UM RASTREADOR VISUAL BASEADO EM REDES NEURAIS SEM PESO E MEMÓRIAS DE PRAZO

Daniel Nunes do Nascimento

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Felipe Maia Galvão França

Rio de Janeiro
Junho de 2015

UM RASTREADOR VISUAL BASEADO EM REDES NEURAIS SEM PESO E MEMÓRIAS DE PRAZO

Daniel Nunes do Nascimento

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Felipe Maia Galvão França, Ph.D.

Prof. Claudio Esperança, Ph.D.

Prof. Rodrigo Tosta Peres, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
JUNHO DE 2015

Nascimento, Daniel Nunes do

Um Rastreador Visual Baseado em Redes Neurais Sem Peso e Memórias de Prazo/Daniel Nunes do Nascimento. – Rio de Janeiro: UFRJ/COPPE, 2015.

XIV, 60 p.: il.; 29, 7cm.

Orientador: Felipe Maia Galvão França

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2015.

Referências Bibliográficas: p. 58 – 60.

1. Rastreamento de objetos. 2. Redes neurais sem peso. 3. Memórias de prazo. I. França, Felipe Maia Galvão. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedico este trabalho a todas as pessoas importantes da minha vida.

Agradecimentos

Agradeço a Deus por todas as oportunidades que tive até aqui e a todos que me acompanharam nesta trajetória, me ajudaram e torceram por esta conquista. Agradeço à minha namorada Amanda, que me incentivou nos momentos mais difíceis deste desafio e foi muito importante para a conclusão deste trabalho; agradeço à minha família que sempre esteve ao meu lado, auxiliando para que pudesse chegar até aqui, em especial ao meu pai, José Valério, minha mãe Rosemery, minha irmã Luiza e meus tios José Maria e Nádia Maria. Agradeço também a todos os parentes e amigos, que sempre proporcionaram momentos importantes na minha vida.

Agradeço ao professor Felipe, pela orientação e pela oportunidade de desenvolver este trabalho; ao Rafael, que me apresentou ao problema de rastreamento de objetos em um momento complicado do mestrado e me auxiliou muito na publicação do artigo que foi o início desta dissertação, e à professora Priscila, que teve a ideia inicial de utilizar estratégias de retreino.

Agradeço também a todos os professores, funcionários, amigos de laboratório, e todas as pessoas que sempre proporcionaram um ambiente muito bom de trabalhar.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM RASTREADOR VISUAL BASEADO EM REDES NEURAIS SEM PESO E MEMÓRIAS DE PRAZO

Daniel Nunes do Nascimento

Junho/2015

Orientador: Felipe Maia Galvão França

Programa: Engenharia de Sistemas e Computação

Um sistema de rastreamento de objetos deve ser capaz de identificar corretamente a localização de um alvo em uma sequência de frames. A tarefa de rastrear objetos envolve diversos problemas, como oclusão, mudança na forma e mudança na escala. Este trabalho apresenta uma solução para o problema de rastreamento, que foi inspirada no funcionamento da memória humana, e na capacidade de armazenamento de memórias de curto e longo prazos. Desta forma, diversos padrões do objeto alvo são aprendidos e armazenados durante o tempo de rastreamento, de modo que padrões que foram aprendidos anteriormente, possam ser recuperados da memória quando necessário.

Esta dissertação apresenta duas propostas de arquitetura baseadas em memórias de prazo e no modelo de redes neurais sem peso WiSARD, que funciona através de uma estrutura de discriminadores, onde cada discriminador é responsável por identificar um padrão. Então, durante o rastreamento, diversos discriminadores são treinados e armazenados para um mesmo objeto que está sendo perseguido, sendo possível recuperar um discriminador antigo da memória e passar a utilizá-lo para identificar o alvo.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A WEIGHTLESS NEURAL NETWORK VISUAL TRACKER BASED ON SHORT AND LONG TERM MEMORIES

Daniel Nunes do Nascimento

June/2015

Advisor: Felipe Maia Galvão França

Department: Systems Engineering and Computer Science

An object tracking system should be able to correctly identify the location of a target in a sequence of frames, and the task of tracking involves many situations such as occlusion, changes of shape and scale. This work presents a solution for the tracking problem, which was inspired by the human memory, and the ability to store short and long-term memories. In this way, many patterns that represent the target are learned and stored during the tracking, so that patterns which were previously learned, can be retrieved from memory when necessary.

This work presents two architectures based on short and long-term memories that use the WiSARD model of weightless neural network, which works with a structure of discriminators, used to identify a pattern. Over the tracking process, several discriminators are trained and stored for the same object being chased, and it's possible to retrieve an old discriminator stored in the memory and start to use it to identify the target.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Rastreamento de objetos é um conhecido problema da área de visão computacional, que envolve técnicas de processamento de imagens e de reconhecimento de padrões, e que já foi bastante estudado e continua a ser explorado através de diversas pesquisas. Este problema consiste no monitoramento automático de um determinado objeto que se movimenta ao longo de uma sequência de quadros presentes em um vídeo, ou seja, uma aplicação de rastreamento deve ser capaz de reconhecer o objeto perseguido, e apontar a localização correta quadro a quadro.

Aplicações de rastreamento de objetos podem ser utilizadas em diversas aplicações reais, e por isso, é uma área que continua sendo bastante pesquisada, e envolve muitos desafios como variação de luminosidade, mudanças na forma do objeto, mudanças na escala e problemas de oclusão do objeto perseguido.

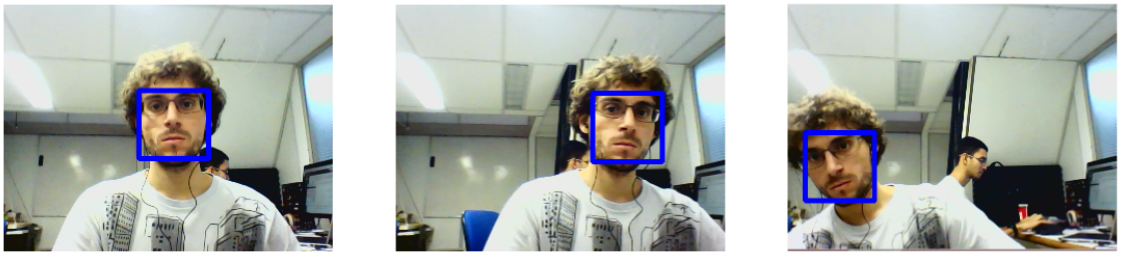

Figura 1.1: Exemplo de rastreamento de rosto: O rastreador deve ser capaz de acompanhar a movimentação do rosto durante o tempo de rastreamento.

1.1 Motivação

Existem muitas aplicações que podem utilizar um rastreador de objetos, e a vasta possibilidade de utilização em aplicações reais, foi a grande motivação para o desenvolvimento deste trabalho. Algumas das aplicações que podem utilizar um rastreador de objetos são listadas abaixo, e mostram a importância do desenvolvimento de pesquisas nessa área.

- **Aplicações de Segurança [3]**
  Muitos lugares possuem câmeras de monitoramento de segurança, e um sistema de rastreamento pode ajudar a monitorar pessoas com comportamento suspeito. Dessa forma, os responsáveis pela segurança de um local podem tomar as atitudes que acharem necessárias, aumentando assim o nível de segurança.

- **Controle por Gestos [4]**
  O rastreamento de objetos pode ser utilizado para rastrear as mãos de uma pessoa, e juntamente com um sistema de reconhecimento, pode-se criar máquinas inteligentes que entendam comandos de sinais, facilitando assim a interação com as pessoas.

- **Reconhecimento de Linguagens de Sinais [5]**
  Da mesma forma, o rastreamento das mãos de uma pessoa pode ser utilizado junto com um sistema de reconhecimento de língua de sinais, como por exemplo, LIBRAS, para auxiliar a comunicação entre as pessoas.

- **Aplicações de Controle por Sinais Enviados Pelo Rosto [6]**
  Um rastreador de objetos, pode ser utilizado para rastrear o rosto de uma pessoa, e através da localização do rosto, extrair sinais que podem ser interpretados por um computador, facilitando a interação de pessoas com necessidades especiais.

- **Monitoramento de Atletas [7]**
  Até mesmo a área de esportes pode se beneficiar de um sistema de rastreamento, pois os atletas podem ser monitorados durante uma partida, através de imagens de câmeras, e então, pode-se obter diversas estatísticas sobre o deslocamento dos jogadores no campo de jogo e dessa forma, trabalhar as estratégias de jogo e de treino.

1.2 Proposta de Trabalho

A proposta deste trabalho é criar uma ferramenta de rastreamento de objetos genéricos utilizando redes neurais sem peso, e um modelo baseado em memórias de prazo, que foi inspirado no funcionamento da memória humana, utilizando conceitos de memórias recentes e memórias de longo prazo. Este modelo visa contornar problemas de oclusão e de mudanças de forma, armazenando diversos padrões aprendidos ao longo do tempo de rastreamento. A localização inicial do objeto a ser perseguido, deve ser passada para o sistema de rastreamento, e a partir deste ponto,

o rastreador deve ser capaz de identificar e localizar corretamente o objeto durante o resto do tempo de rastreamento.

Este trabalho apresenta duas propostas de arquitetura que utilizam redes neurais sem peso e memórias de prazo, e além dos testes que visam avaliar o desempenho do rastreador, foram criados dois sistemas práticos e de tempo real que utilizam as arquiteturas propostas. Foi desenvolvida uma aplicação para interação entre pessoas e computadores, que utiliza comandos extraídos do rosto que são convertidos em comandos para o mouse. Além desta aplicação, foi criado um sistema neuro-simbólico que utiliza o rastreamento dos olhos e da boca, a fim de determinar a pose do rosto de uma pessoa.

1.3 Organização da Dissertação

Este trabalho foi dividido em 6 capítulos. No capítulo 2, o problema de rastreamento de objetos é apresentado com maiores detalhes, em conjunto com alguns trabalhos existentes relacionados com o tema. No capítulo 3, são apresentados conceitos de redes neurais sem peso, dando destaque para o modelo WiSARD. No capítulo 4, são apresentadas as duas arquiteturas de rastreamento propostas, que utilizam redes neurais sem peso e um modelo de memórias de prazo. O capítulo 5, de experimentos e resultados, mostra uma avaliação do desempenho do rastreador, utilizando alguns vídeos de benchmark, além de apresentar duas aplicações que foram desenvolvidas a fim de testar a viabilidade de utilização do rastreador em tempo real. Finalizando, o capítulo 6 apresenta as conclusões e possíveis trabalhos futuros.

Capítulo 2

O Problema do Rastreamento de Objetos

Neste capítulo, o problema do rastreamento de objetos será apresentado em maiores detalhes, considerando-se desafios encontrados, como oclusão, mudança na forma do objeto, mudança de escala e variação de luminosidade. Além disso, ao final do capítulo, alguns trabalhos relacionados com o tema serão descritos brevemente.

2.1 Descrição do Problema

Aplicações de rastreamento de objetos devem ser capazes de identificar a localização de um objeto alvo e acompanhar o seu deslocamento em tempo real. Sendo assim, um sistema desse tipo deve realizar uma busca em cada quadro do vídeo, a fim de encontrar a localização correta do objeto perseguido. O rastreamento é obtido através da identificação de um padrão que representa o objeto alvo, e este padrão é obtido diretamente dos valores obtidos nos pixels das imagens [8].

Para encontrar o objeto alvo, deve existir um sistema capaz de reconhecer o padrão que o represente, onde o treinamento desse sistema é realizado através da informação da localização do objeto de interesse no primeiro quadro do vídeo. A figura a seguir ilustra um exemplo de etapas necessárias para a construção de um sistema de rastreamento de objetos.

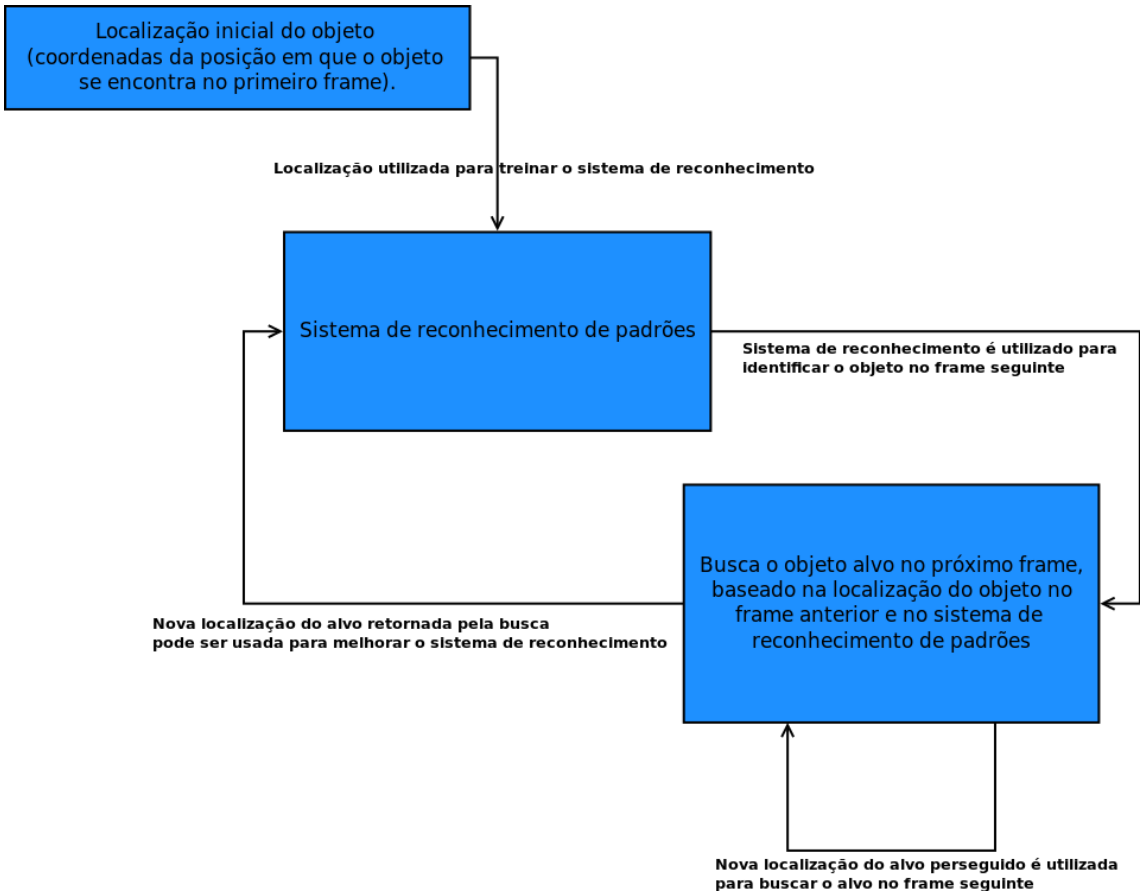

Figura 2.1: Diagrama com possíveis passos para a construção de um sistema de rastreamento de objetos: O sistema de reconhecimento de padrões é treinado a partir da localização inicial do objeto, e então, nos próximos frames, a busca do objeto é realizada, baseando-se na localização do objeto no frame anterior e na resposta retornada pelo sistema de reconhecimento.

2.2 Desafios do Rastreamento

Existem algumas situações que podem prejudicar o bom desempenho de um sistema de rastreamento de objetos, e dentre as dificuldades enfrentadas, as principais são listadas a seguir:

2.2.1 Oclusão

O problema de oclusão ocorre quando o alvo perseguido encontra-se parcialmente ou totalmente encoberto por algum outro objeto presente na cena. Este fator é um complicador para o rastreador, pois quando o objeto fica encoberto, o sistema de reconhecimento pode aprender um padrão errado, como o padrão que representa o objeto que se encontra à frente do objeto originalmente perseguido. Desta forma, o rastreador pode acabar se perdendo, ou ficando preso no objeto da frente, devido ao novo padrão aprendido.

Figura 2.2: Exemplo oclusão: Frame retirado do vídeo 'Occluded Face 2' [1]. O rosto que estava sendo perseguido, se esconde atrás de um livro e de um chapéu. Caso o sistema de reconhecimento aprenda o padrão que representa o livro, um problema que pode ocorrer, é o rastreador passar a perseguir o livro e não conseguir voltar a seguir o rosto.

2.2.2 Mudança na Forma

O problema de mudança na forma acontece quando o objeto perseguido passa a apresentar-se com uma aparência não vista pelo sistema anteriormente. Pode ocorrer quando há uma mudança na posição do objeto em relação à câmera, como por exemplo, quando uma pessoa que estava posicionada com a face apontada na direção da câmera, vira-se de lado e o seu perfil passa a ser observado.

As possíveis modificações na aparência do objeto alvo atrapalham o rastreador, pois caso o objeto passe a se apresentar com uma forma desconhecida do sistema, o rastreador pode acabar perdendo o alvo na cena.

Figura 2.3: Exemplo mudança de forma: Frames retirados do vídeo 'Tiger 1' [1]. O objeto rastreado é um tigre, que quando abre a boca, muda a sua forma de apresentação.

2.2.3 Mudança de Escala

A mudança de escala é um problema muito frequente em aplicações de rastreamento, e ocorre quando o objeto perseguido afasta-se ou aproxima-se da câmera. Essa situação pode causar problemas para o rastreador caso o sistema de reconhecimento esteja apto apenas a reconhecer o alvo a uma determinada distância.

2.2.4 Variação na Luminosidade

A variação de luminosidade também é um dos desafios presentes em aplicações que realizam rastreamento de objetos, pois o ambiente no qual o alvo se encontra, pode ficar mais claro ou mais escuro em diferentes momentos, atrapalhando o sistema de reconhecimento.

Figura 2.4: Exemplo de mudança de forma, escala e luminosidade: Frames retirados do vídeo 'David Indoor' [1]. Neste vídeo, a pessoa rastreada se desloca em um ambiente com variação de luminosidade, e além disso, modifica a sua posição em relação à câmera, causando mudanças de forma e escala.

2.3 Trabalhos Relacionados

Rastreamento de objetos pode ser aplicado em diversas situações, e muitas pesquisas já foram e ainda vêm sendo desenvolvidas na área. Alguns dos trabalhos relacionados são descritos brevemente abaixo:

- **Movement persuit control of an offshore automated platform via a ram-based neural network [9]**
  Neste trabalho, os autores desenvolveram um sistema de visão computacional que utiliza o modelo de redes neurais sem peso WiSARD, a fim de acompanhar a cadência de navios.

- **Online tracking of multiple objects using WiSARD [10]**
  Este trabalho foi uma primeira tentativa de utilização do modelo WiSARD para resolver o problema de rastreamento de objetos, onde os autores utilizaram técnicas para retreinar o sistema em tempo real, com o objetivo de contornar problemas de ruídos de fundo ou alterações no formato do alvo perseguido.

  A partir deste trabalho, surgiu a ideia de construir uma aplicação que armazene diversos discriminadores WiSARD treinados ao longo do rastreamento, que representam diversos padrões de um mesmo objeto. Essas ideias deram origem ao artigo descrito a seguir, e ao desenvolvimento desta dissertação.

- **A WiSARD-based multi-term memory framework for online tracking of objects [11]**
  O desenvolvimento de um rastreador visual baseado em redes neurais sem peso e memórias de prazo, gerou a publicação desse artigo e desta dissertação. Neste trabalho, utilizou-se o armazenamento de diferentes discriminadores WiSARD, treinados ao longo do tempo de rastreamento, onde discriminadores antigos eram utilizados para classificar o objeto quando este passava a se apresentar em uma forma já vista pelo sistema em algum momento. Como este trabalho foi a base para esta dissertação, maiores detalhes serão apresentados durante o decorrer do texto.

- **Robust object tracking with online multiple instance learning [12] [13]**
  Nestes trabalhos, os autores desenvolveram um rastreador que utiliza um algoritmo de *multiple instance learning*, onde os exemplos de treinamento são agrupados em conjuntos, cada um com um rótulo, ao invés de rotular cada instância individualmente.

- **Tracking Learning Detection [14] [15]**
  Nestes trabalhos, o autor desenvolveu um método para rastreamento de objetos, que decompõe a tarefa de rastrear, em três subtarefas: *Tracking, Learning and Detection*. A componente de *Tracking* segue o objeto quadro a quadro do vídeo; o *Detector* localiza aparências vistas durante o rastreamento e corrige o rastreador; e a parte de *Learning* estima erros cometidos pelo *Detector*.

- **Sistema de Rastreamento Visual de Objetos Baseado em Movimentos Oculares Sacádicos [16]**
  A autora deste trabalho utilizou o modelo de redes neurais sem peso VGRAM, para desenvolver um sistema computacional de busca visual, inspirado bio-

logicamente nos movimentos sacádicos dos olhos, que foi incorporado em um sistema de rastreamento automático de objetos.

Capítulo 3

Redes Neurais Sem Peso

Neste capítulo, são apresentados conceitos de redes neurais clássicas e de redes neurais sem peso, dando destaque para o modelo de rede neural sem peso WiSARD, que foi utilizado no desenvolvimento do rastreador de objetos genéricos, como ferramenta para identificação do alvo perseguido.

3.1 Redes Neurais Clássicas

Redes neurais artificiais [17] são modelos computacionais inspirados no funcionamento dos neurônios biológicos, e são utilizadas para a resolução de problemas de difícil formulação matemática ou de problemas aos quais não se conhece uma formulação que os descrevem. São muito aplicadas em situações que envolvam algum tipo de classificação ou reconhecimento de padrões, e para a sua utilização, é necessário executar uma etapa de treinamento, onde a rede neural adquire o conhecimento. Essa etapa de treinamento é realizada a partir de um conjunto de exemplos previamente classificados, que são apresentados para a rede neural. Após o treinamento, a rede é utilizada para classificar exemplos que não foram vistos anteriormente, e essa classificação é feita através do conhecimento adquirido a partir dos exemplos que foram apresentados durante a etapa de treinamento.

O Neurônio Artificial

No modelo tradicional de redes neurais, apresentado por McCulloch e Pitts [18], um neurônio recebe um conjunto de entradas e fornece uma saída que pode ser utilizada como entrada para outros neurônios. A conexão entre os neurônios é feita através dos pesos sinápticos, e cada uma das entradas recebidas pelo neurônio, é multiplicada pelo seu respectivo peso sináptico. Então, o somatório dessas entradas multiplicadas, é aplicado em uma função de propagação, resultando em um novo valor que pode ser usado como entrada de outros neurônios.

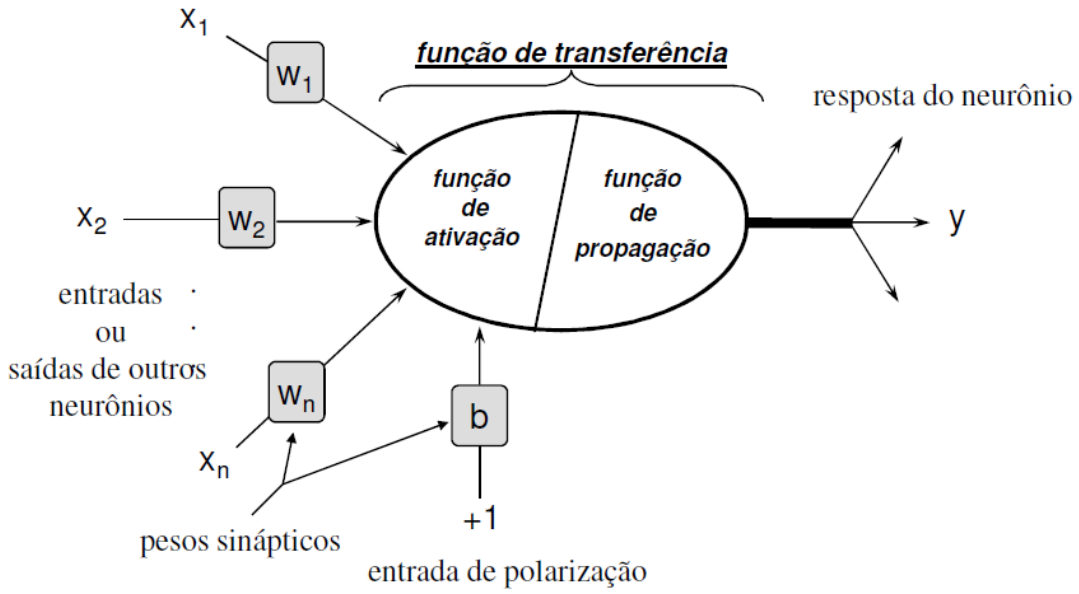

Figura 3.1: Representação do neurônio artificial [2]

Uma rede neural com pesos, é formada por um conjunto de neurônios, conectados uns aos outros através de pesos sinápticos que detêm o conhecimento adquirido através dos exemplos de treinamento previamente classificados. Os valores dos pesos sinápticos são atualizados durante a etapa de treinamento, com base nos exemplos que são apresentados, e então, a rede neural deve ser capaz de generalizar o conhecimento adquirido durante a etapa de treinamento, a fim de classificar corretamente, padrões de entrada desconhecidos.

3.2 Modelos Sem Peso

Diferente das redes neurais tradicionais, as redes neurais sem peso não possuem pesos sinápticos. Neste tipo de rede neural, os neurônios são representados por memórias RAM (Mémórias de Acesso Aleatório) [19], que endereçam uma determinada quantidade de bits, onde o conhecimento fica armazenado nos endereços que são acionados durante a etapa de treinamento.

Neste trabalho, o modelo de redes neurais sem peso utilizado foi a WiSARD (Wilkie, Stonham and Aleksander's Recognition Device) [20], porém, existem outros modelos, como AutoWiSARD [21], VG-RAM [22], SDM [23] e GNU [24].

3.2.1 WiSARD

O modelo WiSARD (Wilkie, Stonham and Aleksander's Recognition Device) [20] utiliza uma estrutura de discriminadores, onde cada discriminador é responsável por

identificar uma única classe.

Um discriminador é composto por um conjunto de memórias RAM, onde todos os endereços de todas as RAMs são inicializados com zeros. Durante a etapa de treinamento, um padrão de entrada é convertido em um vetor de endereços, onde cada endereço representa uma posição de uma determinada memória RAM que terá o seu conteúdo alterado. O primeiro elemento do vetor representa o endereço que terá o seu conteúdo incrementado dentro da primeira RAM, o segundo elemento do vetor representa o endereço na segunda RAM que terá o seu conteúdo incrementado e assim sucessivamente até a última RAM.

Na etapa de reconhecimento, o padrão de entrada é convertido para um vetor de endereços, assim como feito na etapa de treinamento. Entretanto, neste caso, os conteúdos dos endereços não são incrementados, pois ocorre somente uma verificação para saber se a posição já foi acessada anteriormente ou não, e em caso positivo, a RAM retorna o valor 1; caso contrário, retorna valor 0. Então, o discriminador retorna como resposta, o total de neurônios (memórias RAM), que foram ativados para aquele padrão de entrada apresentado.

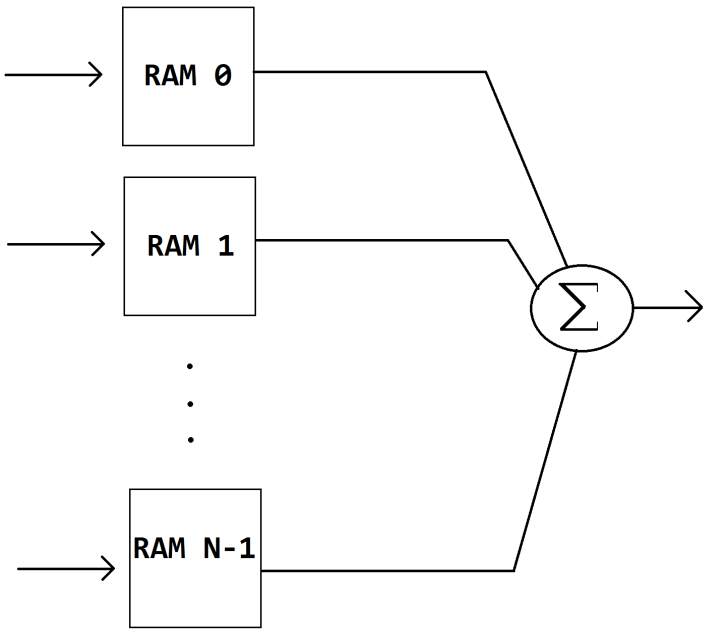

Figura 3.2: Representação de um discriminador com N memórias RAM.

Treinamento

Durante a etapa de treinamento, para cada discriminador, são apresentados exemplos de uma única classe. Para cada exemplo de treinamento de uma classe específica, é extraído um vetor de endereços que é passado para o treinamento do discriminador correspondente. Cada elemento do vetor, corresponde a um endereço

de uma das RAMs do discriminador que terá o seu valor alterado para 1. Caso ocorra de o endereço já possuir valor igual a 1, o conteúdo se mantém inalterado.

A figura a seguir, mostra um exemplo de treinamento de um discriminador através da apresentação de um padrão de entrada representado por uma imagem binarizada, onde pixels brancos e pretos indicam valores iguais a 0 e 1, respectivamente. Como o treinamento do exemplo utiliza 3 bits, cada RAM é capaz de endereçar 3 bits, ou seja, 8 posições de memória. Então, cada endereço é obtido através do sorteio de 3 pixels da imagem. A ordem do sorteio é representada pelos números que estão dentro de cada pixel da imagem. Sendo assim, para os três primeiros pixels sorteados, ou seja, os pixels 0, 1 e 2, os valores correspondentes são 0 (branco), 1 (preto) e 1 (preto), e então, o endereço 011 tem o seu valor alterado dentro da primeira RAM; na segunda RAM, o endereço obtido para ter o seu conteúdo alterado é o endereço 101, na terceira RAM, 100 e na quarta RAM, 111.

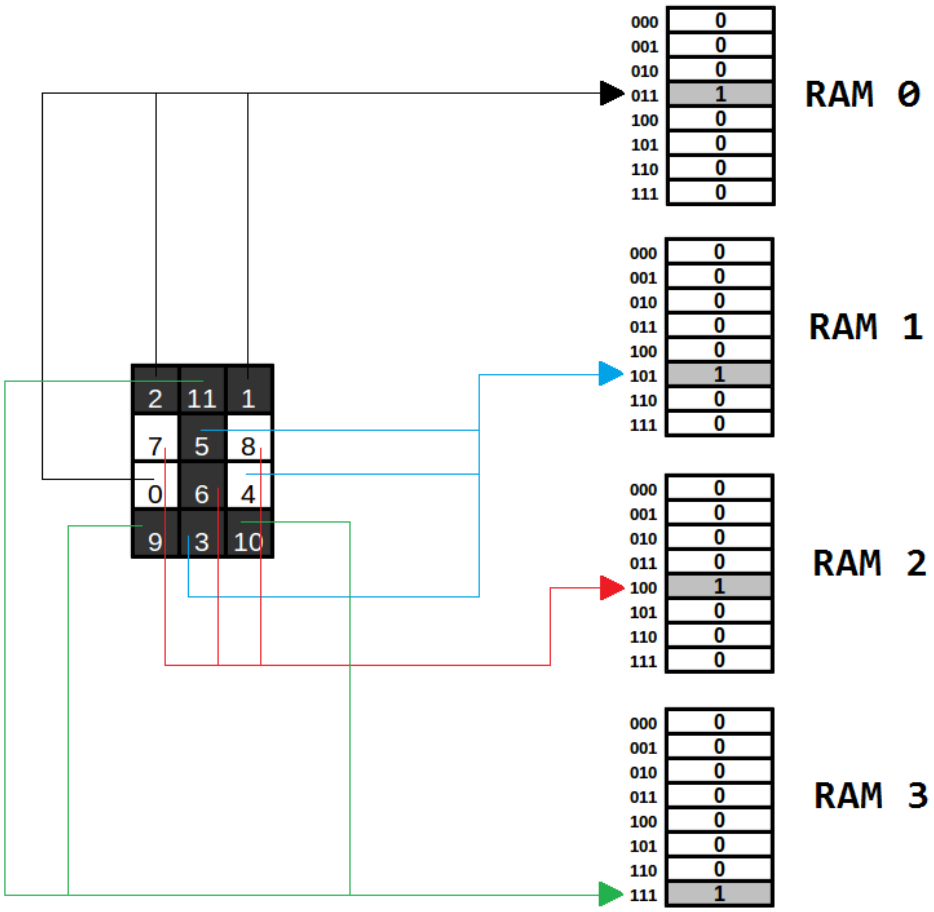

Figura 3.3: Ilustração do treinamento de um discriminador, a partir da apresentação de um padrão de entrada.

Classificação

Após a realização do treinamento dos discriminadores, cada um representando uma classe, estes são utilizados para classificar padrões de entrada não vistos anteriormente. Na etapa de classificação, a partir do padrão que deseja-se obter uma classificação, realiza-se a extração do vetor de endereços da mesma maneira que foi feita na etapa de treinamento, utilizando a mesma ordem de escolha dos pixels. Para cada endereço deste vetor, verifica-se se a RAM correspondente foi ativada. Caso o conteúdo do endereço da RAM correspondente, seja maior que zero, significa que este neurônio foi ativado. A figura a seguir ilustra um discriminador treinado que é utilizado para classificar um padrão de entrada:

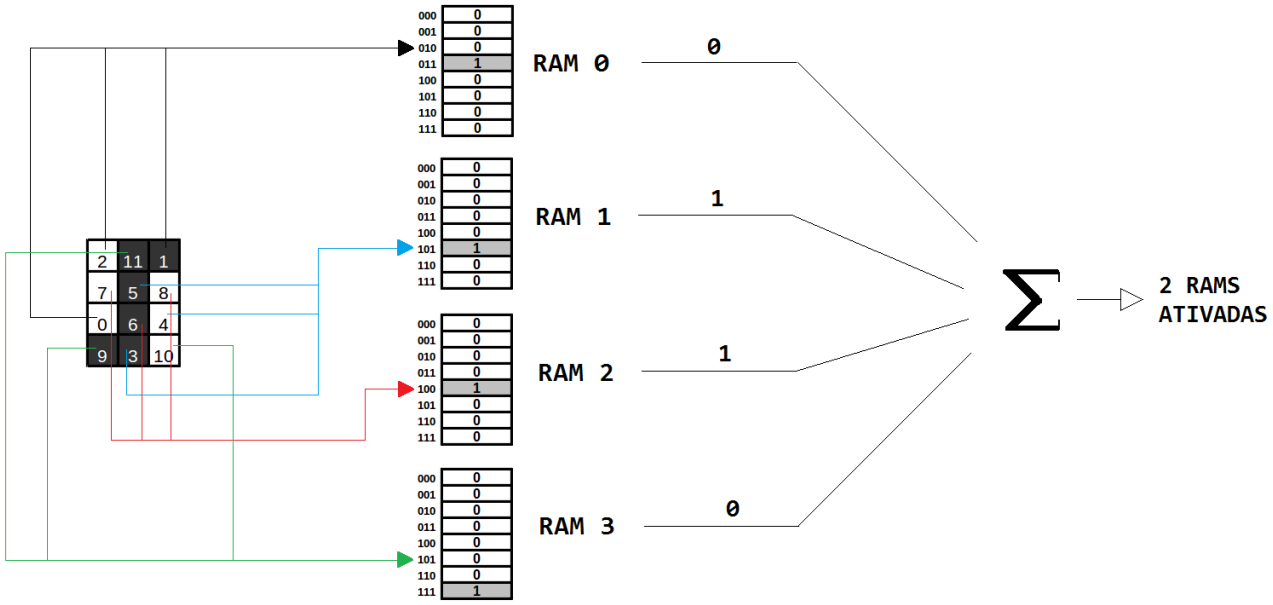

Figura 3.4: Classificação: Dois dos quatro neurônios foram ativados neste discriminador para este padrão de entrada.

O vetor de endereços é passado como entrada para todos os discriminadores da rede, e aquele que retornar o maior número de neurônios ativados é escolhido para dar a classificação para o padrão.

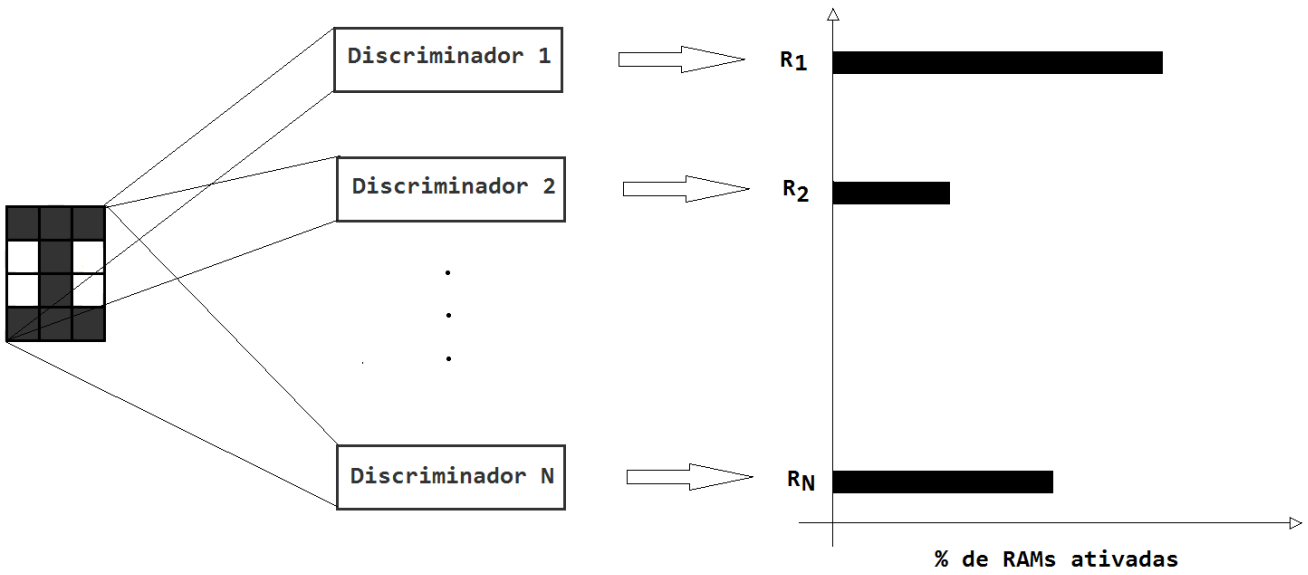

Figura 3.5: Classificação: O padrão de entrada é passado para todos os discriminadores, e aquele com maior número de neurônios ativados, é utilizado como resposta.

Neste trabalho, o modelo WiSARD foi utilizado, pois além de ser muito rápido, tanto na etapa de treinamento, quanto na etapa de classificação, funciona de forma satisfatória para o reconhecimento de objetos de formas variadas, não havendo a necessidade de identificar descritores específicos de cada objeto.

No desenvolvimento deste rastreador, a única informação utilizada para treinar o classificador foi a imagem binarizada, sem considerar formatos de objetos específicos. Então, a WiSARD mostrou-se bastante adequada para este tipo de aplicação, onde o objetivo principal é rastrear diferentes tipos de objetos, independente da sua forma.

Capítulo 4

Rastreador Genérico Utilizando WiSARD e Memórias de Prazo

Neste capítulo, serão apresentados os conceitos de memórias de prazo que inspiraram a criação deste trabalho. Além disso, serão apresentadas duas propostas de arquitetura que foram utilizadas para desenvolver o rastreador de objetos genéricos, ambas baseando-se em memórias de prazo e discriminadores WiSARD. As duas arquiteturas serão apresentadas, mostrando as etapas de treinamento, busca e classificação do alvo perseguido.

4.1 Memórias de Prazo

O desenvolvimento do rastreador de objetos genéricos foi inspirado no funcionamento da memória humana e nos conceitos de memórias de curto e longo prazos [25]. Na memória de curto prazo, as informações são armazenadas durante um curto período de tempo, e podem ser transferidas para a memória de longo prazo, porém, se a transferência para a memória de longo prazo não for realizada neste período de tempo, as informações são perdidas. Na memória de longo prazo, ocorre a consolidação das informações, que podem ser recuperadas mesmo depois de muitos anos, e podem ser transferidas para as memórias de curto prazo.

Os conceitos de memórias de curto e longo prazos inspiraram a criação de um modelo que armazena diferentes discriminadores WiSARD, que são treinados em tempo real a partir de diferentes padrões do mesmo objeto que está sendo rastreado. Em cada quadro do vídeo, todos os discriminadores guardados em memória, são utilizados para tentar encontrar o objeto perseguido, sendo que aquele que retorna uma resposta considerada como a melhor, é escolhido para determinar a localização do objeto naquele quadro.

A ideia principal é que durante o tempo de rastreamento, como o objeto

perseguido pode modificar o seu formato ou então ficar parcialmente ou totalmente escondido por um outro objeto presente na cena, diversos padrões do alvo rastreado devem ser utilizados para treinar diferentes discriminadores, que são guardados na memória para serem utilizados quando possível, no momento em que o objeto volte a apresentar um formato semelhante a algum já visto anteriormente.

4.2 Arquitetura 1

A primeira proposta de arquitetura utiliza uma fila de tamanho limitado para armazenar os discriminadores treinados durante o rastreamento. A cada treinamento realizado, o novo discriminador é inserido no início da fila, e dentre todos os discriminadores presentes, aquele que retorna o melhor resultado é passado para a primeira posição. Desta forma, pode-se fazer uma analogia com a memória humana, onde os discriminadores do início da fila, fazem parte da memória mais recente (de curto prazo), e quando um discriminador se desloca para a primeira posição, ocorre uma recuperação da informação que estava armazenada na memória mais antiga (de longo prazo), que passa a ser considerada como parte da memória mais recente. O fato de recuperar a informação de uma memória antiga e passá-la para a memória recente, faz com que os discriminadores que estão há mais tempo sem utilização, se desloquem em direção ao final da fila e sejam naturalmente descartados. Sendo assim, as informações contidas nestes discriminadores são "esquecidas"pelo sistema.

Após o treinamento inicial do primeiro discriminador, se o objeto mudar de forma, o padrão aprendido pode não ser adequado para conseguir encontrar o objeto que modificou sua forma. Então, treina-se um novo discriminador representando o novo padrão, e a partir desse momento, utiliza-se esse novo discriminador para encontrar o objeto na cena. Porém, o discriminador antigo não é descartado, pois ele pode ser útil em algum momento futuro. Então, ao invés de descartá-lo, armazena-se o discriminador que representa o padrão antigo, em uma fila de discriminadores para ser utilizado novamente quando possível. O rastreamento segue nos quadros seguintes do vídeo utilizando esse novo discriminador, porém, a todo instante, o discriminador antigo que está armazenado continua sendo utilizado para procurar o objeto, então, se em algum momento o discriminador antigo retornar um maior número de neurônios ativados, assume-se que este é mais adequado para encontrar a localização do objeto, e dessa forma, este discriminador passa para a primeira posição da fila para ser utilizado. Da mesma maneira, se nenhum dos dois discriminadores presentes na fila retornar uma pontuação adequada (acima do limiar estabelecido), um terceiro discriminador é treinado e colocado na primeira posição da fila de discriminadores e passa a ser utilizado a partir desse momento. Esse processo continua durante o rastreamento, e o discriminador presente na fila que

retornar a maior pontuação sempre é passado para a primeira posição. Esse procedimento faz com que os discriminadores que estão há mais tempo sem utilização sejam movidos em direção ao final da fila, e como esta possui tamanho fixo, quando estiver cheia e um novo discriminador precisar ser treinado e armazenado, aquele presente na última posição é descartado para liberar espaço.

4.2.1 Treinamento

Inicialmente, a localização do objeto (coordenadas em pixels do retângulo que contém o objeto) que vai ser perseguido pelo rastreador deve ser passada para a aplicação. Dessa forma, o sistema aprende o padrão que representa o objeto na forma como é apresentado no primeiro quadro.

O treinamento consiste na criação de um discriminador para identificar o objeto nos próximos quadros do vídeo, e a entrada para o treinamento consiste da imagem binarizada e das coordenadas que representam a posição do retângulo que contém o objeto. A binarização [8] é feita a partir da imagem em tons de cinza, e o limiar de binarização utilizado é a média da luminância dos pixels presentes dentro do retângulo que compreende o objeto. Assim, pixels com valores abaixo da luminância média recebem valor igual a zero (pixels tornam-se pretos) e pixels com valores acima da luminância média recebem valor igual a um (pixels tornam-se brancos).

Após a realização da binarização, a entrada para o treinamento do discriminador é criada a partir da escolha de pixels aleatórios que formarão os endereços que serão ativados em cada RAM do discriminador. A quantidade de RAMs é determinada pelo número de pixels contidos dentro do retângulo que envolve o objeto, dividido pelo número de bits utilizados. Por exemplo, se o número de bits de cada RAM for definido em 3, então, dentro do retângulo que define o objeto perseguido na imagem binarizada, sorteia-se 3 pixels aleatórios que formarão o endereço que será ativado na primeira RAM; em seguida, sorteia-se 3 outros pixels para formarem o endereço que será ativado na segunda RAM e assim por diante, até chegar no endereço ativado dentro da última RAM.

4.2.2 Busca do Objeto Perseguido

Inicialmente, é realizado o treinamento de um discriminador a partir do padrão apresentado no primeiro quadro. Esse discriminador é utilizado nos próximos quadros do vídeo para procurar o objeto que está sendo perseguido. Então, a partir do segundo quadro do vídeo, faz-se uma busca ao redor da posição em que o objeto encontrava-se no quadro anterior (a região de busca é definida aumentando-se o retângulo que contém o objeto, de uma quantidade de pixels determinada previamente, para cada um dos lados), procurando a região que obtenha a maior pontuação com esse dis-

criminador (maior número de RAMS ativadas). Uma janela de mesmo tamanho do objeto perseguido, desloca-se dentro da região de busca, e para cada posição dessa janela, realiza-se uma classificação utilizando o discriminador treinado. A entrada para a classificação é obtida da mesma forma que a entrada obtida para o treinamento, ou seja, faz-se a binarização da imagem utilizando como limiar, a média da luminância dos pixels presentes na janela de mesmo tamanho do objeto que se desloca dentro da região de busca, e após isto, com a mesma ordem de sorteio de pixels da etapa de treinamento, monta-se um conjunto de endereços que serão verificados em cada uma das RAMs do discriminador. Para o primeiro endereço, verifica-se na primeira RAM do discriminador, se o endereço encontra-se ativado, ou seja, se o valor presente nele é igual a 1; para o segundo endereço, faz-se a verificação na segunda RAM, e assim sucessivamente, até que ocorra a verificação da última RAM.

Um neurônio é ativado, quando a memória RAM possui valor igual a 1 no endereço verificado. Sendo assim, para cada uma das regiões de mesmo tamanho do objeto perseguido, dentro da região de busca, obtém-se uma quantidade de neurônios ativados, que é a pontuação obtida pelo discriminador. Então, aquela região que contiver o maior número de neurônios ativados é a escolhida para representar a nova localização do objeto perseguido.

Para cada quadro do vídeo, atualiza-se a localização do objeto e busca-se ao redor dessa região no quadro seguinte, utilizando o discriminador treinado. Conforme o vídeo avance, pode acontecer de o discriminador treinado retornar pontuações baixas, e quando isso acontece, pode ser necessário treinar um novo discriminador para que o rastreador não se perca (esse novo treinamento é realizado com base na localização que obteve a melhor pontuação). Durante o rastreamento, todos os discriminadores presentes na fila são utilizados na busca do objeto, e aquele com a maior pontuação é escolhido para retornar a posição do objeto.

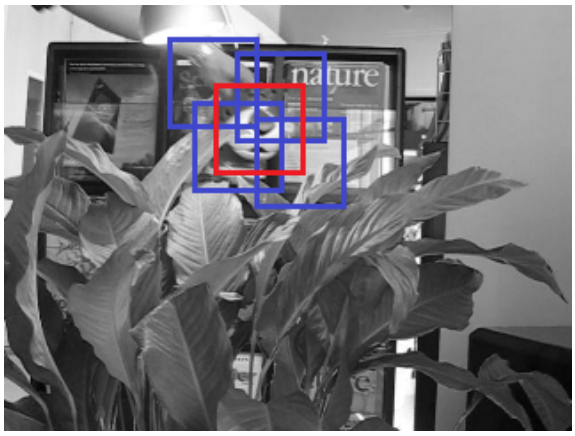

Figura 4.1: Busca local: A busca do alvo ocorre no entorno da posição do objeto no frame anterior. Neste caso, há uma região de busca, cujo tamanho é predefinido, e há uma janela, de mesmo tamanho do objeto perseguido, que se desloca dentro da região de busca. Em cada posição, ocorre uma classificação utilizando-se todos os discriminadores pertencentes à fila, e aquela posição que recebe a melhor pontuação de algum dos discriminadores, é considerada como a nova localização que contém o objeto alvo. Esta localização retornada é utilizada para a busca do objeto no frame seguinte, e assim sucessivamente.

4.2.3 Execução do Rastreador com a Arquitetura 1

Durante o tempo de rastreamento, diferentes discriminadores são treinados para que o sistema possa manter a localização correta do objeto, e ocorre um gerenciamento de uma fila de discriminadores, de forma que aquele que retorna a melhor resposta, sempre é passado para a primeira posição. Além disso, aqueles discriminadores que ficam mais tempo sem utilização, acabam sendo descartados quando surge a necessidade de realizar-se um novo treinamento.

Abaixo encontra-se uma série de quadros, que exemplificam o funcionamento do modelo proposto, utilizando um limiar para realizar um novo treinamento igual a 0,5 das RAMs ativadas:

Figura 4.2: No primeiro frame, a localização do objeto é passada como entrada para o rastreador e o primeiro discriminador é treinado e armazenado na fila.

Figura 4.3: Após alguns frames, o discriminador D1 retorna 0,85 das RAMs ativadas, logo, continua sendo utilizado para classificar o objeto perseguido.

Figura 4.4: Neste frame, 0,45 das RAMs foram ativadas, ficando abaixo do *threshold* predefinido de 0,5. Sendo assim, treinou-se um novo discriminador D2, que foi armazenado no início da fila.

Figura 4.5: Neste momento, a fila possui os discriminadores D1 e D2. Ambos são utilizados para classificar o objeto, sendo que D1 retorna 0,7 das RAMs ativadas e D2 retorna 0,4; então, ocorre uma mudança na ordenação da fila, passando o discriminador D1 novamente para o início da fila.

Figura 4.6: Neste frame, o tigre virou de lado e a sua aparência se tornou diferente das formas vistas pelo sistema anteriormente. Como o discriminador D1 retornou 0,45 de RAMs ativadas e o discriminador D2 retornou 0,3, ambos abaixo do limiar de treinamento, treinou-se o discriminador D3, colocado no início da fila.

Essas constantes atualizações da fila de discriminadores, permitem acompanhar o objeto, pois sempre que o alvo passa a se apresentar de alguma maneira já vista anteriormente pelo sistema, um discriminador antigo é recuperado da memória para ser utilizado. Da mesma forma, também é possível sempre treinar um novo discriminador, para os casos em que o objeto se apresente em alguma forma desconhecida pelo sistema. Essa estratégia rendeu bons resultados, que serão apresentados no próximo capítulo.

4.2.4 Algoritmo

Os passos necessários para realizar o rastreamento utilizando a Arquitetura 1 estão resumidos no algoritmo a seguir:

Algorithm 1 Arquitetura 1

1: Obter coordenadas da localização do objeto no primeiro frame.
2: Treinar primeiro discriminador.
3: Armazenar discriminador treinado na fila de discriminadores.
4: **while** Existem frames para processar **do**
5: Buscar objeto no próximo frame ao redor da localização no frame anterior utilizando todos os discriminadores presentes na fila.
6: A região que obtiver a maior pontuação (porcentagem de neurônios ativados) para um dos discriminadores, será usada como resposta para a localização correta do objeto.
7: **if** Todas as pontuações dos discriminadores para todas as localizações possíveis dentro da região de busca forem menores do que o limiar estabelecido **then**
8: Treinar novo discriminador com a imagem encontrada na localização que obteve a maior pontuação.
9: **if** Fila de discriminadores estiver cheia **then**
10: Descartar discriminador presente na última posição da fila.
11: **end if**
12: Armazenar o discriminador recém-treinado no início da fila de discriminadores.
13: **end if**
14: **end while**

4.3 Arquitetura 2

Nesta seção, a segunda proposta de arquitetura para o desenvolvimento do rastreador de objetos genéricos, chamada de Arquitetura 2, será apresentada. Esta arquitetura também é baseada em memórias de prazo e discriminadores WiSARD, porém, a grande diferença, é que nesta segunda arquitetura, há um limite definido de novos discriminadores que podem ser treinados, e também é permitido realizar o chamado *retreino* de um discriminador. Neste caso, o número de discriminadores novos que são criados, é limitado pelo tamanho da fila, ou seja, se em uma determinada execução do rastreador, a fila de discriminadores possui capacidade de armazenar 10 discriminadores, então, o número máximo de treinamentos de novos discriminadores, será igual a 10, porém, será permitido realizar o retreino, onde um discriminador já existente, recebe outro padrão de entrada para ser aprendido, sem esquecer o conhecimento antigo adquirido. Essa estratégia de retreinar um mesmo discriminador quando necessário, é uma tentativa de fazer com que um mesmo discriminador identifique o objeto em formas e posições distintas.

4.3.1 Estratégia de Retreino

No início do rastreamento, a localização inicial do objeto é passada como entrada para o sistema, assim como feito na Arquitetura 1. Dessa forma, o primeiro padrão de entrada é utilizado para treinar o primeiro discriminador, que é armazenado na fila de discriminadores. Este discriminador é utilizado para identificar o objeto nos frames seguintes, porém, desta vez, existem 2 *thresholds* que são utilizados para determinar três possíveis situações: a primeira ocorre quando deve-se realizar o treinamento de um novo discriminador; a segunda situação ocorre quando existe a necessidade de retreinar um discriminador já existente na fila de discriminadores; e a terceira situação ocorre quando não é necessário treinar um novo discriminador, nem retreinar um discriminador já existente. Os dois *thresholds* serão denominados de *limiar de retreino de discriminador* e *limiar de novo discriminador*.

Enquanto o primeiro discriminador retornar pontuações acima do *limiar de retreino de discriminador*, nenhum novo discriminador é treinado, e a localização retornada por este discriminador é aceita como a localização correta do objeto na cena. Caso este discriminador retorne uma pontuação entre o *limiar de novo discriminador* e o *limiar de retreino de discriminador*, é realizado um retreino desse discriminador, e caso a pontuação retornada seja abaixo do *limiar de novo discriminador*, um novo discriminador é treinado e armazenado na fila de discriminadores.

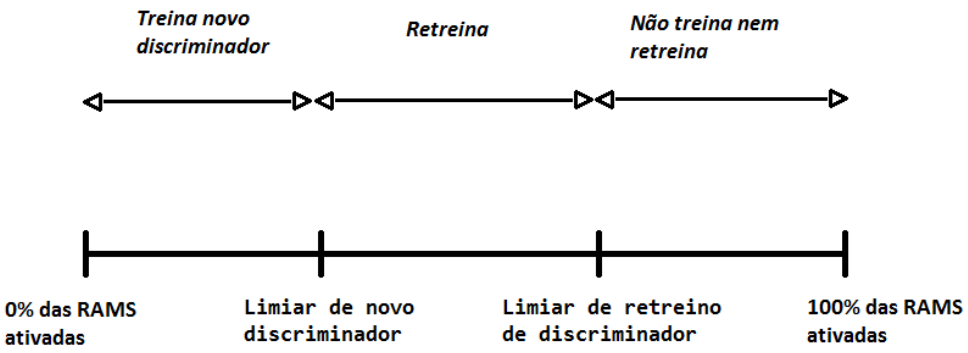

Figura 4.7: Limiares que determinam se um novo discriminador deve ser treinado, se um antigo deve ser retreinado ou se nada deve ser feito.

Em cada frame executado, a busca do objeto alvo é feita utilizando todos os discriminadores presentes na fila. Dentro da região de busca, em cada localização possível, utiliza-se todos os discriminadores para classificar o objeto perseguido, e aquela posição que obtiver a maior pontuação é utilizada como a localização correta do objeto. Caso essa pontuação fique acima do *limiar de retreino de discriminador*,

nenhum treinamento é realizado; caso essa pontuação fique entre o *limiar de novo discriminador* e o *limiar de retreino de discriminador*, o discriminador que retornou esta pontuação é retreinado com o padrão de entrada correspondente à localização retornada; e caso a pontuação seja inferior ao *limiar de novo discriminador*, um novo discriminador é treinado e armazenado se a fila não estiver cheia. O número de retreinos por discriminador deve ser limitado, para evitar que um mesmo discriminador fique com uma grande quantidade de endereços preenchidos, e passe a retornar pontuações altas para partes da cena diferentes do objeto perseguido.

4.3.2 Execução do Rastreador com Arquitetura 2

A seguir, será apresentado um exemplo de execução do rastreador utilizando a Arquitetura 2, mostrando a atualização da fila de discriminadores com a criação de novos e com o retreino de discriminadores antigos. Os frames do vídeo Tiger 1 são usados novamente para ilustrar o processo de rastreamento, o os *thresholds* adotados são: 0,6 para *limiar de retreino de discriminador* e 0,4 para *limiar de novo discriminador*.

Figura 4.8: No primeiro frame, a localização do objeto é passada como entrada para o rastreador e o primeiro discriminador é treinado e armazenado na fila.

Figura 4.9: Após alguns frames, o discriminador D1 retorna 0,85 das RAMs ativadas, ou seja, acima do *limiar de retreino de discriminador*. Logo, continua sendo utilizado para classificar o objeto perseguido.

Figura 4.10: Neste frame, 0,35 das RAMS foram ativadas pelo discriminador D1, e então, como o valor encontra-se abaixo do *limiar de novo discriminador*, treinou-se o discriminador D2, e a partir deste momento, os dois discriminadores passam a ser utilizados na busca pelo objeto alvo.

Figura 4.11: Neste frame, a melhor pontuação foi 0,5 das RAMS ativadas, obtida pelo discriminador D2. Dessa forma, como o valor encontra-se acima do *limiar de novo discriminador* e abaixo do *limiar de retreino de discriminador*, o discriminador D2 recebe o primeiro retreino.

Figura 4.12: Neste frame, novamente a melhor pontuação foi obtida pelo discriminador D2, com valor entre o *limiar de novo discriminador* e o *limiar de retreino de discriminador*. Logo, o discriminador D2 recebe o segundo retreino.

A atualização da fila de discriminadores continua ocorrendo até que todas as posições estejam preenchidas e todos os discriminadores já tenham recebido o número máximo de retreinos permitidos. A partir deste momento, o rastreamento continua utilizando sempre o discriminador que retornar a melhor pontuação na busca pelo alvo, porém, nenhum novo treinamento é realizado.

4.3.3 Algoritmo

O algoritmo que resume a Arquitetura 2, que utiliza a estratégia de retreino, é apresentado a seguir:

Algorithm 2 Arquitetura 2

1: Obter coordenadas da localização do objeto no primeiro frame.
2: Treinar primeiro discriminador.
3: Armazenar discriminador treinado na fila de discriminadores.
4: **while** Existem frames para processar **do**
5: Buscar objeto no próximo frame ao redor da localização no frame anterior utilizando todos os discriminadores presentes na fila.
6: A região que obtiver a maior pontuação (porcentagem de neurônios ativados) para um dos discriminadores, será usada como resposta para a localização correta do objeto.
7: **if** A pontuação do melhor discriminador estiver acima do *limiar de retreino de discriminador* **then**
8: Utilizar a localização como resposta e não realizar nenhum treinamento.
9: **end if**
10: **if** A pontuação do melhor discriminador estiver entre o *limiar de novo discriminador* e o *limiar de retreino de disciminador* **then**
11: **if** Discriminador recebeu um número de retreinos menor do que a *quantidade de retreinos permitida* **then**
12: Retreinar o melhor discriminador com a imagem encontrada na localização que obteve a melhor pontuação.
13: **end if**
14: **end if**
15: **if** A pontuação do melhor discriminador estiver abaixo do *limiar de novo discriminador* **then**
16: **if** Fila de discriminadores não estiver cheia **then**
17: Treinar novo discriminador com a imagem encontrada na localização que obteve a maior pontuação.
18: Armazenar novo discriminador na fila de discriminadores.
19: **end if**
20: **end if**
21: **end while**

Capítulo 5

Experimentos e Resultados

Neste capítulo, serão apresentados os experimentos realizados para a avaliação do rastreador baseado em memórias de prazo e WiSARD. Para avaliar o desempenho do rastreador, foram utilizados alguns vídeos de benchmark, disponíveis em [1]. Além disso, serão apresentadas outras aplicações desenvolvidas a fim de verificar a utilização do rastreador em tempo real, como controle de mouse através de comandos extraídos do rosto e aplicação para definir a pose do rosto através do rastreamento de seus componentes (olhos e boca).

5.1 Métricas de Avaliação de Rastreadores

Algumas das métricas que são utilizadas para a avaliação de sistemas de rastreamento, como coeficiente de Jaccard, erro médio do centro da localização e cálculo da precisão, são descritas a seguir:

5.1.1 Erro médio do Centro

Uma das métricas utilizadas para avaliar a performance do rastreador desenvolvido, foi o erro médio da localização do centro do objeto, medido em pixels e calculado quadro a quadro para cada um dos vídeos, assim como em [12].

Para efetuar-se o cálculo desse erro médio, foram utilizados gabaritos, contendo as posições corretas dos objetos em cada quadro de cada vídeo, marcadas manualmente. As posições esperadas dos objetos, estão organizadas em arquivos chamados de *ground truth*, disponíveis em [1], e foram marcadas somente de 5 em 5 frames. Dessa forma, foi feita uma interpolação linear para completar os gabaritos e determinar as posições esperadas em todos os outros frames, a fim de calcular-se o erro obtido em todos os quadros dos vídeos.

O erro médio foi calculado através das distâncias entre as posições retornadas pelo rastreador e as posições corretas presentes nos arquivos ground truth, e para

cada um dos vídeos, o programa de rastreamento foi executado 5 vezes, devido à aleatoriedade da escolha dos pixels utilizados na etapa de treinamento, o que pode causar diferenças de resultado entre as execuções. Dessa forma, para um mesmo vídeo, em cada uma das execuções, calculou-se o erro médio da localização do centro, e então, foi feita uma média desses valores, para obter-se um valor final.

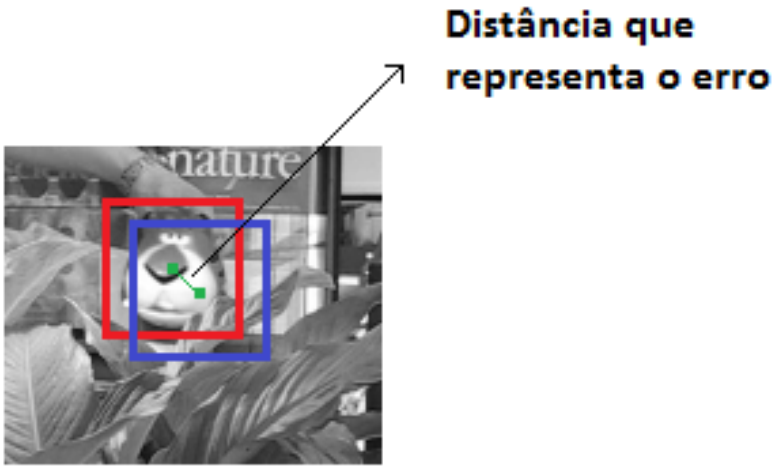

Figura 5.1: Distância entre o centro da localização do objeto retornada pelo rastreador e centro da localização do objeto correta, presente no arquivo *ground truth*.

5.1.2 Coeficiente de Jaccard

O Coeficiente de Jaccard [26], é uma métrica que avalia a porcentagem de interseção entre a área da janela retornada pelo rastreador e a área da janela da localização correta, descrita no *arquivo ground truth*.

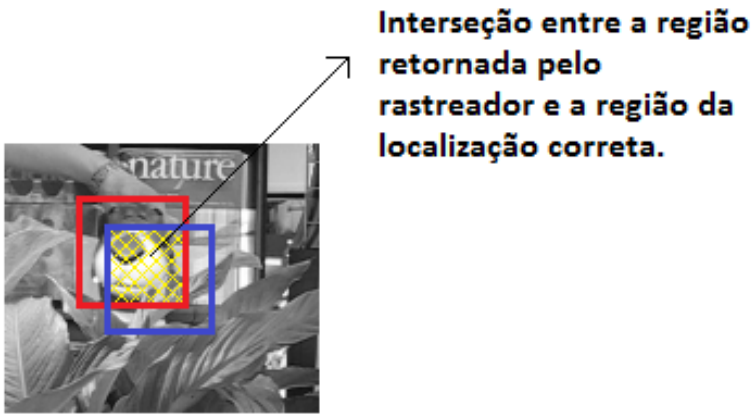

Figura 5.2: Região de interseção entre a região de localização retornada pelo rastreador e região de localização correta.

5.1.3 Precisão

A medida de precisão indica a porcentagem de frames em que o alvo foi rastreado corretamente. Esta é uma métrica importante, pois dependendo da aplicação, pode ser mais importante obter um erro muito pequeno na maior parte dos frames (mesmo que o rastreador se perca durante alguns frames), do que rastrear o objeto com um erro um pouco maior, em todos os frames do vídeo e sem se perder. Neste trabalho, considerou-se como rastreamento correto, os frames em que a distância entre o centro retornado pelo rastreador e o centro da localização correta foram menores que 20 pixels, assim como feito em [12].

5.2 Vídeos de Benchmark

O desempenho do rastreador foi avaliado a partir de um conjunto de vídeos contendo diversas dificuldades que devem ser enfrentadas pelo rastreador. Estes vídeos já foram utilizados para avaliar outros trabalhos, e por esse motivo, fornecem uma boa ideia da eficiência do rastreador desenvolvido, comparando-se com outros existentes.

Os vídeos utilizados para avaliação de desempenho do rastreador são os seguintes:

- **Tiger 1 e Tiger 2**
  Nos vídeos Tiger 1 e Tiger 2, o objeto a ser rastreado é um tigre de pelúcia que é movimentado por uma pessoa. Estes vídeos possuem alto nível de dificuldade, pois ocorrem muitas oclusões, há várias mudanças na forma como

o tigre se apresenta, e a velocidade de deslocamento nos quadros é grande.

Figura 5.3: Frames retirados do vídeo Tiger 1 - Neste vídeo ocorrem oclusões e mudanças na forma do objeto.

- **Coupon Book**
  Neste vídeo, inicialmente existem duas notas sobrepostas, e após a nota de cima ser dobrada, esta se desloca, deixando a outra nota visível. A dificuldade para o rastreador encontra-se justamente na nota que estava escondida e passou a aparecer após o deslocamento da primeira, pois a presença da segunda nota pode fazer com que o sistema passe a acompanhar o objeto errado.

Figura 5.4: Frames retirados do vídeo Coupon Book 1 - Dois objetos próximos e muito parecidos podem confundir o rastreador.

- **Occluded Face e Occluded Face 2**
  Estes vídeos apresentam duas pessoas que têm as suas faces escondidas parcialmente. O primeiro vídeo mostra uma oclusão parcial do rosto, através de uma resvista, e no segundo ocorre uma oclusão parcial do rosto por um livro, e uma oclusão quase que total do rosto, utilizando o livro e um chapéu, onde apenas os olhos se mantêm visíveis.

Figura 5.5: Frames retirados do vídeo Occluded Face - Neste vídeo, parte do rosto fica coberto por uma revista.

Figura 5.6: Frames retirados do vídeo Occluded Face 2 - Neste vídeo, parte do rosto fica encoberto por um livro, e em determinado momento, um chapéu é utilizado juntamente com o livro, para tentar atrapalhar o sistema de rastreamento.

- **Sylvester**

  Este vídeo apresenta uma pessoa movimentando um gato de pelúcia que deve ser rastreado. O desafio ocorre por conta de diversas mudanças no formato em que o objeto se apresenta.

Figura 5.7: Frames retirados do vídeo Sylvester - Diversas mudanças no formato, devido à movimentação do objeto pela pessoa.

- **David Indoor**

  Este vídeo mostra uma pessoa se deslocando em um ambiente com bastante variação de luminosidade, ao mesmo tempo em que há variação na distância entre a pessoa e a câmera do vídeo, onde em alguns momentos, há uma aproximação da câmera e em alguns momentos há um distanciamento da câmera. Além disso, ocorrem mudanças no formato do objeto rastreado, pois em alguns momentos, os óculos são retirados, ou a apresentação para a câmera passa a ser de perfil. O desafio é seguir o rosto, mesmo em um

ambiente com variação de luminosidade, e onde o objeto alvo modifica a sua forma e modifica a sua escala em relação ao ponto de observação.

Figura 5.8: Frames retirados do vídeo David Indoor - Variação de luminosidade, mudanças na forma e na escala do rosto.

5.3 Resultados

Evolução do Erro

As duas arquiteturas propostas para o rastreador, apresentadas no capítulo 4, foram testadas para o conjunto de vídeos de benchmark apresentado. Devido à aleatoriedade do sorteio dos pixels durante a etapa de treinamento, os resultados podem variar de uma execução para a outra, e então, 5 execuções de cada vídeo foram feitas utilizando cada uma das arquiteturas, e foi feita uma média dos resultados de cada rodada. Para testar as duas arquiteturas, todas as execuções de todos os vídeos, utilizaram os mesmos parâmetros de entrada. São estes:

Arquitetura 1 - *número de bits: 5*; *limiar para treinar um novo discriminador: 0,7*; *tamanho da fila de discriminadores: 6*; *área de busca: 12 pixels (Essa quantidade de pixels é utilizada para criar a janela de busca, aumentando-se o retângulo que envolve o alvo, com essa quantidade de pixels para cada um dos lados).*

Arquitetura 2 - *número de bits: 7*; *limiar de retreino de discriminador: 0,6*; *limiar de treino de novo discriminador: 0,35*; *tamanho da fila de discriminadores: 25*; *limite de retreinos de um mesmo discriminador: 3*; *área de busca: 10 pixels.*

Desta forma, a métrica utilizada foi o erro médio do centro do objeto ao longo dos frames. Os gráficos a seguir mostram a evolução do erro da localização do centro do objeto (medida em pixels) para cada um dos vídeos descritos na seção anterior, utilizando a arquitetura 1 e a arquitetura 2, comparando com o MILTrack [12].

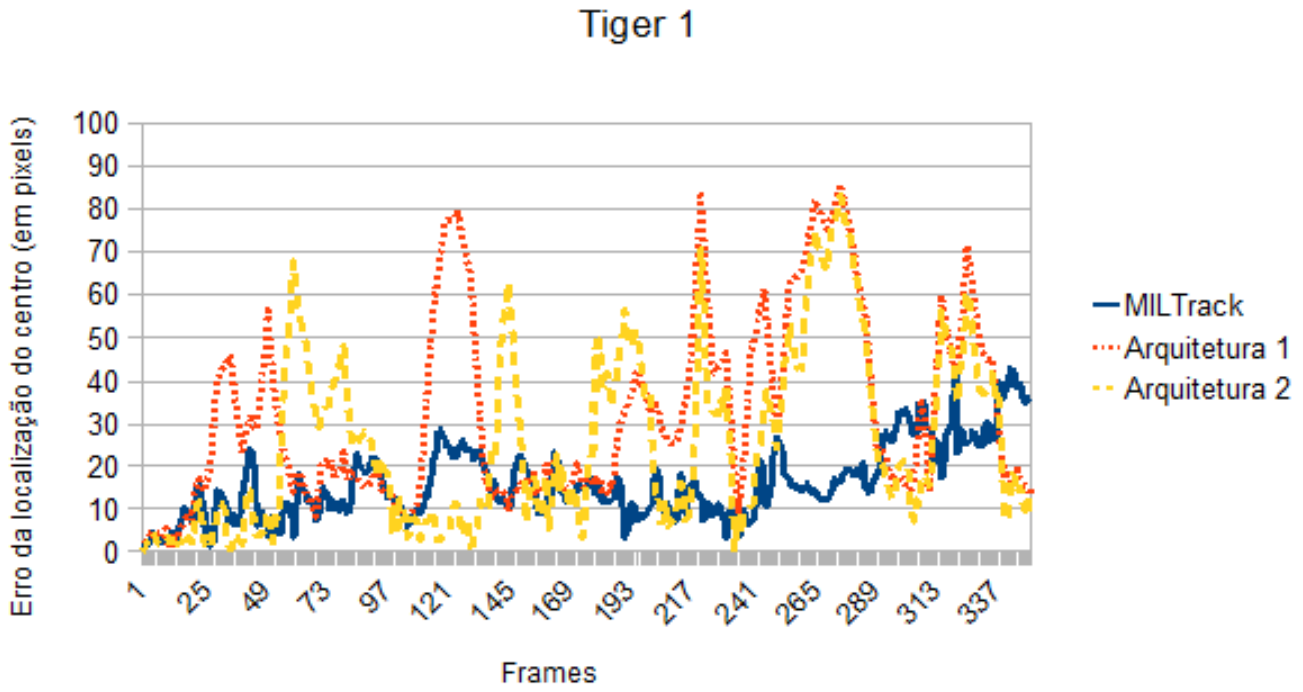

Figura 5.9: Evolução do erro para o vídeo Tiger1

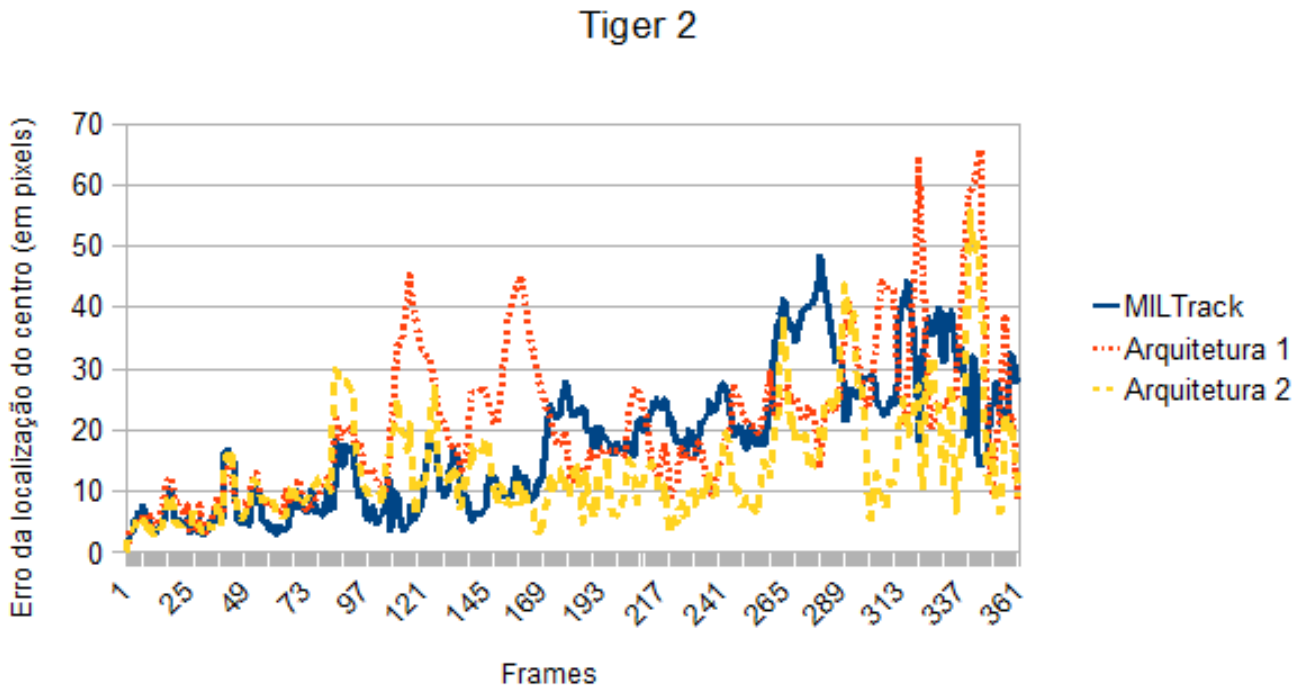

Figura 5.10: Evolução do erro para o vídeo Tiger2

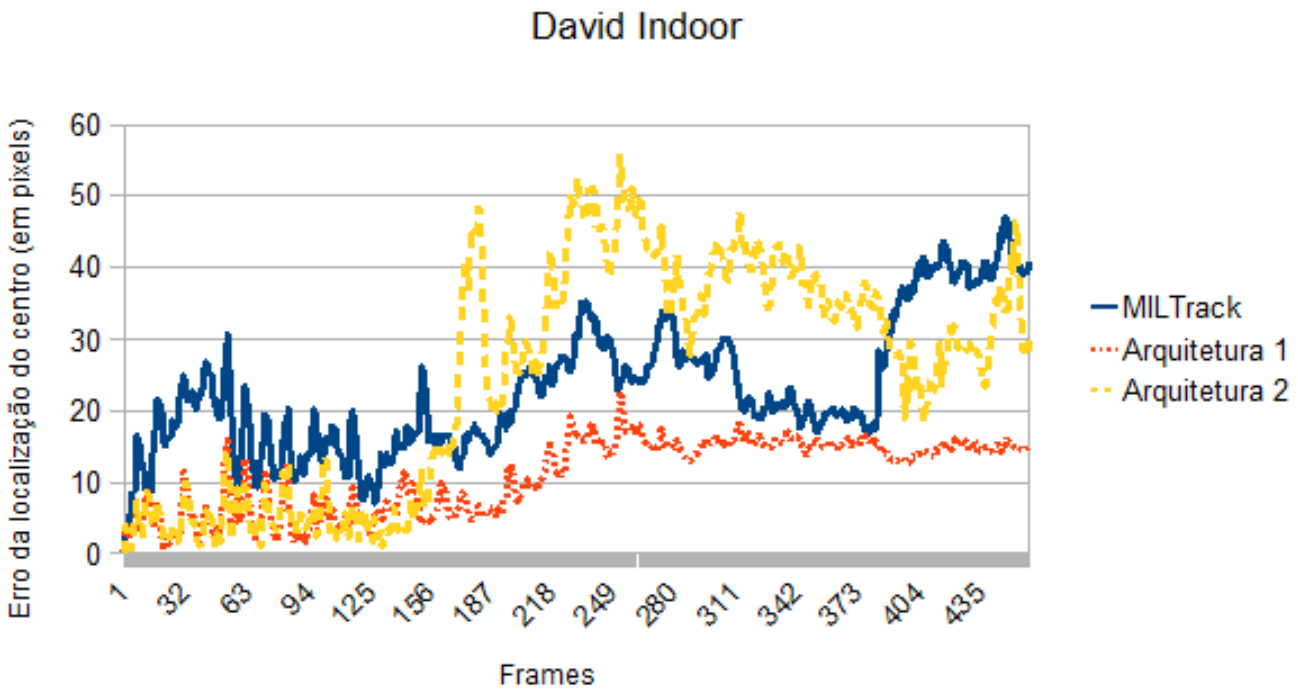

Figura 5.11: Evolução do erro para o vídeo David Indoor

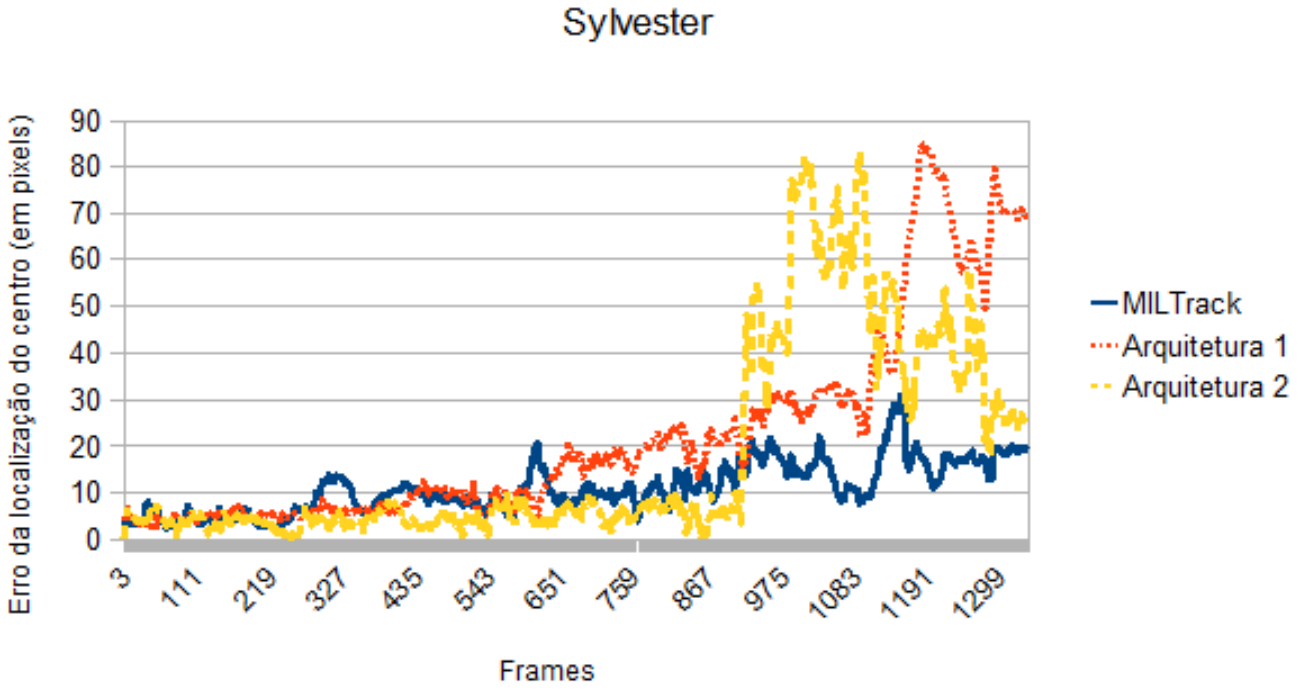

Figura 5.12: Evolução do erro para o vídeo Sylvester

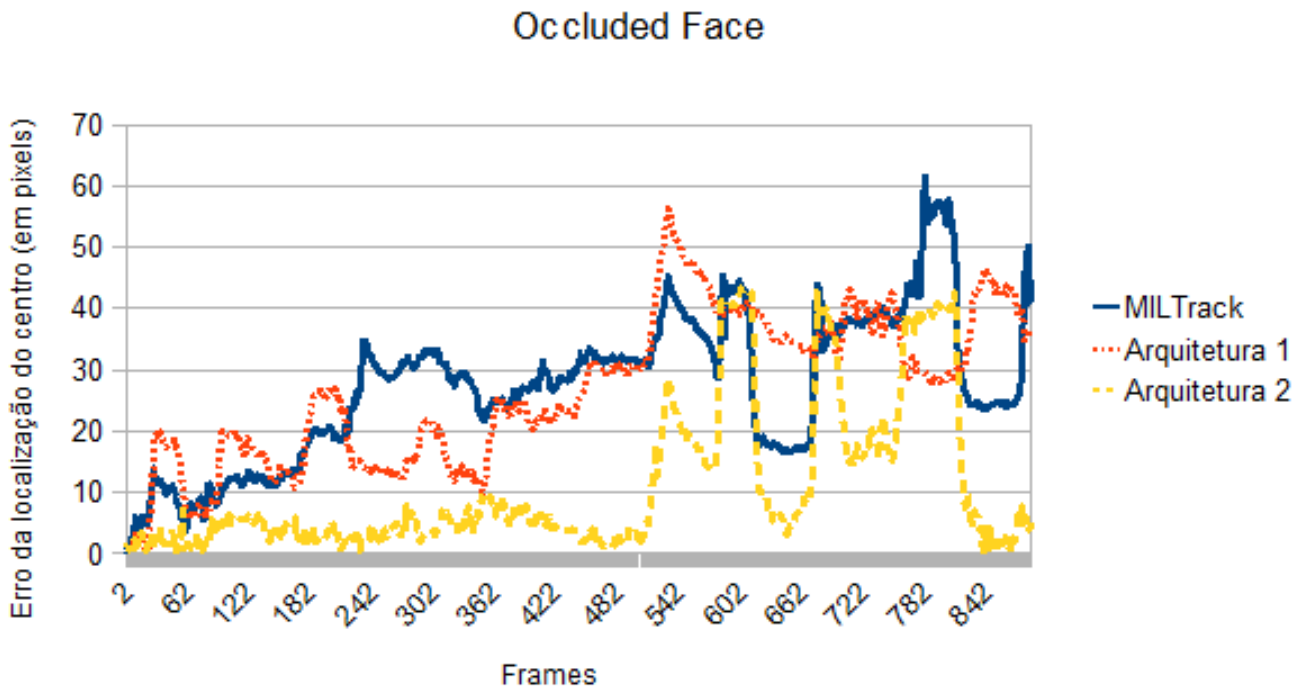

Figura 5.13: Evolução do erro para o vídeo Occluded Face

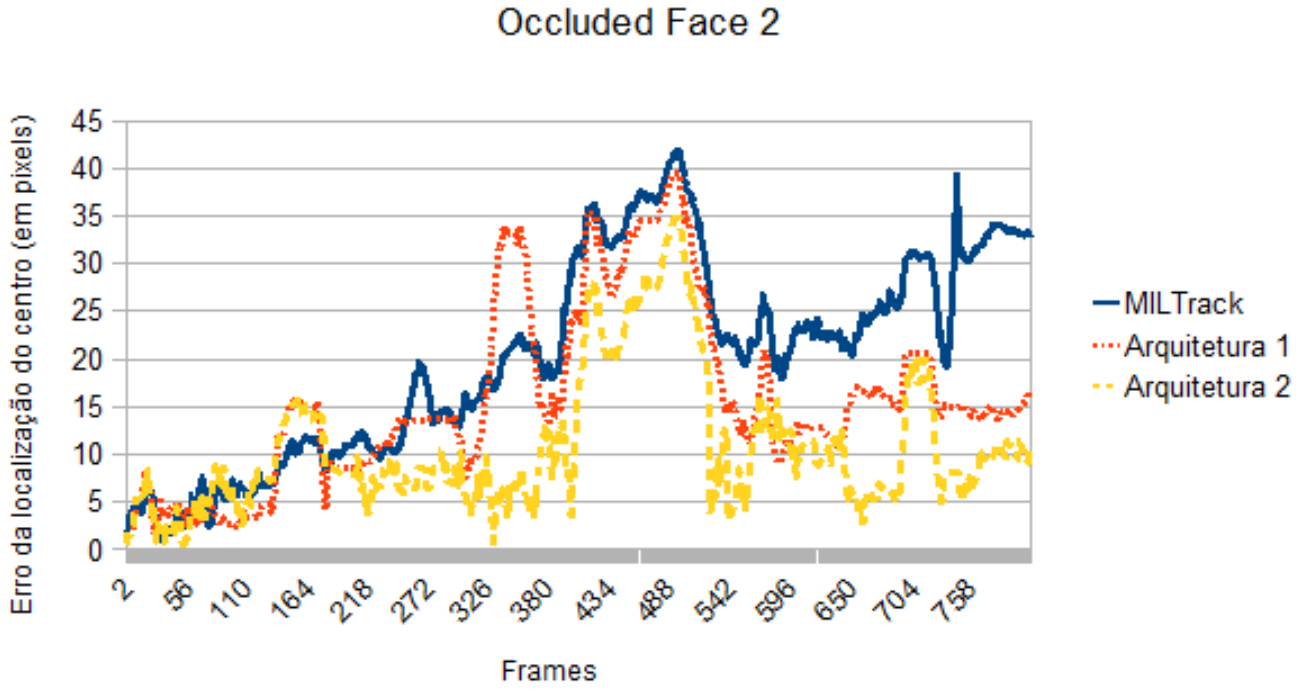

Figura 5.14: Evolução do erro para o vídeo Occluded Face 2

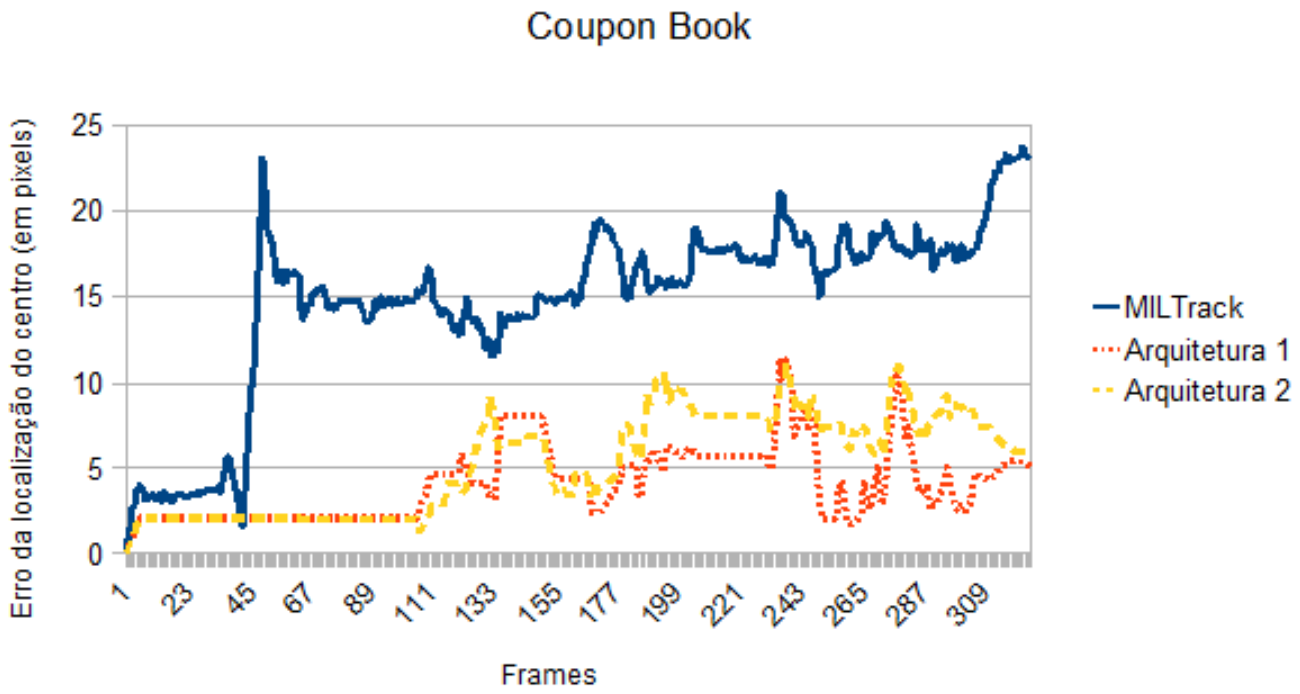

Figura 5.15: Evolução do erro para o vídeo couponBook

Estes resultados estão resumidos na tabela abaixo, juntamente com alguns resultados obtidos pelo TLD [14]:

Tabela 5.1: Erro médio da localização do centro do alvo (em pixels). Resultados marcados com '*' indicam a melhor performance e os marcados em negrito representam as segundas melhores performances.

Video Clip	*TLD*	*MILTrack*	*Arquitetura 1*	*Arquitetura 2*
Sylvester	6*	**11**	22	18
David Indoor	4*	23	**11**	25
Occluded Face	**15**	27	27	11*
Occluded Face 2	**13**	20	16	11*
Tiger 1	6*	**16**	33	25
Tiger 2	-	**18**	21	13*
Coupon Book	-	15	4*	**5**

Analisando os resultados, a Arquitetura 1 obteve 3 resultados melhores que o MILTrack, de um conjunto de 7, empatando em 1 e perdendo em 3, e quando comparado ao TLD, todos os resultados foram piores. Já a arquitetura 2 obteve 4 resultados melhores que o MILTrack, e 3 resultados piores; e comparando-se com o TLD, superou os resultados em 2 dos 5 vídeos.

Ajuste de Parâmetros

Os resultados anteriores foram obtidos utilizando-se parâmetros iguais de entrada para todos os vídeos, e então, foram realizados testes para tentar melhorar o desempenho do rastreador, utilizando parâmetros ajustados para cada um dos vídeos. Estes ajustes de parâmetros foram feitos com o objetivo de testar o potencial de melhora do rastreador, fornecendo uma motivação para estudos futuros sobre como automatizar a determinação de parâmetros de forma a melhorar o desempenho do rastreador. Estes ajustes de parâmetros foram feitos manualmente, e os parâmetros *default* e *ajustados* utilizados para as Arquiteturas 1 e 2 estão listados nas tabelas a seguir:

Tabela 5.2: *Arquitetura 1* - Parâmetros default e parâmetros ajustados usados para cada um dos vídeos. Os vídeos marcados com '*' indicam que uma subtração de background também faz parte dos parâmetros ajustados. *Bits* é o número de bits utilizado nos discriminadores, *Novo disc.* representa o limiar para treinamento de um novo discriminador, *Tam da fila* indica o tamanho da fila de discriminadores e *Busca* representa o número de pixels utilizados para determinar a região de busca.

Video	*Bits*	*Novo disc.*	*Tam da fila*	*Busca*
Default params.	*5*	*0.7*	*6*	*12*
Tiger1*	default	0.35	20	14
Tiger2*	default	0.35	20	16
Occluded Face	3	0.5	10	10
Occluded Face 2	3	0.5	10	10
David Indoor	6	default	default	10
Sylvester*	3	0.8	default	5
Coupon Book	default	default	default	default

Uma subtração de background foi feita para os vídeos marcados com * para auxiliar na detecção do objeto rastreado, pois em alguns momentos, o objeto acaba se deslocando para fora da região de busca.

Tabela 5.3: *Arquitetura 2* - Parâmetros default e parâmetros ajustados usados para cada um dos vídeos. Os vídeos marcados com '*' indicam que uma subtração de background também faz parte dos parâmetros ajustados. *Bits* é o número de bits utilizado nos discriminadores, *Retreino* indica o limiar de retreino, *Novo Disc* indica o limiar para treino de novo discriminador, *Tam da fila* representa o tamanho da fila de discriminadores, *Lim. Retreinos* é o número máximo de retreinos que é permitido para cada discriminador e *Busca* é o número de pixels utilizados para determinar a região de busca ao redor do objeto.

Video	*Bits*	*Retreino*	*Novo Disc*	*Tam da fila*	*Lim. Retreinos*	*Busca*
Default params.	*7*	*0.6*	*0.35*	*25*	*3*	*10*
Tiger1*	default	default	0.4	6	4	20
Tiger2*	default	default	default	default	default	default
Occluded Face	5	default	default	default	default	default
Occluded Face 2	5	default	default	default	default	default
David Indoor	6	0.7	0.45	10	default	14
Sylvester*	3	0.8	0.6	30	default	8
Coupon Book	default	default	default	default	default	default

Os resultados melhorados após a realização do ajuste de parâmetros estão listados na tabela abaixo, e mostram que existe um potencial de melhora para o rastreador, que depende de pesquisas futuras para obter as melhores combinações de parâmetros automaticamente.

Tabela 5.4: Erro médio da localização do centro do alvo (em pixels). Resultados marcados com '*' indicam a melhor performance, enquanto os marcados em negrito representam as segundas melhores performances. *Arq 1** e *Arq 2** representam as arquiteturas 1 e 2 utilizadas com os parâmetros ajustados.

Video Clip	*TLD*	*MILTrack*	*Arq 1*	*Arq 2*	*Arq 1**	*Arq 2**
Sylvester	6*	11	22	18	**8**	11
David Indoor	4*	23	11	25	**8**	12
Occluded Face	15	27	27	**11**	12	7*
Occluded Face 2	13	20	16	**11**	9*	9*
Tiger 1	6*	16	33	25	11	**10**
Tiger 2	-	18	21	13	10*	**11**
Coupon Book	-	15	4*	**5**	4*	**5**

Os gráficos a seguir mostram uma comparação da evolução do erro para as arquiteturas 1 e 2, utilizando parâmetros default e parâmetros ajustados. As comparações das arquiteturas utilizando parâmetros default e parâmetros ajustados mostram que sempre foi possível melhorar os resultados. Dessa forma, o ajuste

automático de parâmetros é um grande desafio a ser enfrentado para possibilitar futuras melhorias no rastreador baseado em WiSARD e memórias de prazo.

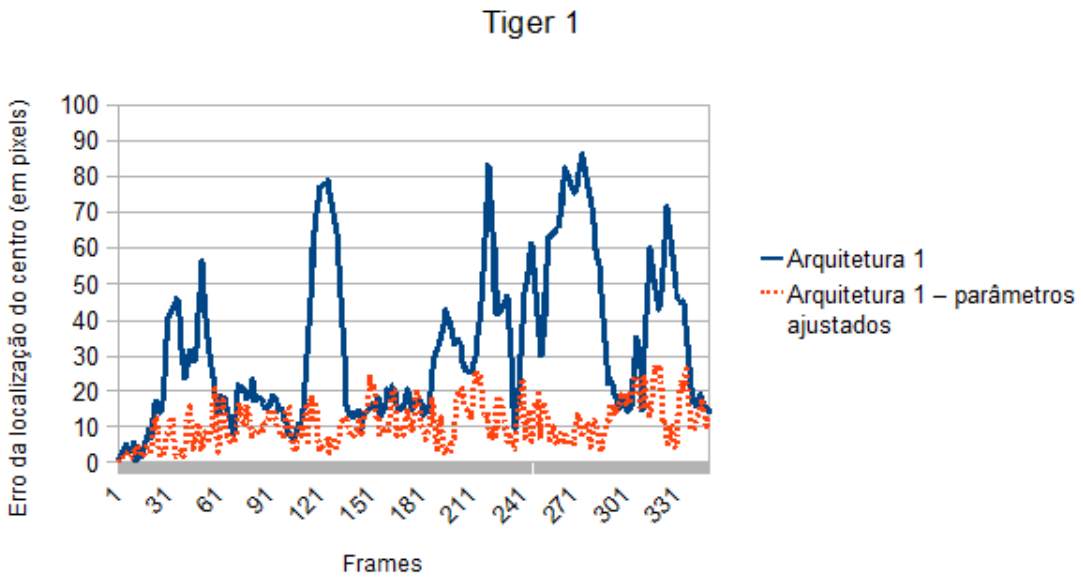

Figura 5.16: Evolução do erro para o vídeo *Tiger 1* utilizando a Arquitetura 1

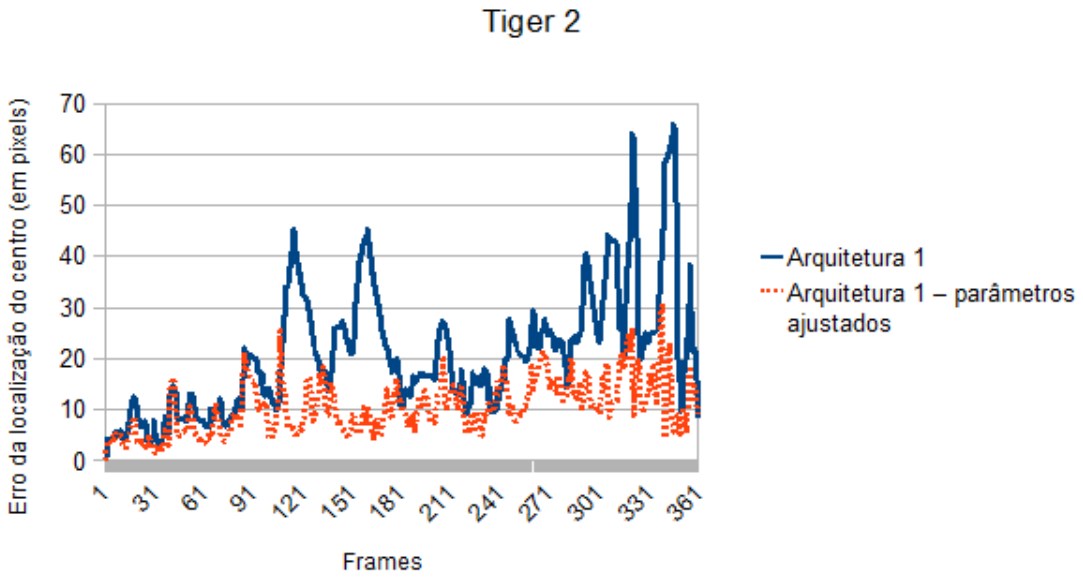

Figura 5.17: Evolução do erro para o vídeo *Tiger 2* utilizando a Arquitetura 1

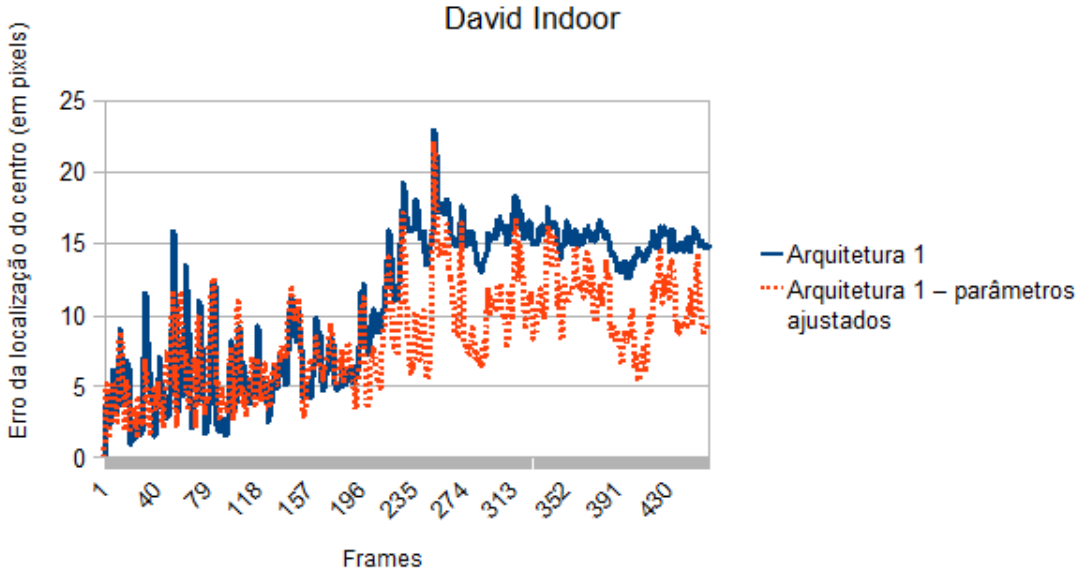

Figura 5.18: Evolução do erro para o vídeo *David Indoor* utilizando a Arquitetura 1

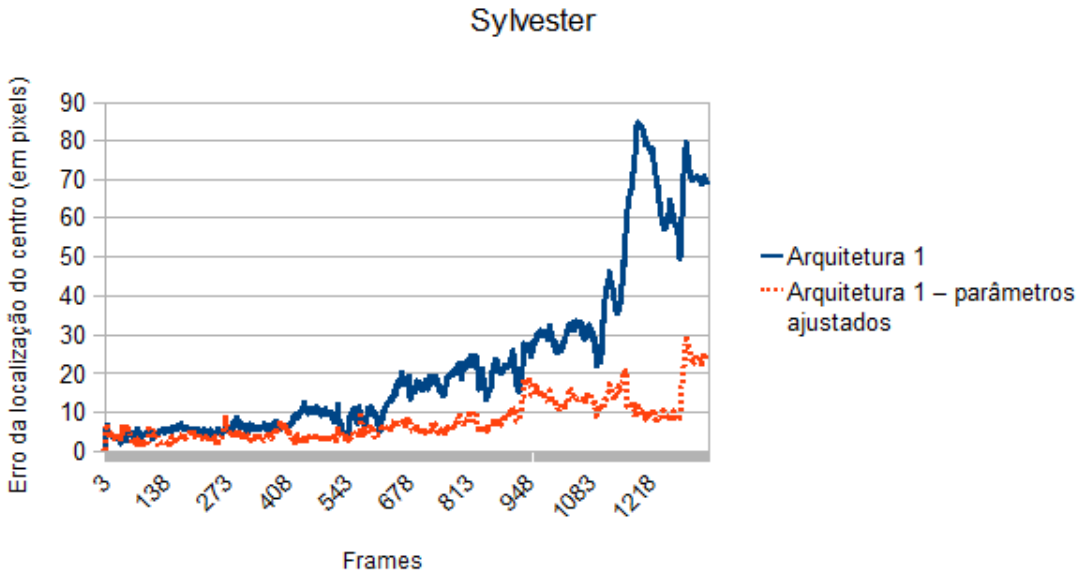

Figura 5.19: Evolução do erro para o vídeo *Sylvester* utilizando a Arquitetura 1

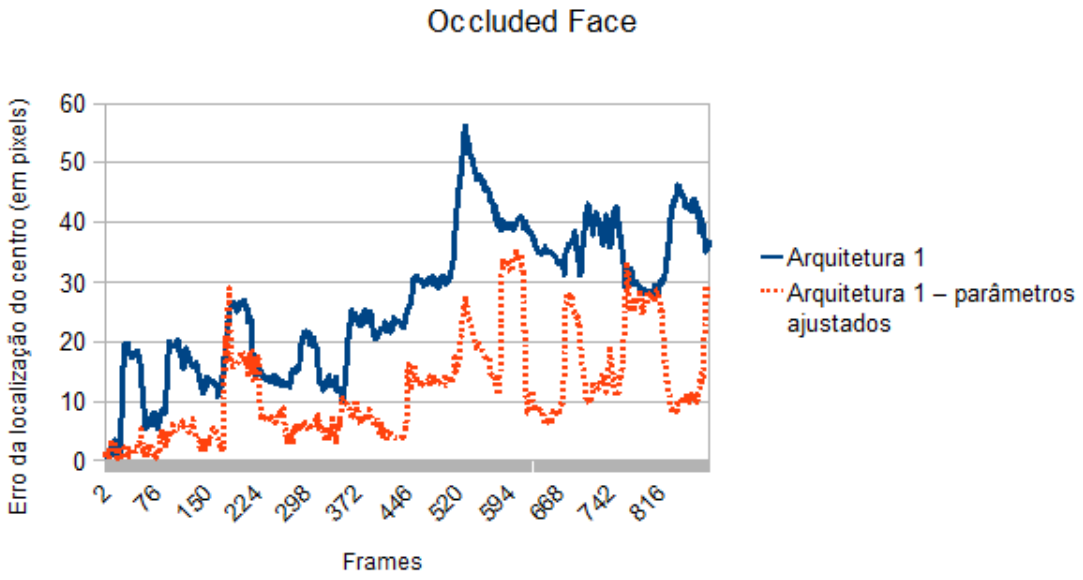

Figura 5.20: Evolução do erro para o vídeo *Occluded Face* utilizando a Arquitetura 1

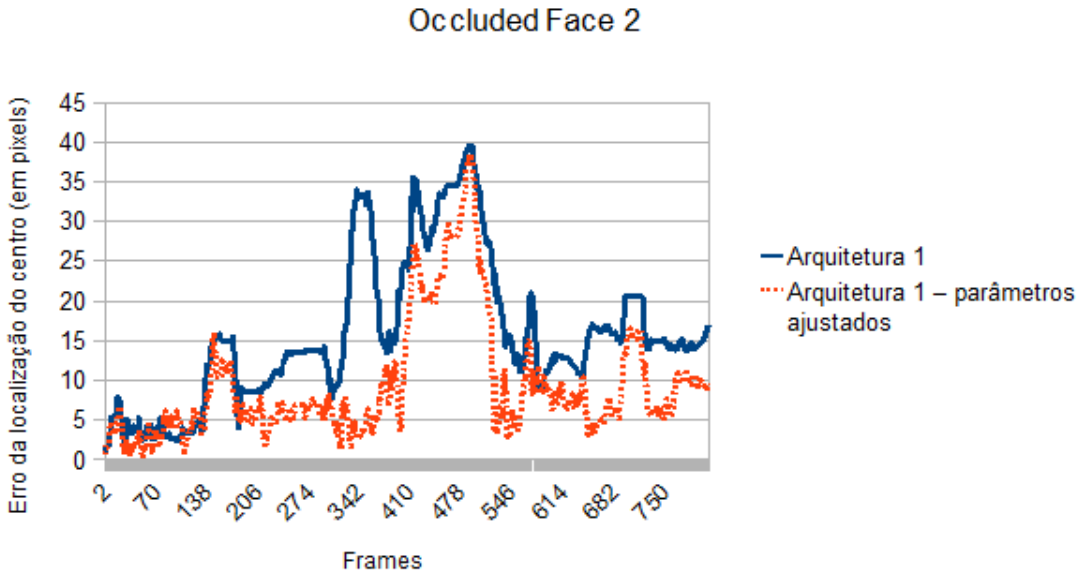

Figura 5.21: Evolução do erro para o vídeo *Occluded Face 2* utilizando a Arquitetura 1

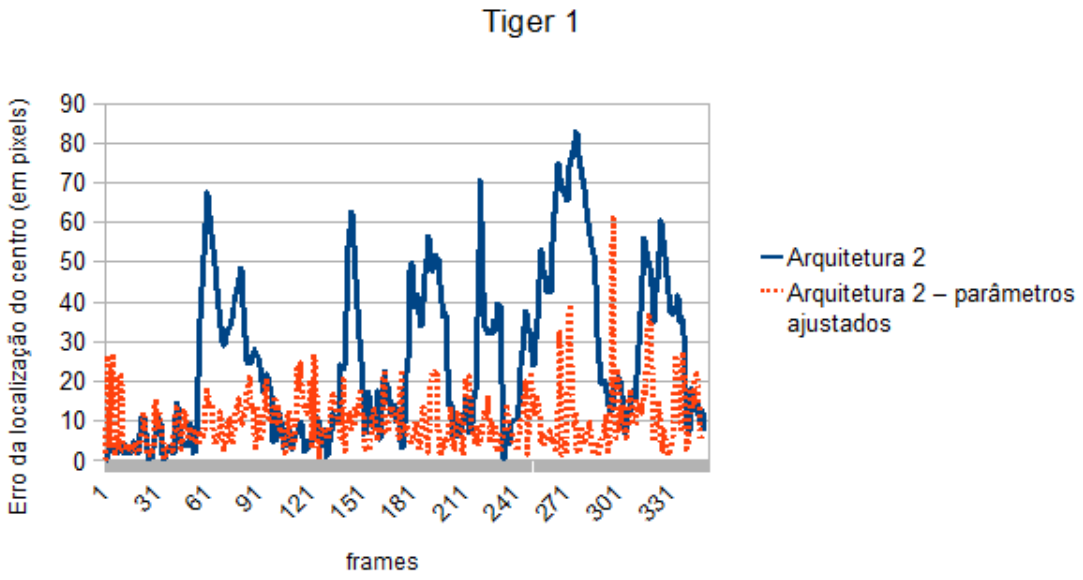

Figura 5.22: Evolução do erro para o vídeo *Tiger 1* utilizando a Arquitetura 2

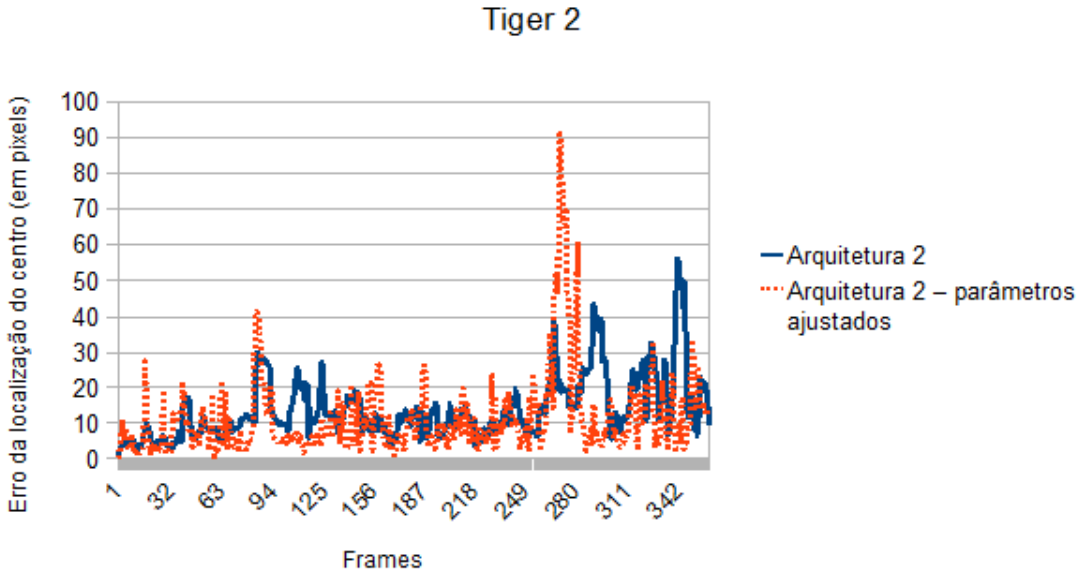

Figura 5.23: Evolução do erro para o vídeo *Tiger 2* utilizando a Arquitetura 2

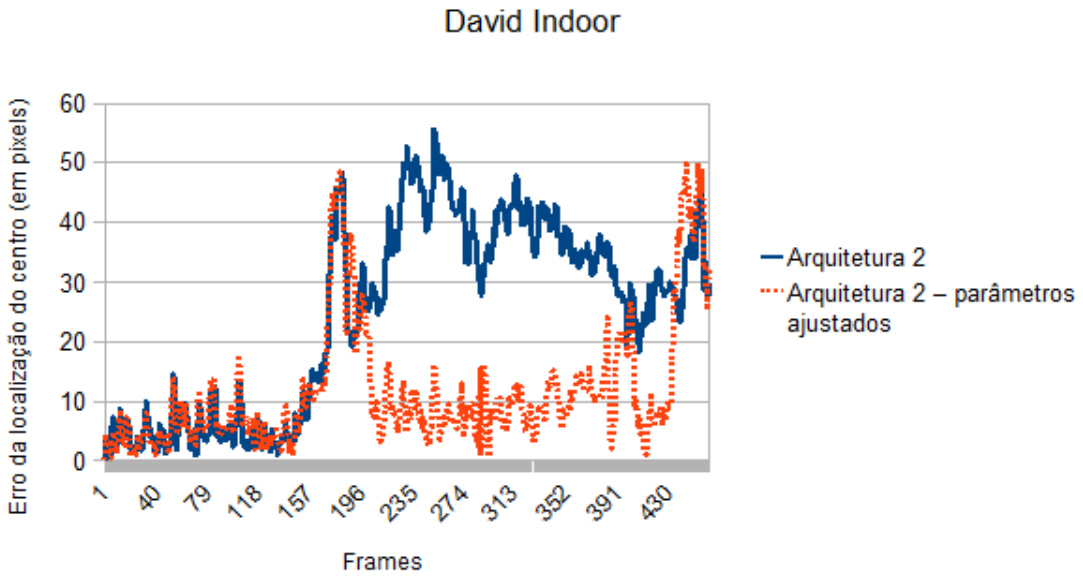

Figura 5.24: Evolução do erro para o vídeo *David Indoor* utilizando a Arquitetura 2

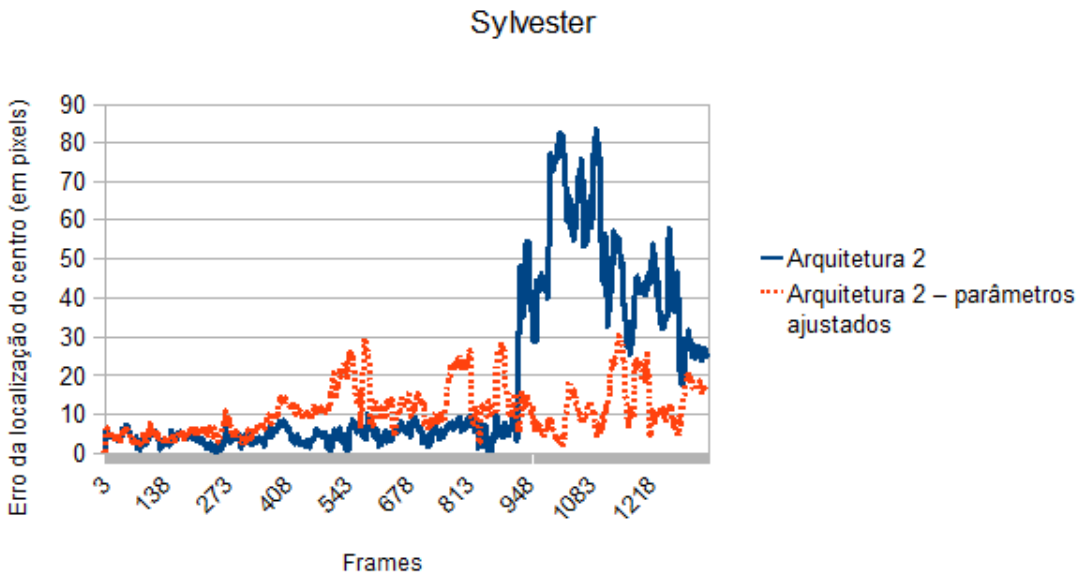

Figura 5.25: Evolução do erro para o vídeo *Sylvester* utilizando a Arquitetura 2

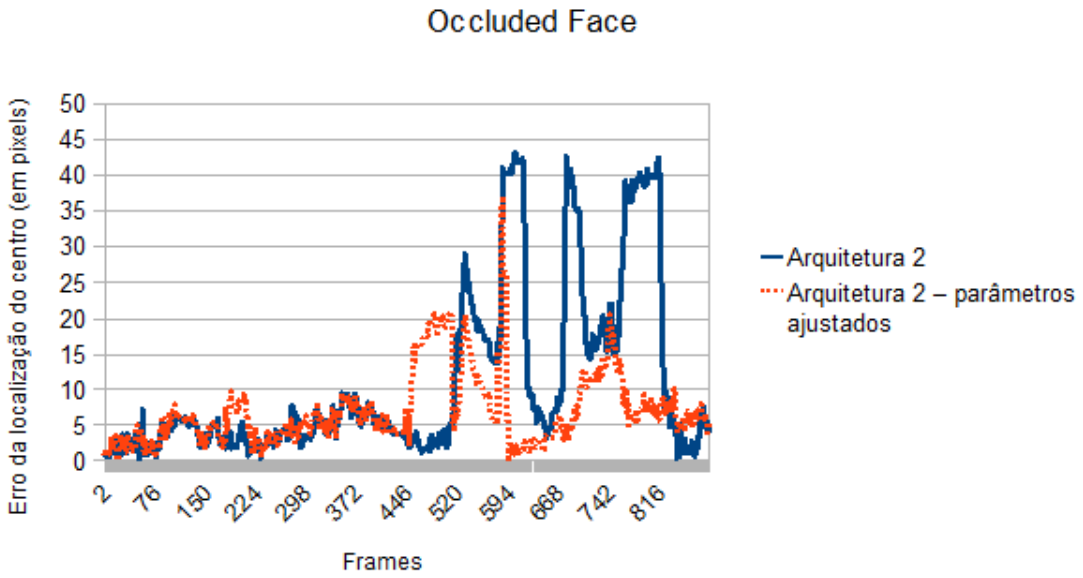

Figura 5.26: Evolução do erro para o vídeo *Occluded Face* utilizando a Arquitetura 2

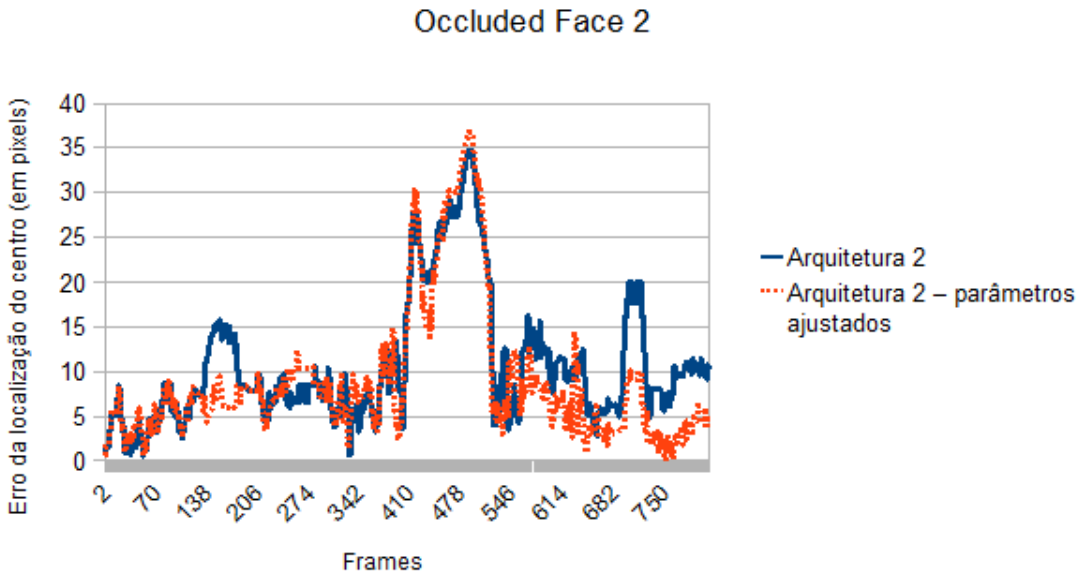

Figura 5.27: Evolução do erro para o vídeo *Occluded Face 2* utilizando a Arquitetura 2

Como pode ser observado, o rastreador que utiliza a Arquitetura 1, conseguiu melhorar os resultados em todas as execuções dos vídeos, quando realizado um ajuste de parâmetros (o vídeo Coupon Book não está representado nos gráficos, pois os melhores resultados foram obtidos utilizando os parâmetros default). O mesmo ocorreu para a Arquitetura 2 que também melhorou o seu desempenho ajustando-se os parâmetros individualmente para cada vídeo. Estes resultados mostram o

potencial do rastreador baseado em memórias de prazo e WiSARD, que pode ser aprimorado através de métodos para tentar ajustar os parâmetros automaticamente e conseguir obter um rastreamento mais confiável dos objetos.

Precisão

O cálculo da precisão mostra a porcentagem de frames em que ocorreu um rastreamento considerado correto, em relação ao total de frames processados. A tabela abaixo mostra que as arquiteturas 1 e 2 obtiveram bons resultados em alguns dos vídeos, porém, sempre foi possível melhorar a precisão realizando um ajuste de parâmetros.

Tabela 5.5: Tabela de Precisão - * indica a arquitetura utilizada com parâmetros ajustados individualmente para cada vídeo.

Video Clip	*MILTrack*	*Arq 1*	*Arq 2*	*Arq 1**	*Arq 2**
Sylvester	0.90	0.65	0.69	0.94	0.88
David Indoor	0.52	0.89	0.38	0.98	0.85
Occluded Face	0.43	0.33	0.83	0.84	0.97
Occluded Face 2	0.60	0.75	0.85	0.8	0.89
Tiger 1	0.81	0.44	0.51	0.89	0.90
Tiger 2	0.83	0.63	0.82	0.93	0.88
Coupon Book	0.69	1.00	1.00	1.00	1.00

As arquiteturas 1 e 2 obtiveram resultados excelentes para todos os vídeos rastreados quando ocorreu o ajuste de parâmetros, com a menor precisão igual a 80% para a Arquitetura 1 e 85% para a Arquitetura 2, e ambas chegaram a obter 100% de acerto no vídeo *Coupon Book*.

5.3.1 Frames por Segundo - FPS

O sistema de rastreamento que armazena diversos discriminadores WiSARD se mostrou bastante adequado para este tipo de problema, pois além de bons resultados de rastreamento, o sistema conseguiu uma boa taxa de frames por segundo. Os resultados obtidos para a Arquitetura 1 estão na tabela abaixo:

Tabela 5.6: Tabela de FPS

Video Clip	*FPS*
Sylvester	87
David Indoor	22
Occluded Face	17
Occluded Face 2	28
Tiger 1	45
Tiger 2	43
Coupon Book	21

Algumas demonstrações do rastreador em funcionamento utilizando a Arquitetura 1 com parâmetros ajustados, podem ser vistas na página do Laboratório de Inteligência Artificial da UFRJ - LABIA [27].

5.4 Outras Aplicações

Esta seção se propõe a mostrar outras aplicações que podem utilizar o rastreador de objetos baseado em memórias de prazo. São aplicações em tempo real que servem para testar o funcionamento do rastreador em situações diferentes das encontradas nos vídeos de benchmark. Como os métodos propostos foram eficientes para rastrear um único objeto em uma determinada cena propôs-se expandir o modelo para rastrear mais de um objeto em um único vídeo. Dessa forma, cada objeto rastreado deve ter uma fila própria de discriminadores, e todas as filas devem ser atualizadas em tempo real, através dos treinamentos. A localização inicial de cada um dos alvos deve ser passada como entrada para o sistema, e a partir desse momento, os treinamentos dos discriminadores e as atualizações de cada uma das filas ocorrem da mesma maneira que foi descrita na Arquitetura 1 e na Arquitetura 2.

5.4.1 Aplicação em Tempo Real para Controle de Mouse Através do Rosto

Um dos experimentos realizados foi o desenvolvimento de um controlador de mouse, através de comandos enviados pelo rosto, sem a necessidade de utilização das mãos. O sistema utiliza imagens capturadas por uma câmera e através do rastreamento do rosto e dos olhos de uma pessoa, é capaz de extrair comandos para controlar o mouse. Essa aplicação além de mostrar a viabilidade da utilização do modelo proposto em uma aplicação de tempo real, onde existe a necessidade de uma resposta rápida; também é uma aplicação muito útil de inclusão social, pois facilita a interação

de pessoas com necessidades especias, permitindo o acesso com maior facilidade e independência.

Cliques dos botões

Inicialmente, deve-se passar para o sistema, a localização do rosto e de cada um dos olhos do usuário, a fim realizar o rastreamento de três objetos diferentes.

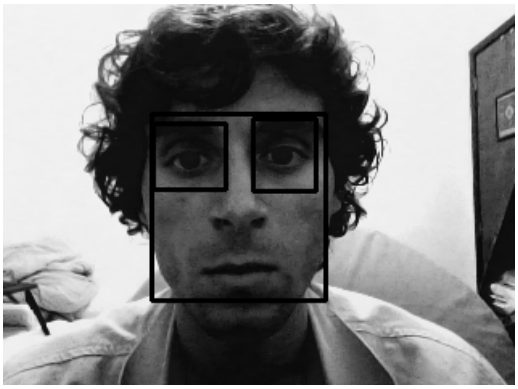

Figura 5.28: Início do rastreamento: As localizações dos olhos e do rosto devem ser informadas.

Neste sistema existem 3 filas de discriminadores: uma para o olho esquerdo, uma para o olho direito e uma para o rosto. O usuário deve inicializar o sistema com os olhos abertos, e dessa forma, o primeiro discriminador presente na fila de discriminadores do olho esquerdo, representa o olho esquerdo aberto; e o primeiro discriminador presente na fila de discriminadores do olho direito, representa o olho direito aberto.

Após o início do rastreamento com os olhos abertos, o usuário pode fechar os olhos. Quando o olho esquerdo for fechado, um novo discriminador será treinado e armazenado na fila de discriminadores do olho esquerdo, ocorrendo o mesmo para o olho direito. Então, pode-se determinar quando os olhos estão abertos ou fechados, pois quando o primeiro discriminador da fila do olho esquerdo retorna a maior pontuação, significa que o olho esquerdo encontra-se aberto, e quando o segundo discriminador retorna a maior pontuação, significa que o olho esquerdo encontra-se fechado. De maneira análoga, ocorre a classificação para o olho direito.

As filas de discriminadores para os olhos são limitadas para armazenar apenas dois discriminadores, um representando o olho fechado e outro representando o olho aberto, e as situações de olho aberto ou fechado são convertidas em cliques do mouse (botão esquerdo ou direito, dependendo do olho). Quando ocorrer uma situação onde o olho esquerdo fique fechado durante 10 frames seguidos, o comando para clique do botão esquerdo do mouse é disparado e quando o olho direito se mantiver fechado

durante 10 frames seguidos, o clique do botão direito é disparado. Essa medida de 10 frames foi adotada para que um comando de clique de botão não seja confundido com uma piscada normal de uma pessoa, evitando que algum clique seja disparado acidentalmente.

Figura 5.29: Cliques: Olho esquerdo fechado indica um comando para disparar o clique do botão esquerdo do mouse, e olho direito fechado indica um comando para disparar o botão direito do mouse.

Movimentação da seta

Os comandos de direção da seta do mouse, são enviados através da posição em que o rosto da pessoa se encontra em relação à posição inicial. Se o rosto virar mais para a direita da posição inicial, a seta do mouse se desloca para a direita; se o rosto virar mais para a esquerda da posição inicial, a seta se desloca para a esquerda; se o rosto se deslocar para cima, a seta do mouse move-se para cima; e se o rosto se deslocar para baixo; a seta se move para baixo.

Esses comandos de movimento foram extraídos apenas com base na informação da localização do rosto, utilizando-se o deslocamento em pixels do rosto em relação à posição inicial de treinamento. Adotou-se um padrão de 7 pixels de deslocamento em relação ao centro da posição inicial do rosto, a fim de considerar-se que houve um deslocamento com intenção de movimentar o mouse.

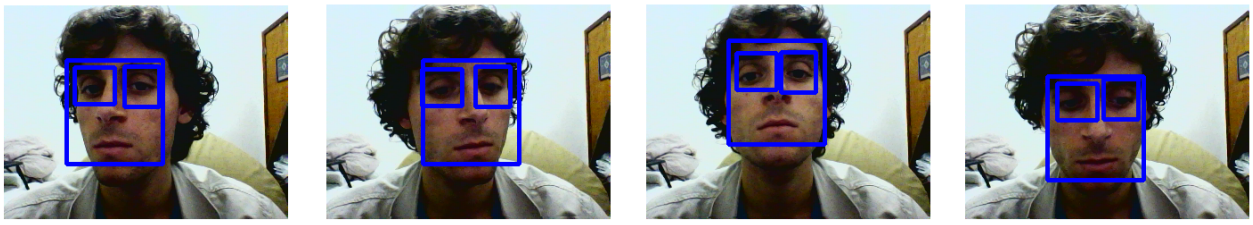

Figura 5.30: Movimentação: A seta do mouse se move quando ocorre um deslocamento do rosto de em relação à posição inicial.

Uma demonstração do sistema desenvolvido pode ser acessada em [27].

5.4.2 Identificação de Poses do Rosto Através de Componentes

Esta aplicação tem o objetivo rastrear cada um dos olhos e a boca de uma pessoa separadamente, e com base nas informações de localização, extrair algumas regras que auxiliem na identificação da pose do rosto. Este sistema de identificação de poses é um sistema neuro-simbólico [28], pois utiliza as saídas de um sistema neural, para extrair regras lógicas que são utilizadas para definir a pose do rosto (face inclinada para a esquerda ou para a direita, e rosto se afastando ou se aproximando da câmera).

Raciocínio Neuro-Simbólico para Determinação da Pose

Algumas regras lógicas para determinar a pose do rosto foram definidas com base nas saídas do sistema de rastreamento baseado em redes neurais sem peso:

- **Aproximação da câmera**
  Quando os retângulos que delimitam cada um dos olhos e a boca, se afastam, significa que o rosto aproximou-se da câmera, pois quando o rosto se aproxima, a distância entre os centros de cada olho e da boca, aumentam na imagem. Então, a tendência é que o sistema de rastreamento retorne localizações mais afastadas para os olhos e boca, em relação às distâncias iniciais.

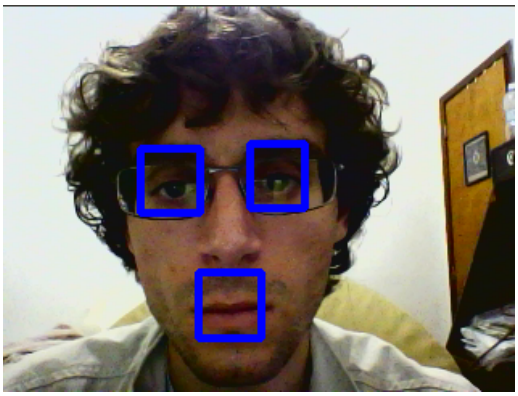

Figura 5.31: Aproximação da câmera.

- **Afastamento da câmera**
  Quando os retângulos que delimitam cada um dos olhos e a boca aproximam-se, significa que o rosto afastou-se da câmera, pois quando o rosto se afasta, a distância entre os centros de cada olho e da boca, diminuem na imagem. Então, a tendência é que o sistema de rastreamento retorne localizações mais próximas para os olhos e boca, em relação às distâncias iniciais.

Figura 5.32: Afastamento da câmera.

- **Inclinação da face para a direita**
  Caso as posições relativas entre os dois olhos e boca se modifiquem, significa que ocorreu uma rotação da face. Se o olho esquerdo ficou acima do olho direito, ocorreu um caso de rotação da face para a direita.

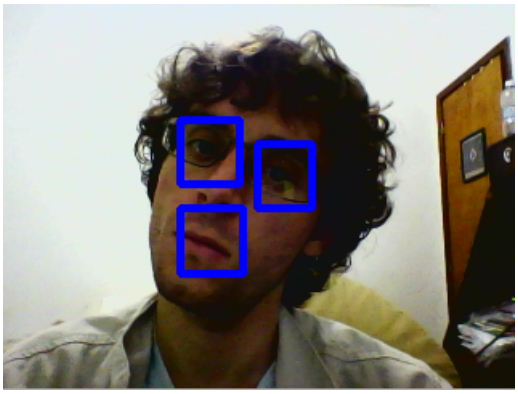

Figura 5.33: Inclinação da face para a direita.

- **Inclinação da face para a esquerda**
  Se em algum momento, o olho direito passar a localizar-se acima do olho esquerdo, significa que hove um caso de rotação da face para a esquerda.

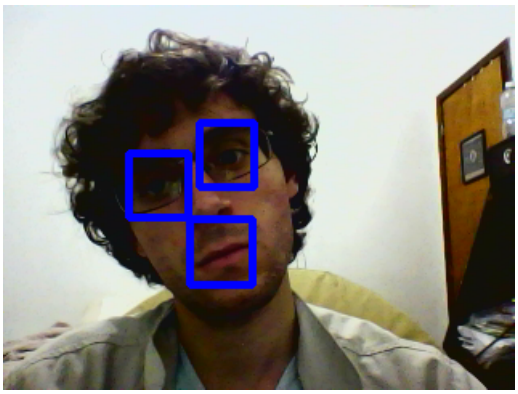

Figura 5.34: Inclinação da face para a esquerda.

Capítulo 6

Conclusão e Trabalhos Futuros

Este trabalho apresentou duas propostas de arquitetura para um rastreador de objetos genéricos. Ambas as arquiteturas são baseadas em memórias de prazo e utilizam o modelo de redes neurais sem peso WiSARD. Os resultados apresentados no capítulo anterior, mostraram que o rastreador de objetos genéricos conseguiu bons resultados nos vídeos de benchmark e também em aplicações de tempo real, onde foi necessário rastrear múltiplos objetos em um mesmo vídeo.

O modelo WiSARD foi muito importante para o sucesso do rastreador, devido à sua velocidade tanto na etapa de treinamento, quanto na etapa de classificação, e juntamente com os modelos baseados em memórias de prazo, foi possível armazenar diversos discriminadores WiSARD, treinados em tempo real e que representam diferentes vistas do objeto rastreado, possibilitando assim, recuperar memórias antigas a fim de realizar o rastreamento de forma adequada, mesmo que ocorram oclusões ou modificações no formato do objeto perseguido.

Para melhorar o desempenho do rastreador, alguns trabalhos futuros podem ser desenvolvidos, como por exemplo, desenvolver um método para ajuste automático dos parâmetros, de acordo com o tipo de objeto que se deseja rastrear e tipo de cena em que o objeto se encontra. Além disso, pode-se desenvolver pesquisas a fim de tentar melhorar a estrutura de armazenamento dos discriminadores, como por exemplo, rever qual o discriminador que vai ser descartado quando a fila de memórias estiver cheia (no caso da Arquitetura 1), dar pesos diferentes para os discriminadores mais antigos, ou até mesmo, criar diversas filas de discriminadores ao longo do tempo de rastreamento, para tentar definir uma melhor forma de escolha do discriminador que classifica corretamente o objeto.

No capítulo de experimentos, foi mostrada uma aplicação de determinação de poses do rosto através do rastreamento dos seus componentes, e foi possível identificar quando um rosto se aproxima ou se afasta da câmera e essa é uma informação útil que pode desenvolver outras técnicas para melhorar o rastreamento. A partir da informação de aproximação ou afastamento, um trabalho futuro que pode

ser desenvolvido é o de utilizar as arquiteturas 1 e 2, com uma modificação para armazenar filas de discriminadores de tamanho diferentes. Assim, caso o rosto se aproxime, uma fila de discriminadores maiores pode ser utilizada e quando o rosto se afastar, pode ser utilizada uma fila de discriminadores menores. Dessa forma, existem pesquisas para melhorar a precisão do rastreador, considerando-se as mudanças de escala. Ainda utilizando-se da possibilidade de rastreamento de múltiplos objetos, é possível desenvolver métodos para identificar poses de outros objetos, e para isso, basta saber quais partes do objeto deve-se rastrear separadamente e identificar relações entre as partes rastreadas.

Uma das principais motivações para o desenvolvimento deste trabalho, foi a possibilidade de utilização de uma ferramenta de rastreamento de objetos, em diversas áreas e em diversas aplicações que podem ser úteis para a sociedade. Dessa forma, existem diversos possíveis trabalhos futuros que podem utilizar o rastreador para resolver algum tipo de problema. Um trabalho futuro muito importante e muito motivador, seria o desenvolvimento de aplicações de inclusão social, que facilitem a vida de pessoas com necessidades especiais.

Neste trabalho, foi desenvolvida uma primeira ferramenta com este objetivo, que foi o controlador de mouse através de comandos extraídos do rosto de uma pessoa, e um possível trabalho futuro, seria utilizar o rastreador desenvolvido para acompanhar os olhos de uma pessoa, utilizando-se uma câmera de maior resolução, a fim de controlar um computador somente através dos olhos, sem a necessidade de utilizar movimentos de cabeça, como feito neste trabalho.

Outra área de pesquisa interessante, que poderia utilizar dos benefícios do rastreador, é a área de reconhecimento de gestos. Pode-se utilizar o trabalho desenvolvido para rastrear as mãos de uma pessoa, e em conjunto com algum sistema de reconhecimento, identificar diferentes gestos. Desta forma, é possível criar sistemas que entendam comandos enviados pelas mãos, ou até mesmo que ententam alguma linguagem de sinais, como LIBRAS. Existem ainda diversas áreas e uma infinidade de aplicações que podem ser desenvolvidas a partir deste trabalho.

Neste trabalho foi desenvolvida uma primeira versão de um rastreador de objetos genéricos, que armazena diversos discriminadores WiSARD ao longo do tempo de rastreamento, a fim de recuperar discriminadores antigos quando possível e superar dificuldades como oclusão e mudança de forma. Este método foi bastante eficaz, e bons resultados foram obtidos, mas ainda existem espaços para melhorias do rastreador, que até este ponto já é utilizável em diversas aplicações de tempo real.

Referências Bibliográficas

[1] BABENKO, B. "MILTrack Project", Disponível em: <http://vision.ucsd.edu/~bbabenko/project_miltrack.html>.

[2] THOMÉ, A. C. G. "Material didático da disciplina de Redes Neurais - UFRJ", 2008. Disponível em: <http://equipe.nce.ufrj.br/thome/p_grad/nn_ic/transp/T3_fundamentos.pdf>.

[3] SONG, B., CHOI, H., LEE, H. S. "Surveillance Tracking System Using Passive Infrared Motion Sensors in Wireless Sensor Network", *Information Networking, 2008. ICOIN 2008. International Conference on*, pp. 1–5, 2008.

[4] KAURA, H. K., HONRAO, V., PATIL, S., et al. "Gesture Controlled Robot using Image Processing", *International Journal of Advanced Research in Artificial Intelligence*, v. 2, n. 5, 2013.

[5] CARNEIRO, A. T. S., CORTEZ, P. C., COSTA, R. C. S. "Reconhecimento de Gestos da LIBRAS com Classificadores Neurais a partir dos Momentos Invariantes de Hu", .

[6] LU, T., YUAN, K. "Head gesture recognition for hands-free control of an intelligent wheelchair", *Industrial Robot: An International Journal, Vol. 34 Iss 1*, pp. 60–68, 2007.

[7] DU, W., HAYET, J.-B., PIATER, J., et al. "Collaborative Multi-Camera Tracking of Athletes in Team Sports", .

[8] GONZALES, R. C., WOODS, R. E. *Processamento Digital de Imagens*. Pearson, 2009.

[9] FRANÇA, H., DA SILVA, J., DE GREGORIO, M., et al. "Movement persuit control of an offshore automated platform via a RAM-based neural network". In: *Control Automation Robotics Vision (ICARCV), 2010 11th International Conference on*, pp. 2437 –2441, dec. 2010.

[10] CARVALHO, R., CARVALHO, D. S., MORA-CAMINO, F., et al. "Online tracking of multiple objects using WiSARD", *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 541–546, abr. 2014.

[11] NASCIMENTO, D. N., CARVALHO, R. L., MORA-CAMINO, F., et al. "A WiSARD-based multi-term memory framework for online tracking of objects", *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 19–24, 2015.

[12] BABENKO, B., BELONGIE, M.-H. Y. S. "Robust Object Tracking with Online Multiple Instance Learning". 2011.

[13] BABENKO, B., YANG, M.-H., BELONGIE, S. "Visual Tracking with Online Multiple Instance Learning". In: *CVPR*, 2009.

[14] KALAL, Z., MIKOLAJCZYK, K., MATAS, J. "Tracking-Learning-Detection". 2010.

[15] KALAL, Z. *Tracking Learning Detection*. Tese de D.Sc., Centre for Vision, Speech and Signal Processing Faculty of Engineering and Physical Sciences University of Surrey, 2011.

[16] ANDRADE, M. B. *Sistema da Rastreamento Visual da Objetos Baseado em Movimentos Oculares Sacádicos*. Tese de D.Sc., PPGI/UFES, Espírito Santo, ES, Brasil, 2015.

[17] HAYKIN, S. *Redes Neurais - Princípios e Prática*. bookman, 2001.

[18] MCCULLOCH, W., PITTS, W. "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, pp. 115–133, 1943.

[19] ALEKSANDER, GREGORIO, M. D., FRANÇA, F., et al. "A brief introduction to Weightless Neural Systems", *European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning*, 2009.

[20] ALEKSANDER, I., MORTON, H. *An introduction to Neural Computing*. Second edition ed. Berkshire House, London, UK, Thomson Computer Press, 1995.

[21] WICKERT, I., FRANÇA, F. M. G. "AUTOWISARD: Unsupervised Modes for the WISARD", *Lecture Notes in Computer Science*, v. 2048.

[22] ALEKSANDER, I. "From WISARD to MAGNUS: A Family of Weightless Virtual Neural Machines", *RAM Based Neural Networks - World Scientific*, pp. 18–30, 1998.

[23] DENNING., P. J. "Sparse Distributed Memory", *American Scientist 77*, pp. 333–335, 1989.

[24] ALEKSANDER, I., MORTON, H. "General Neural Unit: Retrieval Performance", *Electronics Letters*, v. 27, n. 19, pp. 1776–1778, 1991.

[25] IZQUIERDO, I. A., DE CARVALHO MYSKIW, J., BENETTI, F., et al. "Memória: tipos e mecanismos – achados recentes", *REVISTA USP - São Paulo*, , n. 98, pp. 9–16, 2013.

[26] GARDNER, A., INKO KANNO, DUNCAN, C. A., et al. "Measuring Distance Between Unordered Sets of Different Sizes", *Computer Vision Foundation*, 2014.

[27] NASCIMENTO, D. N. "Weightless Hierarchy Memory Tracker", Disponível em: <`http://labia.cos.ufrj.br/publicacoes/artigos/weightless-hierarchy-memory-tracker`>.

[28] CORAGGIO, P., GREGORIO, M. D. "A neurosymbolic hybrid approach for landmark recognition and robot localization", *Computer Vision Foundation*.