efetuaria o rastreamento no vídeo, enquanto outro calcularia a trajetória futura do alvo, ou até mesmo o guiamento da câmera que capta as imagens do ambiente de busca.

REFERÊNCIAS

ABDEL-KADER, R. F.; ATTA, R.; EL-SHAKHABE, S. An efficient eye detection and tracking system based on particle swarm optimization and adaptive block-matching search algorithm. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 31, p. 90–100, 2014.

AHUJA, K.; TULI, P. Object recognition by template matching using correlations and phase angle method. *International Journal of Advanced Research in Computer and Communication Engineering*, v. 2, n. 3, p. 1368–1373, 2013.

ALBA, E.; NAKIB, A.; SIARRY, P. *Metaheuristics for dynamic optimization*. Berlin, Germany: Springer, 2013. ISBN 9783642306648.

BAE, C. et al. A novel real time video tracking framework using adaptive discrete swarm optimization. *Expert Systems with Applications*, Elsevier, v. 64, p. 385–399, 2016.

BISWAS, A. et al. A synergy of differential evolution and bacterial foraging optimization for global optimization. *Neural Network World*, Institute of Computer Science, v. 17, n. 6, p. 607, 2007.

BOUDAHER, E.; HOORFAR, A. Fireworks algorithm: A new swarm intelligence technique for electromagnetic optimization. In: IEEE. *Antennas and Propagation (APSURSI), 2016 IEEE International Symposium on*. Fajardo, Puerto Rico, 2016. p. 575–576.

BRANKE, J. Memory enhanced evolutionary algorithms for changing optimization problems. In: IEEE. *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*. Washington DC, USA, 1999. v. 3, p. 1875–1882.

BROWN, L. G. A survey of image registration techniques. *ACM computing surveys (CSUR)*, ACM, v. 24, n. 4, p. 325–376, 1992.

CARDOSO, A. de V.; NEDJAH, N.; MOURELLE, L. de M. Hardware/software co-design for template matching using cuckoo search optimization. In: SPRINGER. *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Montreal, Canada, 2018. p. 16–21.

CARDOSO, A. de V. et al. Co-design system for template matching using dedicated co-processor and modified elephant herding optimization. In: IEEE. *Circuits & Systems (LASCAS), 2018 IEEE 9th Latin American Symposium on*. Puerto Vallarta, Mexico, 2018.

CARDOSO, A. de V. et al. Co-design system for template matching using dedicated co-processor and cuckoo search. *International Journal of Swarm Intelligence Research (IJSIR)*, IGI Global, v. 9, n. 1, p. 58–74, 2018.

CHEN, J. et al. An improved fireworks algorithm with landscape information for balancing exploration and exploitation. In: IEEE. *Evolutionary Computation (CEC), 2015 IEEE Congress on*. Sendai, Japan, 2015. p. 1272–1279.

CIVICIOGLU, P.; BESDOK, E. A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review*, Springer, v. 39, n. 4, p. 315–346, 2013.

COLLINS, R.; ZHOU, X.; TEH, S. K. An open source tracking testbed and evaluation web site. In: *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2005)*. Breckenridge, USA: Citeseer, 2005. v. 2, p. 35.

COMANICIU, D.; RAMESH, V.; MEER, P. Kernel-based object tracking. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 25, n. 5, p. 564–577, 2003.

DATTA, T. et al. Improved adaptive bacteria foraging algorithm in optimization of antenna array for faster convergence. *Progress In Electromagnetics Research C*, EMW Publishing, v. 1, p. 143–157, 2008.

ENGELBRECHT, A. P. *Fundamentals of computational swarm intelligence*. Chichester, England: John Wiley & Sons, 2006.

ERTAŞ, G. et al. Breast mr segmentation and lesion detection with cellular neural networks and 3d template matching. *Computers in biology and medicine*, Elsevier, v. 38, n. 1, p. 116–126, 2008.

FISTER, I.; YANG, X.-S.; BREST, J. A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation*, Elsevier, v. 13, p. 34–46, 2013.

FUNDED, E. *CAVIAR project/IST 2001 37540*. 2014.

GAO, M.-L. et al. Object tracking using firefly algorithm. *IET Computer Vision*, IET, v. 7, n. 4, p. 227–237, 2013.

GAO, M.-L. et al. Firefly algorithm (fa) based particle filter method for visual tracking. *Optik-International Journal for Light and Electron Optics*, Elsevier, v. 126, n. 18, p. 1705–1711, 2015.

GAO, M.-L. et al. A novel visual tracking method using bat algorithm. *Neurocomputing*, Elsevier, v. 177, p. 612–619, 2016.

GOLDBERG, D. E.; DEB, K. A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms*, v. 1, p. 69–93, 1991.

GOLLAPUDI, S. V. et al. Intelligent bacterial foraging optimization technique to calculate resonant frequency of rma. *International Journal of Microwave and Optical Technology*, v. 4, n. 2, 2009.

GOLLAPUDI, S. V. et al. Bacterial foraging optimization technique to calculate resonant frequency of rectangular microstrip antenna. *International Journal of RF and Microwave Computer-Aided Engineering*, Wiley Online Library, v. 18, n. 4, p. 383–388, 2008.

GUSTAFSSON, F. et al. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on signal processing*, IEEE, v. 50, n. 2, p. 425–437, 2002.

ILUNGA-MBUYAMBA, E. et al. Active contours driven by cuckoo search strategy for brain tumour images segmentation. *Expert Systems with Applications*, Elsevier, v. 56, p. 59–68, 2016.

JARRAH, A.; JAMALI, M. M.; HOSSEINI, S. S. S. Optimized fpga based implementation of particle filter for tracking applications. In: IEEE. *NAECON 2014-IEEE National Aerospace and Electronics Conference*. Dayton, US, 2014. p. 233–236.

JENKINS, M. D. et al. Extended fast compressive tracking with weighted multi-frame template matching for fast motion tracking. *Pattern Recognition Letters*, Elsevier, v. 69, p. 82–87, 2016.

JIHANG, C. et al. Study on image target interpretation decision-making based on analytic hierarchy process and clustering analysis. In: IEEE. *Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on*. Changchun, China, 2012. p. 2045–2049.

JR, I. F. et al. A brief review of nature-inspired algorithms for optimization. *Elektrotehniski Vestnik*, v. 80, n. 3, p. 116–122, 2013.

KAR, A. K. Bio inspired computing–a review of algorithms and scope of applications. *Expert Systems with Applications*, Elsevier, v. 59, p. 20–32, 2016.

KARABOGA, D. *An idea based on honey bee swarm for numerical optimization*. Kayseri, Turkey, 2005.

KARABOGA, D.; BASTURK, B. Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In: SPRINGER. *International Fuzzy Systems Association World Congress*. Cancun, Mexico, 2007. p. 789–798.

KARTHIKEYAN, S. et al. A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems. *International Journal of Bio-Inspired Computation*, Inderscience Publishers (IEL), v. 7, n. 6, p. 386–401, 2015.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: IEEE. *Proceedings of IEEE International Conference on Neural Networks 95*. Perth, Australia, 1995. v. 1000.

KORANI, W. M.; DORRAH, H. T.; EMARA, H. M. Bacterial foraging oriented by particle swarm optimization strategy for pid tuning. In: IEEE. *Computational Intelligence in Robotics and Automation (CIRA), 2009 IEEE International Symposium on*. Daejeon, Korea, 2009. p. 445–450.

KULKARNI, A.; VARGANTWAR, M.; VIRULKAR, S. Video based tracking and optimization using mean-shift, kalman filter and swarm intelligence. In: IEEE. *Computing Communication Control and Automation (ICCUBEA), 2015 International Conference on*. Pune, India, 2015. p. 629–633.

LIU, H. et al. Thermal feature extraction of servers in a datacenter using thermal image registration. *Infrared Physics & Technology*, Elsevier, 2017.

LIU, W. et al. Rfid network scheduling using an adaptive bacterial foraging algorithm. *Journal of Computational Information Systems*, v. 7, n. 4, p. 1238–1245, 2011.

LIU, W.; CHEN, H.; MA, L. Moving object detection and tracking based on zynq fpga and arm soc. In: IET. *IET International Radar Conference 2015*. Hangzhou, China, 2015. p. 1–4.

LJOUAD, T.; AMINE, A.; RZIZA, M. A hybrid mobile object tracker based on the modified cuckoo search algorithm and the kalman filter. *Pattern Recognition*, Elsevier, v. 47, n. 11, p. 3597–3613, 2014.

MA, H.; SIMON, D. Blended biogeography-based optimization for constrained optimization. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 24, n. 3, p. 517–525, 2011.

MAJI, D. et al. Hierarchical clustering for segmenting fused image using discrete cosine transform with artificial bee colony optimization. In: IEEE. *Computational Intelligence & Communication Technology (CICT), 2016 Second International Conference on*. Ghaziabad, India, 2016. p. 54–59.

MERAD, D. et al. Tracking multiple persons under partial and global occlusions: Application to customers' behavior analysis. *Pattern Recognition Letters*, Elsevier, 2016.

MISRA, P. R.; SI, T. Image segmentation using clustering with fireworks algorithm. In: IEEE. *Intelligent Systems and Control (ISCO), 2017 11th International Conference on*. Coimbatore, India, 2017. p. 97–102.

MORAIS, R. G. *Proposta, Implementação e Análise de um Algoritmo de Otimização Global Inspirado no Comportamento dos Pássaros de Hitchcock*. Dissertação (Mestrado) — UERJ, Rio de Janeiro, Brazil, 2017.

NEDJAH, N.; MOURELLE, L. d. M. *Co-design for system acceleration: a quantitative approach*. Dordrecht, The Netherlands: Springer Science & Business Media, 2007.

PASSINO, K. M. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE control systems*, IEEE, v. 22, n. 3, p. 52–67, 2002.

PEKDEMIR, H.; TOPCUOGLU, H. R. Enhancing fireworks algorithms for dynamic optimization problems. In: IEEE. *Evolutionary Computation (CEC), 2016 IEEE Congress on*. Vancouver, Canada, 2016. p. 4045–4052.

PERVEEN, N.; KUMAR, D.; BHARDWAJ, I. An overview on template matching methodologies and its applications. *IJRCCT*, v. 2, n. 10, p. 988–995, 2013.

REYNOLDS, A. M.; FRYE, M. A. Free-flight odor tracking in drosophila is consistent with an optimal intermittent scale-free search. *PloS one*, Public Library of Science, v. 2, n. 4, p. e354, 2007.

RYAN, M.; HANAFIAH, N. An examination of character recognition on id card using template matching approach. *Procedia Computer Science*, Elsevier, v. 59, p. 520–529, 2015.

SALMOND, D. Tracking and guidance with intermittent obscuration and association uncertainty. In: IEEE. *Information Fusion (FUSION), 2013 16th International Conference on*. Istanbul, Turkey, 2013. p. 691–698.

SARDARI, F.; MOGHADDAM, M. E. A hybrid occlusion free object tracking method using particle filter and modified galaxy based search meta-heuristic algorithm. *Applied Soft Computing*, Elsevier, v. 50, p. 280–299, 2017.

SHA, F. et al. A categorized particle swarm optimization for object tracking. In: IEEE. *2015 IEEE Congress on Evolutionary Computation (CEC)*. Sendai, Japan, 2015. p. 2737–2744.

SHARMA, V.; PATTNAIK, S.; GARG, T. A review of bacterial foraging optimization and its applications. In: IEEE. *National Conference on Future Aspects of Artificial intelligence in Industrial Automation, NCFAAIIA*. New Delhi, India, 2012. p. 9–12.

SHI, J. et al. Multiple cells tracking by firework algorithm. In: IEEE. *Control, Automation and Information Sciences (ICCAIS), 2015 International Conference on*. Changshu, China, 2015. p. 508–511.

SHI, W. et al. Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey. *Integration, the VLSI Journal*, Elsevier, v. 59, p. 148–156, 2017.

STORN, R.; PRICE, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Springer, v. 11, n. 4, p. 341–359, 1997.

STRENS, M. J.; GREGORY, I. N. Tracking in cluttered images. *Image and Vision Computing*, Elsevier, v. 21, n. 10, p. 891–911, 2003.

STUBBERUD, S. C.; KRAMER, K. A.; STUBBERUD, A. R. Navigation using image tracking. In: IEEE. *Science of Electrical Engineering (ICSEE), IEEE International Conference on the*. Eilat, Israel, 2016. p. 1–5.

TAN, Y.; ZHU, Y. Fireworks algorithm for optimization. In: SPRINGER. *International Conference in Swarm Intelligence*. Beijing, China, 2010. p. 355–364.

TAVARES, Y. M. *Sistema Integrado de Hardware Software para Rastreamento de Alvos*. Dissertação (Mestrado) — UERJ, Rio de Janeiro, Brazil, 2016.

TAVARES, Y. M.; NEDJAH, N.; MOURELLE, L. de M. Utilização de otimização por enxame de partículas e algoritmos genéticos em rastreamento de padrões. 2015.

TAVARES, Y. M.; NEDJAH, N.; MOURELLE, L. de M. Embedded implementation of template matching using correlation and particle swarm optimization. In: SPRINGER. *International Conference on Computational Science and Its Applications*. Beijing, China, 2016. p. 530–539.

WALIA, G. S.; KAPOOR, R. Intelligent video target tracking using an evolutionary particle filter based upon improved cuckoo search. *Expert Systems with Applications*, Elsevier, v. 41, n. 14, p. 6315–6326, 2014.

WALTON, S. et al. Modified cuckoo search: a new gradient free optimisation algorithm. *Chaos, Solitons & Fractals*, Elsevier, v. 44, n. 9, p. 710–718, 2011.

WANG, G.-G.; DEB, S.; COELHO, L. d. S. Elephant herding optimization. In: IEEE. *Computational and Business Intelligence (ISCBI), 2015 3rd International Symposium on*. Bali, Indonesia, 2015. p. 1–5.

WANG, H. et al. Firefly algorithm with neighborhood attraction. *Information Sciences*, Elsevier, v. 382, p. 374–387, 2017.

WEBER, J.; LEFÈVRE, S. Spatial and spectral morphological template matching. *Image and Vision Computing*, Elsevier, v. 30, n. 12, p. 934–945, 2012.

YANG, X.-S. Firefly algorithms for multimodal optimization. In: SPRINGER. *International Symposium on Stochastic Algorithms*. Sapporo, Japan, 2009. p. 169–178.

YANG, X.-S. *Nature-inspired metaheuristic algorithms*. United Kingdom: Luniver press, 2010. ISBN 9781905986286.

YANG, X.-S.; DEB, S. Cuckoo search via lévy flights. In: IEEE. *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. Coimbatore, India, 2009. p. 210–214.

YILMAZ, A.; JAVED, O.; SHAH, M. Object tracking: A survey. *Acm computing surveys (CSUR)*, Acm, v. 38, n. 4, p. 13, 2006.

YOO, J. et al. Scale-invariant template matching using histogram of dominant gradients. *Pattern Recognition*, Elsevier, v. 47, n. 9, p. 3006–3018, 2014.

YU, C.; TAN, Y. Fireworks algorithm with covariance mutation. In: IEEE. *Evolutionary Computation (CEC), 2015 IEEE Congress on*. Sendai, Japan, 2015. p. 1250–1256.

YUAN, C. et al. Unmanned aerial vehicle based forest fire monitoring and detection using image processing technique. In: IEEE. *Guidance, Navigation and Control Conference (CGNCC), 2016 IEEE Chinese*. Nanjing, China, 2016. p. 1870–1875.

ZHANG, B. et al. Fireworks algorithm with enhanced fireworks interaction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, IEEE, 2015.

ZHANG, Y.; WU, L.; WANG, S. Bacterial foraging optimization based neural network for short-term load forecasting. *Journal of Computational Information Systems*, v. 6, n. 7, p. 2099–2105, 2010.

ZITOVA, B.; FLUSSER, J. Image registration methods: a survey. *Image and vision computing*, Elsevier, v. 21, n. 11, p. 977–1000, 2003.

APÊNDICE A – Resultados Numéricos

Este apêndice apresenta os valores de tempo de processamento, número de iterações e taxa de acerto obtidas nas execuções apresentadas nas Seções 5.3 e 5.4.

A.1 Resultados da Operação sem Coprocessador

As Tabelas 2 a 19 exibem os resultados para as configurações das técnicas feitas para as metas de tempo de processamento de até 33 ms e taxa de acerto mínima de 60%.

Tabela 2: Comparação de tempo de processamento (ms) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	145,52	138,80	396,88	336,40	168,11	318,70	218,53
Pickup – I_2	134,64	153,50	379,36	308,49	166,81	426,62	212,68
Sedan – I_3	103,34	127,81	335,04	296,63	152,70	369,44	182,76
IR1 – I_4	117,58	122,88	280,69	280,87	136,29	275,03	176,04
IR2 – I_5	127,21	153,88	408,45	401,53	185,98	513,44	238,64
IR3 – I_6	89,77	128,43	303,91	233,91	144,76	385,21	182,83
Truck – I_7	109,97	133,27	355,15	258,82	148,69	399,21	195,63
Rcar – I_8	119,87	142,76	396,76	307,17	169,82	285,39	196,47

Tabela 3: Comparação de desvio padrão do tempo de processamento (ms) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	70,21	50,22	200,20	156,75	91,52	222,09
Pickup – I_2	65,63	58,43	201,54	148,57	90,37	259,27
Sedan – I_3	56,14	60,64	217,81	165,18	90,60	279,73
IR1 – I_4	68,04	55,88	186,36	167,25	61,85	242,62
IR2 – I_5	57,67	51,10	197,18	144,94	89,46	218,35
IR3 – I_6	41,48	51,11	285,46	124,51	70,59	194,42
Truck – I_7	54,14	59,36	209,25	137,07	72,80	271,01
Rcar – I_8	63,07	59,85	217,93	160,49	88,92	198,60

Tabela 4: Comparação do número de iterações sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	6,48	6,23	6,95	7,03	3,15	4,68	5,64
Pickup – I_2	6,02	6,86	6,53	6,44	2,97	5,75	5,48
Sedan – I_3	4,71	5,48	5,49	6,07	2,62	5,52	4,61
IR1 – I_4	5,86	5,90	4,50	5,79	2,32	3,77	4,44
IR2 – I_5	6,43	7,37	6,95	8,39	2,95	3,76	6,24
IR3 – I_6	4,53	6,31	5,15	4,71	2,32	4,72	4,65
Truck – I_7	4,98	5,88	5,70	5,15	2,58	5,68	4,97
Rcar – I_8	5,43	6,39	6,78	6,42	2,54	3,94	5,00

Tabela 5: Comparação de desvio padrão do número de iterações sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	2,84	2,64	3,73	3,52	2,96	3,44
Pickup – I_2	2,65	2,56	3,69	3,34	2,71	3,54
Sedan – I_3	2,13	2,34	3,59	3,50	2,52	4,13
IR1 – I_4	3,07	2,67	2,95	3,57	1,67	3,11
IR2 – I_5	2,78	2,69	3,53	3,19	1,81	3,49
IR3 – I_6	1,98	2,63	3,28	2,64	1,78	2,27
IR3 – I_6	1,82	2,63	3,28	2,64	2,11	2,27
Truck – I_7	2,05	2,47	3,57	3,78	2,11	4,97
Rcar – I_8	2,50	2,65	3,93	5,31	1,70	2,86

Tabela 6: Comparação da taxa de acerto (%) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	69,94	84,94	62,42	62,66	85,88	81,88	95,4
Pickup – I_2	80,36	88,00	67,50	73,10	88,90	78,70	95,9
Sedan – I_3	93,60	95,62	79,68	75,40	91,74	72,46	99,7
IR1 – I_4	69,36	88,98	91,96	76,12	87,36	91,54	99,6
IR2 – I_5	70,20	79,40	66,12	48,52	79,48	57,34	91,5
IR3 – I_6	95,52	87,92	84,32	91,94	93,62	97,68	99,8
Truck – I_7	97,12	94,42	81,84	93,12	89,96	68,88	100
Rcar – I_8	86,34	87,94	59,10	67,22	84,98	89,98	99,1

Tabela 7: Comparação de desvio padrão da taxa de acerto (%) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	10,70	3,22	0,50	4,78	3,28	3,74
Pickup – I_2	3,65	2,84	4,49	4,20	2,99	4,27
Sedan – I_3	2,32	2,21	3,84	4,03	2,90	1,94
IR1 – I_4	3,64	2,99	2,57	3,54	2,43	0,37
IR2 – I_5	4,79	3,64	3,52	4,84	4,10	5,03
IR3 – I_6	2,08	3,08	4,00	2,87	2,31	1,40
Truck – I_7	1,53	1,90	3,29	2,66	2,79	1,04
Rcar – I_8	3,40	3,74	4,47	5,04	3,19	2,96

A.2 Resultados da Operação com Auxílio de Coprocessador Serial

Tabela 8: Comparação do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	21,95	26,74	72,25	61,75	30,84	57,77	39,14
Pickup – I_2	19,52	28,94	69,85	57,37	29,32	79,19	37,84
Sedan – I_3	13,57	24,03	61,96	55,26	27,40	67,94	31,88
IR1 – I_4	17,96	22,45	53,29	52,10	24,76	50,08	32,09
IR2 – I_5	19,14	31,07	76,12	73,33	34,24	95,05	43,48
IR3 – I_6	11,28	23,94	56,23	49,19	25,63	56,05	33,52
Truck – I_7	14,08	24,37	66,85	45,55	26,75	72,29	34,20
Rcar – I_8	15,95	25,13	71,06	57,07	31,03	52,32	35,16

Tabela 9: Comparação de desvio padrão do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	14,55	10,48	35,05	26,95	17,17	41,32
Pickup – I_2	13,08	11,33	34,22	25,91	14,52	47,26
Sedan – I_3	10,56	11,45	38,04	28,99	14,49	51,29
IR1 – I_4	13,97	10,93	32,65	29,44	11,63	46,49
IR2 – I_5	12,27	11,20	34,65	25,85	17,49	40,81
IR3 – I_6	7,91	9,89	31,88	22,46	11,89	32,63
Truck – I_7	9,29	10,94	36,70	24,24	13,10	49,79
Rcar – I_8	12,58	10,01	37,60	28,05	14,47	36,17

Tabela 10: Comparação do número de iterações com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	7,11	7,19	6,90	7,25	3,15	4,59	5,66
Pickup – I_2	6,47	7,43	6,59	6,75	2,62	5,89	5,47
Sedan – I_3	4,96	5,99	6,45	6,35	2,42	5,52	4,51
IR1 – I_4	6,59	6,05	4,50	6,01	2,43	3,80	4,53
IR2 – I_5	7,09	8,01	7,04	8,54	3,15	7,71	6,34
IR3 – I_6	4,74	6,83	5,10	5,21	2,22	3,79	4,80
Truck – I_7	5,12	6,33	5,82	5,58	2,49	5,65	4,88
Rcar – I_8	4,85	7,48	6,72	6,69	2,68	3,96	5,04

Tabela 11: Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	3,35	2,88	3,78	3,34	2,93	3,47
Pickup – I_2	3,24	2,71	3,68	3,23	1,87	3,54
Sedan – I_3	2,43	2,62	3,57	3,39	1,84	4,13
IR1 – I_4	3,84	2,79	2,94	3,46	1,92	3,23
IR2 – I_5	3,56	2,80	3,52	3,11	2,16	3,50
IR3 – I_6	2,18	2,93	3,28	2,64	1,55	2,01
Truck – I_7	2,14	2,73	3,44	2,74	1,87	3,98
Rcar – I_8	3,00	3,08	3,92	3,44	2,00	2,85

Tabela 12: Comparação da taxa de acerto (%) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	72,58	75,46	61,82	63,32	86,20	98,42	94,8
Pickup – I_2	82,74	79,72	67,28	71,98	88,50	78,06	96,2
Sedan – I_3	93,72	90,84	79,82	75,24	90,30	70,90	99,2
IR1 – I_4	69,24	87,90	91,58	76,28	88,30	91,56	99,7
IR2 – I_5	70,98	69,12	65,40	47,02	80,26	57,66	91,6
IR3 – I_6	94,66	77,80	85,12	91,00	93,72	98,42	98,3
Truck – I_7	98,36	86,66	81,08	92,40	89,10	69,20	99,7
Rcar – I_8	87,36	66,90	59,86	67,14	85,68	90,12	98,9

Tabela 13: Comparação de desvio padrão da taxa de acerto (%) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	4,43	4,61	4,54	4,75	3,53	1,10
Pickup – I_2	3,34	4,09	3,75	4,23	2,97	3,93
Sedan – I_3	2,61	2,70	4,15	4,26	3,03	4,99
IR1 – I_4	3,85	2,64	3,03	4,32	3,06	3,01
IR2 – I_5	5,23	4,73	4,39	5,75	4,20	5,59
IR3 – I_6	2,41	3,65	3,17	3,19	2,19	1,10
Truck – I_7	1,20	3,87	4,34	2,49	2,53	4,33
Rcar – I_8	3,21	4,41	5,66	1,88	4,33	2,71

A.3 Resultados da Operação com Auxílio de Coprocessador *Pipeline*

Tabela 14: Comparação de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	15,86	27,98	72,63	59,63	29,82	59,33	17,71
Pickup – I_2	13,85	28,34	71,09	55,54	29,70	78,80	17,52
Sedan – I_3	14,44	23,01	62,28	53,25	26,84	68,90	14,46
IR1 – I_4	13,21	23,41	54,40	50,87	24,76	50,75	14,26
IR2 – I_5	14,25	28,76	75,85	71,86	33,81	95,21	18,98
IR3 – I_6	12,85	23,52	56,76	43,16	25,89	70,69	15,02
Truck – I_7	14,73	22,52	68,21	47,23	26,94	71,75	15,95
Rcar – I_8	14,53	26,01	70,68	54,65	31,40	52,97	15,86

Tabela 15: Comparação de desvio padrão de tempo de processamento (ms) com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	10,66	10,72	34,97	28,15	14,91	40,66
Pickup – I_2	8,18	11,08	34,96	26,64	14,80	46,73
Sedan – I_3	10,33	10,48	37,38	29,80	14,34	51,18
IR1 – I_4	7,61	11,74	33,09	29,67	11,38	44,76
IR2 – I_5	7,16	9,97	34,18	26,84	17,36	40,74
IR3 – I_6	7,42	10,30	32,32	22,31	12,40	35,99
Truck – I_7	9,80	9,82	36,64	24,12	13,16	49,02
Rcar – I_8	9,66	11,15	37,66	28,95	17,69	35,93

Tabela 16: Comparação de número de iterações com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	3,98	7,02	6,84	6,95	2,68	3,98	5,78
Pickup – I_2	5,85	7,12	6,65	6,45	2.64	5,85	5,68
Sedan – I_3	5,67	5,75	5,40	6,05	2,36	5,67	4,61
IR1 – I_4	3,78	5,93	4,54	5,79	2,41	3,78	4,53
IR2 – I_5	7,78	7,52	6,93	8,29	3,07	7,78	6,21
IR3 – I_6	4,73	6,00	5,08	4,83	2,26	4,73	4,81
Truck – I_7	5,66	5,82	5,90	5,24	2,53	5,66	5,12
Rcar – I_8	3,98	6,57	6,60	6,34	2,73	3,98	5,08

Tabela 17: Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	2,84	2,56	3,73	3,51	1,98	2,84
Pickup – I_2	3,54	2,58	3,72	3,31	1,88	3,54
Sedan – I_3	4,17	2,42	3,51	3,50	1,80	4,17
IR1 – I_4	3,11	2,64	2,95	3,51	1,93	3,11
IR2 – I_5	3,50	2,69	3,53	3,24	2,12	3,50
IR3 – I_6	2,31	2,43	3,26	2,63	1,61	2,31
Truck – I_7	3,99	2,45	3,44	2,72	1,92	3,99
Rcar – I_8	2,84	2,62	3,92	3,56	2,05	2,84

Tabela 18: Comparação da taxa de acerto (%) com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA	PSO
Cars – I_1	74,46	85,94	63,58	63,74	85,96	81,68	90,5
Pickup – I_2	86,64	86,72	65,52	73,12	88,18	78,36	92,9
Sedan – I_3	94,28	94,76	81,32	75,62	91,20	68,90	97,6
IR1 – I_4	67,56	90,96	92,06	76,94	88,22	92,10	98,5
IR2 – I_5	72,46	78,32	66,70	49,62	81,64	57,02	89,4
IR3 – I_6	85,44	94,36	85,38	92,16	92,82	97,02	96,9
Truck – I_7	98,22	94,35	80,02	93,34	88,12	68,76	96,2
Rcar – I_8	78,64	88,18	61,54	68,84	84,66	90,44	97,5

Tabela 19: Comparação de desvio padrão da taxa de acerto (%) com auxílio de coprocessador operando em modo *pipeline*

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	0,44	3,72	4,28	4,43	3,07	3,12
Pickup – I_2	3,94	3,51	4,84	3,93	3,22	3,53
Sedan – I_3	2,26	2,59	3,85	4,21	2,93	5,00
IR1 – I_4	3,76	2,96	2,91	3,77	3,97	2,40
IR2 – I_5	4,91	4,04	4,17	5,10	4,02	4,59
IR3 – I_6	3,09	1,86	3,35	2,57	2,52	1,69
Truck – I_7	1,34	2,09	3,83	1,97	3,12	5,58
Rcar – I_8	3,68	2,98	4,19	4,55	3,94	2,97

A.4 Resultados para Novas Configurações para as Técnicas

As Tabelas 20 a 31 mostram os resultados para as configurações das técnicas feitas para a nova meta taxa de acerto mínima de 95%.

Tabela 20: Comparação do tempo de processamento (ms), sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	246,72	240,99	695,15	623,76	221,98	829,33
Pickup – I_2	237,42	247,87	653,42	592,87	216,19	882,05
Sedan – I_3	205,34	190,29	466,97	488,98	203,29	782,82
IR1 – I_4	184,19	203,22	410,34	441,54	190,36	825,97
IR2 – I_5	195,73	324,18	765,06	907,73	247,39	877,10
IR3 – I_6	174,42	210,43	465,89	464,95	191,92	1135,42
Truck – I_7	217,78	201,42	568,68	496,83	191,88	758,39
Rcar – I_8	229,98	210,19	611,42	542,86	224,03	1067,72

Tabela 21: Comparação de desvio padrão do tempo de processamento (ms), sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	81,06	98,87	491,16	353,89	116,03	391,62
Pickup – I_2	64,99	102,22	458,75	300,02	106,09	472,06
Sedan – I_3	69,17	90,47	410,73	334,12	101,74	530,70
IR1 – I_4	95,63	110,54	341,60	310,90	71,84	584,26
IR2 – I_5	54,12	123,06	495,02	499,59	130,68	514,22
IR3 – I_6	43,49	95,79	350,08	243,64	79,23	672,69
Truck – I_7	61,39	93,92	451,73	274,62	76,23	488,25
Rcar – I_8	62,49	90,23	496,24	294,15	115,75	541,69

Tabela 22: Comparação do número de iterações sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	3,13	4,47	3,56	2,94	1,79	3,20
Pickup – I_2	3,04	4,61	3,32	2,78	1,78	3,44
Sedan – I_3	2,93	3,61	2,46	2,37	1,65	3,33
IR1 – I_4	3,07	4,01	2,18	2,18	1,61	3,13
IR2 – I_5	3,21	5,52	3,84	4,34	2,00	3,94
IR3 – I_6	2,81	4,11	2,46	2,21	1,48	4,53
Truck – I_7	2,82	3,87	2,85	2,34	1,58	3,05
Rcar – I_8	2,94	3,92	3,16	2,54	1,68	4,23

Tabela 23: Comparação de desvio padrão do número de iterações sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	1,07	1,47	2,47	1,72	1,70	0,23
Pickup – I_2	0,70	1,47	2,27	1,37	1,68	1,87
Sedan – I_3	0,88	1,53	1,75	1,45	1,56	2,24
IR1 – I_4	1,78	1,49	1,36	1,37	1,54	1,72
IR2 – I_5	1,19	1,87	2,45	2,54	1,63	2,72
IR3 – I_6	0,52	1,36	1,54	1,01	0,89	2,74
Truck – I_7	0,56	1,36	1,96	1,14	1,03	2,16
Rcar – I_8	0,62	1,25	2,47	1,32	1,21	2,00

Tabela 24: Comparação da taxa de acerto (%) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	98,80	99,60	96,14	98,74	96,84	99,94
Pickup – I_2	99,86	99,92	97,20	98,80	96,70	98,56
Sedan – I_3	99,98	99,80	99,52	99,64	97,30	97,12
IR1 – I_4	95,78	99,20	99,74	99,76	97,34	99,98
IR2 – I_5	98,28	99,20	97,00	95,06	90,76	94,72
IR3 – I_6	99,98	100,00	99,78	99,98	98,06	99,44
Truck – I_7	100,00	99,66	99,30	100,00	97,14	95,40
Rcar – I_8	99,92	99,66	95,76	99,20	95,02	99,88

Tabela 25: Comparação de desvio padrão da taxa de acerto (%) sem uso de coprocessador

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	1,10	0,63	1,65	1,08	1,98	0,23
Pickup – I_2	0,40	0,27	1,78	0,40	1,65	1,14
Sedan – I_3	0,14	0,14	0,70	0,63	1,64	1,84
IR1 – I_4	2,08	0,34	0,52	0,47	1,27	0,14
IR2 – I_5	1,24	0,91	1,58	2,38	4,24	2,45
IR3 – I_6	0,14	0,00	0,46	0,14	1,15	0,64
Truck – I_7	0,00	0,55	0,95	0,00	1,57	2,22
Rcar – I_8	0,27	0,35	1,98	0,70	2,03	0,32

Tabela 26: Comparação do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	31,04	48,84	138,34	127,66	38,91	155,20
Pickup – I_2	29,40	41,59	131,68	119,44	38,80	165,76
Sedan – I_3	27,01	33,87	99,46	100,33	36,56	158,18
IR1 – I_4	27,92	34,79	87,29	92,05	33,78	168,00
IR2 – I_5	25,88	53,65	150,76	175,52	46,52	160,76
IR3 – I_6	22,28	36,39	97,16	95,16	34,48	220,17
Truck – I_7	28,25	35,21	114,98	99,81	34,74	142,10
Rcar – I_8	28,25	42,20	125,51	110,01	40,35	199,14

Tabela 27: Comparação de desvio padrão do tempo de processamento (ms) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	17,56	21,02	89,78	65,56	18,22	74,08
Pickup – I_2	10,19	17,75	82,03	55,33	17,99	91,18
Sedan – I_3	5,22	16,71	75,94	64,90	17,13	105,54
IR1 – I_4	23,95	18,96	62,42	60,48	12,66	120,84
IR2 – I_5	15,09	20,11	87,89	92,66	25,28	97,01
IR3 – I_6	5,22	17,03	61,69	46,49	14,81	131,49
Truck – I_7	6,91	16,31	82,35	52,33	13,52	92,76
Rcar – I_8	8,98	18,76	88,92	57,48	21,05	101,97

Tabela 28: Comparação do número de iterações com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	2,29	4.96	3,57	3,41	1,70	1,32
Pickup – I_2	2,19	4.32	3,35	3,20	1,69	1,88
Sedan – I_3	2,04	3.60	2,49	2,74	1,60	2,36
IR1 – I_4	2,45	3,85	2,16	2,55	1,73	1,97
IR2 – I_5	2,36	5,64	3,81	4,68	2,17	2,53
IR3 – I_6	2,05	3,95	2,43	2,59	1,48	2,71
Truck – I_7	2,08	3,76	2,85	2,72	1,58	2,17
Rcar – I_8	2,12	4.44	3,22	2,94	1,70	1,98

Tabela 29: Comparação de desvio padrão do número de iterações com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	1,07	1,67	2,56	1,76	1,37	1,32
Pickup – I_2	0,53	1,44	2,28	1,40	1,25	1,88
Sedan – I_3	0,21	1,17	1,76	1,54	1,21	2,36
IR1 – I_4	1,78	1,42	1,38	1,48	1,53	1,97
IR2 – I_5	1,33	1,97	2,42	2,55	1,80	2,53
IR3 – I_6	0,25	1,34	1,46	1,07	0,98	2,71
Truck – I_7	0,30	1,24	1,95	1,23	1,11	2,17
Rcar – I_8	0,46	1,52	2,53	1,43	1,35	1,98

Tabela 30: Comparação da taxa de acerto média (%) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	98,76	99,46	95,84	98,58	95,16	99,96
Pickup – I_2	99,90	99,80	97,50	99.78	96,42	98,92
Sedan – I_3	100,0	100,0	99,56	99,72	96,74	96,94
IR1 – I_4	95,94	99,98	99,98	99,60	93,34	99,96
IR2 – I_5	97,94	97,42	97,14	95,16	89,94	96,30
IR3 – I_6	100,00	99,92	99,84	100,00	98,24	98,94
Truck – I_7	100,00	99,94	99,56	100,00	97,64	95,40
Rcar – I_8	99,90	99,74	95,04	99,32	95,52	99,94

Tabela 31: Comparação de desvio padrão da taxa de acerto média (%) com auxílio de coprocessador operando em modo serial

	CS	ABC	EHO	BFOA	FFA	FWA
Cars – I_1	1,13	0,67	1,95	1,38	2,14	0,19
Pickup – I_2	0,30	0,45	1,70	0,46	1,72	0,94
Sedan – I_3	0,00	0,00	0,70	0,57	1,44	1,63
IR1 – I_4	1,98	0,14	0,14	0,60	2,70	0,19
IR2 – I_5	1,40	1,76	2,05	2,35	3,29	2,42
IR3 – I_6	0,00	0,27	0,37	0,00	1,30	1,05
Truck – I_7	0,00	0,23	0,64	0,00	1,71	2,42
Rcar – I_8	0,30	0,48	2,06	0,68	1,90	0,23