CENTRO UNIVERSITÁRIO FEI

FERNANDO AZEVEDO FARDO

METODOLOGIA PARA RECONHECIMENTO DE OBJETOS UTILIZANDO PADRÕES BINÁRIOS LOCAIS COM SENSORES BASEADOS EM EVENTOS

São Bernardo do Campo

2021

FERNANDO AZEVEDO FARDO

METODOLOGIA PARA RECONHECIMENTO DE OBJETOS UTILIZANDO PADRÕES BINÁRIOS LOCAIS COM SENSORES BASEADOS EM EVENTOS

Tese de Doutorado apresentado ao Centro Universitário FEI para obtenção do título de Doutor em Engenharia Elétrica, orientado pelo Prof. Paulo Sérgio Silva Rodrigues

São Bernardo do Campo

2021

Fardo, Fernando Azevedo.
Metodologia para reconhecimento de objetos utilizando padrões binários locais com sensores baseados em eventos / Fernando Azevedo Fardo. São Bernardo do Campo, 2021.
109 p.

Tese - Centro Universitário FEI.
Orientador: Prof. Dr. Paulo Sérgio Silva Rodrigues.

1. LBP. 2. classificação. 3. eventos. I. Rodrigues, Paulo Sérgio Silva, orient. II. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

centro universitário FEI

APRESENTAÇÃO DE TESE
ATA DA BANCA EXAMINADORA

Programa de Pós-Graduação Stricto Sensu em Engenharia Elétrica

Doutorado

PGE-10

Aluno: Fernando Azevedo Fardo **Matrícula:** 516202-9

Título do Trabalho: METODOLOGIA PARA RECONHECIMENTO DE OBJETOS UTILIZANDO PADRÕES BINÁRIOS LOCAIS CAPTURADOS COM SENSORES DINÂMICOS.

Área de Concentração: Processamento de Sinais e Imagens

Orientador: Prof. Dr. Paulo Sérgio Silva Rodrigues

Data da realização da defesa: 21/05/2021

ORIGINAL ASSINADA

Avaliação da Banca Examinadora

O aluno foi arguido e respondeu aos questionamentos da banca e foi aprovado por unanimidade. No entanto, esta solicitou que fossem realizadas alterações no texto de acordo com as sugestões encaminhadas por cada examinador, ficando o orientador encarregado da certificação. Também, o título foi alterado para: Metodologia para Reconhecimento de Objetos Utilizando Padrões Binários Locais Com Sensores Baseados em Eventos.

São Bernardo do Campo, 21/ 05 /2021.

MEMBROS DA BANCA EXAMINADORA	
Prof. Dr. Paulo Sérgio Silva Rodrigues	Ass.: _____
Prof. Dr. Flavio Tonidandel	Ass.: _____
Prof. Dr. Vagner Bernal Barbeta	Ass.: _____
Prof. Dr. Jaime Santos Cardoso	Ass.: _____
Prof. Dr. Silvio Jamil Ferzoli Guimarães	Ass.: _____

A Banca Examinadora acima-assinada atribuiu ao aluno o seguinte:

APROVADO ☒ REPROVADO ☐

VERSÃO FINAL DA TESE

ENDOSSO DO ORIENTADOR APÓS A INCLUSÃO DAS RECOMENDAÇÕES DA BANCA EXAMINADORA

_____

Aprovação do Coordenador do Programa de Pós-graduação

_____

Prof. Dr. Carlos Eduardo Thomaz

A Deus, minha esposa, meu filho e meus pais.

AGRADECIMENTOS

Inicialmente agradeço a Deus por ter me dado paciência e energia quando estas foram necessárias. Aos meus pais por também me apoiarem na decisão de ingressar no curso de mestrado.

Ao Dr Paulo Sérgio da Silva Rodrigues por acreditar em mim para realização deste trabalho e pela paciência e apoio dados no decorrer da dissertação.

Ao Centro Universitário da FEI por oferecer a infraestrutura necessária e pelo apoio financeiro para realização do curso.

Agradeço também aos diversos amigos e colegas pelas experiências compartilhadas que contribuíram na realização deste trabalho.

Agradeço em especial a minha esposa pelo grande apoio, incentivo e compreensão nos momentos mais difíceis. Também agradeço ao meu amado filho que em sua inocência me trouxe as palavras mais encorajadoras possíveis: "Vai dar certo, papai".

“Now the bells of hope are ringing. Angels cry again.”

Rafal Bittencourt

RESUMO

Recentemente, novos sensores com pixels ativos foram colocados no mercado. Estes sensores exportam variações locais de intensidade luminosa na forma de eventos assíncronos com baixa latência. Uma vez que o formato de saída dos dados é um fluxo de eventos endereçáveis e não uma imagem de intensidades completa, novos algoritmos são necessários para problemas conhecidos na área de Visão Computacional, como segmentação, VO, SLAM, reconhecimento de objetos e cenas. Algumas propostas para estes novos algoritmos foram aplicadas à navegação de veículos autônomos e mostraram bom desempenho para manobras em alta velocidade. Foram propostas também algumas metodologias para classificação de objetos utilizando métodos convencionais, adaptações para uso de redes profundas e redes neurais de terceira geração baseadas em *spikes*. No entanto, métodos utilizando redes profundas ou redes *spike*, frequentemente requerem recursos de *hardware* específicos e de difícil miniaturização. Além disso, diversos operadores e descritores tradicionais utilizados na área de Visão Computacional tem sido negligenciados no contexto de eventos e poderiam contribuir para metodologias mais leves para reconhecimento de objetos e símbolos. Esta tese propõe um algoritmo para extração de padrões binários locais em estruturas esparsas tipicamente encontrados em capturas por eventos com complexidade linear demonstrada experimentalmente. Para sustentar a plausibilidade da adoção deste operador, esta tese propõe a duas metodologias utilizando padrões binários locais aplicados a capturas com sensores basados em eventos para o problema de reconhecimento de objetos. A primeira metodologia, tira proveito do conhecimentos sobre os movimentos realizados pelo sensor, enquanto a segunda é agnóstica a movimentos. É demonstrado experimentalmente que o LBP é uma alternativa viável, rápida e leve e que possibilita a redução de variáveis usando algoritmo PCA em alguns casos. Demonstramos que é possível reduzir o tamanho do vetor final utilizado para classificação em até $99{,}73\%$ em relação a métodos convencionais considerados estado da arte e ainda manter acurácia comparável aos mesmos.

Palavras-chave: LBP; classificação; eventos

ABSTRACT

Recently, new sensors with active pixels were brought to market. These sensors export local variations of light intensity in the form of asynchronous events with low latency. Since the data output format is a stream of addressable events and not a complete image of light intensities, new algorithms are required for known problems in the field of Computer Vision, such as segmentation, VO, SLAM, object, and scene recognition. Some proposals for such algorithms were applied for autonomous vehicle navigation and showed good performance in high-speed maneuvers. There are also proposed methodologies for object recognition using conventional methods, deep learning, and third-generation neural networks based on spikes. However, deep learning networks and spike neural networks require specific hardware for processing, hard to miniaturize. Also, several traditional Computer Vision operators and feature descriptors have been neglected in the context of event sensors and could contribute to lighter methodologies in object recognition. This thesis proposes an algorithm for local binary pattern extraction in sparse structures, typically found in event-based sensor captures, with linear complexity demonstrated experimentally to sustain the plausibility of adopting this operator. This thesis also proposes two methodologies using local binary patterns to captures with event-based sensors for object recognition. The first methodology exploits the known motion performed by the sensor, while the second is motion agnostic. It is demonstrated experimentally that the LBP operator is a fast and light alternative that enables variable reduction using PCA in some cases. The experiments also show that it is possible to reduce the final feature vector for classification by up to $99{,}73\%$ when compared to conventional methods considered state-of-the-art while maintaining comparable accuracy.

Keywords: LBP; classification; events

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS

AER	*Addressable Event Representation*
AHLBP	*Accumulated Histograms of Local Binary Patterns*
CNN	*Convolutional Neural Network*
HATS	*Histogram of averaged Time Surfaces*
LBP	*Local Binary Pattern*
LSTM	*Long Short Term Memory*
MSLBP	*Multi Saccade Local Binary Pattern*
PCA	*Principal Component Analysis*
PCA-RECT	*Histogram of averaged Time Surfaces*
PWM	*Pulse Width Modulation*
SAE	*Surface of Active Events*
SLAM	*Simultaneous Locatilzation and Mapping*
SVM	*Support Vector Machines*
VO	*Visual Odometry*
YOLO	*You Only Look Once*

SUMÁRIO

1 INTRODUÇÃO

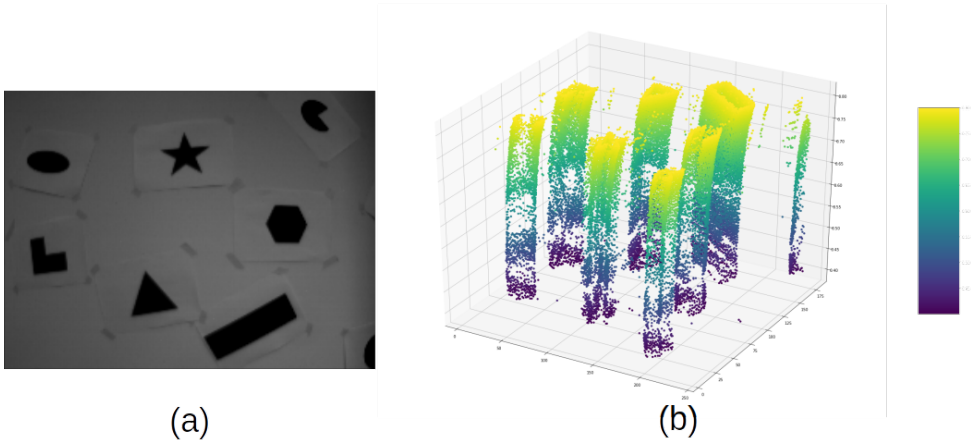
Na área de visão computacional, a análise de cenas em movimento envolve o processamento de várias imagens em sequência, muitas vezes contendo informações irrelevantes e redundantes. Tal processamento se torna computacionalmente custoso em imagens com alta resolução, uma vez que todos os pixels da imagem precisam ser processados à procura de regiões de interesse, padrões característicos ou observação de propriedades emergentes de padrões locais. Uma alternativa para isso é uma mudança de paradigma na origem dos dados, de modo que o sensores exportem somente as alterações locais e desta forma reduzam a quantidade de informação a ser processada.

Recentemente, novos sensores baseados em pixels ativos foram introduzidos ao mercado. Tratam-se de sensores que registram alterações locais de intensidade luminosa e as exportam na foram de eventos endereçáveis (BRANDLI et al., 2014). Nessa nova tecnologia, cada pixel do sensor registra variações de intensidade, independente dos demais, na forma de eventos discretos no tempo. Estas variações ocorrem em função do movimento dos objetos na cena ou do movimento da câmera em si, tornando esta tecnologia bastante atrativa para aplicações baseadas na detecção de movimento. Este formato de saída difere bastante dos sensores de imagem convencionais onde as intensidades luminosas de todos os pixels são registrados em intervalos regulares em cenas com ou sem movimento. A Figura 1 mostra um exemplo de uma captura de um poster de uma base de eventos desenvolvida por Mueggler et al. (2017) contendo formas geométricas em formato de imagem convencional (a) e um gráfico 3D com uma amostra do fluxo de eventos. Nesta captura, o sensor realiza um movimento de rotação no eixo Z.

A tecnologia de sensores baseados em eventos possui algumas vantagens em relação aos sensores convencionais. Uma vez que os pixels são independentes, não existe a necessidade de sincronizar a varredura de toda a matriz de pixels do sensor, resultando em uma baixa resolução temporal. Além disso, o fato dos eventos serem endereçáveis permite um tratamento mais rápido das informações obtidas, uma vez que reduz a necessidade de buscar áreas de interesse em uma matriz como no caso dos sensores convencionais.

Esta nova forma de informação visual representa uma mudança de paradigma na área de Visão Computacional uma vez que os algoritmos tradicionais nessa área não são facilmente adaptados a este tipo de estrutura de dados. Novos algoritmos baseados no processamento do fluxo de eventos se fazem necessários para problemas já conhecidos como classificação de imagens, rastreamento de pontos fiduciais, segmentação, entre outros.

Figura 1 – Exemplo de captura de eventos de um poster com formas geométricas.



Fonte: Autor.
Legenda: Em (a) é mostrada uma imagem convencional do poster capturada pelo sensor DAVIS240C. Em (b) é mostrado uma gráfico $3D$ de uma amostragem do fluxo de eventos.

No caso específico de cirurgias minimamente invasivas, a adoção de sensores baseados em eventos pode representar uma vantagem principalmente para situações onde existe deformação das estruturas observadas. Além disso, eventos causados por artefatos poderiam ser mais facilmente tratados e descartados por conta da baixa resolução temporal para detecção de eventos.

Atualmente, existem diversas propostas para solução de VO (*Visual Odometry*) e SLAM (*Simultaneous Locatization and Mapping*) utilizando sensores baseados em eventos (REBECQ et al., 2017b; REBECQ; HORSTSCHAEFER; SCARAMUZZA, 2017; ZHU; ATANASOV; DANIILIDIS, 2017; MUEGGLER; HUBER; SCARAMUZZA, 2014). Tais propostas mostraram bons resultados em problemas de navegação de veículos aéreos não tripulados em ambientes bem estruturados. No entanto, segundo levantamento bibliográfico para esta tese, muitas destas propostas fazem uso de imagens de intensidade concomitantes com o fluxo de eventos ou sensores de inércia. Algumas destas técnicas são voltadas para potenciais subproblemas de VO como a correspondência e rastreamento de pontos fiduciais. Estas técnicas utilizam modelos probabilísticos, alinhamento com formas primitivas ou aprendizado de máquina para reconhecimento de tais pontos em diferentes intervalos de tempo.

Na literatura científica sobre o assunto, também foram publicados métodos para reconhecimento de objetos e extração de características de cenas que variam desde adaptações de algoritmos conhecidos como o FAST (MUEGGLER; BARTOLOZZI; SCARAMUZZA, 2017), quanto a aplicação de hardwares neuromórficos específicos para lidar com este tipo de sinal. No entanto, a maioria dos algoritmos faz uso de técnicas que envolvem a projeção dos eventos em uma estrutura espacial $2D$ similar a uma imagem convencional contendo intensidades de pixels,

e desta forma aplica-se algoritmos convencionais como uso de redes convolucionais e LSTM para classificação de objetos, com resultados bastante variados.

O treinamento de redes profundas é custoso em bases com muitas classes e atualmente inviável em sistemas embarcados. Frequentemente o aprendizado em redes profundas tem demandado investimento em recursos de *hardware* específico (LECUN, 2019). No entanto, o custo de produção de tais componentes tem aumentado exponencialmente e o número de fabricantes capazes de produzi-los tem diminuído ao longo dos anos (THOMPSON; SPANUTH, 2018). Existe portanto uma demanda por operadores, descritores de características e modelos mais leves para aplicações em veículos não tripulados como *drones*. Entretanto, a família de operadores de padrões binários locais ou LBP (*Local Binary Patterns*) poderia atender estes requisitos, porém tem sido praticamente negligenciada neste contexto em pesquisas recentes.

Assim, nesta tese é demonstrado um algoritmo para extração de padrões binários locais capaz de tirar proveito da natureza esparsa dos eventos em estruturas não matriciais. A plausibilidade de adoção de padrões binários locais é demonstrada com a classificação de objetos e caracteres em experimentos utilizando bases de *benchmark*, bem como novas bases sintéticas com adição de ruído aditivo e degenerativo. Os resultados obtidos se mostraram próximos ao estado da arte, porém com um custo muito inferior em termos de memória e processamento. Abordagens para processamento de dados esparsos já foram explorados em operações de convolução (MESSIKOMMER et al., 2020). No entanto, com base no levantamento bibliográfico, esta tese apresenta a primeira implementação esparsa de extração de LBPs conhecida.

1.1 Objetivo

O objetivo deste trabalho é propor uma metodologia voltada para o reconhecimento de objetos utilizando padrões binários locais aplicados a capturas utilizando câmeras de eventos, sem a utilização de *hardware* específico para processamento.

1.2 Principais contribuições

Este trabalho tráz a primeira proposta para uso de uma versão simplificada do operador $\mathrm{LBP}^{riu2}_{P,R}$ aplicado a reconhecimento de objetos capturados utilizando uma câmera baseada em eventos.

A contribuição principal deste trabalho é a exploração do uso de sensores baseados para reconhecimento de objetos sem uso de redes convolucionais profundas e sem criação prévia de *codebooks*, utilizando um descritor baseado no operador LBP, com baixo uso em memória.

Também como contribuição, esta tese apresenta uma variação do algoritmo para extração de padrões binários locais aplicável a dados esparsos, com complexidade linear em representações utilizando tabelas *hash*.

1.3 Organização do trabalho

Esta tese é organizada da seguinte forma: O Capítulo 2 compreende a revisão dos trabalhos relacionados ao reconhecimento de objetos utilizando sensores baseados em eventos. O Capítulo 3 descreve os fundamentos teóricos da tecnologia de captura de eventos, suas limitações e vantagens e faz uma descrição do operador LBP. O Capítulo 4 descreve a metodologia proposta para atender o objetivo, bem como a proposta para reconhecimento de objetos de pontos fiduciais utilizando padrões locais de eventos. O Capítulo 5 descreve os experimentos realizados e as bases de dados que foram utilizados. O capítulo 6 contém uma análise dos resultados encontrados nos experimentos e o Capítulo 7 traz conclusão e possibilidades para futuros trabalhos.

2 REVISÃO BIBLIOGRÁFICA

Recentemente, novos sensores baseados em pixels ativos, também conhecidos como sensores de visão dinâmica ou sensores baseados em eventos, foram apresentados ao mercado (BRANDLI et al., 2014). Um evento é uma variação de intensidade luminosa em um determinado pixel da câmera em um instante de tempo. A saída deste tipo de sensor é uma sequência de eventos com seus respectivos registros de tempo, coordenadas (x,y) e polaridade (positiva ou negativa). Desta forma, diferem bastante das câmeras convencionais que registram intensidades luminosas de todos os pixels em intervalos regulares.

Sensores baseados em eventos representam uma mudança de paradigma com relação aos sensores de imagem convencionais. Algoritmos tradicionais da Visão Computacional não são facilmente adaptáveis ao fluxo de eventos gerados por estes sensores. Por conta disso, esta tecnologia frequentemente depende da introdução de novos algoritmos para aplicações onde câmeras convencionais já são utilizadas. Alguns sensores mais novos como o DAVIS 240C (BRANDLI et al., 2014) têm como dados de saída não só o fluxo de eventos, como também uma imagem em intervalos regulares, tal como uma câmera convencional. Este novo modelo de câmera híbrido estimulou diversos estudos relacionados à percepção de movimento utilizando sensores com pixels ativos. Desde então, foram propostas algumas metodologias para os problemas conhecidos da Visão Computacional. A Seção 2.1 descreve alguns exemplos de algoritmos aplicados a problemas de navegação e a Seção 2.2 descreve alguns exemplos na área de reconhecimento de objetos.

2.1 APLICAÇÕES EM PROBLEMAS DE NAVEGAÇÃO

Censi e Scaramuzza (2014) propuseram a primeira implementação de uma metodologia para VO (*Visual Odometry*) utilizando sensores baseados em eventos. Esta metodologia no entanto, usa os eventos para estimativa do deslocamento relativo partindo de uma imagem de um sensor RGBD, tratando-se portanto de um método híbrido. Mueggler, Huber e Scaramuzza (2014) propuseram uma metodologia para estimativa de pose de uma câmera em seis graus de liberdade utilizando detecção de linhas por integração de eventos, aplicável a manobras em alta velocidade em veículos aéreos quadrirotores.

Milford et al. (2015) propuseram uma metodologia para SLAM (*Simultaneous Localization and Mapping*) utilizando sensores baseados em eventos, com uso do algoritmo SeqSLAM proposto

por Milford e Wyeth (2012) com imagens de intensidade sintéticas geradas a partir dos eventos gerados por um sensor DVS128. Esta técnica foi testada em ambientes abertos para obtenção de mapas de trajetória, não estruturas 3D.

Kueng et al. (2016) propuseram uma metodologia para VO baseada na detecção e rastreamento de pontos fiduciais. Tais pontos fiduciais são obtidos da imagem de intensidade de um sensor DAVIS 240C (BRANDLI et al., 2014) e são rastreados com base no fluxo de eventos, tratando-se portanto de uma metodologia híbrida.

Kim, Leutenegger e Davison (2016) propuseram uma metodologia SLAM em tempo real utilizando somente fluxo de eventos para uso em cenas não estruturadas desconhecidas. Esta abordagem envolve a estimativa de um mapa inverso de profundidade e uma estrutura probabilística usando filtros estendidos de Kalman, distribuídos em processos paralelos com uso de GPU.

Gallego et al. (2017) propuseram uma metodologia para estimativa de posição da câmera em seis graus de liberdade, utilizando rastreamento de eventos em mapas de profundidade fotométricos pré-existentes.

Rebecq, Horstschaefer e Scaramuzza (2017) propuseram uma metodologia para SLAM baseada exclusivamente no fluxo de eventos. Esta técnica combina rastreamento de pontos fiduciais baseado em fluxo de eventos com um alinhamento imagem-modelo 3D. Os pontos 3D da cena são estimados utilizando o método proposto por Rebecq, Gallego e Scaramuzza (2016). Posteriormente, Rebecq et al. (2017a) propuseram uma metodologia pare reconstrução 3D utilizando novamente o método Rebecq, Gallego e Scaramuzza (2016), obtendo bons resultados em cenários que sensores convencionais normalmente apresentam dificuldades como em alto movimento ou com alto alcance dinâmico HDR (*High Dynamic Range*).

Vidal et al. (2017) propuseram uma metodologia para SLAM que combina fluxo de eventos, imagem de intensidade luminosa e informações de um sensor IMU embarcados em um veículo quadrirotor não tripulado. Trata-se portanto de uma metodologia híbrida multisensorial.

Também foram propostas metodologias para problemas locais onde não existe manutenção de mapas globais, que podem ser aplicadas a algoritmos de VO como estimativa de fluxo óptico e rastreamento de pontos fiduciais.

Benosman et al. (2012) propuseram a primeira metodologia conhecida para estimativa de fluxo óptico baseada em fluxo de eventos utilizando derivação parcial de eventos em uma vizinhança. Posteriormente, Benosman et al. (2014) propuseram uma metodologia para estimativa do fluxo óptico utilizando derivação local na superfície de co-eventos ativos. Barranco, Fermüller

e Aloimonos (2014) propuseram uma metodologia híbrida para estimativa de fluxo óptico combinando informações de eventos com informações de intensidade luminosa. Em sua dissertação de mestrado, Mirus (2016) propôs a utilização de eventos para cálculo do fluxo óptico utilizando o algoritmo proposto por Benosman et al. (2014) com dados de uma simulador que gera um fluxo de eventos emulando um sensor do tipo DAVIS (BRANDLI et al., 2014), demonstrando a eficácia no uso de dados simulados para desenvolvimento deste tipo de algoritmo. Ridwan e Cheng (2017) propuseram uma metodologia para cálculo de fluxo óptico a partir do fluxo de eventos baseada em detectores Reichardt, que são estruturas neuro-inspiradas no sistema visual de moscas.

Outras abordagens para sensores ativos envolvem a detecção de pontos fiduciais de forma análoga aos algoritmos já propostos para câmeras convencionais para adoção em problemas de navegação como VO e SLAM. Lagorce et al. (2015) propuseram uma metodologia para rastreamento de grupos de eventos baseados em múltiplos *kernels*. Vasco, Glover e Bartolozzi (2016) propuseram a detecção de pontos fiduciais Harris aplicada a fluxo de eventos na superfície de eventos ativos. Posteriormente, Mueggler, Bartolozzi e Scaramuzza (2017) propuseram uma abordagem similar com adaptação do detector FAST. Estas duas abordagens podem ser utilizadas para rastreamento de pontos fiduciais. Zhu, Atanasov e Daniilidis (2017) propuseram uma metodologia iterativa com base em uma estrutura probabilística para rastreamento de pontos fiduciais a partir do fluxo de eventos. Ramesh et al. (2017) propuseram um descritor de pontos fiduciais para rastreamento e reconhecimento de cenas e objetos em cenas a partir do fluxo de eventos. Tais pontos fiduciais são descritos por modelos em vizinhanças radiais que são aprendidos utilizando máquinas de vetores de suporte (SVM). Apesar de aplicado a um problema diferente, tal metodologia, segundo Ramesh et al. (2017), poderia ser aplicada aos problemas de VO e SLAM.

Algumas das abordagens descritas utilizam uma combinação de dados de eventos com unidade de movimento por inércia (IMU) como complemento para estimativa de movimento. No entanto trabalhos baseados em sensores adicionais fogem do escopo desta tese. Notavelmente, as abordagens que mostraram melhores resultados até o momento não são puramente baseadas em eventos e frequentemente realizam processamento em conjunto com imagens de intensidade ou complementam os dados com sensores de inércia.

Aplicações para a navegação em diversas áreas poderiam se beneficiar deste tipo de sensor devido à baixa latência dos pixels ativos. Essa característica pode contribuir para um baixo erro estocástico em cenas não estruturadas como as encontradas em procedimentos cirúrgicos onde

existe grande transformação na estrutura observada. No entanto, aplicações na área médica deste tipo de sensor ainda não são conhecidas, possivelmente por conta da baixa resolução deste tipo de sensor.

2.2 APLICAÇÕES EM RECONHECIMENTO DE OBJETOS

Diversos trabalhos foram propostos para reconhecimento de objetos e extração de características utilizando sensores baseados em eventos. Serrano-Gotarredona et al. (2009) propuseram uma solução baseada em uma rede neural implementada em hardware específico para operação com eventos e realizando convoluções espaciais para detecção e rastreamento de objetos em movimento. Belbachir et al. (2010) propuseram uma metodologia para detecção e reconhecimento de objetos se movendo em altas velocidades, com base nas informações de contornos dos objetos capturados por um sensor de eventos.

Li, Li e Shi (2016) propuseram uma metodologia para reconhecimento de posturas humanas e caracteres utilizando florestas aleatórias com representação dos eventos em histogramas $2D$ quantizados em intervalos de $10ms$. A base de caracteres é uma base própria MNIST-DVS que é uma conversão da base MNIST em fluxo de eventos. Nesta base de dados, atingiram $88{,}26\%$ de precisão.

Lagorce et al. (2016) propusera um modelo hierárquico de histogramas de superfícies de tempo e demonstram este método para classificação de caracteres, obtendo bons resultados para bases de naipes de cartas de baralho, poses e caracteres.

Clady et al. (2017) propuseram um descritor de características para representação de objetos capturados com sensores baseados em eventos a partir da distribuição do fluxo óptico extraído das capturas.

Sironi et al. (2018) propuseram um descritor baseado no histograma de superfícies de tempo acumuladas, obtendo excelentes resultados na base NMNIST e introduziram a base N-CARS, sendo até o presente momento o modelo com melhor acurácia para a base NMNIST sem uso de redes profundas.

Ramesh et al. (2018) propuseram uma metodologia para detecção e reconhecimento de objetos utilizando um descritor estatístico de características, consistindo em um histograma $2D$ dos eventos mais recentes em uma FIFO, com sub amostragens. O método também envolve um filtro de ruídos usando algoritmo PCA aplicado diretamente aos histogramas e a criação de um dicionário com busca utilizando KD-Tree para composição de um vetor de características

representando o objeto a ser classificado. Este método conseguiu obter $72{,}30\%$ de precisão na base N-Caltech e $98{,}9\%$ na base N-MNIST utilizando classificadores SVM lineares.

2.2.1 Classificação de objetos com métodos baseados em *Deep Learning*

Algumas metodologias aproveitam os recentes avanços das redes de aprendizado profundo para tarefas de reconhecimento de objetos, executando uma conversão de eventos em uma representação $2D$ ou outra estrutura matricial compatível com redes neurais clássicas.

Camunas-Mesa et al. (2011) propuseram um modelo embarcado para convolução de eventos para reconstrução 2D e demonstraram possibilidades de classificação de naipes de cartas de baralho quando exibidas rapidamente para um sensor de eventos.

Ghosh et al. (2014) propuseram o uso de uma rede convolucional implementada em FPGA para reconhecimento de objetos utilizando duas formas de conversão de eventos em imagens sendo uma, simplesmente acumulando o total de ocorrência em cada coordenada (x,y) e a outra uma representação ternária com valores representando eventos negativos, eventos positivos e ausência de eventos.

Sullivan e Lawson (2017) propuseram o uso de redes convolucionais para classificação de ações realizadas por humanos, obtendo bons resultados na base Weizmann e em uma base própria coletada em laboratório.

Hongmin Li et al. (2018) propuseram uma metodologia de representação de objetos usando redes convolucionais com representação dos eventos utilizando um modelo de codificação temporal baseada em neurônios LIF. Após extração da representação é feita classificação usando um modelo SVM. Esta metodologia obteve bons resultados nas bases Poker-DVS, MNIST-DVS e Posture.

Cannici et al. (2019b) propuseram duas metodologias para detecção de objetos, uma utilizando uma rede LSTM e outra usando a rede DRAW (GREGOR et al., 2015) para formação de imagens a partir dos eventos, obtendo resultados promissores no aprimoramento de redes CNN.

Bi et al. (2019) Propuseram uma metodologia utilizando representação do fluxo de eventos em grafos residuais e classificação usando redes convolucionais para reconhecimento de objetos, obtendo resultados comparáveis com os principais métodos do estado da arte utilizando as principais bases de *benchmark*.

Damien, Hubert e Frederic (2019) propuseram o uso de redes convolucionais para classificação de veículos utilizando imagens integradas a partir do fluxo de eventos. Seus resultados se mostraram inferiores ao método HATS (SIRONI et al., 2018).

Gehrig et al. (2019) propuseram o uso de redes convolucionais para classificação de objetos capturados com câmeras de eventos. O método EST adota a representação do fluxo de eventos em um tensor $4D$ onde cada célula contem as informações de coordenadas (x,y), registro de tempo t e polaridade p dos eventos. Os experimentos mostraram resultados superiores ao PCA-RECT na base N-Caltech e superiores ao HATS na base N-Cars.

Camuñas-Mesa, Linares-Barranco e Serrano-Gotarredona (2019) propuseram um modelo de redes neurais convolucionais com quatro camadas que pode ser implementado em FPGA tendo como meta baixo consumo, validado com a base *Poker Cards* e obtendo precisão de até $97{,}5\%$.

Chen et al. (2019b) Propos um modelo que combina representação em superfícies de eventos ativos, frequência e neurônios LIF utilizando três redes YOLO sendo uma para cada modalidade. Apesar de promissores os experimentos foram realizados em bases próprias e o trabalho não oferece comparações com outros métodos.

Li e Shi (2019) propuseram uma metodologia utilizando uma representação dos eventos em superfície com valores de contraste temporal, combinando filtros e redes convolucionais para rastreamento de objetos em ambientes reais. Apesar de os autores sustentarem que isto representa bons resultados, o trabalho não faz comparativos com outras metodologias.

Cannici et al. (2019a) propuseram uma adaptação da rede YOLO para classificação de objetos capturados com sensores baseados em eventos obtendo resultados promissores. Nesta metodologia, os eventos são acumulados em uma superfície inspirada nos neurônios LIF de redes *spike*.

Chen et al. (2019a) propuseram um modelo combinando uma rede neural residual (RNN) com um modelo oculto de Markov para reconhecimento de gestos.

Jiang et al. (2019) propuseram uma metodologia combinando captura de eventos e imagens de intensidades para detecção de pedestres com uma câmera montada em um veículo automotivo, sendo o primeiro trabalho neste domínio de aplicação.

Giannone et al. (2020) propuseram um modelo utilizando equações diferenciais ordinárias neurais (NODE) para classificação de objetos. Apesar de não utilizar redes *spike*, os eventos são tratados no domínio do tempo, não sendo previamente agregados em uma superfície. Este

método foi avaliado em experimentos com as bases N-MNIST e com um subconjunto da base N-Caltech mostrando bons resultados comparáveis ao uso de LSTM.

Bisulco et al. (2020) propuseram uma metodologia para detecção de pedestres utilizando uma rede convolucional implementada em FPGA atingindo acurácia de 83% com capturas de $450ms$. Também apresentam uma compressão dos dados que reduzem a quantidade de informação do fluxo de eventos em 99.6% para processamento.

Ojeda et al. (2020) Propuseram uma metodologia utilizando redes neurais binárias (BNN) com representação utilizando agregação dos eventos em mapas binários para classificação de pedestres, obtendo uma melhora em 23% quando comparado a outros métodos para a mesma base.

Deng, Li e Chen (2020) propuseram um auto codificador agnóstico a movimento utilizando as redes ResNet-34 proposta por He et al. (2016) para classificação de objeto e ResNet-18 pra aprendizado dos parâmetros, os métodos EST e HATS nas bases N-Caltech e N-MNIST, mostrando resultados superiores ao método EST.

2.2.2 Classificação de objetos com métodos baseados em redes *Spike*

Também são observadas abordagens que utilizam redes neurais de terceira geração baseadas em pulsos (*spikes*). Frenquentemente estas redes são implementadas em *hardware* neuromórfico específico como o *SpiNNaker* (FURBER et al., 2014) ou o TrueNorth (AKOPYAN et al., 2015). Nestas redes os eventos são aplicados diretamente aos neurônios de entrada da rede, podendo antes passar por algoritmos de filtragem ou *clustering*.

Zhao et al. (2014) propuseram um modelo de classificação utilizando redes *Spike* para classificação de objetos atingindo precisão de $88{,}14\%$ na base MNIST-DVS.

Serrano-Gotarredona et al. (2015) propuseram o uso de redes convolucionais de 6 camadas implementadas em um hardware neuromórfico SpiNNaker recebendo o fluxo de eventos diretamente nos neurônios de entrada em uma área restrita a 32×32 pixels. A aplicação do modelo CNN foi feita em uma base contendo capturas de naipes de cartas, obtendo precisões entre $90{,}1\%$ e $91{,}6\%$. Cohen et al. (2016) propuseram uma metodologia para reconhecimento de caracteres utilizando um *framework* de engenharia neural NEF desenvolvido por Stewart (2012) e uma base de dados N-MNIST convertida de sua versão original MNIST em fluxo de eventos. Utilizando uma rede baseada em *spikes* com 10000 neurônios ocultos, obtiveram uma precisão

de $92{,}87\%$. Negri et al. (2018) propuseram uma metodologia para classificação de ambientes utilizando redes *spike* profundas em uma base contendo 4 ambientes obtendo bons resultados.

Nan et al. (2019) propuseram um modelo utilizando redes *spike* convolucionais passando a informação espaço-temportal do fluxo de eventos inicialmente por uma camada de filtros gabor com resultados promissores nos *datasets*: *AER Posture*, *Poker Cards*, *Letters and Digits* e *Event-based ORL Dataset*.

Barua, Miyatani e Veeraraghavan (2016) propuseram uma metodologia para reconstrução de imagens a partir de um fluxo de eventos e demonstraram esta metodologia em um experimento para detecção de faces, obtendo resultados experimentais ainda inferiores à classificação de imagens reconstruídas.

Moeys et al. (2016) propuseram uma metodologia para fazer um robô reconhecer e perseguir outro robô, utilizando reconhecimento com redes neurais convolucionais com representação dos eventos em histogramas $2D$ normalizados. Neste método os autores aproveitam a natureza do domínio da aplicação em seu favor de modo que o campo de visão do robô é dividido em segmentos verticais que são separados e conectados a entradas isoladas permitindo um processamento mais leve e inferência sobre a posição espacial do alvo.

Yousefzadeh et al. (2017) propuseram um modelo de plasticidade de pulsos dependentes no tempo implementado em hardware para extração de características visuais e demonstraram os *kernels* aprendidos em uma base de caracteres. Este modelo foi posteriormente utilizado em um estudo sobre o impacto de diferentes formas de decaimento em representação de superfícies de tempo (AFSHAR et al., 2019)

Ghosh et al. (2019) propuseram um modelo de aprendizado de características lentas a partir do fluxo de eventos e demonstraram este modelo no rastreamento de pontos característicos. Os eventos são agrupados em um mapa $3D$ e comprimidos usando algoritmo PCA para utilização em redes convolucionais.

Jianing Li et al. (2019) propuseram uma metodologia que combina análise de fluxo de eventos com imagens para detecção de pedestres utilizando redes *spike* convolucionais, obtendo resultados superiores a métodos anteriores com a base DDD17.

Xiao et al. (2019) também propuseram uma metodologia usando redes *spike* em uma estrutura de aprendizado temporal para classificação de objetos e símbolos obtendo bons resultados com a base MNIST-DVS, mas ainda inferiores ao HATS e PCA-RECT na base N-MNIST, atingindo cerca de $93{,}26\%$ de acurácia.

Liu et al. (2020) Propuseram um modelo de redes neurais *spike* combinando camadas de filtros gabor e um novo algoritmo para treinamento SPA *Segmented Probability-Maximization*, obtendo bons resultados com destaque para $96{,}3\%$ na base N-MNIST e $100{,}0\%$ na base *Poker Cards*.

Ramesh e Yang (2020) propuseram um método para detecção de faces em cenas capturadas com eventos utilizando filtros com kernel amplificados, atingindo acurácia de detecção na ordem de 87% em uma base própria.

Lu et al. (2020) propuseram uma metodologia para classificação de objetos utilizando redes neurais *spike* com uma camada adicional de *pooling* no final, permitindo uma redução dos parâmetros utilizados para classificação de objetos.

Perot et al. (2020) Propuseram a primeira metodologia para detecção de veículos utilizando sensores baseados em eventos com resolução 1280×720, obtendo bons resultados

2.3 PADRÕES BINÁRIOS LOCAIS

Esta seção contém um levantamento de diferentes algoritmos e operadores de padrões binários locais ou LBP. Todos estes operadores são aplicados em imagens convencionais no formato de matrizes, não possuindo até o presente momento aplicações utilizando fluxo de eventos ou representações esparsas.

Padrões binários locais é o nome dado a operadores que realizam operações de transformações ou comparações locais em uma imagem e retornam um descritor binário com base nessas operações. Inicialmente Ojala, Pietikainen e Harwood (1994) propuseram o primeiro operador com esse nome que realiza 8 comparações em uma vizinhança de tamanho fixo 3×3 pixels, onde cada pixel é comparado com o pixel central. Cada comparação corresponde a um bit em uma palavra de 8 bits que representa as orientações de gradiente daquele padrão. Posteriormente, Pietikäinen, Ojala e Xu (2000) propuseram uma variação invariante a rotação do operador LBP original, ao observar que alguns padrões são mais discriminantes que outros, reduzindo o número de possíveis padrões de 256 para 59 em uma região de tamanho 3×3.

Yao e Chen (2003) adaptaram o operador LBP original para extração de características de textura de imagens coloridas, com robustez a rotação e escala. Mostraram bons resultados para segmentação de objetos utilizando textura, quando comparados à segmentação por histograma de cores.

Zhang et al. (2005) propuseram o uso do LBP em images processadas com filtros gabor e criaram um operador em duas etapas para classificação de imagens de face obtendo resultados melhores que as metodologias FERET (PHILLIPS et al., 2000) e usando o LBP original (AHONEN; HADID; PIETIKÄINEN, 2004).

Heikkilä, Pietikäinen e Schmid (2006) propuseram uma variação dos padrões binários locais, onde os pixels são comparados com seus pares opostos em relação ao centro (CS-LBP), resultando em um descritor de apenas 4 bits independente do valor do pixel central, e demonstraram resultados superiores ao SIFT (LOWE, 2004) para classificação de objetos.

Hafiane, Seetharaman e Zavidovique (2007) propuseram uma variação do LBP com comparação com a mediana da região, ao invés de comparação com o pixel central, operador que se mostrou mais robusto em variações de intensidade luminosa dos pixels quando comparados ao LBP original e a outras metodologias para classificação de texturas.

Zhao e Pietikainen (2007) propuseram dois operadores de padrões binários locais para extração de texturas dinâmicas. Uma das propostas consiste em combinar mapas de LBP em uma estrutura volumétrica enquanto a outra consiste em decompor a textura em três componentes para cada par de dimensões (X,Y; X,T; Y,T). Ambas propostas mostraram resultados superiores aos modelos anteriores na classificação de texturas dinâmicas encontradas nas bases DynTex e MIT.

Tan e Triggs (2007) propuseram uma variante do LBP adotando valores ternários (1, 0 e -1) para operações de classificação de face em condições de iluminação desafiadoras, mostrando bons resultados com a base CMU PIE.

Liao e Chung (2007) propuseram um operador LBP alongado (ELBP) para captura de caracterísitcas anisotrópicas em imagens de face, além de uma variação que opera com média de gradientes ao invés de comparações locais (AMDGM). Os experimentos mostraram bons resultados com as bases ORL e FERET. Este método foi demonstrado superior a demais métodos em análises de bases médicas no estudo conduzido por Nanni, Lumini e Brahnam (2010).

Guo, Zhang e Zhang (2010) propuseram uma operador LBP completo (CLBP) que envolve múltiplas etapas. Inicialmente, são extraídos os mapas LBP que são decompostos em mapas de sinais e mapas de magnitude, além de extrair informações sobre a intensidade dos pixels envolvidos. Os experimentos mostraram resultados superiores a outros métodos contemporâneos para classficação de padrões de textura.

Outras propostas de operadores LBP procuram extrair informação em múltiplas escalas, não se limitando a uma vizinhança de tamanho fixo. Qian et al. (2011) propuseram uma varição do LBP de tamanho fixo aplicado a uma pirâmide de imagens em várias escalas

(PLBP). Posteriormente, He, Sang e Gao (2013) propuseram uma versão multiestrutura do operador LBP (MS-LBP) que permite operação em várias escalas e concatenando os histogramas de ocorrência de LBPs. Além disso, este operador também permite combinar valores de diferentes escalas dando controle do número de valores distintos nos histogramas finais. Ambas metodologias demonstraram resultados superiores a métodos utilizando somente uma resolução para classificação de imagens e texturas.

Bilodeau, Jodoin e Saunier (2013) propôs um operador para extração de padrões de similaridade binários locais. Neste operador são extraídas diferenças absolutas locais em uma imagem e comparadas com uma imagem de referência para extração de fundo em sequencias de vídeo, mostrando resultados superiores a remoção por cores e relativamente superior a outros métodos considerados estados da arte.

Vishnyakov et al. (2014) propuseram um operador iLBP que combina o operador LBP original com valores de intensidade para criação de um modelo rápido para extração de fundo em sequências de vídeo, demonstrando bons resultados com ganhos de velocidade na ordem de centenas de vezes.

Silva, Bouwmans e Frélicot (2015) propuseram uma extensão do CS-LBP (XCS-LBP) combinando com o operador LBP original, demonstrando uma aplicação de remoção de fundo em sequencias de video. Os resultados sugerem robustez a variações de iluminação. Verma e Raman (2015) também propuseram uma adaptação ao CS-LBP para reconhecimento de faces e texturas, ao combinar mapas de CS-LBP com matrizes de co-ocorrências de tons de cinza, mostrando resultados superiores a outros métodos considerados estado da arte na classificação de texturas.

2.4 BASES DE DADOS E SIMULADORES

Recentemente, pesquisas voltadas para novos algoritmos aplicados a sensores baseados em eventos levaram à criação de bases de dados contendo cenas em ambientes conhecidos como escritórios, ruas e elementos mais simples como formas geométricas ou caracteres e à implementação de simuladores para geração destes dados (MUEGGLER et al., 2017). Este simulador de sensores baseados em eventos possibilita o estudo da aplicabilidade deste tipo de sensor em cenas artificiais simulando condições adversas encontradas em vários domínios de aplicação.

Para reconhecimento de objetos, Orchard et al. (2015) propuseram uma metodologia para converter bases de imagens conhecidas como a Caltech101 (BISHOP et al., 2006) e MNIST (LECUN; CORTES, 2010) em bases de eventos. O método utiliza um aparato consistindo em um sensor DVS240 montado em um conjunto eletromecânico que realiza pequenos movimentos sacádicos em frente a uma tela de LCD que exibe as imagens de uma base de imagens em sequencia. Hongmin Li et al. (2017) empregaram um método similar à base CIFAR-10, utilizando uma câmera DVS128 fixa e movimentando a imagem na tela de LCD. Sironi et al. (2018) desenvolveram a base N-Cars para classificação de binária, contendo capturas em cenas reais de carros em ambientes abertos.

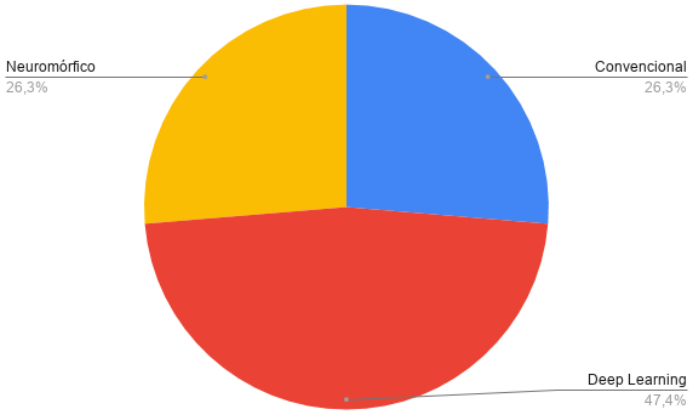
2.4.1 Resumo e estado da arte

A partir do levantamento bibliográfico, é possível notar algumas tendências nas pesquisas mais recentes. Na área de detecção e reconhecimento de objetos, observa-se um grande número de trabalhos envolvendo redes neurais de aprendizado profundo, ou inciativas que adotam hardware neuromórfico como o Spinnaker. A Figura 2 mostra um comparativo entre as técnicas levantadas agrupadas em três categorias, correspondendo a métodos convencionais, métodos de aprendizado profundo e métodos utilizando hardware neuromórfico. Métodos de aprendizado profundo compreendem redes neurais com um grande número de camadas que frequentemente requerem uso de GPUs para treinamento, tais como redes convolucionais (CNN), redes neurais residuais (RNN), NODE, LSTM, YOLO e ReLU e estão descritos na Seção 2.2.1. Métodos neuromórficos compreendem técnicas que utilizam redes neurais de terceira geração baseadas em pulsos (SNN), estejam já implementadas ou não em hardware neuromórfico e estão descritos na Seção 2.2.2.

Outro ponto que chama atenção é que o domínio de aplicação das pesquisas mais recentes se concentraram em reconhecimento de pedestres e aplicações em carros autônomos (JIANG et al., 2019; LI, J. et al., 2019; BISULCO et al., 2020; OJEDA et al., 2020).

Apesar das acurácias reportadas, nota-se também que existem poucas bases de dados a disposição para comparação das diferentes metodologias, levando os autores a criarem suas próprias bases (SIRONI et al., 2018; RAMESH et al., 2017; RAMESH; YANG, 2020). A falta de acesso à câmera potencialmente limita pesquisa em situações que ainda não foram capturadas por câmeras de eventos, como cenas na área médica. Uma outra limitação está relacionada a falta de padronizações claras nas bases de dados. Diferentes sensores possuem formatos de arquivos

Figura 2 – Distribuição de técnicas levantadas para reconhecimento de objetos ou representação



Fonte: Autor

diferentes; diferentes características de ganho, latência e resolução; variações nos movimentos; variações nos tempos de gravação.

Também não é possível afirmar com segurança que as comparações entre todos os métodos que utilizam a base N-Caltech, por exemplo, foram feitos nas mesmas condições, uma vez que esta base não está balanceada e não está separada em conjunto de treino e teste, ficando a critério do autor fazer tal divisão. No material suplementar, o autor do método HATS adota a relação $2/3$ da base para treino e $1/3$ para teste. O material disponível pelo autor do método PCA-RECT (RAMESH et al., 2018) sugere uma seleção de trinta entradas para treino e o restante para teste, além de excluir a classe de fundo. Além disso, o material fornecido pelo autor do método EST (GEHRIG et al., 2019) contém uma versão da base N-Caltech já convertida em tensores de quatro dimensões e já dividida em treino, teste e validação.

Nota-se também uma tendência em adotar representações dos eventos em superfícies, mais próximas às imagens em formato convencional, e a partir daí adaptar metodologias tradicionais da Visão Computacional. Apesar disso, não foram encontradas durante o levantamento propostas para adaptação de alguns operadores, detectores e descritores de pontos característicos tradicionais, como BRIEF (CALONDER et al., 2010), BRISK (LEUTENEGGER; CHLI; SIEGWART, 2011), ORB (RUBLEE et al., 2011), SIFT (LOWE, 2004) e as diversas variantes dos padrões binários locais levantados na Seção 2.3.

3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica da tecnologia de sensores dinâmicos baseados em eventos, algumas formas de operação, além de fazer uma breve introdução ao operador LBP, ao PCA e ao algoritmo SVM utilizado para classificação nesta tese. Este capítulo também faz uma breve introdução as metodologias HATS e PCA-RECT que são utilizadas para comparativo com as metodologias propostas nesta tese. Além disso, descreve as bases de dados utilizadas nos experimentos.

3.1 SENSORES BASEADOS EM EVENTOS

Sensores baseados em eventos são câmeras pertencentes a uma nova classe de sensores de imagem, que registram variações locais de intensidade luminosa em uma cena que contenha movimento. Estas câmeras possuem algumas vantagens em relação a câmeras convencionais como baixa latência e baixo consumo de energia. Eventos são caracterizados por mudanças relativas na intensidade luminosa dos pixels do sensor registradas no tempo. Diferentemente do que ocorre com sensores convencionais, os dados capturados com sensores de eventos não apresentam *blurring* por conta de movimentos rápidos. Além disso, o fato destas câmeras exportarem os dados relacionados a variações significativas de intensidade dos pixels, reduzem drasticamente o processamento redundante de informações pouco relevantes de uma cena em movimento. Outra vantagem do uso de fluxo de eventos dá-se pelo fato dos eventos serem endereçáveis, possuindo assim um marcação de tempo, coordenadas x e y e polaridade, dispensando, portanto a necessidade de rastreamento para detecção de informações importantes em uma imagem, reduzindo, assim, a ordem de complexidade dos algoritmos propostos.

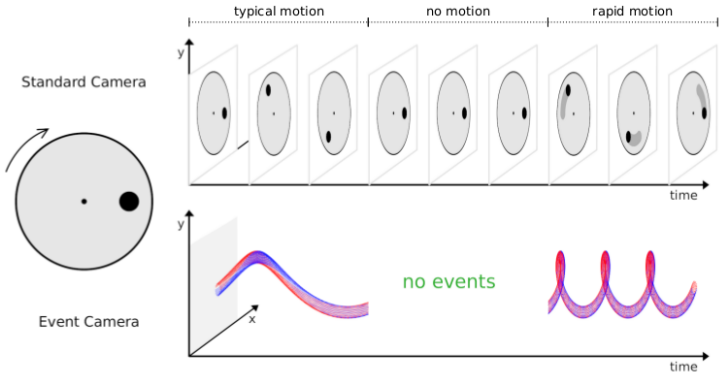
A Figura 3 mostra um comparativo entre sensores convencionais e sensores baseados em eventos. O circulo à esquerda representa um estímulo, caracterizado por um objeto circular com um círculo preto próximo à beirada em rotação. À direita do estímulo são exibidas as saídas para os tipos de câmera convencional acima e baseados em eventos abaixo em função do tempo, demonstrando situações com movimento, sem movimento e movimento rápido.

Na situação com movimento, uma câmera convencional reproduz uma imagem instantânea do estímulo completo em intervalos regulares, enquanto a câmera baseada em eventos registra cada alteração de pixel independentemente dos demais em uma taxa superior. Quando não há

movimento, a câmera convencional continua gerando imagens idênticas com a marcação na mesma posição, enquanto a câmera baseada em eventos não transmite nenhuma informação.

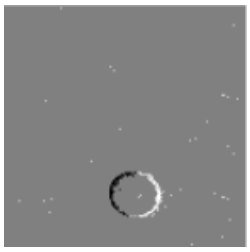
Na situação de movimento rápido do estímulo, a câmera tradicional produz um rastro característico (*motion blur*) gerado pela latência da câmera, enquanto a câmera baseada em eventos gera somente os eventos sem rastro. Uma representação gráfica de eventos acumulados em um intervalo de tempo é mostrada na Figura 4, onde pixels brancos representam eventos positivos, pixels pretos representam eventos negativos e os pixels cinzas representam ausência de eventos.

Figura 3 – Comparativo de tecnologias de captura de imagem com sensor do tipo *global shutter* e sensores baseados em eventos.



Fonte: Kim, Leutenegger e Davison, 2016

Figura 4 – Visualização de grupo de eventos

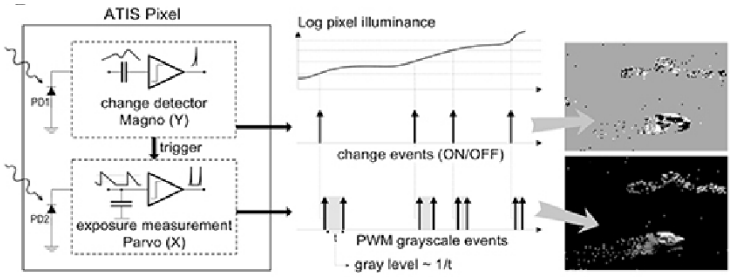


Fonte: Kim, Leutenegger e Davison, 2016

Alguns sensores baseados em eventos como o ATIS (*Asynchronous Time-based Image Sensor*) também podem exportar informação de intensidade luminosa dos pixels por meio de eventos, modulados por largura de pulso (PWM) (RUECKAUER; DELBRUCK, 2016). A Figura 5 mostra esta representação em maiores detalhes.

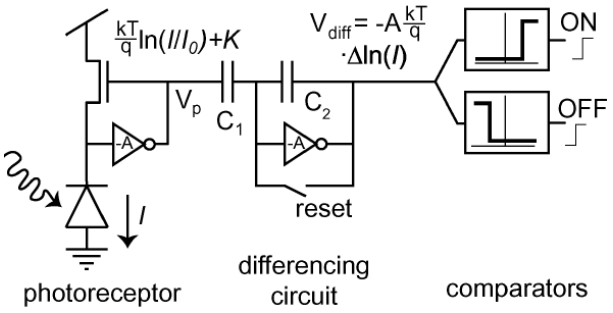
Esta nova tecnologia representa uma mudança de paradigma com relação a câmeras convencionais. Os algoritmos tradicionais de Visão Computacional não são diretamente

Figura 5 – Tecnologia da câmera ATIS.



Fonte: Rueckauer e Delbruck, 2016.
Legenda: Além do registro de eventos e suas polaridades, este sensor também produz informações relacionadas à intensidade luminosa, codificadas em PWM.

Figura 6 – Circuito de pixel ativo.



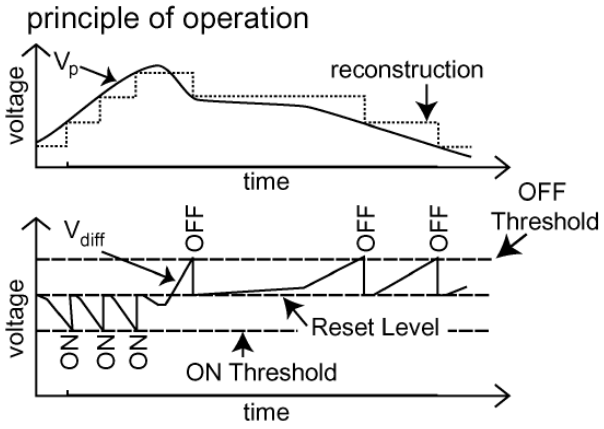
Fonte: Lichtsteiner, Posch e Delbruck, 2008

compatíveis com processamento de fluxo de eventos. Portanto, novos algoritmos na área de Visão Computacional se fazem necessários para explorar o potencial desta recente tecnologia.

Uma representação simplificada do circuito do pixel ativo é mostrada na Figura 6, que mostra a composição dos três componentes básicos: foto-receptor, circuito diferenciador e comparadores. O circuito foto-receptor é responsável pela conversão de luz em potencial elétrico. O circuito diferenciador é responsável pela detecção da ultrapassagem de um limiar para registro de eventos, além de definir o novo limiar. Os comparadores são responsáveis pela determinação da polaridade do evento, sendo positivo para subida e negativo para descida, caracterizados por um pulso momentâneo de sinal positivo ou negativo na saída do circuito, dependendo do tipo do evento.

Um evento pode ser positivo ou negativo, dependendo da variação da tensão medida pelo circuito. A cada evento, o valor de referência é redefinido e um novo limiar para detecção de eventos é estabelecido. Desta forma, uma subida gradativa na tensão do circuito pode gerar uma sequência de eventos de mesma polaridade para um mesmo pixel, conforme mostra a Figura 7, onde V_p é o nível de tensão no photo-receptor e V_{diff} é o nível de tensão no circuito diferenciador.

Figura 7 – Princípio de funcionamento.



Fonte: Lichtsteiner, Posch e Delbruck, 2008.
Legenda: O gráfico superior mostra varições de tensão V_p no foto receptor e ajuste do limiar de comparação a cada evento detectado. O gráfico inferior mostra a variação de tensão no circuito diferenciador V_{diff}.

3.1.1 Formato AER

O padrão de saída de eventos adotado para este tipo de sensor na indústria é o padrão de representação por eventos endereçáveis AER (*Addressable Event Representation*). Nesta representação, um evento r contém informações espaço-temporais sobre uma alteração significativa de intensidade de um pixel ocorrida, expressa no formato $e = (t, x, y, p)$, onde t é o tempo partindo de um instante inicial $t_0 = 0$, x é a coordenada do evento no eixo horizontal, y é a coordenada do evento no eixo vertical e p é a polaridade do evento, podendo ser positiva ou negativa.

O fluxo de eventos pode ser lido de um arquivo para testes e validação de algoritmos *a posteriori*, contendo uma lista de eventos neste formato. A Tabela 1 mostra um exemplo de fluxo de eventos registrada em arquivo. Tal como descrito na Seção 3.1, eventos podem ocorrer de forma independente. No entanto, isso não impede que dois ou mais eventos sejam registrados no mesmo instante t.

Uma vez que a taxa de eventos por unidade de tempo é muito maior do que a taxa de exibição da maioria dos monitores convencionais, frequentemente é necessário agrupar eventos em intervalos de tempo fixos para poder exibi-los posteriormente em uma imagem. A prática mais comum para representação visual dos eventos é adotar valores distintos de pixels em tons de cinza, conforme descrito na Tabela 2. A utilização de códigos de cores também é comum.

Tabela 1 – Sequência de eventos no formato AER registradas em arquivo. Eventos são endereçáveis por um registro de tempo e coordenadas x e y. A polaridade p indica se é um evento positivo (1) ou negativo (0)

Tempo (s)	x	y	p
0.000000000	43	140	1
0.000082999	43	141	1
0.000149999	43	139	1
0.000157000	43	141	1
0.000161999	43	141	1
⋮	⋮	⋮	⋮

Fonte: Autor

Tabela 2 – Códigos comuns para valores em tons de cinza para representação visual de eventos.

Tipo	Valor
Negativo	0
Sem evento	127
Positivo	255

Fonte: Autor

3.1.2 Superfície de eventos ativos

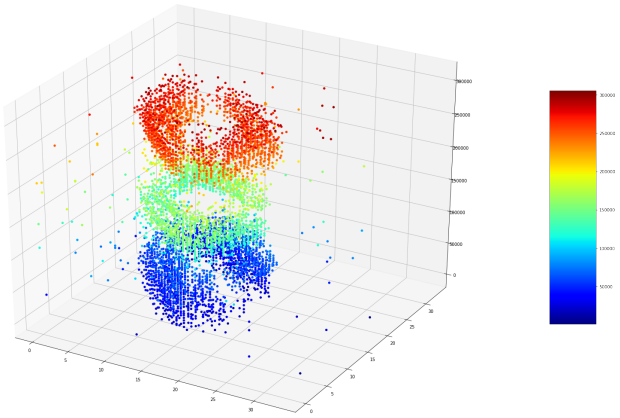
Uma das formas mais simples e amplamente utilizada quando se trabalha com sensores baseados em eventos é a adoção da chamada superfície de eventos ativos ou SAE (*Surface of Active Events* (LAGORCE et al., 2016; MUEGGLER; BARTOLOZZI; SCARAMUZZA, 2017; SIRONI et al., 2018), de modo que os eventos sejam representados em uma estrutura $2D$, o que em algumas situações permite a aplicação de alguns algoritmos de visão computacional.

Esta tese aborda duas formas de construção da SAE, com registro do evento mais recente ou com soma dos registros de tempo. A forma com registro do evento mais recente constitui no mapeamento direto do registro de tempo (TS) em uma matriz (x,y), de modo que o evento mais recente substituí um evento registrado na mesma posição. Desta forma, o registo de tempo é usado como valor de intensidade luminosa do pixel (MUEGGLER; BARTOLOZZI; SCARAMUZZA, 2017). A segunda forma com soma dos registros de tempo envolve a somatória das assinaturas de tempo em cada coordenada (x,y).

A representação com somatória é um pouco mais robusta a ruído, porém frequentemente requer normalização da superfície (SIRONI et al., 2018). Por outro lado, a representação com registro mais recente aparenta trazer superfícies mais bem definidas. A Figura 8 mostra um

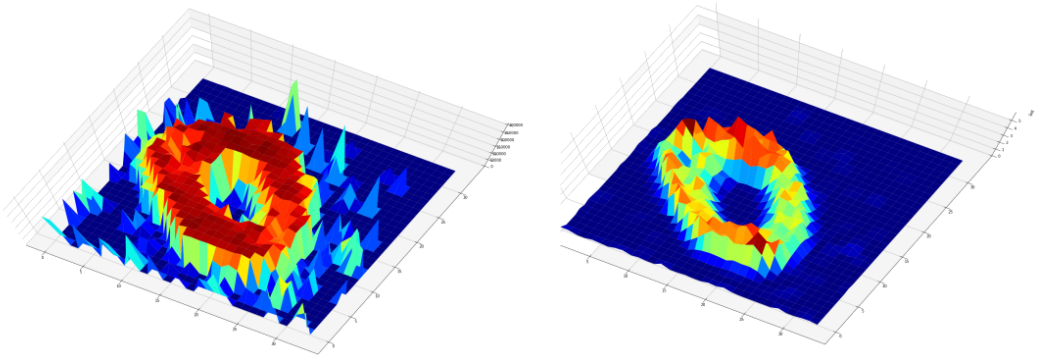
exemplo de fluxo de eventos da base N-MNIST e, assim como a Figura 9, mostra um comparativo das duas formas de construção da SAE.

Figura 8 – Fluxo de eventos de uma captura da base N-MNIST



Fonte: Autor

Figura 9 – Comparativo entre duas formas de formação da superfície de eventos ativos.



Fonte: Autor

A adoção da superfície de eventos ativos possui algumas dificuldades que precisam ser endereçadas para cada aplicação. Se os eventos mais antigos da superfície não forem suprimidos, existe a possibilidade da mesma ficar totalmente coberta por eventos, eliminando qualquer informação visual que se deseja recuperar. Outro problema enfrentado é o acúmulo estocástico de ruído na superfície que, se não for eliminado, pode afetar a qualidade dos modelos empregados na análise de eventos. Uma das técnicas comumente utilizadas para remoção destes eventos é a adoção de um fator de decaimento para cada evento. Neste método, cada novo evento do fluxo é introduzido na SAE com um valor máximo global de intensidade arbitrário. Uma função de decaimento é aplicada a todos os eventos da superfície em função do tempo, enquanto os valores

de intensidade vão diminuindo até se tornarem desprezíveis ou serem substituídos por novos eventos (AFSHAR et al., 2019). Outra técnica aplicada é a utilização de filas de eventos de tamanhos fixos, onde eventos mais antigos são substituídos por eventos mais recentes (RAMESH et al., 2017, 2018).

3.2 PADRÕES BINÁRIOS LOCAIS

Padrões binários locais ou LBP (*Local Binary Pattern*) é o nome de uma técnica criada por Ojala, Pietikainen e Harwood (1994), que codifica pequenas variações locais nas intensidades dos pixels de uma imagem em palavras binárias, permitindo assim a representação de uma vizinhança por um valor numérico. Tais palavras podem ser computadas em um modelo estatístico que descreve uma superfície em função destes padrões locais. Esta técnica tem forte inspiração no modelo do Espectro de Texturas TS (*Texture Spectrum*), proposto por He e Wang (1991) podendo ser considerado um caso particular do mesmo. A diferença principal é que o padrão resultante do TS é uma palavra ternária, enquanto o LBP gera uma palavra binária. Para uma vizinhança de 3×3 pixels, isto representa uma redução de 6561 possibilidades para apenas 256. Existe uma versão mais atual, simplificada, do TS que reduz as possibilidades de 6561 para apenas 15 (HE; WANG, 2010). No entanto, o LBP ganhou bastante popularidade para aplicações de análise de texturas nos últimos anos (PIETIKÄINEN; ZHAO, 2015).

Seja uma vizinhança V de tamanho 3×3 pixels de centro g_c, conforme exemplo da Figura 10. Neste caso, o LBP é obtido a partir da comparação de cada pixel g_i da vizinhança, com o pixel central g_c. O resultado de cada comparação resulta em um bit B_i, conforme Equação (1).

Figura 10 – Vizinhança de tamanho 3×3 pixels.

g0	g1	g2
g3	gc	g4
g5	g6	g7

Fonte: Autor

$$B(i) = \begin{cases} 1 \; se \; g_i \geq g_c \\ 0 \; se \; g_i < g_c \end{cases} \tag{1}$$

Cada bit B_i possui um peso de valor 2^i, conforme mostra a Figura 11. Matematicamente, o LBP pode ser descrito em termos de uma soma de todos os pesos $B(i) \cdot 2^i$ obtidos com a aplicação da Equação (1), como mostra a Equação 2.

Figura 11 – Mapeamento de pesos binários em uma vizinhança de tamanho 3×3 pixels.

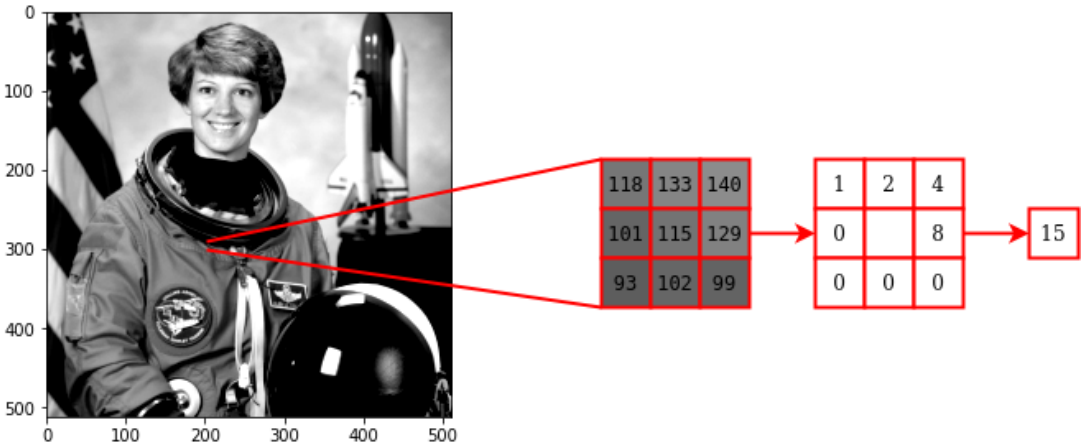
1	2	4
8		16
32	64	128

Fonte: Autor

$$LBP = \sum_{i=0}^{7} B(i) \cdot 2^i \tag{2}$$

A Figura 12 mostra um exemplo de extração do LBP em uma vizinhança 3×3. Neste exemplo, os bits resultantes das comparações com o pixel central de valor 115 são mapeados aos valores e calculados usando a Equação 1. A somatória destes pesos resulta no valor 15 que representa a vizinhança.

Figura 12 – Exemplo de computação do LBP em uma vizinhança 3×3.



Fonte: Autor

As ocorrências de LBP de cada pixel de uma imagem podem ser posteriormente computadas em um histograma para composição de um modelo estatístico da imagem que está sendo analisada. Tal histograma pode ser então utilizado para comparações com outros modelos, através de técnicas de aprendizado de máquina como o SVM ou Redes Neurais, ou através de alguma heurística para o cálculo similaridade como a distância χ^2 ou a distância Kullback-Leiber.

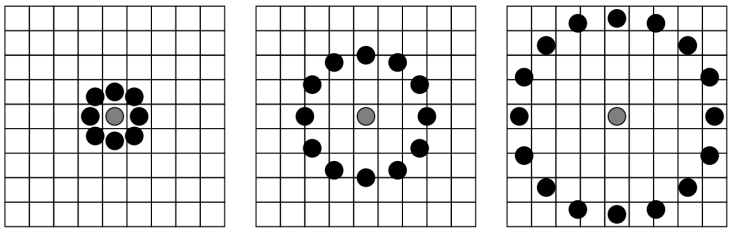
Apesar de prático e relativamente leve, o LBP possui algumas limitações como variância a rotação e escala.

3.2.1 LBP Multi Resolução

Uma das principais limitações do LBP em sua proposta original é o fato de que alguns padrões não são observáveis em um espaço tão pequeno quanto 3×3 pixels (MÄENPÄÄ; PIETIKÄINEN, 2003), demandando portanto um operador que consiga extrair informações de uma área maior. Além disso, pequenas variações na imagem como ruídos podem gerar códigos de LBP que não correspondem ao comportamento geral do padrão observado.

Para minimizar a limitação de escala, Ojala, Pietikäinen e Mäenpää (2002) propuseram o operador $\text{LBP}_{P,R}$, onde P corresponde ao nível de quantização angular e R é a resolução espacial do operador. Com os parâmetros P e R, é possível controlar a quantidade de bits que irão compor o LBP e a distância dos pixels vizinhos em relação ao pixel central. A Figura 13 mostra alguns exemplos de vizinhanças (P,R).

Figura 13 – Exemplos de vizinhanças com diferentes valores de P e R.



Fonte: Mäenpää e Pietikäinen, 2003

Neste modelo, uma vizinhança V é dada por $V = (g_c,g_0, ...,g_{P-1})$ e as coordenadas (y,x) de um pixel g_i em relação ao pixel central g_c de coordenadas $(0,0)$ são obtidas pelas Equações (3) e (4), respectivamente.

$$x(i) = R \cdot cos(2\pi(i-1)/P) \tag{3}$$

$$y(i) = -R \cdot sen(2\pi(i-1)/P) \tag{4}$$

Assim, o valor em tons de cinza de um pixel g_i que não estiver exatamente no centro de um pixel é estimado por interpolação. Desta forma, uma estrutura $\text{LBP}_{8,1}$ é equivalente ao LBP

clássico, com a diferença de que os valores dos pixels nas diagonais são obtidos por interpolação. O LBP resultante é obtido da mesma forma que o LBP, considerando P pixels vizinhos a g_c, conforme a Equação (5), onde $B(i)$ é o operador de comparação descrito na Equação 1.

$$LBP_{(P,R)} = \sum_{i=0}^{P-1} B(i) \cdot 2^i \tag{5}$$

Este operador é necessário para garantir simetria circular necessária para invariância por rotação. No entanto, com o aumento do número de bits, aumentam também os diferentes valores de padrões LBP e uma representação de um padrão por ocorrências de LBP terá um descritor de tamanho final maior.

3.2.2 LBP Multi Resolução e Invariância por Rotação

Alguns anos após a proposta do LBP, Pietikäinen, Ojala e Xu (2000) propuseram melhorias para o LBP de modo a tratar algumas de suas limitações, sobretudo à invariância por rotação.

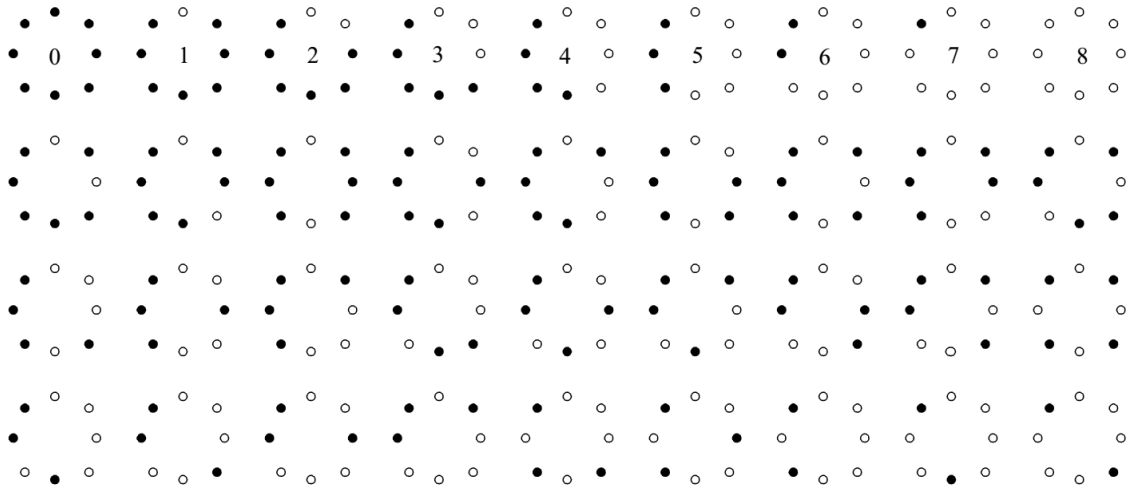
Seja I uma imagem com uma textura natural qualquer e I' o resultado de uma operação de rotação em I com um ângulo diferente de 360 graus. É bem provável que, apesar das texturas de ambas as imagens serem as mesmas, os padrões LBP resultantes para I e I' sejam distintos. Isso ocorre porque a rotação fará com que todos os bits resultantes do LBP sejam deslocados em função do ângulo. Pietikäinen, Ojala e Xu (2000) propuseram uma versão simplificada do LBP, denominada LBPROT , onde são registrados somente os padrões binários que podem ser representados em qualquer direção.

Enquanto o LBP registra as comparações entre cada bit vizinho ao bit central em posições fixas, o LBPROT registra a ocorrência das possíveis sequencias únicas de bits 0 e 1 no sentido horário. Por exemplo, os padrões 00000001_2 e 01000000_2 possuem a mesma sequência de bits 0 e 1 no sentido horário. Tais sequencias compõem uma tabela contendo 36 padrões binários únicos que são invariantes por rotação dos bits, conforme mostra a Figura 14.

Para obter um código LBPROT, obtém-se antes o código LBP clássico. Em seguida, os bits do LBP são rotacionados até que seja encontrado um padrão LBPROT tabelado que possua uma sequência de bits correspondente. O valor resultante do LBPROT é o próprio índice da tabela.

No entanto, o operador LBPROT se mostrou experimentalmente pouco eficaz para garantir invariância por rotação, devido à alta variância na ocorrência dos 36 padrões do LBPROT e a baixa quantização de 45 graus do espaço angular (OJALA; PIETIKÄINEN; MÄENPÄÄ, 2002).

Figura 14 – Os 36 padrões únicos que são invariantes por rotação.



Fonte: Pietikäinen, Ojala e Xu, 2000.
Legenda: Os pontos brancos representam bit 1 e os pontos pretos representam o bit 0. Os padrões na primeira linha numerados de 0 a 8 são denominados padrões uniformes.

Alguns conceitos do LBPROT foram aplicados ao $\text{LBP}_{P,R}$, discutido na Seção 3.2.1 para a criação de uma novo operador que é invariante por rotação e possui suporte a múltiplas resoluções (OJALA; PIETIKÄINEN; MÄENPÄÄ, 2002), denominado $\text{LBP}^{riu2}_{P,R}$.

Ojala, Pietikäinen e Mäenpää (2002) notaram que existem padrões que alguns padrões representam propriedades fundamentais em imagens de texturas, uma vez que conseguem representar as microestruturas que as compõem. Tais padrões são denominados padrões uniformes e possuem estruturas circulares com poucas alternâncias de bits 0 e 1. Em uma imagem de textura, os padrões uniformes correspondem a microestruturas como pontos, linhas e áreas sólidas.

Como forma de determinar a uniformidade do padrão, Ojala, Pietikäinen e Mäenpää (2002) propuseram o operador U descrito pela Equação (6), que determina o número de alterações de bits 0 e 1 em uma palavra binária de P bits. Os padrões considerados uniformes são os que possuem no máximo duas alternâncias de bits.

$$U(LBP_{P,R}) = |B(i)(g_{P-1} - g_c) - B(i)(g_0 - g_c)| + \sum_{i=1}^{P-1} |B(i)(g_{P-1} - g_c) - B(i)(g_0 - g_c)| \tag{6}$$

O operador $\text{LBP}^{riu2}_{P,R}$ descrito pela Equação (7) registra as ocorrências dos padrões uniformes observáveis em P ângulos a uma distância R do centro. O termo $riu2$ significa uniforme invariante por rotação (*rotation invariant uniform*) e 2 indica o número máximo de

alternâncias de bits permitido.

$$LBP_{P,R}^{riu2} = \begin{cases} \sum_{i=0}^{P-1} B(i) \ \ se \ \ U(LBP_{P,R}) \leq 2 \\ P+1 \ \ se \ \ U(LBP_{P,R}) > 2 \end{cases} \tag{7}$$

A forma usual de se trabalhar com o $\text{LBP}_{P,R}^{riu2}$ é construindo previamente uma tabela para consultas dos possíveis padrões uniformes em função de P e R. As ocorrências de cada padrão são então registradas em um histograma que pode ser então utilizado em classificadores de dados como o SVM ou outras ferramentas estatísticas.

3.3 Descrição de cenas utilizando LBP

Em análise preliminar das bases contendo cenas de eventos, observa-se que pontos com potencial para serem considerados pontos fiduciais apresentam um comportamento recorrente quando observados em movimento por um sensor baseado em eventos. Por serem pontos físicos de área relativamente pequena, é comum observar uma sequência de eventos de polaridades alternadas nos pixels na vizinhança em torno do local onde eles são observados. Padrões de sequência de eventos são detectados e reconhecidos em sequências de grupos de eventos usando modelos probabilísticos como o DART, proposto por Ramesh et al. (2017).

Em contraste com métodos baseados em aprendizado para extração de características, esta tese propõe o uso de um operador mais simples para extração de padrões binários locais uniformes ou $\text{LBP}_{P,R}^{riu2}$ proposto por Ojala, Pietikäinen e Mäenpää (2002).

Existe alta probabilidade de o evento mais recente em uma vizinhança local ser o de valor mais alto em uma SAE (MUEGGLER; BARTOLOZZI; SCARAMUZZA, 2017). Desta forma, aplicar o LBP tal como proposto originalmente nas coordenadas do evento mais recente, provavelmente resultará em pouca informação extraída, uma vez que as comparações dos pixels vizinhos com o pixel central provavelmente resultará em bit 0. Desta forma, esta tese propõe a aplicação do LBP com amostragens em grade sobre uma superfície de eventos ativos. A classificação de imagens com LBP aplicado em grade é demonstrado experimentalmente.

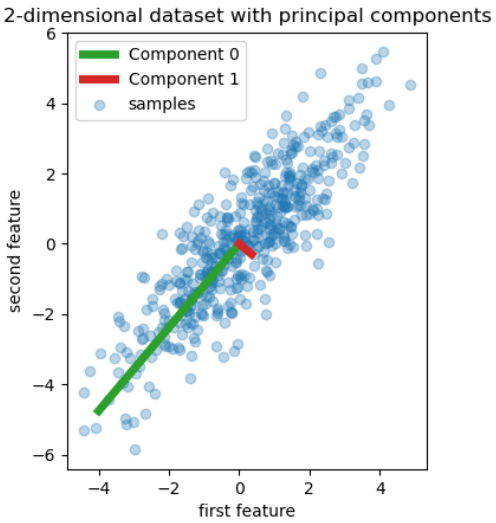
3.4 ANÁLISE DE COMPONENTES PRINCIPAIS

A técnica de análise de componentes principais é uma importante ferramenta da estatística multivariada, proposta por Karl Pearson (PEASON, 1901). Trata-se de um procedimento que

visa explicar a covariança estrutural de um conjunto de variáveis por um conjunto menor de transformações lineares.

A análise PCA é feita partir do cálculo dos autovalores e autovetores da matriz de covarianças das variáveis das quais os dados foram coletados. A partir destes dados é criado um novo sistema de coordenadas para representação de tais variáveis, cujos eixos representam as direções com maior variabilidade entre os dados. Deste modo, os dados passam a ser representados em um novo sistema de coordenadas ϕ_1 e ϕ_2 ortogonais e independentes. A Figura 15 mostra um exemplo de PCA aplicado a um conjunto de dados distribuídos por uma função gaussiana multivariada, com suas respectivas componentes principal e secundária indicadas pelas setas. Variáveis de menor relevância para representação destes dados poderiam portanto ser suprimidas.

Figura 15 – Análise PCA aplicada a uma gaussiana multivariada.



Fonte: scikit, 2021a

O cálculo do PCA segue um procedimento relativamente simples. Dado um conjunto de variáveis X, inicialmente é feito o cálculo da matriz de covarianças C. A partir desta matriz, são obtidos os autovalores e autovetores da mesma. Os autovalores e autovetores são então reordenados em ordem decrescente dos autovalores, sendo que a variável com maior autovalor a componente principal e a segunda maior a componente secundária.

3.5 MÁQUINAS DE VETORES DE SUPORTE (SVM)

Uma máquina de vetores de suporte (*Support Vector Machine*) (SVM) é um classificador de dados de treinamento supervisionado proposto por Vapnik (CORTES; VAPNIK, 1995) (SCHÖLKOPF et al., 1998) que ganhou popularidade nas últimas décadas por sua robustez e capacidade de discriminação de dados não linearmente separáveis.

O funcionamento do SVM é baseado na determinação de hiperplanos ótimos de separação entre os dados de duas classes (VAPNIK; KOTZ, 1982). O algoritmo de treinamento do SVM busca tais hiperplanos de modo que a margem de separação entre os dados seja máxima.

3.5.1 Dados linearmente separáveis

Um classificador linear pode ser expresso por uma função linear que divide o espaço e separa os dados em duas classes. Seja um conjunto de dados linearmente separáveis em duas dimensões. Existem diversas funções de separação possíveis que atendem a esse critério inclusive algumas que se encontram muito próximas a alguns desses pontos. Apesar destas funções estarem separando corretamente os dados, a generalização pode não ser muito eficaz. Desta forma, um dado apresentado para classificação próximo a um limiar de separação poderia ser incorretamente classificado como pertencente à classe errada.

O algoritmo de treinamento do SVM procura os hiperplanos que maximizam a margem de separação entre os dados mais próximos pertencentes a classes diferentes. A Figura 16 mostra os hiperplanos ótimos obtidos para um conjunto de pontos linearmente separáveis.

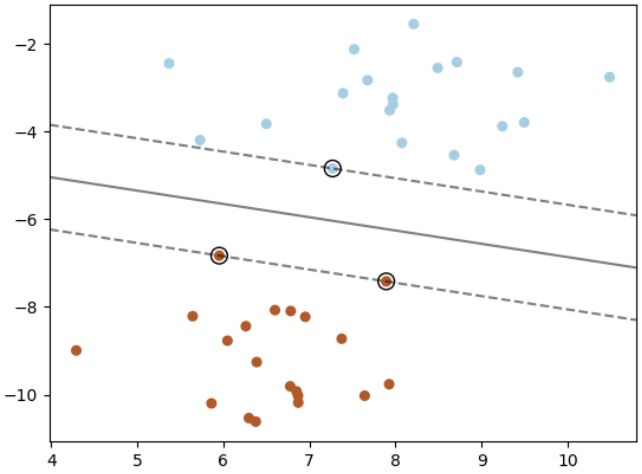
Sejam os rótulos que categorizam os exemplos denominados positivos e negativos, representados pelos valores $+1$ e -1, respectivamente. Um hiperplano de separação linear no SVM é definido pelo conjunto de pontos x que satisfazem a Equação (8), onde b é o parâmetro de corte (*bias*), w é a normal ao hiperplano, $|b| \,/\, \|w\|$ é a distância perpendicular do hiperplano até a origem e $\|w\|$ é a norma euclidiana de w.

$$w \cdot x + b = 0 \tag{8}$$

Sejam os hiperplanos de separação H_+ e H_- expressos pelas Equações (9) e (10), respectivamente, sendo que H_+ e H_- são paralelos e não há pontos de treinamento entre eles.

$$x_i \cdot w + b \geq +1 \quad para \quad y = +1 \tag{9}$$

Figura 16 – Exemplo de um treinamento utilizando algoritmo SVM com determinação de hiperplanos ótimos de separação entre os dados.



Fonte: scikit, 2021c.
Legenda: Os pontos com anéis compõem os chamados vetores de suporte que definem o hiperplano de separação entre as classes vermelho e azul.

$$x_i \cdot w + b \leq -1 \quad para \quad y = -1 \tag{10}$$

Ambas as Equações (9) e (10) podem ser combinadas na Inequação (11).

$$y_i \cdot (x_i \cdot w) + b \geq 1 \tag{11}$$

Seja d_+ a menor distância entre o hiperplano de separação e o dado positivo mais próximo, e seja d_- a menor distância entre o hiperplano de separação e o dado negativo mais próximo. A margem máxima pode ser definida por $d = d_+ + d_-$. Para dados linearmente separáveis o SVM busca w e b de modo a maximizar d. Esta distância também pode ser expressa por $|2| \,/\, \|w\|$.

O treinamento do SVM pode ser descrito portanto como um problema de otimização de ordem quadrática que visa minimizar $\|w\|^2$. Para minimizar a complexidade do treinamento, é introduzida a função Lagrangiana definida em termos de w e b, resultando na Equação (12) onde α_i são os multiplicadores de Lagrange para as $i = 1,...,n$ restrições da Inequação (11).

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i y_i (x_i \cdot w + b) + \sum_{i=1}^{n} \alpha_i \tag{12}$$

O resultado da introdução da função Lagrangiana é dada pela Equação (13).

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{13}$$

Um ponto importante da Equação (13) é que o valor dos coeficientes α_i é igual a 0 para todos os pontos de treinamento, exceto para os pontos que pertencem aos hiperplanos H_+ e H_-. Tais pontos compõem os chamados Vetores de Suporte.

A predição de novos dados apresentados ao classificador SVM é dada pela Equação (14), onde sgn é a função sinal, conforme a Equação (15).

$$g(x) = sgn\left(\sum_{i=1}^{n} \alpha_i y_i (x_i \cdot x) + b\right) \tag{14}$$

$$sgn(x) = \begin{cases} -1 \; se \; x < 0 \\ 0 \; se \; x = 0 \\ +1 \; se \; x > 0 \end{cases} \tag{15}$$

Utilizando a Equação (14), o resultado da classificação de um padrão x depende somente do produto interno $x \cdot x_i$. O resultado da previsão de um novo dado será $+1$ ou -1 para as classes positiva ou negativa, respectivamente.

3.5.2 Dados não linearmente separáveis

Em casos onde os dados não são linearmente separáveis existem algumas medidas que podem ser adotadas para minimizar ou mesmo evitar erros na classificação dos dados. Uma delas é utilizar o conceito de margens suaves (VAPNIK; VAPNIK, 1998). Neste caso, os classificadores continuam sendo lineares, sendo admitida uma tolerância em troca da minimização de $\|w\|^2$. Isso é realizado através da introdução de variáveis de relaxamento ξ às equações dos hiperplanos H_+ e H_-. Assim, obtém-se então as novas Equações (16) e (17) dos hiperplanos H_+ e H_-, respectivamente.

$$x_i \cdot w + b \geq 1 + \xi_i \;\; para \;\; y = +1 \tag{16}$$

$$x_i \cdot w + b \leq -1 + \xi_i \;\; para \;\; y = -1 \tag{17}$$

Tais variáveis permitem a violação das restrições da Inequação (11) em troca de alguns erros na classificação.

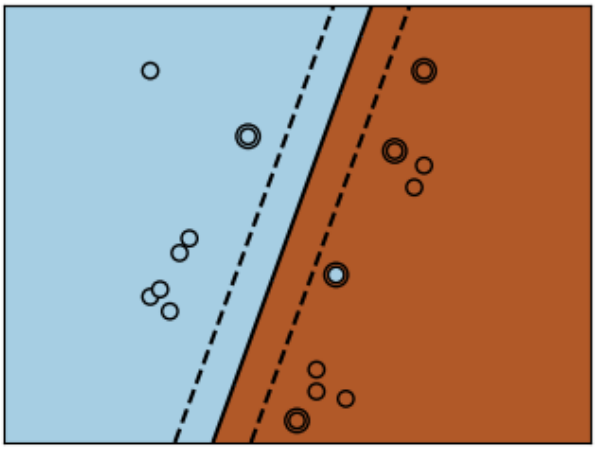
O conceito de variáveis de relaxamento tem sua origem na teoria da otimização (BOYD; VANDENBERGHE, 2004). A função a ser otimizada é descrita pela Equação (18), onde C é

uma constante definida empiricamente que determina as contribuições relativas das variáveis relaxadas e de $\|w\|^2$.

$$J = \|w\|^2 + C\left(\sum_{i=1}^{n} \xi_i\right) \tag{18}$$

A solução dos hiperplanos continua sendo a Equação (13), porém, os coeficientes α_i que serão computados incluem não somente os pontos pertencentes aos hiperplanos, mas também os pontos onde $\xi_i > 0$ não pertencentes aos hiperplanos. A previsão dos dados com margem suave é realizada utilizando a Equação (14), porém considerando agora os novos hiperplanos. Em cenários reais podemos encontrar situações em que existem dados de ambas as classes misturados em uma região comum tornando o conjunto de dados linearmente inseparável. Esta situação pode ocorrer devido a ruídos na aquisição, mas também por conta da própria natureza dos dados. A Figura 17 exibe o resultado de um treinamento considerando variáveis de relaxamento. Os pontos com anéis pretos pertencem aos vetores de suporte. Porém, é possível notar alguns pontos que também compõem os vetores de suporte mas que não pertencem aos mesmos, indicados com anéis. Estes pontos possuem $\xi_i > 0$. Nesta situação, a grande maioria dos dados é classificada corretamente, com uma certa tolerância em função dos dados não serem linearmente separáveis em sua totalidade.

Figura 17 – Exemplo de SVM treinado a partir dos dados não linearmente separáveis considerando variáveis de relaxamento.



Fonte: scikit, 2021b.
Legenda: Os pontos que possuem anéis são os que compõem os vetores de suporte, sendo que os pontos que possuem anéis se encontram fora do hiperplano e possuem $\xi_i > 0$.

3.5.3 *Kernels*

Outra medida bastante eficaz para classificação de dados não linearmente separáveis é a aplicação dos chamados truques de *kernel* (VAPNIK, 1963) para a criação de separadores não lineares. Tais técnicas possibilitam uma generalização melhor de dados não linearmente separáveis. Além disso, podem existir situações onde os dados não podem ser discriminados por um classificador linear devido à própria natureza dos mesmos.

Os truques de *kernel* utilizam funções reais ϕ_i para mapear os dados de treinamento no domínio do espaço de entradas para um espaço de características. As funções ϕ_i podem ser não lineares.

Seja F o espaço de características de dimensão N_F. Seja x uma entrada mapeada em F com o mapeamento $\phi : R^d \rightarrow F$, ou seja, F é o espaço de características definido pelo mapeamento ϕ. O treinamento envolve a construção dos hiperplanos no espaço de características F. Este procedimento é realizado calculando-se os produtos internos no espaço de características na forma $\phi(x_i) \cdot \phi(x_j)$ com as chamadas funções *kernel*.

Uma função *kernel* k, tal que $k(x_i,x_j) = \phi(x_i) \cdot \phi(x_j)$, é uma função real que recebe dois pontos x_i e x_j do espaço de entradas e os transforma para um espaço de características. Tal medida pode fazer com que alguns dados que normalmente não são linearmente separáveis no domínio do espaço de entrada, se tornem linearmente separáveis no espaço de características. Para que uma função possa ser utilizada como *kernel* ela precisa pertencer a um domínio que permita o cálculo de produtos internos (LORENA; CARVALHO, 2007). As funções *kernel* mais comumente utilizadas são as funções polinomiais, Gaussiana ou RBF (*Radial Basis Function*) e a função Sigmoide, conforme descritas nas Equações (19), (20) e (21), respectivamente.

$$k(x,y) = (x_i^T \cdot x_j + 1)^p \tag{19}$$

$$k(x,y) = e^{-\frac{\|x-y\|^2}{2 \cdot \sigma^2}} \tag{20}$$

$$k(x,y) = tanh(\beta_0 x_i. \cdot x_j + \beta_1) \tag{21}$$

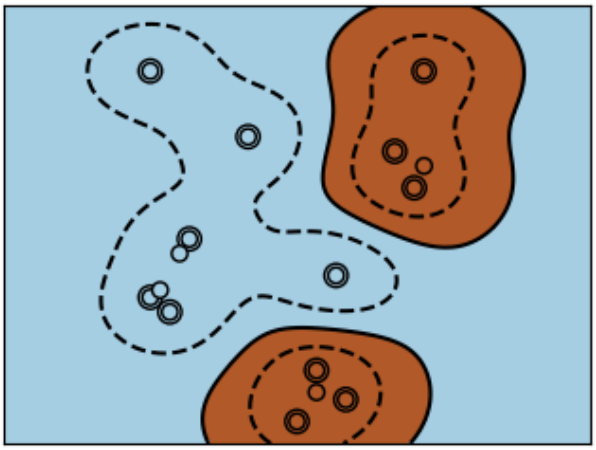
É importante notar que algumas funções *kernel* possuem parâmetros que devem ser ajustados pelo usuário. Tomando o *kernel* polinomial como exemplo, é necessário definir os coeficientes T e p. Para o *kernel* gaussiano é necessário definir o parâmetro σ e para o *kernel* sigmoid é necessário definir os coeficientes β_0 e β_1.

Com a função *kernel* definida, é possível reescrever a função de previsão dos novos dados, conforme mostra a Equação 22.

$$g(x) = sgn\left(\sum_{i=1}^{n} \alpha_i y_i \cdot k(x_i, x) + b\right) \tag{22}$$

A Figura 18 mostra um exemplo de uma SVM usando *kernel* RBF, treinada com os mesmo dados não linearmente separáveis mostrados na Figura 17.

Figura 18 – Exemplo de SVM treinado com *kernel* RBF.



Fonte: scikit, 2021b.
Legenda: Os pontos com anéis compõem os vetores de suporte não lineares.

3.5.4 SVM Multi-Classe

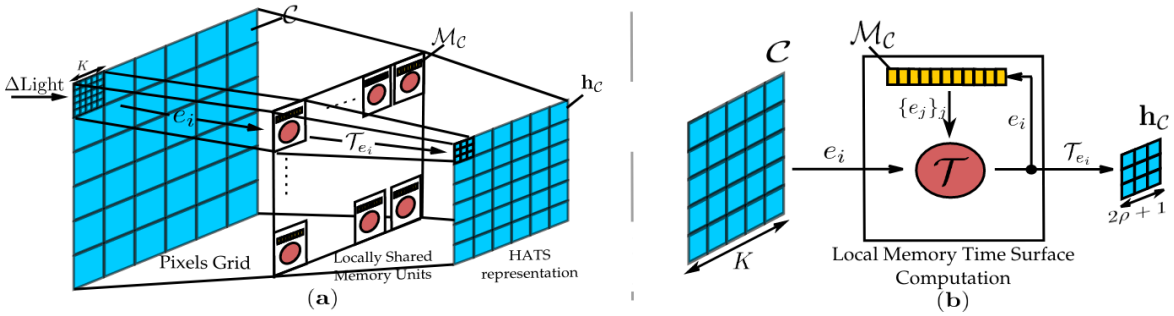
O classificador SVM em sua implementação original é um classificador binário, ou seja, só permite classificar objetos em duas classes. Para problemas de três ou mais classes, existem duas abordagens que podem ser adotadas para aplicação do SVM. A primeira delas é dividir o problema em $\frac{k(k-1)}{2}$ classificadores binários que comparam cada classe com cada outra classe (*One vs One*). A segunda envolve rotular internamente os valores de um classe com valor $+1$ e agrupar os demais valores de todas as outras classes como -1 (*One vs Rest*). Ambas as abordagens necessitam de um sistema de votação para eleger a melhor classificação dentre todas.

3.6 METODOLOGIA HATS

A metodologia HATS (*Histogram of Accumulated Time Surfaces*) é uma metodologia para reconhecimento de objetos utilizando eventos proposta por Sironi et al. (2018). Nesta metodologia, eventos são agrupados por coordenadas (x,y) em células C de tamanho $K\times$K e por um intervalo de tempo Δ_t. Para cada novo evento e_i, é construída uma superfície de tempo τ_{e_i} partir dos eventos anteriores mais recentes até $-\Delta_t$ presentes na mesma unidade de memória compartilhada M_c. Os valores de tempo obtidos de τ_{e_i} são somados em um histograma $2D$ de tamanho $2\rho+1$, onde ρ é o raio do operador espacial.

Os histogramas $2D$ obtidos são acumulados em estruturas matriciais. Ao término do processamento, é calculada a média dos valores somados dos histogramas $2D$ e estes são serializados em um histograma final com a representação do objeto. A Figura 19 mostra um diagrama desta metodologia em (a) e um detalhamento da construção de τ_{e_i} e do histograma local em (b).

Figura 19 – Diagrama geral da metodologia HATS.



Fonte: Sironi et al., 2018.
Legenda: (a) Mostra a arquitetura geral com mapeamento dos eventos em unidades compartilhadas de memória. (b) mostra o detalhamento das construções das superfícies de eventos ativos

O Algoritmo 1 mostra o pseudo-código da metodologia HATS com unidades compartilhadas de memória.

3.7 METODOLOGIA PCA-RECT

A metodologia PCA-RECT é uma metodologia para reconhecimento de objetos utilizando eventos proposta por Ramesh et al. (2018). Esta metodologia é dividida em duas etapas sendo

Algoritmo 1 – HATS com unidades de memória compartilhadas (SIRONI et al., 2018)

Data: $\varepsilon = \{e_i\}_{i=1}^{I}$: Eventos
Data: K: Tamanho da célula de memória C
Data: Δ_t: Intervalo de tempo para construção de T_{e_i}
Data: ρ: Raio do operador espacial
Result: H: Histograma final com a representação do objeto
forall l **do**
  $h_{C_l} \leftarrow 0$, $|C_l| \leftarrow 0$, $M_{C_l} \leftarrow \varnothing$
end
for $i = 1 \dots I$ **do**
  $C_l \leftarrow getCell(x_i, y_i)$;
  $\tau_{e_i} \leftarrow computeTimeSurface(e_i, M_{C_l})$;
  $h_{C_l} \leftarrow h_{C_l} + \tau_{e_i}$;
  $M_{C_l} \leftarrow M_{C_l} \cup e_i$;
  $C_l \leftarrow C_l + 1$;
end
$H \leftarrow [h_{C_1}/C_1, \dots, h_{C_l}/C_l]^T$

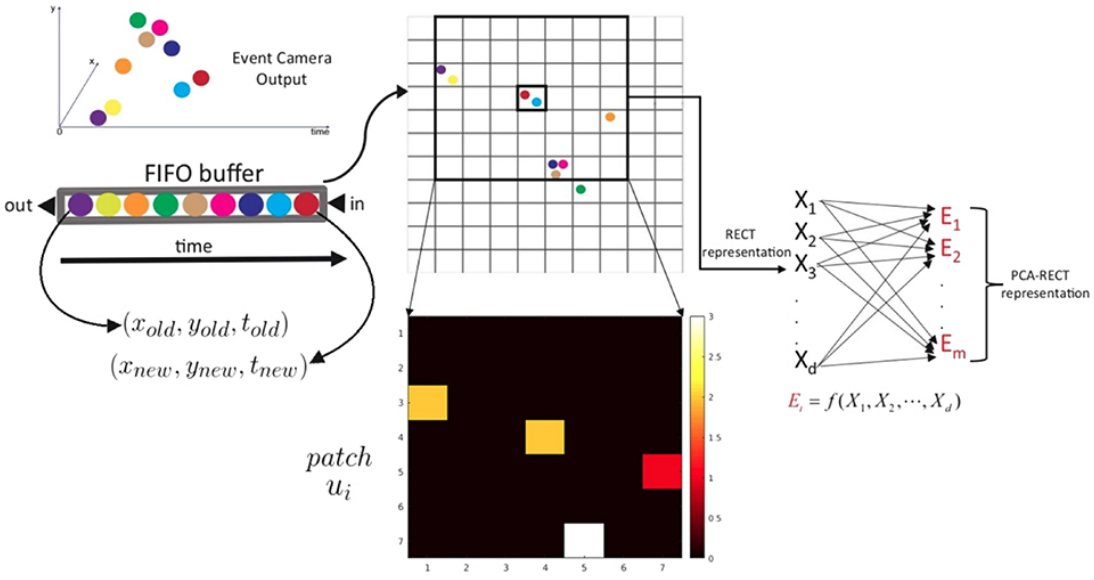
uma de detecção e uma de classificação de objetos. Esta seção só irá descrever a etapa de classificação uma vez que a detecção foge do escopo desta tese.

Nesta metodologia, os n eventos mais recentes são armazenados em uma fila. Cada novo evento $e = (x,y,t,p)$ é contabilizado em um histograma 2D de centro em (x,y) pertencente a uma região de sub-amostragem RECT. Desta forma, o evento mais recente na fila, é retirado e sua ocorrência é descontada de seu histograma correspondente. A estes histogramas é aplicado um algoritmo PCA para retirada de ocorrências pouco discriminantes. A Figura 20 mostra um diagrama geral da construção dos descritores PCA-RECT.

A partir de um conjunto de histogramas obtido previamente, é feita a construção de um *codebook* utilizando um algoritmo de agrupamento como o K-Means. Os centros obtidos com o K-Means são armazanados em uma tabela e uma árvore KD-Tree é construída para rápida consulta dos descritores PCA-RECT.

Com o codebook construído, a representação de um objeto é obtida a partir das ocorrências de índices deste codebook obtidos com buscas de todos os pontos característicos obtidos de uma captura de eventos em um histograma. Os histogramas são então usados para treinamento e testes de classificadores lineares como o SVM.

Figura 20 – Diagrama da representação do descritor PCA-RECT



Fonte: Ramesh et al., 2018

3.8 BASES DE DADOS

Sensores baseados em eventos ainda não são equipamentos acessíveis para o público geral. Para realização de pesquisas envolvendo análise de eventos, algumas bases de imagens conhecidas foram convertidas em bases de eventos. Esta tese faz uso de três bases bastante utilizadas para *benchmark* de classificação: N-Caltech, N-MNIST e N-Cars.

3.8.1 N-Caltech

A base N-Caltech é uma base não balanceada contendo capturas de eventos de 100 classes de objetos. Trata-se de uma base derivada da base de imagens Caltech-101 (FEI-FEI; FERGUS; PERONA, 2004). Como as imagens já estavam previamente capturadas por câmeras convencionais, foi empregado um aparato consistindo em um sensor DVS240 montado em um conjunto eletromecânico em frente a uma tela de LCD. Durante a captura, as imagens da base Caltech-101 são exibidas em sequência na tela, sendo que o motor realiza um movimento sacádico artificial para produzir variações de intensidade locais, produzindo assim a geração de eventos.

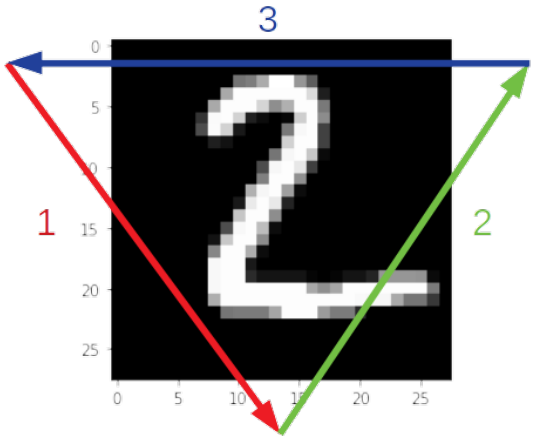
A Figura 21 mostra uma exemplo desse aparato e a Figura 22 mostra o movimento realizado pela câmera para captura dos eventos em frente a um caractere da base N-MNIST com respectiva ordem indicada pelos números 1,2 e 3. Os três movimentos levam em média $100ms$

para serem realizados com intervalo de $5ms$ entre cada um, totalizando $315ms$ de captura para cada objeto.

Figura 21 – Aparato usado para captura das bases N-Caltech e N-MNIST

Fonte: Orchard et al. (2015)

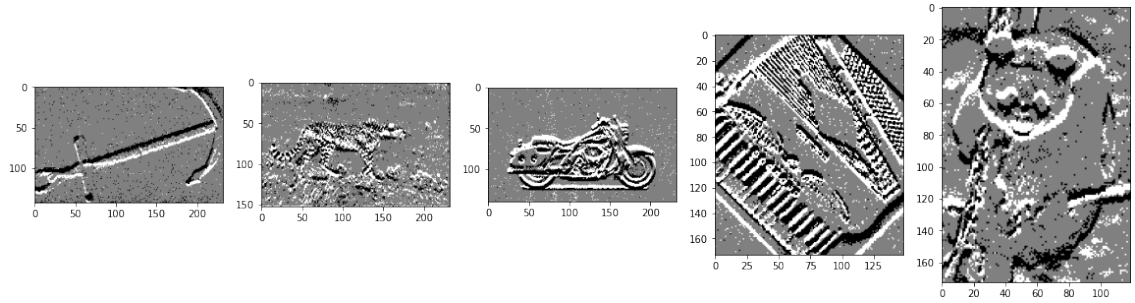
Figura 22 – Representação dos movimento sacádicos realizados pela câmera em frente a um caractere da base N-MNIST



Fonte: Autor

A Figura 23 mostra uma amostra de alguns objetos da base N-Caltech renderizados em uma superfície após $50ms$ de captura de eventos. A base contém 8242 capturas de aproximadamente 300×200 pixels divididas em 100 classes de objetos e uma classe adicional de fundo contendo 467 capturas. As capturas possuem anotação da área de interesse permitindo uma seleção dos eventos que correspondem ao objeto, exceto pela classe de fundo.

Figura 23 – Amostra da base N-Caltech



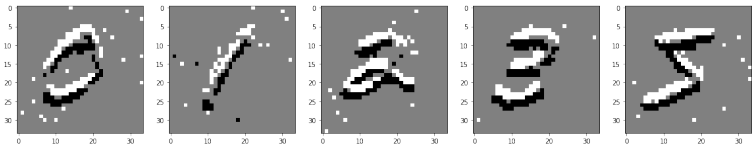
Fonte: Autor

3.8.2 N-MNIST

A base N-MNIST é uma base balanceada contendo caracteres numéricos escritos a mão, obtida a partir do serviço de correios norte-americano. Tal como a N-Caltech. A base N-MNIST deriva da base MNIST que comumente é usada para testes de algoritmos de classificação. Para esta base, foi empregado o mesmo aparato utilizada na criação da base N-Caltech. No entanto, as capturas foram escaladas para capturas de 35×35 pixels.

A Figura 24 mostra exemplos de renderizações de capturas da base N-MNIST. A base contém 60000 capturas de treino e 10000 capturas de teste de 35×35 pixels divididas em 10 classes correspondentes aos algarismos de 0 a 9.

Figura 24 – Amostra da base N-MNIST



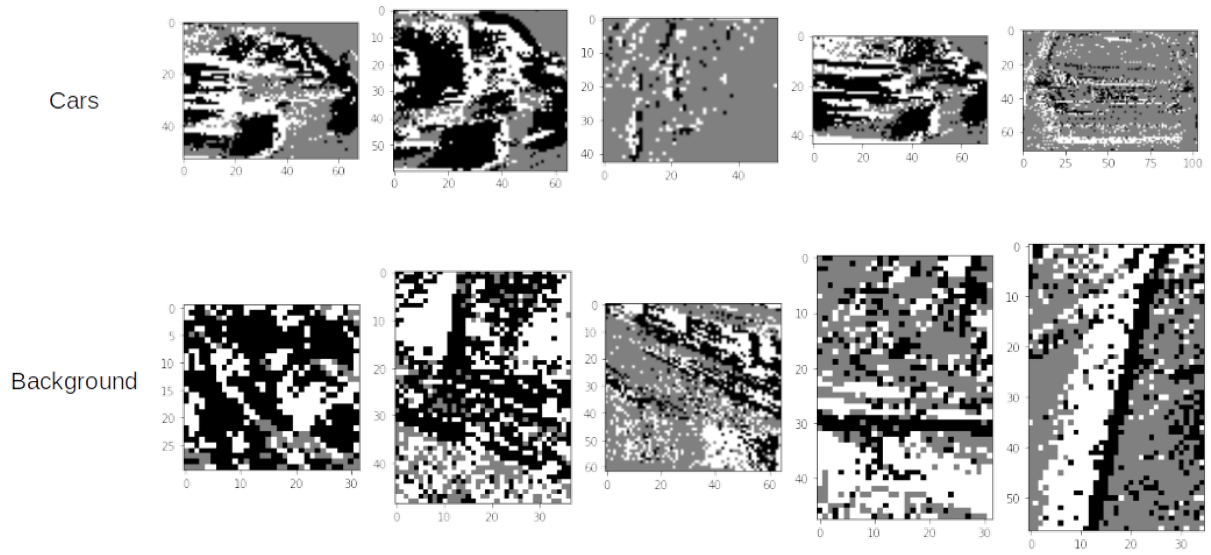
Fonte: Autor

3.8.3 N-Cars

A Base N-Cars foi criada para testes com a metodologia HATS Sironi et al. (2018) utilizando um sensor ATIS Prophesee em cenas reais. Esta base contém 24029 capturas de objetos em duas classes correspondendo a carros e fundo e está dividia em subconjuntos de treinos e testes. Cada captura contém aproximadamente $100ms$ de gravação, sem realização de movimentos sacádicos como ocorre nas bases N-MNIST e N-Caltech.

A Figura 25 mostra uma amostra das classes *cars* e *background* após $100ms$ de captura. Trata-se de uma base desafiadora, uma vez que tanto a classe *cars* quanto a classe *bacgkround* possuem bastante variação intra-classe. Carros estão em diferentes posições, objetos que compõem o fundo variam de captura para captura e também existe variação no número total de eventos entre as capturas.

Figura 25 – Amostra da base N-Cars



Fonte: Autor

3.8.4 Noisy-N-MNIST

A base Noisy-N-MNIST foi criada especificamente para esta tese. Trata-se de um conjunto de bases N-MNIST com adição de ruido branco na forma de eventos aleatórios que são inseridos na captura. O Algoritmo 2 descreve o método de geração de ruído. Para cada evento e de uma lista de eventos E, é gerado um número aleatório $r \in [0,1]$. Se $r > P$ onde P é o fator de probabilidade de novo evento P, um novo evento com coordenadas e polaridade aleatórios com o mesmo registro de tempo é gerado e inserido no fluxo de eventos. Para $P > 1.0$, um laço é excetuado de modo que, para cada evento aleatório gerado, é descontado 1.0 de P. Deste modo, é garantida a geração de pelo menos um evento aleatório para cada $1{,}0$ descontados de P. Exemplo: seja $P = 3.2$, serão gerados 3 eventos aleatórios para cada evento do fluxo e o quarto evento terá 20% de chances de ocorrer. O Algoritmo 2 descreve o processo para geração de ruído aditivo.

Algoritmo 2 – Geração de ruído aditivo

Data: P: Probabilidade de evento ruidoso
Data: E: Lista de eventos
Data: mX: Valor máximo admitido para X
Data: mY: Valor máximo admitido para Y
Result: R Lista de ventos com ruído adicionado
$R \leftarrow \{\}$;
forall e *in* E **do**
    R.append(e);
    $p \leftarrow P$;
    while$p > 0$;
    $lp \leftarrow$ min(1.0, p);
    r $\leftarrow$ random(0, 1.0);
    if *r > 1.0 -* lp **then**
        $NE \leftarrow$ novoEventoAleatório(mX, mY, e.Ts) ;
        R.append(NE)
    end
    $p \leftarrow p - 1.0$;
end

A Tabela 3 contem as versões com ruído da base N-MNIST geradas para os experimentos nesta tese e a Figura 26 exibe uma amostra da base Noisy-N-MNIST com evolução da superfície de eventos ativos no tempo em intervalos de $63ms$ para o mesmo caractere em diferentes níveis de ruído.

3.8.5 D-Noisy-N-MNIST

A simples inserção de eventos aleatórios pode não representar bem todos os cenários. Considerando que fontes de ruído poderiam além de produzir novos eventos, impedir a geração de eventos legítimos, é proposto o Algoritmo 3 para geração de ruído degenerativo. O Algoritmo 3 descreve a geração de ruído degenerativo para um fluxo de eventos.

Com este algoritmo, foram construídas versões da base N-MNIST para valores de probabilidade de degeneração de eventos entre 5 e 100% mostratos na Tabela 4. A Figura 27 exibe uma amostra da base D-Noisy-N-MNIST com evolução da superfície de eventos ativos no tempo em intervalos de $63ms$ para o mesmo caractere em diferentes níveis de ruído. A Tabela 4 mostra as variações criadas utilizando o Algoritmo 3.

Tabela 3 – Bases com ruído geradas a partir da N-MNIST

Base de Treino	Base de Teste	Probabilidade de Ruído
Train_001	Test_001	1%
Train_005	Test_005	5%
Train_010	Test_010	10%
Train_015	Test_015	15%
Train_020	Test_020	20%
Train_025	Test_025	25%
Train_030	Test_030	30%
Train_035	Test_035	35%
Train_040	Test_040	40%
Train_045	Test_045	45%
Train_050	Test_050	50%
Train_055	Test_055	55%
Train_060	Test_060	60%
Train_065	Test_065	65%
Train_070	Test_070	70%
Train_075	Test_075	75%
Train_080	Test_080	80%
Train_085	Test_085	85%
Train_090	Test_090	90%
Train_095	Test_095	95%
Train_100	Test_100	100%
Train_125	Test_125	125%
Train_150	Test_150	150%
Train_175	Test_175	175%
Train_200	Test_200	200%
Train_225	Test_225	225%
Train_250	Test_250	250%
Train_275	Test_275	275%
Train_300	Test_300	300%
⋮	⋮	⋮
Train_475	Test_475	475%

Fonte: Autor

Figura 26 – Amostra da base Noisy-N-MNIST contendo diferentes superfícies de eventos ativos obtidas após acumulo de eventos em diferentes instantes.

Fonte: Autor
Legenda: Cada linha corresponde à probabilidade de ruído aditivo p e cada coluna corresponde ao registro de tempo máximo dos evento na SAE.

Algoritmo 3 – Geração de ruído degenerativo

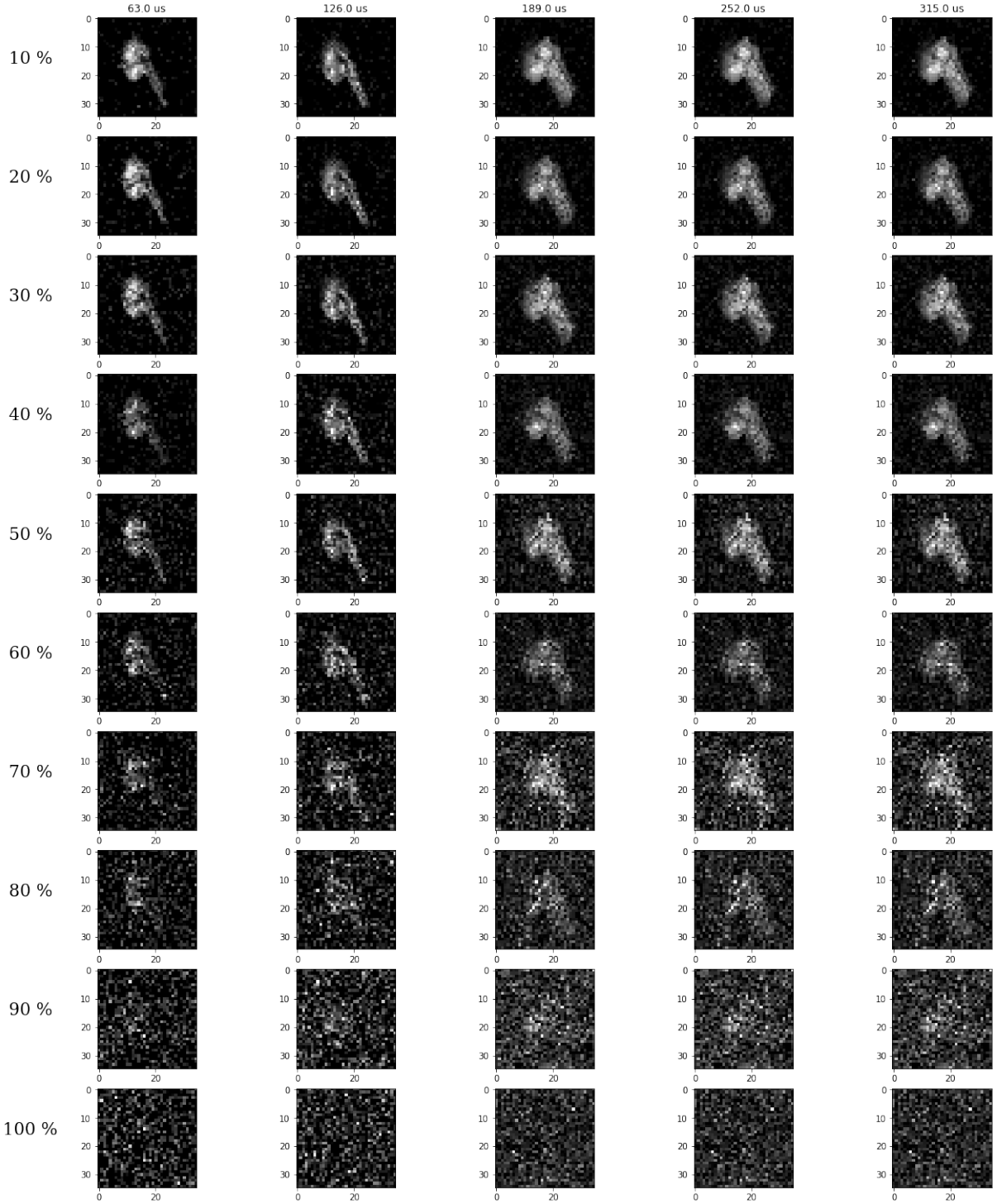
Data: P: Probabilidade de evento ruidoso
Data: E: Lista de eventos
Data: mX: Valor máximo admitido para X
Data: mY: Valor máximo admitido para Y
Result: R Lista de ventos com ruído
$R \leftarrow \{\}$;
forall e *in* E **do**
    R.append(e);
    r $\leftarrow$ random(0, 1.0);
    if *r > 1.0 -* P **then**
        $NE \leftarrow$ novoEventoAleatório(mX, mY, e.Ts) ;
        R.append(NE)
    end
    else
        R.append(e)
    end
end

Tabela 4 – Bases com ruído degenerativo geradas a partir da N-MNIST

Base de Treino	Base de Teste	Probabilidade de Ruído Degenerativo
Train_005	Test_005	5%
Train_010	Test_010	10%
Train_015	Test_015	15%
Train_020	Test_020	20%
Train_025	Test_025	25%
Train_030	Test_030	30%
Train_035	Test_035	35%
Train_040	Test_040	40%
Train_045	Test_045	45%
Train_050	Test_050	50%
Train_055	Test_055	55%
Train_060	Test_060	60%
Train_065	Test_065	65%
Train_070	Test_070	70%
Train_075	Test_075	75%
Train_080	Test_080	80%
Train_085	Test_085	85%
Train_090	Test_090	90%
Train_095	Test_095	95%
Train_100	Test_100	100%

Fonte: Autor

Figura 27 – Amostra da base D-Noisy-N-MNIST contendo diferentes superfícies de eventos ativos obtidas após acumulo de eventos em diferentes instantes.



Fonte: Autor
Legenda: Cada linha corresponde à probabilidade de ruído degenerativo p e cada coluna corresponde ao registro de tempo máximo dos evento na SAE.

4 Metodologia

Este capítulo descreve as metodologias utilizadas para reconhecimento de objetos utilizando sensores baseados em eventos. Tratam-se de duas metodologias que utilizam uma versão simplificada do operador $\text{LBP}^{riu2}_{P,R}$ e elas diferem em dois pontos. A primeira tira proveito do conhecimento do movimento da câmera, do qual os eventos podem ser separados em função do tempo, nesta tese, identificada por MSLBP (*Multi Saccade Local Binary Pattern*). A segunda não depende de conhecimento prévio do movimento e não leva em consideração tal separação, identificada por AHLBP (*Accumulated Histograms of Local Binary Patterns*).

Algumas etapas descritas neste capítulo são comuns a ambas as metodologias. Inicialmente na Seção 4.1 é descrita a metodologia MSLBP e as etapas correspondentes são descritas em suas subseções. Na Seção 4.2 é descrita a metodologia AHLBP, e suas etapas específicas são descritas em suas subseções.

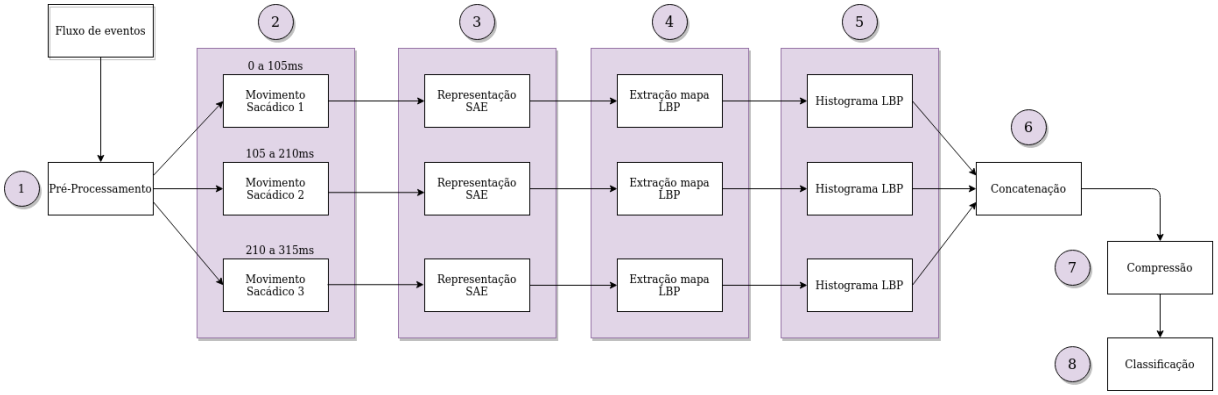
4.1 Metodologia MSLBP

A metodologia MSLBP tira proveito de uma propriedade existente nas bases de dados N-Caltech e N-MNIST, que é o fato de todos os objetos terem sido capturados utilizando o mesmo padrão de movimentos sacádicos. A Figura 28 mostra o diagrama geral da metodologia. Na etapa 1 é feito um pré-processamento da imagem para correção do movimento sacádico e eliminação de ruído. Na etapa 2 os eventos são separados em função do movimento sacádico realizado. Na etapa 3 é construída a representação dos eventos em superfícies de eventos ativos. Na etapa 4 é feita a extração dos padrões binários locais. Na etapa 5 é feita a construção dos histogramas de padrões binários locais em grade. Na etapa 6 é feita a concatenação dos histogramas em um histograma final. Na etapa 7 é feita a compressão dos vetores e na etapa 8 é feita a classificação utilizando classificador SVM.

4.1.1 Pré-processamento

A etapa de pré-processamento consiste na filtragem de ruído branco introduzido pelo próprio sensor de eventos, seguido de uma compensação do movimento sacádico da câmera quando aplicável. O procedimento para filtragem de ruído é o mesmo utilizado no trabalho

Figura 28 – Metodologia para classificação de objetos utilizando capturas com eventos



Fonte: Autor

proposto por Padala, Basu e Orchard (2018) e utilizado pelo método PCA-RECT (RAMESH et al., 2018).

A filtragem dos eventos é feita em duas etapas, a primeira etapa consiste em aplicar um filtro refratário. Trata-se de um mecanismo de inspiração biológica onde uma célula ou neurônio não exibem resposta imediata após um estímulo. Para que possa responder a um novo estímulo, tais células necessitam de um tempo denominado tempo refratário para retornar a um estado de repouso e só então apresentar resposta a um novo estímulo. Seja um evento $e = (x,y,ts,p)$ e seja $T^{anterior}_{(x,y)}$ o registro de tempo do evento mais recente em um fluxo de eventos em uma coordenada (x,y). Seja $\Delta T_{(x,y)}$ o período registrado entre o evento atual e o último evento válido em (x,y), conforme descrito pela Equação 23.

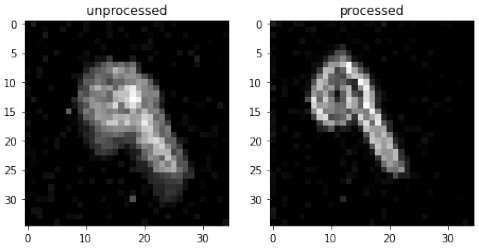
$$\Delta T_{(x,y)} = t - T^{anterior}_{(x,y)} \tag{23}$$

Seja um período refratário T_{ref}, um novo evento será considerado válido desde que satisfaça a Inequação 24.

$$\Delta T_{(x,y)} > T_{ref} \tag{24}$$

A segunda etapa consiste em um filtro baseado no vizinho mais próximo. Seja um n-ésimo evento $e^n_{(x,y)}$ com registro de tempo $T^n_{(x,y}$. Seja uma vizinhança de distância D a partir de $e^n_{(x,y)}$ e T^n o registro de tempo mais recente nesta vizinhança, excluindo $e^n_{(x,y)}$, será considerado válido um novo evento nesta vizinhança se a diferença de tempo entre $T^n_{(x,y)}$ e T^n não superar um limiar de corte T_{NNb}. Desta forma, um novo evento será considerado válido desde que satisfaça

Figura 29 – Exemplo de dado da base N-MNIST antes e depois do algoritmo de estabilização



Fonte: Autor

a Inequação 25.

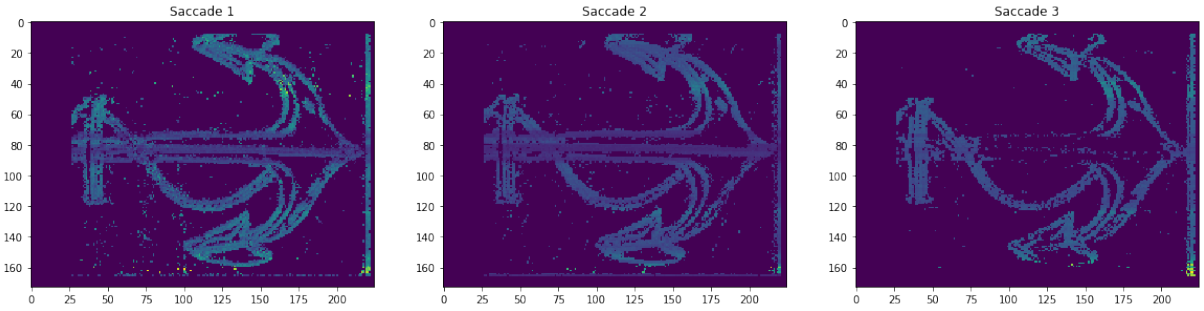
$$T^{n}_{(x,y)} - T^{n} < T_{NNb} \tag{25}$$

Para a etapa de estabilização, é aplicado uma algoritmo simples para compensar os movimentos realizados pela câmera durante a captura. Este algoritmo foi desenvolvido e disponibilizado por Orchard (2018), mesmo autor das bases N-Caltech e N-MNIST. Este algoritmo compensa os movimentos da câmera aplicando uma transformação espacial nas coordenadas dos eventos em função do intervalo de tempo correspondente a cada um dos três movimentos sacádicos realizados. Por conta disso, não se trata de um algoritmo de estabilização de uso geral e não pode ser aplicado a qualquer base de eventos. A Figura 29 mostra um exemplo um captura da base N-MNIST com e sem aplicação da estabilização.

4.1.2 Separação dos movimentos sacádicos

Esta etapa envolve a separação dos eventos em 3 movimentos sacádicos. Durante a captura, a câmera montada sobre um aparato eletromecânico, executa um movimento de padrão triangular conforme descrito na Seção 3.8. Cada movimento sacádico leva aproximadamente 105 ms, totalizando 315 ms por captura. Como o sensor de eventos é sensível a alterações locais de intensidade, é esperado que alguns padrões espaciais sejam atenuados quando o objeto se movimenta em uma determinada direção. A Figura 30 mostra uma quantidade menor de eventos no terceiro movimento sacádico na região central da imagem. O terceiro movimento que é realizado na direção horizontal, tende a amplificar eventos em estruturas alinhadas verticalmente e atenuar eventos em estruturas orientadas horizontalmente.

Figura 30 – Exemplo de superfícies de eventos ativos separadas por movimento sacádico



Fonte: Autor

4.1.3 Representação da SAE utilizando tabelas *hash*

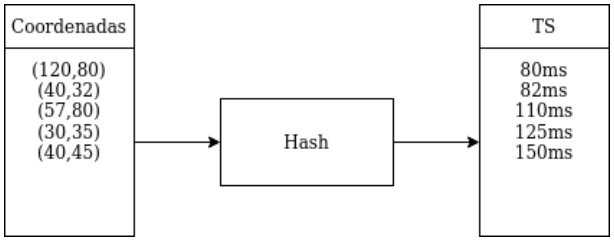
A aplicação do LBP em dados de eventos requer a construção de uma representação $2D$ contendo as ocorrências de eventos em um intervalo de tempo. No entanto, eventos não possuem valor de intensidade e estão limitados a somente dois possíveis valores correspondentes às polaridades positiva ou negativa. Uma representação adequada para extração de informação do fluxo de ventos utilizando LBP é o uso das chamadas Superfícies de Eventos Ativos.

Conforme descrito na Seção 3.1.2, cada sequência capturada pode ser dividida em subsequências separadas com duração $\Delta_t ms$ e representada em uma superfície 2D, de modo que para um par de coordenadas (x,y) exista somente um valor correspondente, seja o valor de tempo do evento mais recente, a somatória de tempos naquela coordenada ou outra forma de representação.

A representação da SAE comumente é feita como uma matriz, tal como representações de imagens de câmeras convencionais. Em contraste com as principais propostas, neste trabalho, a representação em memória da SAE é feita usando tabelas *hash*, onde as coordenadas (x,y) são combinadas para formar a chave e o registro de tempo do evento TS é o valor armazenado. A Figura 31 mostra uma representação dessa estrutura.

A principal vantagem de se mapear os eventos e padrões binários locais neste formato é o baixo consumo em memória, pois a ausência de uma chave (x,y) pode ser usada para representar um valor 0 ou a ausência de um registro de tempo naquela coordenada. Nesta forma, a informação de polaridade dos eventos não é preservada. Isso pode ser contornado utilizando duas tabelas

Figura 31 – Representação da superfície de eventos ativos como uma tabela utilizando uma função *hash*.



Fonte: Autor.
Legenda: O par de coordenadas (x,y) define a chave e o tempo do evento TS é o valor armazenado

hash, sendo uma para cada polaridade, ou adotando-se a polaridade como parte da chave que passaria a ter o formato (x,y,p).

Dada uma SAE obtida a partir de uma sequência limitada de movimentos sacádicos, é simples demonstrar que uma parte considerável da informação corresponde a ausência de eventos. Considerando duas bases de *benchmark* N-MNIST e N-Caltech, utilizadas nesta tese. Seja TP_e o total de pixels na SAE que possui pelo menos 1 evento, e seja TP o total de pixels da SAE, a relação de eventos pela área total de todas as capturas c de uma base B pode ser dada pela Equação 26.

$$TP_e/TP \tag{26}$$

Os resultados desse levantamento são mostrados na Tabela 5. Sem aplicação de pré-processamento, quando os eventos são projetados em uma superfície de eventos ativos, a área preenchida total corresponde a $66{,}028\%$ para a base N-Caltech e $66{,}028\%$ para a base N-MNIST. Após a aplicação do pré-processamento, esses números caem para $42{,}562\%$ e $42{,}563\%$, respectivamente. Estes valores são obtidos considerando a captura completa, totalizando $315ms$ de gravação para cada objeto e com total acúmulo de ruído branco coletado durante a captura. Nestas condições, mais da metade dos eventos capturados corresponde a silêncio. É esperado, por tanto, que quando as superfícies de eventos ativos são quantizadas em intervalos menores de tempo, tal como discutido na Seção 4.1.2, essas porcentagens sejam inferiores. Esta observação justifica a adoção de tabelas *hash* para representação de superfícies de eventos ativos como solução para reduzir o uso em memória.

Tabela 5 – Razão de eventos em relação ao total de eventos representados em uma SAE

Pré-processamento	N-Caltech	N-MNIST
Não	66,028%	66,028%
Sim	42,562%	42,563%

Fonte: Autor

4.1.4 Extração dos padrões binários locais uniformes em estrutura esparsa

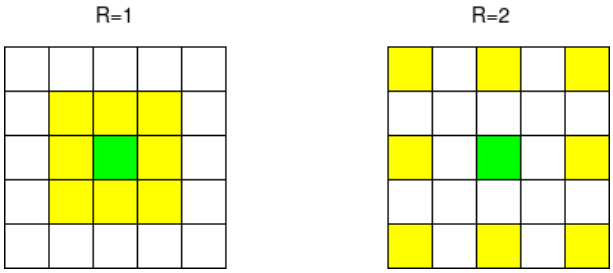
A representação da superfície de eventos ativos em tabelas *hash* tem como principal benefício o baixo uso em memória, especialmente em dados esparsos. Tradicionalmente, o algoritmo de extração de padrões LBP executa uma varredura por todos os pixels da imagem realizando as comparações locais em cada uma das vizinhanças com o pixel central em (x,y). Em uma estrutura esparsa representada por uma tabela *hash*, uma parte considerável de tais comparações representa um desperdício computacional.

Como parte das contribuições, esta tese apresenta uma variação do algoritmo de extração de padrões binários locais de uma superfície aplicado a dados esparsos em uma estrutura de tabelas *hash*, denominado Sparse-LBP. Este algoritmo adota como premissa a necessidade de realizar somente as comparações com centro (x,y) nas localidades dos valores na superfície e no conjunto de localidades (x,y) correspondentes às vizinhanças dos respectivos valores. Desta forma, evitam-se comparações em vizinhanças que não possuem nenhum valor.

Seja uma SAE S em formato de tabela *hash* $(x,y) \longrightarrow v$, para cada chave $c = (x,y) \in S$, devem ser feitas as comparações de cada valor vizinho a c com o valor da chave c. Para manter consistência com o algoritmo LBP original, os pixels vizinhos a c também devem ser avaliados. Em uma vizinhança retangular como a que é adotada nesta tese, é possível calcular o bit resultante para cada posição vizinha de c em função de c, eliminando assim verificações adicionais nestas posições. Para isso, utiliza-se um mapa auxiliar de pesos de modo que cada posição (x,y) vizinha a c possua o peso de seu bit correspondente e o peso que a posição central representa em relação ao pixel vizinho.

Seja uma superfície de eventos ativos S representado por uma tabela *hash*, para cada chave $c = (x,y) \in S$ é obtido o valor central v. Seja n uma estrutura auxiliar que representa um pixel vizinho a c no formato $n = \{\Delta_x, \Delta_y, w, o\}$, onde Δ_x é a distância horizontal entre n e e, Δ_y é a distância vertical entre n e o, w é o bit correspondente do padrão LBP resultante em (x,y) e o o bit resultante em $(x + n_{\Delta_x}, y + n_{\Delta_y})$. Seja N uma lista representando a vizinhança completa em torno de c. Seja $n_p = (x + n_{\Delta_x} y + n_{\Delta_y})$ e seja v_n o valor obtido em $S(n_p)$, podemos obter o

Figura 32 – Detalhamento do parâmetro R



Fonte: Autor.
Legenda: Mapeamento das vizinhanças ao pixel central em verde, marcadas em amarelo para $R = 1$ e $R = 2$.

valor LBP para cada coordenada $c = (x,y) \in S$ a partir de comparações de v com cada valor v_n. Seja L o mapa de LBP resultante da extração em formato de tabela *hash*. Seja l o valor resultante das comparações nas vizinhanças em torno de c. Se $v > v_n$ então l é acrescido de n_w. Caso contrário, $L(n_p)$ é acrescido de n_o. Esta operação não deve ser realizada para todo $n_p \in S$ uma vez que o bit correspondente a n pode já estar setado em $L(np)$ para os casos onde $np \in S$, e a soma de n_o em resultaria em valores inconsistentes.

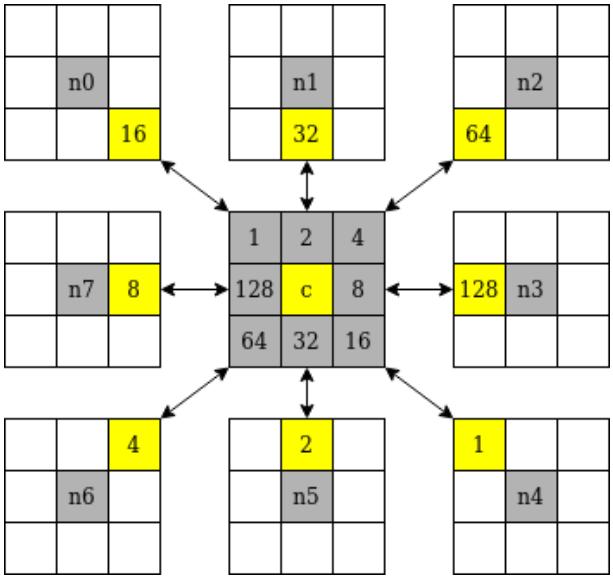
O modelo de vizinhança empregado nesta tese utiliza uma versão simplificada do $\text{LBP}^{riu2}_{P,R}$ com um padrão retangular de oito posições independente do raio, para evitar cálculos de interpolação que podem elevar o custo computacional da extração. O parâmetro R define a distância em x e y a partir do pixel central. A Figura 32 mostra o mapeamento dos pixels vizinhos a um pixel central em função de R nesta forma simplificada.

Com a distância R definida, é possível construir o mapeamento de pesos e pesos opostos. A Figura 33 mostra uma vizinhança com $R = 1$ com o mapeamento de pesos. Os pesos de cada bit na posição c são mapeados em cinza. Os pesos dos bits de c nas posições vizinhas estão marcadas em amarelo.

A partir deste mapeamento, é possível construir N conforme mostra a Tabela 6.

O Algoritmo 4 mostra uma implementação em pseudo-código do funcionamento do Sparse-LBP. A partir o mapa resultante L, é necessário executar mais uma iteração por todas as posições $(x,y) \in L$ para converter os valores de LBP em padrões uniformes previamente mapeados, conforme descrito na Seção 3.2.2.

Figura 33 – Mapeamento de pesos e vizinhanças com raio $R = 1$.



Fonte: Autor.
Legenda: Os pesos para os bits vizinhos a um pixel central c são marcados em cinza. O peso de c em cada uma das localizações vizinhas é mapeado em amarelo.

Tabela 6 – Construção de tabela N para uma vizinhança retangular de 8 posições de tamanho $R \times R$

Posição	Δ_x	Δ_y	w	o
n_0	$-R$	$-R$	1	16
n_1	0	$-R$	2	32
n_2	R	$-R$	4	63
n_3	R	0	8	128
n_4	R	R	16	1
n_5	0	R	32	2
n_6	$-R$	R	64	4
n_7	$-R$	0	128	8

Fonte: Autor

4.1.4.1 Custo computacional

O processo de extração de padrões LBP ocorre somente nas localizações onde existem valores e nas localidades que correspondem à vizinhança destes valores. Uma vez que cada chave é avaliada uma única vez, podemos definir a ordem de complexidade do Sparse-LBP como $O(N \times P)$ para o pior caso, onde N é número de chaves na SAE e P é o número de posições vizinhas a serem avaliadas. A Tabela 7 mostra o custo e número de repetições de cada instrução do algoritmo Sparse-LBP.

Algoritmo 4 – Sparse-LBP - Extração esparsa de mapa de padrões binários locais

```
1  Data: S: Tabela hash (x,y) ⟶ v
2  Data: R: Raio
3  Data: P: Número de posições vizinhas
4  Result: L: Mapa de padrões LBP
5  forall c ∈ S do
6      N = buildNeighborhood(c, P, R) ;
7      v = S(c) ;
8      l = 0 ;
9      forall n ∈ N do
10         n_p = (c_x − n_{Δ_x}, c_y + n_{Δ_y}) ;
11         v_n = S(n_p) ;
12         if v > v_n then
13             l = l + n_w ;
14         end
15         else
16             if n_p ∉ S then
17                 L(n_p) = L(n_p) + n_o ;
18             end
19         end
20     end
21     L(c) = l ;
22 end
```

Tabela 7 – Análise de custo do algoritmo Sparse-LBP

Linha	Instrução	Custo	Frequência
5	**forall** $c \in S$ {	c1	$n+1$
6	N = buildNeighborhood(c,P,R)	c2	n
7	$v = S(c)$	c3	n
8	$l = 0$	c4	n
9	**forall** $n \in N$	c5	$n(\cdot P+1)$
10	$n_p = (c_x - n_{\Delta_x}, c_y + n_{\Delta_y})$	c6	$n \cdot P$
11	$v_n = S(n_p)$	c7	$n \cdot P$
12	**if** $v > v_n$ **then**	c8	$n \cdot P$
13	$l = l + n_w$	c9	$n \cdot P$
14	**end**	c10	0
15	**else**	c11	0
16	**if** $n_p \notin S$ **then**	c12	$n \cdot P$
17	$L(n_p) = L(n_p) + n_o$	c13	$n \cdot P$
18	**end**	c14	0
19	**end**	c15	0
20	**end**	c16	0
21	$L(c) = l$	c17	n

Fonte: Autor

A partir do levantamento dos custos e frequências, o custo do algoritmo $T(n)$ pode ser descrito conforme a Equação 27.

$$T(n) = c1(n+1) + n(c2+c3+c4+c17) + c5(n(\cdot P+1)) + (n \cdot P)(c6+c7+c8+c9) \quad (27)$$

Aplicando-se a distributivas de $n(\cdot P + 1)$, podemos reescrever o custo $T(n)$ conforme a Equação 28

$$T(n) = c1(n+1) + n(c2+c3+c4+c17) + c5(n \cdot P+n) + (n \cdot P)(c6+c7+c8+c9) \quad (28)$$

Por fim, aplicando a distributiva do custo $c5$, temos um termo $c5 \cdot n$ e um termo $c5(n \cdot P)$, e desta forma podemos reescrever $T(n)$ conforme a Equação 29.

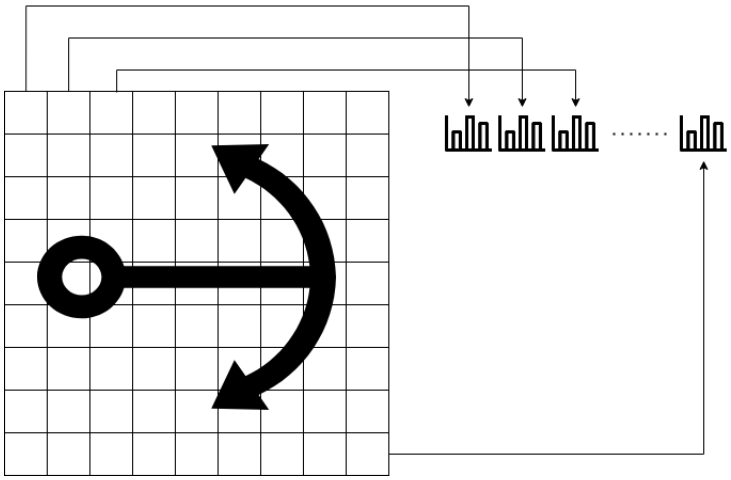
$$T(n) = n(c1+c2+c3+c4+c5+c17) + (n \cdot P)(c5+c6+c7+c8+c9) + c1 \quad (29)$$

O termo $(n \cdot P)(c5+c6+c7+c8+c9)$ sugere um limite assintótico $O(N \times P)$. Deste modo, o impacto de P não pode ser desprezado uma vez que os custos $c5$, $c6$, $c7$, $c8$ e $c9$ dependem do mesmo. No entanto, em aplicações práticas P é uma constante de tamanho fixo referente somente ao operador LBP adotado normalmente inferior a n, podendo ser interpretado como um custo fixo. Uma vez fixado P, o custo computacional do Sparse-LBP passa a depender somente de n e, desta forma, podemos considerar o seu custo computacional como $O(n)$.

4.1.4.2 Comparação com extração na versão matricial

Em sua implementação original (OJALA; PIETIKAINEN; HARWOOD, 1994), a extração do mapa de LBP tem ordem de complexidade $O(N \times M \times P)$ onde N é o número de linhas, M é o número de colunas da imagem e P é o número de posições vizinhas. Tal como no Sparse-LBP, P pode ser também interpretado como um custo fixo e desta forma $O(N \times M)$. Apesar de linear, há um custo elevado em memória, na ordem de $O(N \times M)$. Em casos de dados esparsos como os encontrados em cenas capturadas com poucos eventos, isso representa um uso de espaço desnecessário e processamento de operações redundantes em áreas sem atividade. Tais afirmações são sustentadas experimentalmente. Uma vez que o Sparse-LBP depende da SAE construída como uma tabela *hash*, o pior cenário para seu uso seria em uma situação onde todos as chaves correspondentes às coordenadas (x,y) válidas em uma SAE estivessem ocupadas. Nestas condições a extração dos padrões de LBP seria equivalente ao de uma varredura completa com o custo adicional do cálculo da função *hash*. É possível, portanto, que exista um valor para

Figura 34 – Extração de histogramas em grade a partir de uma superfície de eventos ativos.



Fonte: Autor
Legenda: Um histograma de 59 posições é extraído para cada janela de tamanho $L \times L$. Todos os histogramas são concatenados em um histograma final que representa a superfície.

a taxa de ocupação, tal como definida na Seção 4.1.3, para a qual o custo computacional do Sparse-LBP se iguala ao custo de extração em estrutura matricial com varredura completa. Este cenário é explorado experimentalmente com dados sintéticos.

4.1.5 Extração dos histogramas de LBPs

Com os mapas de LBP extraídos, são então obtidos os histogramas de padrões binários locais. Os mapas são divididos em $N \times M$ regiões de amostragem de tamanhão $L \times L$ pixeis. Para cada coordenada (X,Y) válida, é extraído o padrão $\text{LBP}^{riu2}_{P,R}$ com raio R e seu valor registrado no histograma da região de amostragem (n,m).

Como os histogramas são extraídos a partir de regiões de amostragem retangulares, para levantar todas as ocorrências de valores, seria necessário consultar todas as coordenadas (x,y) da região, o que não é vantajoso em estruturas esparsas e pode aumentar o tempo de processamento das capturas. Desta forma, o histograma local pode ser obtido também de forma esparsa. Seja L uma tabela *hash* com chave (x,y) contendo valores de LBP extraídos usando o algoritmo Sparse-LBP descrito na Seção 4.1.4. Para cada chave em L, obtém-se as janelas de amostragem nas quais o respectivo valor deve ser registrado nos histogramas h.

Operando desta forma, elimina-se a necessidade de processamentos redundantes para coordenadas ausentes em L que podem ser interpretadas como valor de LBP 0. No entanto, para

estar em conformidade com o algoritmo original, as ocorrências de valor 0 também devem ser contabilizadas nos histogramas. Isso pode ser obtido com complexidade $O(1)$ descontando o total de possíveis valores $(n \times m)$, de todas as ocorrências já registradas, inclusive os valores 0 que estão presentes atualizando portanto o histograma na posição 0 (h'_0), conforme mostra a Equação 30 onde k é o tamanho do histograma.

$$h'_0 = h_0 + ((n \cdot m) - \sum_{i=0}^{k} h_i) \tag{30}$$

4.1.6 Concatenação dos histogramas de LBPs

Os histogramas dos três movimentos sacádicos são combinados em um histograma final, com a representação completa do objeto capturado. Com base nos parâmetros N e M é possível determinar o tamanho final do histograma dado pela Equação 31, onde N e M são o número de linhas e o número de colunas de amostragem, h é o tamanho do histograma de LBP e s é o número de movimentos sacádicos realizados.

$$H = N \times M \times h \times s \tag{31}$$

O tamanho final não depende portanto da janela de amostragem de tamanho L. No entanto, aumentar as linhas ou colunas de amostragem pode resultar em extração de padrões em uma área maior que a desejada caso não seja admitida sobreposição de janelas de amostragem. Se a sobreposição de janelas de amostragem for admitida, aumentar N ou M pode resultar em obtenção de valores redundantes com o custo de um vetor maior. A escolha de N e M deve ser considerada com base no tamanho das áreas e a relação custo benefício entre o tamanho do vetor e a a precisão da classificação. Neste caso, pode ser possível uma diminuição do tamanho da janela de amostragem L.

4.1.7 Compressão

A concatenação de histogramas de LBP uniformes, mesmo que utilizando somente os valores distintos correspondentes aos padrões uniformes, pode resultar em um descritor de tamanho elevado. Seja um vetor de características H concatenado obtido com a seguinte configuração: 3 movimentos sacádicos, 20×16 janelas de amostragem, 2 raios de LBP e histograma de tamanho 59, conforme definição de padrões uniformes descritos na Seção 3.2.2. O

tamanho final de H é um histograma de 113280 posições. Tal configuração é computacionalmente custosa para o classificador SVM.

Para tornar o problema da classificação menos custoso, esta tese propõe o uso de um método de compressão que leva em consideração o descarte de padrões LBP que não ocorrem em nenhuma cena observada em uma mesma grade de amostragem e uma transformação em um espaço de características envolvendo uma etapa de escalonação seguida da aplicação do algoritmo PCA.

A escalonação envolve uma simples binarização que substitui valores com ocorrência maior que 0 por um valor arbitrário. Seja uma histograma H de N posições e um resultante H' também de tamanho N e seja V_i o valor do histograma H na posição i e V'_i o valor no vetor de características H' também na posição i. O novo valor V'_i a ser armazenado em H' é dado pela Equação 32.

$$V'_i = \begin{cases} 1 \; se \; V_i > 0 \\ 0 \; se \; V_i = 0 \end{cases} \tag{32}$$

Ao vetor escalonado é aplicado o algoritmo PCA para redução de de dimensões e transformação dos pontos característicos em um novo espaço de representação. A eficácia da redução é demonstrada experimentalmente no Capítulo 5.

4.1.8 Classificação

Dada uma cena descrita, pelo histograma definido na Seção 3.3 e comprimido utilizando o método, os histogramas são utilizados em um modelo SVM multi-classe para classificação dos objetos com *kernel* RBF, conforme descrito na Seção 3.5.3.

4.1.8.1 Calibração dos modelos

Os modelos SVM resultantes dependem de três parâmetros a serem determinados: o número de componentes do PCA para etapa de compressão, o parâmetro C e o parâmetro γ da função radial utilizada como *kernel* de transformação. A busca exaustiva pelos valores destes parâmetros para se obter uma boa classificação é computacionalmente custosa. Para reduzir o tempo de busca foi utilizada uma busca heurística com algoritmo genético adotando maximização da precisão da classificação como função de otimização.

4.2 Metodologia AHLBP

A segunda metodologia proposta para trabalhar com padrões binários locais utilizando sensores baseados em eventos envolve a criação de várias superfícies de eventos ativos separadas em intervalos de tempo, extração de padrões binários locais, e acúmulo das ocorrências de padrões LPB em grade.

Esta metodologia possui algumas diferenças em relação à metodologia MSLBP descrita na Seção 4.1. Enquanto a metodologia MSLBP tira proveito dos movimentos conhecidos, esta metodologia não depende do movimento do sensor. Outra diferença se dá pelo fato de que as superfícies são formadas exclusivamente por soma dos eventos mais recentes. Além disso, os eventos são separados em função de sua polaridade.

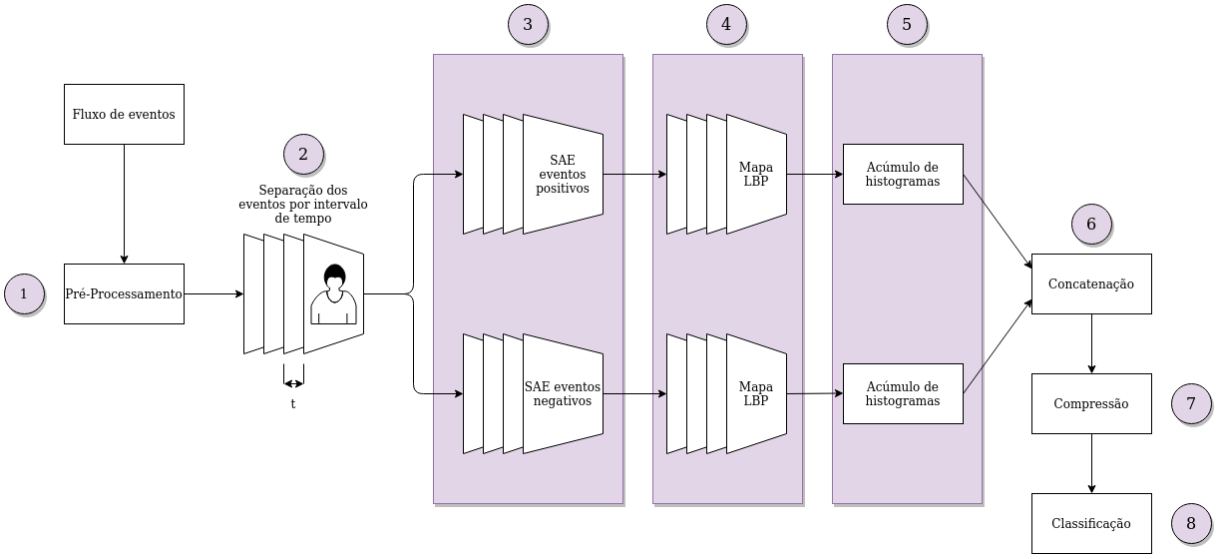
O mesmo procedimento de estabilização dos movimentos e filtragem de ruído é aplicado ao fluxo de eventos antes da formação das superfícies quantizadas. A representação das superfícies de eventos ativos, extração dos padrões binários locais e construção dos histogramas é feita da mesma forma, conforme descrito nas Seções 4.1.3, 4.1.4 e 4.1.5, respectivamente. No entanto, a construção das superfícies não adota separação por movimentos sacádicos, mas por intervalos de tempo fixos.

A Figura 35 mostra uma visão geral desta metodologia. Na etapa 1 é realizado o mesmo procedimento de pré-processamento descrito na Seção 4.1.1. Na etapa 2 é feita a separação dos eventos em intervalos de tempos regulares. Na etapa 3 é feita a construção das superfícies de eventos ativos separadas por polaridade. Na etapa 4 são extraídos os padrões binários locais utilizando o Algoritmo 4 descrito na Seção 4.1.4. Na etapa 5 são extraídos e acumulados os histogramas de padrões binários locais. Na etapa 6 é feita a concatenação dos histogramas obtidos. Na etapa 7 é realizado o procedimento de compressão descrito na Seção 4.1.7 e na etapa 8 é feita a classificação.

4.2.1 Separação dos eventos em intervalos de tempo

Diferente da metodologia MSLBP que separa os eventos em função dos movimentos realizados pelo sensor, a metodologia AHLBP inicialmente divide os eventos em intervalos de tempo Δt regulares. A separação ainda é possível utilizando valores corretos para Δt, mas as superfícies de eventos ativos construídas não são tratadas de forma isolada como na metodologia MSLBP, de modo que duas superfícies afetam a mesma seção do histograma final.

Figura 35 – Diagrama geral da metodologia AHLBP



Fonte: Autor

4.2.2 Construção das superfícies de eventos ativos

Nesta metodologia os eventos são separados em função da polaridade, positiva ou negativa. A mesma estratégia é aplicada ao método HATS (SIRONI et al., 2018) que separar as superfícies de tempo em função da polaridade.

Conforme descrito na Seção 4.1.3, para separação dos eventos em função da polaridade, esta metodologia utiliza duas tabelas *hash*, uma para eventos positivos e uma para eventos negativos, conforme mostra a Figura 35 item 3.

4.2.3 Acúmulo de histogramas LBP

Tal como no método MSLBP, os histogramas são obtidos em formato de grade, conforme descrito na Seção 4.1.5. No entanto, como as superfícies de eventos são processadas em sequencia, é necessária a manutenção de dois mapa de $N \times M$ histogramas de 59 posições, até que a última superfície de tempo tenha sido processada.

4.2.4 Concatenação dos histogramas

Nesta etapa os histogramas acumulados separados por polaridade são concatenados no histograma final representando o objeto. Uma vez que são utilizadas duas polaridades, o tamanho final do histograma é dado pela Equação 33

$$H = N \times M \times h \times 2 \tag{33}$$

Esta metodologia produz um histograma relativamente menor do que a metodologia MSLBP descrita na Seção 4.1 mas ainda assim de tamanho considerável para treinamento de modelos SVM. No entanto, com a aplicação do método de compressão descrito na Seção 4.1.7, o vetor final pode ser reduzido um tamanho na ordem centenas de valores distintos.

5 EXPERIMENTOS

Este capítulo descreve os experimentos realizados para validação da plausibilidade da adoção de padrões binários locais para reconhecimento de objetos, além de validação de complexidade do algoritmo Sparse-LBP.

5.1 CUSTO COMPUTACIONAL DO ALGORITMO SPARSE-LBP

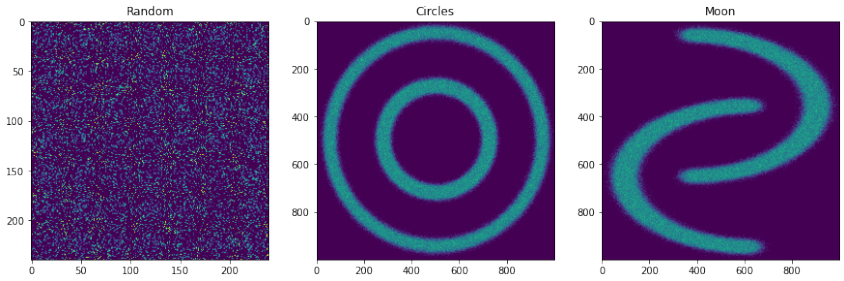
Para avaliar a complexidade do algoritmo Sparse-LBP, foram realizados três experimentos com dados sintéticos. Foram gerados eventos aleatórios esparsos, eventos em formato de círculo e eventos em formato de luas utilizando geradores de pontos. Os eventos gerados possuem coordenadas (x,y) e registro de tempo t. A polaridade dos eventos não é necessária para medição dos tempos e não foi considerada para estes experimentos.

Para os experimentos com ruído aleatório, foi utilizado um gerador que recebe o tamanho n da matriz resultante $n \times n$ contendo a superfície de eventos ativos, além de um parâmetro de probabilidade de geração de eventos p ajustada em 20%. Para cada coordenada (x,y) em uma matriz $n \times n$, é testada a possibilidade de geração de um evento com um valor aleatório entre 0 e 1. Se este for superior a $1 - p$, é gerado um evento aleatório.

Para os experimentos com dados em formato de círculos e luas foram utilizados geradores de dados presentes na biblioteca *scikit-learn*. Estes geradores costumam ser utilizado em demonstrações de algoritmos de agrupamento e geram pares de valores i,j entre -1 e 1 com dois rótulos que são desprezados. Com base nestes geradores, foram criados geradores de eventos, que recebem como parâmetro a quantidade de eventos n a gerar. Os algoritmos geradores retornam uma matriz de tamanho $n \times n$ contendo os eventos gerados, bem como sua representação esparsa em formato de tabela *hash* conforme descrito na Seção 4.1.3. Durante a geração dos eventos, os mesmos são escalonados e suas coordenadas corrigidas, de modo que os valores mínimos e máximos sejam $(0,0)$ e (n,n), respectivamente. Todos os geradores foram ajustados para gerar eventos ocupando em torno de 20% da área total. A Figura 36 mostra alguns exemplos de dados sintéticos usados nos testes.

Com os dados gerados são executadas as versões matricial e esparsa do algoritmo LBP para cada valor total de chaves na tabela *hash* e são registrados os tempos em segundos e espaços ocupados em unidades de memória para cada versão do algoritmo. O número de chaves é usado

Figura 36 – Exemplos de dados aleatórios, em formato de círculos e luas, gerados para os experimentos de validação da complexidade e análise temporal



Fonte: Autor

no lugar de n uma vez que a quantidade de eventos na superfície resultante pode ser menor que n, caso dois ou mais eventos gerados ocupem posições iguais.

5.1.1 Impacto da taxa de ocupaçãp

Além dos experimentos com superfícies sintéticas, também foi avaliado o impacto da taxa de ocupação dos valores na SAE. Para este experimento, foi utilizado uma superfície de tamanho fixo (1000×100) com eventos aleatórios sendo inseridos tanto na estrutura matricial quanto na SAE em formato de tabela *hash*. Os mapas de padrões LBP para ambos os formatos da SAE foi extraído com a versão matricial e com o Sparse-LBP e os tempos de execução foram registrados em função da taxa de ocupação.

5.2 CLASSIFICAÇÃO BASE N-CALTECH

Os experimentos descritos nesta seção consistem na verificação da acurácia do classificação de objetos utilizando as metodologias propostas. Os resultados são então comparados com metodologias existentes.

Para a classificação de objetos foi utilizada a base N-Caltech descrita na Seção 3.8.1. A base foi dividida seguindo a razão 66% para treino e 33% para teste, respeitando as proporções de cada classe. A classe *Background* não foi utilizada nestes experimentos.

Para todos os experimentos onde o pré processamento pode ser aplicado, foram utilizados os mesmos parâmetros de filtragem, conforme mostra a Tabela 8, conforme recomendado por Padala, Basu e Orchard (2018).

Tabela 8 – Parâmetros de pré processamento

Parâmetro	Valor
T_{ref}	$1ms$
T_{NBB}	$5ms$

Fonte: Autor

O tamanho da janela utilizado foi 16×16 pixels. A grade de amostragem adotada foi de 16 linhas por 20 colunas e raios para para extração dos padrões LBP de tamanho 1 e 2. Para o algoritmo AHLBP foi adotado um intervalo de $\Delta t = 50ms$ utilizado na construção das superfícies de eventos ativos.

Para o método MSLBP nos experimento envolvendo classificação foram utilizadas as duas formas de representação da superfície de eventos ativos descritas na Seção 3.1.2. A primeira adotando o registro de tempo do evento mais recente em uma coordenada (x,y) identificada por MSLBP e a segunda como a soma de registros de tempo em cada coordenada identificada como MSLBP-AS. Nestas metodologias também foram filtrados os eventos que correspondem a fundo da imagem, com base nas anotações fornecidas.

Para ajuste dos parâmetros C e γ do algoritmo SVM, bem como o número de componentes n utilizados pelo algoritmo PCA, foi utilizado o método de calibração descrito na Seção 4.1.8.1. A Tabela 9 mostra os parâmetros calibrados para os modelos classificadores.

Tabela 9 – Parâmetros calibrados para classificação utilizando a base N-Caltech

Parâmetro	MSLBP	MSLBP-AS	AHLBP
n	169	189	161
C	662,7611	504,80	556,4911
γ	0,004118	0,02846	0,003390

Fonte: Autor

5.3 CLASSIFICAÇÃO BASE N-MNIST

Para a classificação de dígitos manuscritos foi utilizada a base N-MNIST. Ao contrário da base N-Caltech, a base N-MNIST já é separada em conjunto de treino e conjunto de teste, não sendo necessária a realização de divisão da base ou estratificação.

Neste experimento foram utilizados valores diferentes entre os métodos MSLBP e AHLBP para janelas de amostragem e tamanho. Estes parâmetros foram determinados de forma empírica

em testes preliminares. Foram utilizados raios para o LBP 1 e 2. Para o algoritmo AHLBP foi adotado um intervalo de $\Delta t = 50ms$ para construção das superfícies de eventos ativos.

O mesmo método de calibração utilizado nos experimentos com a base N-Caltech foi aplicado nos experimentos com a base N-MNIST, bem como os mesmos parâmetros de filtragem. A Tabela 10 mostra os parâmetros utilizados nos modelos classificadores.

Tabela 10 – Parâmetros calibrados para classificação utilizando a base N-MNIST

Parâmetro	MSLBP	MSLBP-AS	AHLBP
$janela$	16×16	16×16	7×7
$grade$	4×4	4×4	5×5
C	721,25	721,25	5836,05
n	298	298	172
γ	0,00312	0,00312	0,0046

Fonte: Autor

5.4 CLASSIFICAÇÃO BASE N-CARS

Para classificação na base N-CARS somente o método AHLBP foi utilizado, uma vez que nesta base não são realizados movimentos sacádicos pela câmera. O tamanho da janela utilizado foi 16×16 pixels. A grade de amostragem adotada foi de 5 linhas por 5 colunas e os raios para o LBP adotados foram 1 e 2. O intervalo de tempo para quantização das superfícies de tempo de $100ms$, similar ao adotado no método HATS.

Tal como naos experimentos com a base N-Caltech e N-NNIST, foi aplicado o processo de calibração descrito na Seção 4.1.8.1 para determinação do número de componentes PCA e valores C e γ do modelo SVM com *kernel* RBF. Os valores utilizados são mostrados na Tabela 11. OS dados foram pré processados com os mesmos parâmetros utilizados nos experimentos anteriores, porém sem realização de estabilização de movimentos, uma vez que os movimentos realizados não são conhecidos.

Tabela 11 – Parâmetros calibrados para classificação utilizando a base N-CARS e método AHLBP

Parâmetro	Valor
C	3,32
n	104
γ	0,0091285

Fonte: Autor

5.5 RESISTÊNCIA A RUÍDO

Para verificar a resistência das metodologias a ruído aditivo, os experimentos de classificação da base N-MNIST foram realizados com as bases Noisy-MNIST e D-Noisy-NMNIST geradas para esta tese, descritas nas Seções 3.8.4 e 3.8.5. Para cada valor de probabilidade de ruído aditivo p foram extraídos os dados nas mesmas condições do experimento descrito na Seção 5.3. Após a extração, foram treinados e avaliados modelos utilizando compressão PCA e classificado SVM, com os mesmos parâmetros C , γ e N calibrados no experimento descrito na Seção 5.3. Foram registrados os valores de acurácia do classificador para cada valor de p para comparação.

Neste experimento o método MSLBP também é avaliado com as duas formas de representação da superfície de eventos ativos descrita na Seção 3.1.2, identificadas por MSLBP e MSLBP-AS, tal como nos experimentos descritos na Seção 5.2.

As metodologias MSLBP e AHLBP foram comparadas com o algoritmo HATS, que foi selecionado como sendo a metodologia convencional que possui melhor precisão na base N-MNIST, segundo levantamento bibliográfico desta tese na Seção 2.2. Para o experimento utilizando a base Noisy-MNIST, foi também realizada uma análise de tempo, uma vez que o número de eventos acrescidos é proporcional à probabilidade de geração de ruído aditivo. Para a comparação de tempo o algoritmo HATS foi executado com as superfícies de tempo τ_{e_i} em formato matricial e com uso de tabelas *hash*. A versão com tabelas hash é identificada nos resultados como HATS-SPARSE.

Os parâmetros utilizados no método HATS foram os mesmos reportados no artigo original (SIRONI et al., 2018) com exceção do parâmetro ρ e são mostrados na Tabela 12.

Tabela 12 – Parâmetros utilizados para o método HATS

Parâmetro	Valor
K	5
ρ	4
$\tau(\mu s)$	10^{11}
$\Delta t(ms)$	100

Fonte: Autor

O parâmetro ρ originalmente reportado como 2 precisou ser alterado devido a uma limitação na função que constrói as superfícies de tempo locais com base nos eventos selecionados a partir do mapeamento das células de memória. O algoritmo não permite o registro direto de

um evento no histograma caso a relação entre a largura ou altura da nela dividido pelo tamanho da célula K seja maior ou igual a $2 \cdot \rho + 1$, o que é verdade para $K = 5$, $\rho = 2$ e o tamanho da imagem capturada na base N-MNIST de 35×35 pixels. Desta forma seria necessário um mapeamento secundário ou cálculos adicionais para determinação da posição final dos eventos das superfícies de tempo o que poderia resultar em custo de tempo adicional. Apesar disso, esta alteração não produz grande impacto direto no tempo de extração, tendo efeito somente no tamanho final do descritor e consequentemente na qualidade da classificação.

5.6 RECURSOS DE *HARDWARE* e *SOFTWARE* UTILIZADOS NOS EXPERIMENTOS

Os experimentos de análise de complexidade foram feitos em um Notebook com processador Intel Core i5, $8GB$ de memória RAM utilizando sistema operacional Ubuntu 18.04 de 64 bits. Os experimentos foram feitos utilizando implementações do algoritmo Sparse-LBP e extração por varredura em estruturas matriciais em linguagem Python 3.7.

Os experimentos de classificação de objetos, descritos nas Seções 5.2, 5.3 e 5.4 foram realizados em um computador com processador AMD Ryzen 7 de oito núcleos, $32GB$ de memória RAM, utilizando sistema operacional Ubuntu Linux 20.04 64-bits. Os códigos de extração de dados foram implementados em linguagem Go 1.14. Os *scripts* para realização das etapas de calibração compressão e classificação foram feitos em Python 3.7, utilizando as bibliotecas scikit-learn para implementações do PCA e do SVM e a biblioteca Platypus (HADKA, 2020) para implementação do algoritmo genético.

Para comparação com o algoritmo HATS (SIRONI et al., 2018), foi feita uma implementação em Go 1.14 com base em código fonte disponível em Mosca (2020).

6 AVALIAÇÃO DOS RESULTADOS

Este capítulo contém a análise dos resultados obtidos nos experimentos realizados conforme descrição no Capítulo 5. A Seção 6.1.1 contém os resultados da análise temporal e custo computacional do algoritmo Sparse-LBP, enquanto a

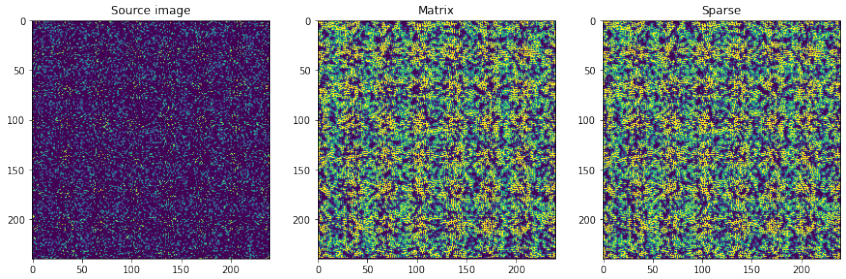
6.1 CUSTO COMPUTACIONAL ALGORITMO SPARSE-LBP

Esta seção contém os resultados de análise de tempo utilizando dados sintéticos. Conforme descrito na Seção 5.1, foram gerados conjuntos eventos sintéticos aleatórios, em formato de círculos e em formato de luas. Os resultados são analisados separadamente em função do tipo de dado gerado nas subseções a seguir.

6.1.1 Eventos aleatórios esparsos

O primeiro experimento foi realizado utilizando eventos aleatórios. A Figura 37 mostra um exemplo de mapa de eventos aleatórios, seguido dos mapas de padrões binários locais resultantes, usando as versões matricial e esparsa do algoritmo de extração.

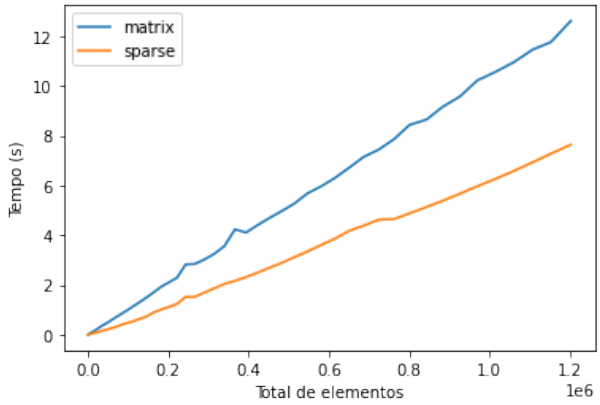
Figura 37 – Dados de eventos sintéticos aleatórios (esquerda), mapa LBP extraído com algoritmo tradicional (centro) e mapa LBP extraído com novo algoritmo Sparse-LBP (direita).



Fonte: Autor

A Figura 38 mostra a evolução do tempo em função de do número eventos presentes na estrutura esparsa para ambas as versões de algoritmo. É possível observar comportamento linear para ambas as versões. Além disso o algoritmo Sparse-lbp leva menos tempo para processar a mesma quantidade de pontos em uma estrutura esparsa.

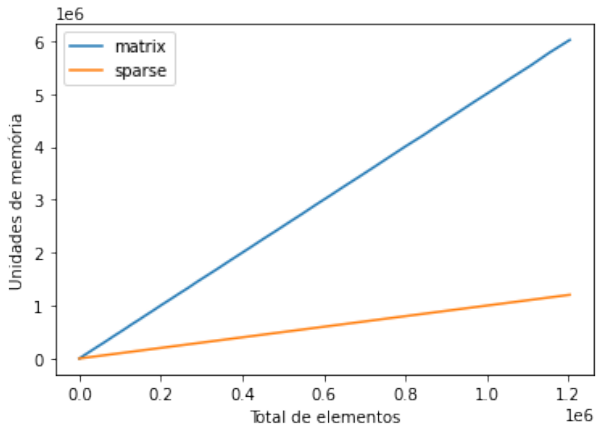
Figura 38 – Comparativo de tempo para extração de padrões LBP na versão esparsa em função do número de eventos e na versão matricial em função do tamanho da diagonal.



Fonte: Autor

Além disso, é possível observar também um considerável ganho em espaço utilizado. A Figura 39 mostra a evolução do espaço utilizado para cada versão em função de do número de eventos.

Figura 39 – Comparativo de espaço utilizado para extração de padrões LBP na versão esparsa e na versão matricial.



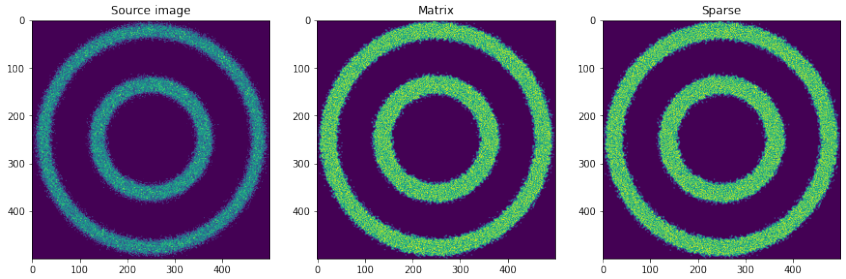
Fonte: Autor

Este resultado sugere um bom potencial para o algoritmo Sparse-LBP em situações onde existe restrição de uso em memória e os dados são suficientemente esparsos.

6.1.2 Círculos

Para o segundo experimento foi utilizado um gerador de eventos em formatos de círculos. A Figura 40 mostra um exemplo de mapa de eventos em formato de círculo, seguido do resultado LBP usando a versão matricial e do resultado da versão esparsa.

Figura 40 – Dados de eventos sintéticos aleatórios em formato de círculos (esquerda), mapa LBP extraído com algoritmo tradicional (centro) e mapa LBP extraído com novo algoritmo Sparse-LBP (direita)



Fonte: Autor

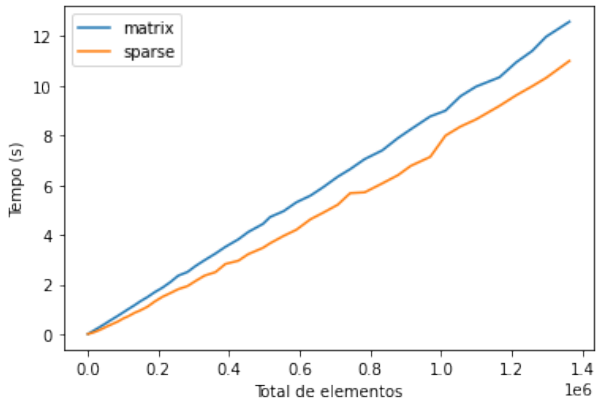
A Figura 38 mostra a evolução do tempo em função do número de eventos na SAE para ambas as versões de algoritmo. É possível observar comportamento linear para ambas as versões. Também se observa um maior no tempo para a versão esparsa se comparado aos dados aleatórios, apesar de ligeiramente inferior à versão matricial. Isso pode ser explicado devido aos dados em formato de círculo estarem menos esparsos, o que pode ser também observado na comparação do espaço utilizado, mostrado na Figura 39. Ainda assim, nota-se um uso em memória bastante inferior quando comparado à versão matricial.

6.1.3 Luas

Para o terceiro experimento foi utilizado um gerador de eventos em formatos de luas. A Figura 43 mostra um exemplo de mapa de eventos em formato de luas, seguido do resultado LBP usando a versão matricial e do resultado da versão esparsa.

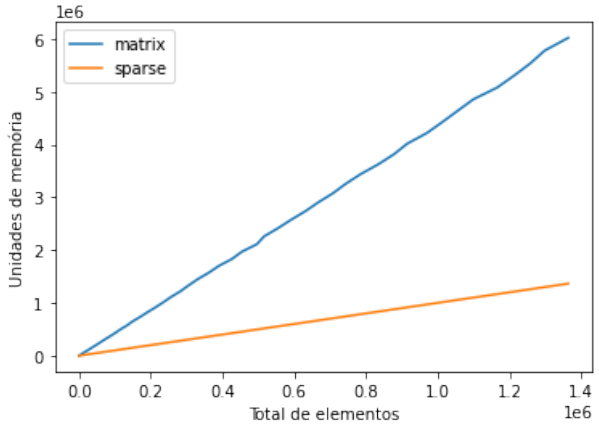
A Figura 44 mostra a evolução do tempo em função de do número de eventos na SAE, para ambas as versões de algoritmo. É possível observar comportamento linear para ambas as versões. Além disso, é possível observar que os tempos de execução para a versão esparsa se encontram inferiores à versão matricial.

Figura 41 – Comparativo de tempo para extração de padrões LBP na versão esparsa em função do número de eventos e na versão matricial em função do tamanho da diagonal para dados em formato de círculo.



Fonte: Autor

Figura 42 – Comparativo de espaço utilizado para extração de padrões LBP na versão esparsa e na versão matricial para dados em formato de círculo.

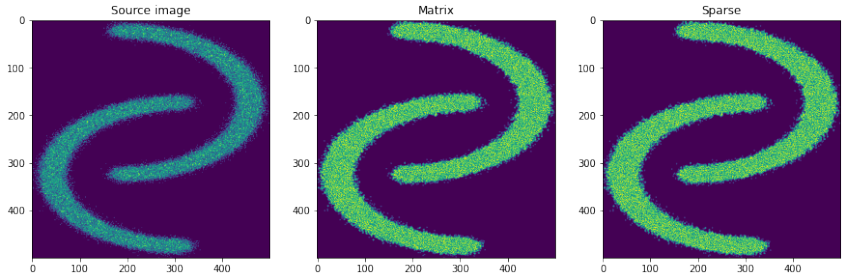


Fonte: Autor

A Figura 45 mostra a evolução do espaço utilizado para cada versão em função do número de eventos. É possível observar um custo em espaço muito inferior à versão matricial. Além disso, o espaço dos dados em forma de lua é menor do que o utilizado pelos dados em forma de círculo sugerindo que neste conjunto os eventos estão mais esparsos.

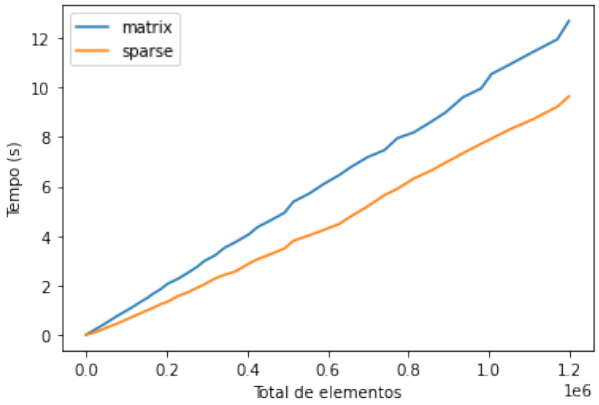
Os resultados dos experimentos conduzidos sugerem que o algoritmo Sparse-LBP possui custo computacional linear em função do número de eventos sustentando a hipótese descrita na Seção 5.1. No entanto, observa-se que o custo unitário das operações em tabelas *hash* pode tornar

Figura 43 – Dados de eventos sintéticos aleatórios em formato de círculos (esquerda), mapa LBP extraído com algoritmo tradicional (centro) e mapa LBP extraído com novo algoritmo Sparse-LBP (direita)



Fonte: Autor

Figura 44 – Comparativo de tempo para extração de padrões LBP na versão esparsa em função do número de eventos e na versão matricial em função do tamanho da diagonal para dados em formato de luas.



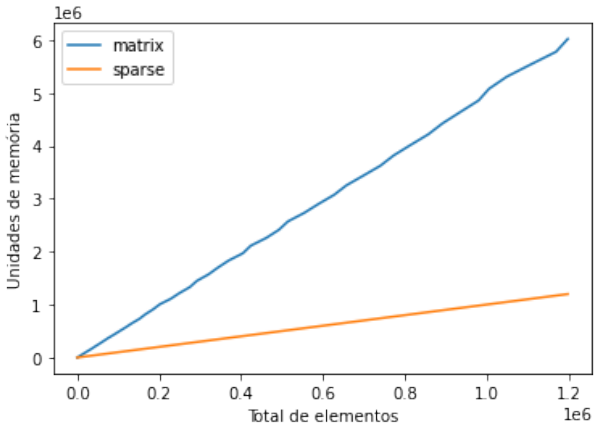
Fonte: Autor

este algoritmo relativamente custoso em cenários onde os dados na SAE não são suficientemente esparsos.

6.1.4 Impacto da taxa de ocupação

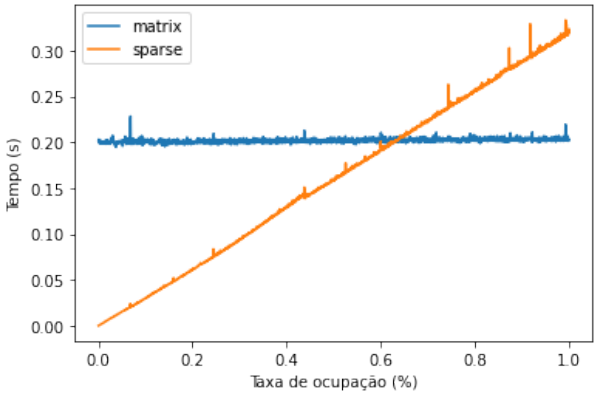
A Figura 46 mostra a evolução do tempo necessário para execução das versões matricial e esparsa do algoritmo de extração de padrões LBP em função da taxa de ocupação. É possível observar o comportamento linear para o Sparse-LBP em função da taxa de ocupação que é diretamente proporcional ao número de chaves na tabela *hash*.

Figura 45 – Comparativo de espaço utilizado para extração de padrões LBP na versão esparsa e na versão matricial para dados em formato de luas.



Fonte: Autor

Figura 46 – Comparativo de espaço utilizado para extração de padrões LBP na versão esparsa e na versão matricial em função da taxa de ocupação.



Fonte: Autor

Uma vez que o tamanho da superfície é fixo e todas as coordenadas devem ser avaliadas, o tempo necessário para extração se mantém próximo de uma constante para a versão matricial. Entretanto, com o aumento no número de chaves na superfície, é possível observar o aumento no tempo necessário para execução do Sparse-LBP, chegando a tempos superiores ao tempo necessário para a versão matricial a partir de 60% da taxa de ocupação. Estes resultados sugerem que em cenários onde os dados são pouco esparsos, o custo computacional do Sparse-LBP não é vantajoso.

6.2 CLASSIFICAÇÃO DE OBJETOS

Esta seção contém o comparativo geral das metodologias apresentadas com metodologias consideradas estado da arte. Nesta tese, os resultados são comparados formalmente aos resultados obtidos pelos métodos PCA-RECT e HATS. Os métodos AMAE e EST apesar de apresentarem acurácia superior, são listados aqui apenas como referência, pois ambos usam redes profundas e requerem recursos de *hardware* específico para treinamento de tais redes. A Tabela 13 mostra um comparativo entre os métodos citados.

Tabela 13 – Resultados da classificação para as bases de dados de *benchmark*

Modelo	N-Caltech	N-MNIST	N-Cars
MSLBP (proposto)	65,99%	97,89%	-
MSLBP-AS (proposto)	64,89%	97,94%	-
AHLBP (proposto)	57,73%	96,49%	85,81%
HATS (SIRONI et al., 2018)	64.20%	99,1%	90,2%
PCA-RECT (RAMESH et al., 2018)	72,30%	98,95%	-
EST (GEHRIG et al., 2019)	83,7%	98,95%	92,5%
AMAE (DENG; LI; CHEN, 2020)	85,1%	99,93%	95,5%

Fonte: Autor

Os resultados sustentam a plausibilidade do uso do operador LBP para classificação de objetos e dígitos com precisão comparável aos principais métodos no estado da arte, especialmente para a base N-MNIST. Para a base N-Caltech temos um resultado superior ao HATS e próximo ao PCA-RECT com diferença de acurácia em torno de 6%, ficando atrás dos métodos AMAE e EST que utilizam redes profundas. O método AHLBP, se mostrou também bastante apto na base N-MNIST. Nota-se também que o MSLBP mostrou resultados superiores ao AHLBP para as bases N-MNIST e N-Caltech, o que sugere que a disponibilidade da informação sobre o movimento realizado pelos sensor é relevante. Além disso, os métodos MSLBP e AHLBP conseguem atingir esse nível de acurácia com um descritor de tamanho muito inferior ao PCA-RECT e ao HATS.

Para a base NMNIST o vetor final reportado pelo método HATS possui tamanho 2450, enquanto o MSLBP após a aplicação do algoritmo de compressão, possui um vetor final de tamanho 298. Isso corresponde a uma redução de $87{,}83\%$ de uso em memória para o classificador quando comparado ao método HATS e de $90{,}01\%$ quando comparado ao PCA-RECT que possui tamanho fixo de 3000. O método AHLBP por sua vez mostrou desempenho muito similar ao MSLBP, porém com um vetor ainda menor de tamanho 172, representando uma redução de

$92{,}97\%$ de uso em memória quando comparado ao método HATS e $94{,}27\%$ se comparado ao PCA-RECT.

Considerando a base N-Caltech, o vetor final para o método MSLBP ficou com tamanhos 169 e 189 para os formatos de SAE com evento mais recente e soma dos tempos, respectivamente. Em comparação, o PCA-RECT que possui vetor de tamanho fixo de 3000. Para o método MSLBP, isso representa uma redução entre $99{,}57\%$ e $99{,}51\%$ quando comparados ao método HATS e entre $96{,}37\%$ e $93{,}7\%$ quando comparado ao PCA-RECT. O método AHLBP ficou com tamanho final 161, porém com resultados bastante inferiores ao PCA-RECT.

Para os experimentos com a base N-Cars, o vetor final do AHLBP ficou de tamanho 104, o que representa uma redução de $99{,}73\%$ no tamanho final do vetor e com diferença de precisão inferior a 5%. O método PCA-RECT não foi reportado com a base N-Cars que foi criada posteriormente à sua publicação.

A Tabela 14 mostra um comparativo dos tamanhos finais dos descritores utilizados na etapa de classificação em comparação com os métodos PCA-RECT e HATS utilizando os parâmetros reportados.

Tabela 14 – Comparativo de tamanho dos descritores para as bases de *benchmark*

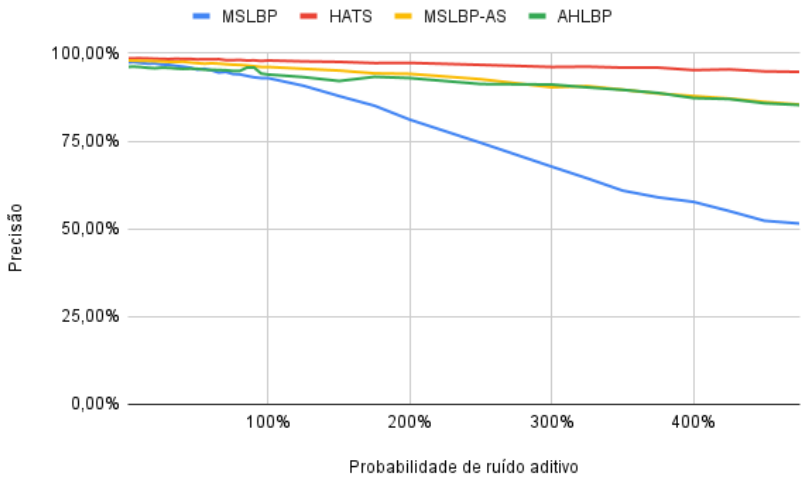
Modelo	N-Caltech	N-MNIST	N-Cars
MSLBP (proposto)	**169**	**298**	-
MSLBP-AS (proposto)	**189**	**298**	-
AHLBP (proposto)	**161**	**172**	**104**
HATS (SIRONI et al., 2018)	39200	2450	39200
PCA-RECT (RAMESH et al., 2018)	3000	3000	-

Fonte: Autor

6.3 IMPACTO DO RUÍDO ADITIVO

O operador LBP tradicionalmente é conhecido por não ser robusto a ruído (SILVA; BOUWMANS; FRÉLICOT, 2015). Apesar disso, os resultados se mostraram próximos aos métodos considerados estado da arte para classificação de dígitos manuscritos. Para o método MSLBP, foram comparadas duas formas de representação da SAE, uma utilizando o registro de tempo do evento mais recente, e uma utilizando a soma dos registros de tempo em cada coordenada (x,y). A Figura 47 mostra a precisão de cada método em função da probabilidade de geração de ruído aditivo na base Noisy-N-MNIST.

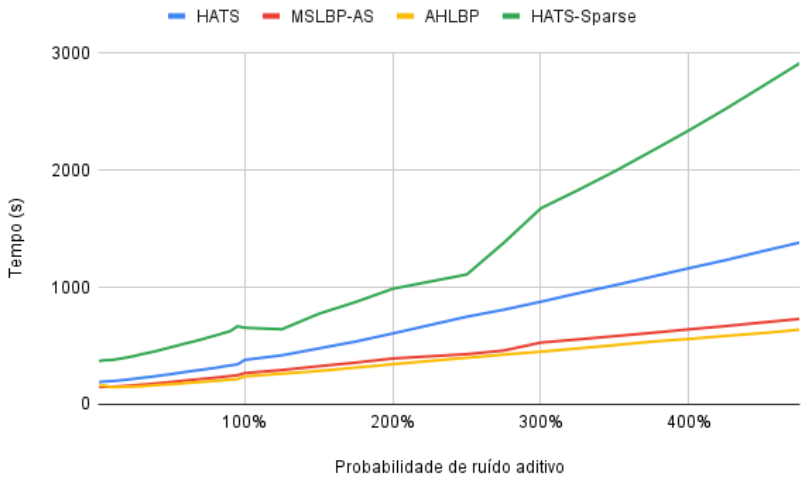
Figura 47 – Resultados de resistência a ruído aditivo para os métodos MSLBP, AHLBP e HATS.



Fonte: Autor

O algoritmo HATS se mostrou pouco superior em relação ao MSLBP com relação à resistência a ruído. No entanto, o custo computacional do método HATS é maior. A Figura 48 mostra os tempos em segundos necessários para processar a base de testes Noisy-MNIST para cada percentual de probabilidade de ruído.

Figura 48 – Tempo de processamento em segundos da base de testes Noisy-MNIST utilizando os métodos MSLBP, AHLBP, HATS e HATS-SPARSE.



Fonte: Autor

Para evitar qualquer interferência nos tempos de execução, durante os experimentos somente um processo de extração foi executado por vez, sem escrita em disco e sem etapas de normalização dos dados. Conforme descrito na Seção 3.8.4, o aumento da probabilidade de ruído resulta em mais eventos a serem processados.

Os resultados mostram que o método AHLBP supera os algoritmos avaliados em termos de tempo de execução. Tanto as metodologias AHBLP quanto MSLBP mostram um comportamento linear em função do acréscimo de eventos na forma de ruído e inferiores ao método HATS, com boa margem.

O algoritmo HATS tem um bom desempenho na faixa inicial de ruído em sua forma matricial. No entanto, é possível notar uma tendência não linear em função do ruído adicional. Esse custo pode ser explicado devido à necessidade de se construir superfícies de tempo locais para cada evento processado. Além disso, tanto o MSBLP quanto o AHLBP utilizam tabelas *hash* que possuem um custo computacional unitário para inserções e leitura relativamente maior que estruturas matriciais. Quando o método HATS é executado utilizando superfícies de tempo em tabelas *hash*, identificado na Figura 48 como HATS-SPARSE, o custo computacional se mostra bastante elevado em função do ruído adicional.

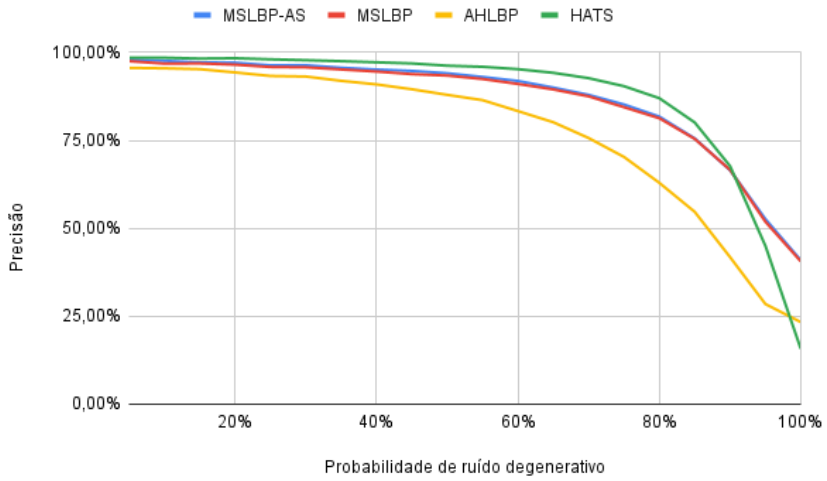
6.4 IMPACTO DO RUÍDO DEGENERATIVO

O algoritmo HATS se mostrou pouco superior em relação ao MSLBP até 50% de ruído. A partir deste valor, nota-se um decaimento maior na qualidade do MSLBP em relação ao HATS. No entanto, a partir de 80% o algoritmo HATS começa a demonstrar maior influência do ruído degenerativo na qualidade da classificação. Já o método AHLBP se mostrou menos eficaz na classificação de dígitos com ruído degenerativo, onde a deterioração do modelo é observada já a partir de 50% de probabilidade de ruído.

Quando comparadas as formas de construção da superfície de eventos ativos, observa-se pouca variação para o ruído degenerativo. A forma MSLBP-AS atingiu níveis de precisão em média de $0{,}5\%$ acima do MSLBP, o que não representa um ganho muito expressivo.

Neste experimento, os mesmos benefícios de uso em memória são observados quando em comparação com o método HATS. No entanto, como a quantidade de eventos é a mesma para todos os níveis de ruído, a comparação de tempo não se faz necessária para cada um deles. A versão HATS-SPARSE não foi avaliada neste experimento, pois a estrutura utilizada para

Figura 49 – Resultados de resistência a ruído degenerativo para os métodos AHLBP, MSLBP, MSLBP-AS e HATS.



Fonte: Autor

construção da superfície de tempo T_{e_i} não interfere no vetor resultante, tendo impacto somente no tempo.

7 CONCLUSÃO

Esta tese propôs a adaptação do operador LBP em cenas capturadas com sensores baseados em eventos e uma potencial aplicação para reconhecimento de objetos trazendo as seguintes contribuições:

a) Algoritmo Sparse-LBP para extração esparsa de padrões binários locais em tabelas *hash* com complexidade linear.
b) Metodologia para reconhecimento de objetos com conhecimento do movimento realizado pelo sensor com baixo uso em memória.
c) Metodologia para reconhecimento de objetos sem conhecimento do movimento realizado pelo sensor com baixo uso em memória.
d) Sem dependência de hardware específico (GPU ou hardware neuromórfico).

O algoritmo Sparse-LBP tira proveito da natureza esparsa dos dados com representação da superfície de eventos ativos em tabelas *hash* com complexidade linear. O operador LBP utilizado é uma versão simplificada do operador $\text{LBP}^{riu2}_{P,R}$ proposto por Ojala, Pietikäinen e Mäenpää (2002), utilizando somente vizinhanças retangulares e, portanto não realiza operações de interpolação que são custosas. As metodologias propostas para classificação, bem como a comprovação do custo computacional foram demonstrados experimentalmente.

Os resultados obtidos nos experimentos sustentam a plausibilidade da adoção de padrões LBP, ou outros extratores de características autorais em contraste com a corrente tendência de utilização de redes convolucionais com aprendizado profundo, onde tais características são aprendidas de maneira *offline*. Trata-se de um operador de simples implementação, baixo custo computacional, e em sua implementação esparsa, possui baixo consumo em memória, o que o torna atrativo para ambientes embarcados ou com baixos recursos computacionais. Com aplicação do método de compressão utilizado, observamos redução no tamanho dos vetores de características em até $99{,}73\%$ e ainda mantendo qualidade na classificação comparável a métodos convencionais considerados estado da arte. Tratam-se de resultados promissores para adoção desse tipo de operador em contextos envolvendo sensores de eventos. Além disso, a extração de características com o Sparse-LBP mostrou tempo de execução menor do que os demais métodos comparados, reduzindo em até $53{,}87\%$ o tempo de extração para o pior caso com ruído aditivo.

No entanto, deve-se levar em consideração algumas limitações desta metodologia. Em cenários onde os dados na superfície de eventos ativos não forem suficientemente esparsos, o

custo computacional de operações unitárias na tabela *hash* pode se tornar proibitivo. Metologias que realizam muitas construções de superfícies de eventos ativos devem também levar esse custo unitário em consideração, conforme observado nos resultados dos experimentos com ruído aditivo. Além disso, deve-se considerar o impacto do número de janelas de amostragem no tempo total do reconhecimento, caso seja empregado o método de compressão utilizando PCA.

Nota-se também que o operador LBP se saiu melhor em situações onde a separação é feita em função do movimento executado, tirando proveito dos padrões distintos provocados por cada um deles. Em aplicações onde o movimento é bem controlado, o operador LBP pode ser utilizado como uma alternativa de baixo custo computacional para tarefas de reconhecimento. Operadores mais sensíveis a padrões direcionais podem se basear nessa premissa e obter resultados melhores, mas não sem levar em consideração o custo no aumento do vetor de características. Neste caso, também é recomendado que o movimento da câmera seja capturado por um sensor de inércia (IMU) quando disponível.

Apesar da literatura recente indicando que o operador LBP não é um operador robusto a ruído (SILVA; BOUWMANS; FRÉLICOT, 2015), os experimentos tanto com ruído aditivo, quanto com ruído degenerativo mostraram um bom desempenho dos modelos. Observa-se no entanto, que o método de formação da superfície de eventos ativos gera um impacto considerável na classificação. Superfícies criadas com registro do evento mais recente são mais suscetíveis a ruídos, mesmo tendo as bordas mais homogêneas e bem definidas. Aplicações práticas podem fazer uso da metodologia MSLBP no reconhecimento de objetos estáticos utilizando sensores de eventos montados em *drones* em ambientes de alto risco para seres humanos como escombros ou áreas contaminadas.

Trabalhos futuros poderiam avaliar diferentes operadores LBP e verificar a possibilidade de adaptação de algoritmos de extração eficientes, além de avaliar a acurácia nas bases de *benchmark* e realizar comparações com as metodologias existentes. Além disso, ainda existe a possibilidade de adaptação de outros descritores binários tal como descrito na Seção 2.3. Em particular, os métodos LBP-TOP e VLBP (ZHAO; PIETIKAINEN, 2007) poderiam fazer uso das informações de tempo obtidas dos eventos. Outros metodologias poderiam também tirar proveito na natureza esparsa dos eventos na SAE para extração rápida de mapas de LBP. Além disso, não foram encontrados estudos de análise de textura utilizando sensores baseados em eventos, onde as variações do operador LBP são comumente estudadas. Novos trabalhos poderiam incluir a conversão de bases existentes de textura utilizando a mesma metodologia proposta por Orchard et al. (2015) sem afastar a necessidade da criação de bases de textura utilizando objetos reais

capturados por um sensor de eventos. Além disso, a adoção de padrões binários locais para classificação de objetos podem ser avaliados em domínios ainda pouco explorados.

REFERÊNCIAS

AFSHAR, Saeed et al. Investigation of event-based surfaces for high-speed detection, unsupervised feature extraction, and object recognition. **Frontiers in neuroscience**, Frontiers, v. 12, p. 1047, 2019.

AHONEN, Timo; HADID, Abdenour; PIETIKÄINEN, Matti. Face recognition with local binary patterns. In: SPRINGER. EUROPEAN conference on computer vision. [S.l.: s.n.], 2004. P. 469–481.

AKOPYAN, Filipp et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. **IEEE transactions on computer-aided design of integrated circuits and systems**, IEEE, v. 34, n. 10, p. 1537–1557, 2015.

BARRANCO, Francisco; FERMÜLLER, Cornelia; ALOIMONOS, Yiannis. Contour motion estimation for asynchronous event-driven cameras. **Proceedings of the IEEE**, IEEE, v. 102, n. 10, p. 1537–1556, 2014.

BARUA, Souptik; MIYATANI, Yoshitaka; VEERARAGHAVAN, Ashok. Direct face detection and video reconstruction from event cameras. In: IEEE. 2016 IEEE winter conference on applications of computer vision (WACV). [S.l.: s.n.], 2016. P. 1–9.

BELBACHIR, Ahmed Nabil et al. High-speed embedded-object analysis using a dual-line timed-address-event temporal-contrast vision sensor. **IEEE Transactions on Industrial Electronics**, IEEE, v. 58, n. 3, p. 770–783, 2010.

BENOSMAN, Ryad et al. Asynchronous frameless event-based optical flow. **Neural Networks**, Elsevier, v. 27, p. 32–37, 2012.

BENOSMAN, Ryad et al. Event-based visual flow. **IEEE transactions on neural networks and learning systems**, IEEE, v. 25, n. 2, p. 407–417, 2014.

BI, Yin et al. Graph-based object classification for neuromorphic vision sensing. In: PROCEEDINGS of the IEEE/CVF International Conference on Computer Vision. [S.l.: s.n.], 2019. P. 491–501.

BILODEAU, Guillaume-Alexandre; JODOIN, Jean-Philippe; SAUNIER, Nicolas. Change detection in feature space using local binary similarity patterns. In: IEEE. 2013 International Conference on Computer and Robot Vision. [S.l.: s.n.], 2013. P. 106–112.

BISHOP, Christopher M et al. **Pattern recognition and machine learning**. [S.l.]: springer New York, 2006. v. 4.

BISULCO, Anthony et al. Near-chip Dynamic Vision Filtering for Low-Bandwidth Pedestrian Detection. In: IEEE. 2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). [S.l.: s.n.], 2020. P. 234–239.

BITTENCOURT, Rafael. **Running Alone**. [S.l.]: Paradoxx Music, 2001. CD Album Rebith, faixa 9.

BOYD, Stephen; VANDENBERGHE, Lieven. **Convex optimization**. [S.l.]: Cambridge university press, 2004.

BRANDLI, Christian et al. A 240× 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. **IEEE Journal of Solid-State Circuits**, IEEE, v. 49, n. 10, p. 2333–2341, 2014.

CALONDER, Michael et al. Brief: Binary robust independent elementary features. **Computer Vision–ECCV 2010**, Springer, p. 778–792, 2010.

CAMUNAS-MESA, Luis et al. An event-driven multi-kernel convolution processor module for event-driven vision sensors. **IEEE Journal of Solid-State Circuits**, IEEE, v. 47, n. 2, p. 504–517, 2011.

CAMUÑAS-MESA, Luis A; LINARES-BARRANCO, Bernabé; SERRANO-GOTARREDONA, Teresa. Low-power hardware implementation of snn with decision block for recognition tasks. In: IEEE. 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS). [S.l.: s.n.], 2019. P. 73–76.

CANNICI, Marco et al. Asynchronous convolutional networks for object detection in neuromorphic cameras. In: PROCEEDINGS of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. [S.l.: s.n.], 2019. P. 0–0.

CANNICI, Marco et al. Attention mechanisms for object recognition with event-based cameras. In: IEEE. 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). [S.l.: s.n.], 2019. P. 1127–1136.

CENSI, Andrea; SCARAMUZZA, Davide. Low-latency event-based visual odometry. In: IEEE. ROBOTICS and Automation (ICRA), 2014 IEEE International Conference on. [S.l.: s.n.], 2014. P. 703–710.

CHEN, Guang et al. FLGR: Fixed length gists representation learning for RNN-HMM hybrid-based neuromorphic continuous gesture recognition. **Frontiers in neuroscience**, Frontiers, v. 13, p. 73, 2019.

CHEN, Guang et al. Multi-cue event information fusion for pedestrian detection with neuromorphic vision sensors. **Frontiers in neurorobotics**, Frontiers, v. 13, p. 10, 2019.

CLADY, Xavier et al. A motion-based feature for event-based pattern recognition. **Frontiers in neuroscience**, Frontiers, v. 10, p. 594, 2017.

COHEN, Gregory K et al. Skimming digits: neuromorphic classification of spike-encoded images. **Frontiers in neuroscience**, Frontiers, v. 10, p. 184, 2016.

CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

DAMIEN, Joubert; HUBERT, Konik; FREDERIC, Chausse. Convolutional neural network for detection and classification with event-based data. In: SCITEPRESS-SCIENCE e TECHNOLOGY PUBLICATIONS. 14TH International Conference on Computer Vision Theory and Applications. [S.l.: s.n.], 2019. P. 200–208.

DENG, Yongjian; LI, Youfu; CHEN, Hao. AMAE: Adaptive Motion-Agnostic Encoder for Event-Based Object Classification. **IEEE Robotics and Automation Letters**, IEEE, v. 5, n. 3, p. 4596–4603, 2020.

FEI-FEI, Li; FERGUS, Rob; PERONA, Pietro. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: IEEE. 2004 conference on computer vision and pattern recognition workshop. [S.l.: s.n.], 2004. P. 178–178.

FURBER, Steve B et al. The spinnaker project. **Proceedings of the IEEE**, IEEE, v. 102, n. 5, p. 652–665, 2014.

GALLEGO, Guillermo et al. Event-based, 6-DOF camera tracking from photometric depth maps. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2017.

GEHRIG, Daniel et al. End-to-end learning of representations for asynchronous event-based data. In: PROCEEDINGS of the IEEE/CVF International Conference on Computer Vision. [S.l.: s.n.], 2019. P. 5633–5643.

GHOSH, Rohan et al. Real-time object recognition and orientation estimation using an event-based camera and CNN. In: IEEE. 2014 IEEE Biomedical Circuits and Systems Conference (BioCAS) Proceedings. [S.l.: s.n.], 2014. P. 544–547.

GHOSH, Rohan et al. Spatiotemporal Feature Learning for Event-Based Vision. **arXiv preprint arXiv:1903.06923**, 2019.

GIANNONE, Giorgio et al. Real-time Classification from Short Event-Camera Streams using Input-filtering Neural ODEs. **arXiv preprint arXiv:2004.03156**, 2020.

GREGOR, Karol et al. Draw: A recurrent neural network for image generation. In: PMLR. INTERNATIONAL Conference on Machine Learning. [S.l.: s.n.], 2015. P. 1462–1471.

GUO, Zhenhua; ZHANG, Lei; ZHANG, David. A completed modeling of local binary pattern operator for texture classification. **IEEE transactions on image processing**, IEEE, v. 19, n. 6, p. 1657–1663, 2010.

HADKA, David. **HATS**. [S.l.: s.n.], 2020. Disponível em: https://platypus.readthedocs.io/en/latest/. Acesso em: em 20 abr. 2021.

HAFIANE, Adel; SEETHARAMAN, Guna; ZAVIDOVIQUE, Bertrand. Median binary pattern for textures classification. In: SPRINGER. INTERNATIONAL Conference Image Analysis and Recognition. [S.l.: s.n.], 2007. P. 387–398.

HE, Dong Chen; WANG, Li. Simplified texture spectrum for texture analysis. **Journal of Communication and Computer**, David Publishing Company, v. 7, n. 8, p. 44–53, 2010.

HE, Dong Chen; WANG, Li. Texture features based on texture spectrum. **Pattern Recognition**, Elsevier, v. 24, n. 5, p. 391–399, 1991.

HE, Kaiming et al. Deep residual learning for image recognition. In: PROCEEDINGS of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. P. 770–778.

HE, Yonggang; SANG, Nong; GAO, Changxin. Multi-structure local binary patterns for texture classification. **Pattern Analysis and Applications**, Springer, v. 16, n. 4, p. 595–607, 2013.

HEIKKILÄ, Marko; PIETIKÄINEN, Matti; SCHMID, Cordelia. Description of interest regions with center-symmetric local binary patterns. In: COMPUTER vision, graphics and image processing. [S.l.]: Springer, 2006. P. 58–69.

JIANG, Zhuangyi et al. Mixed frame-/event-driven fast pedestrian detection. In: IEEE. 2019 International Conference on Robotics and Automation (ICRA). [S.l.: s.n.], 2019. P. 8332–8338.

KIM, Hanme; LEUTENEGGER, Stefan; DAVISON, Andrew J. Real-time 3D reconstruction and 6-DoF tracking with an event camera. In: SPRINGER. EUROPEAN Conference on Computer Vision. [S.l.: s.n.], 2016. P. 349–364.

KUENG, Beat et al. Low-latency visual odometry using event-based feature tracks. In: IEEE. INTELLIGENT Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. [S.l.: s.n.], 2016. P. 16–23.

LAGORCE, Xavier et al. Asynchronous event-based multikernel algorithm for high-speed visual features tracking. **IEEE transactions on neural networks and learning systems**, IEEE, v. 26, n. 8, p. 1710–1720, 2015.

LAGORCE, Xavier et al. Hots: a hierarchy of event-based time-surfaces for pattern recognition. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 39, n. 7, p. 1346–1359, 2016.

LECUN, Yann. 1.1 deep learning hardware: Past, present, and future. In: IEEE. 2019 IEEE International Solid-State Circuits Conference-(ISSCC). [S.l.: s.n.], 2019. P. 12–19.

LECUN, Yann; CORTES, Corinna. MNIST handwritten digit database, 2010. Disponível em: http://yann.lecun.com/exdb/mnist/.

LEUTENEGGER, Stefan; CHLI, Margarita; SIEGWART, Roland Y. BRISK: Binary robust invariant scalable keypoints. In: IEEE. 2011 International conference on computer vision. [S.l.: s.n.], 2011. P. 2548–2555.

LI, Hongmin; LI, Guoqi; SHI, Luping. Classification of spatiotemporal events based on random forest. In: SPRINGER. INTERNATIONAL Conference on Brain Inspired Cognitive Systems. [S.l.: s.n.], 2016. P. 138–148.

LI, Hongmin; SHI, Luping. Robust event-based object tracking combining correlation filter and cnn representation. **Frontiers in neurorobotics**, Frontiers, v. 13, p. 82, 2019.

LI, Hongmin et al. Cifar10-dvs: an event-stream dataset for object classification. **Frontiers in neuroscience**, Frontiers, v. 11, p. 309, 2017.

LI, Hongmin et al. Deep representation via convolutional neural network for classification of spatiotemporal event streams. **Neurocomputing**, Elsevier, v. 299, p. 1–9, 2018.

LI, Jianing et al. Event-based vision enhanced: A joint detection framework in autonomous driving. In: IEEE. 2019 IEEE International Conference on Multimedia and Expo (ICME). [S.l.: s.n.], 2019. P. 1396–1401.

LIAO, Shu; CHUNG, Albert CS. Face recognition by using elongated local binary patterns with average maximum distance gradient magnitude. In: SPRINGER. ASIAN conference on computer vision. [S.l.: s.n.], 2007. P. 672–679.

LICHTSTEINER, Patrick; POSCH, Christoph; DELBRUCK, Tobi. A 128x128 120 dB 15us Latency Asynchronous Temporal Contrast Vision Sensor. **IEEE journal of solid-state circuits**, IEEE, v. 43, n. 2, p. 566–576, 2008.

LIU, Qianhui et al. Effective AER Object Classification Using Segmented Probability-Maximization Learning in Spiking Neural Networks. In: 02. PROCEEDINGS of the AAAI Conference on Artificial Intelligence. [S.l.: s.n.], 2020. v. 34, p. 1308–1315.

LORENA, Ana Carolina; CARVALHO, André CPLF de. Uma introdução às support vector machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43–67, 2007.

LOWE, David G. Distinctive image features from scale-invariant keypoints. **International journal of computer vision**, Springer, v. 60, n. 2, p. 91–110, 2004.

LU, Junwei et al. An Event-based Categorization Model Using Spatio-temporal Features in a Spiking Neural Network. In: IEEE. 2020 12th International Conference on Advanced Computational Intelligence (ICACI). [S.l.: s.n.], 2020. P. 385–390.

MÄENPÄÄ, Topi; PIETIKÄINEN, Matti. Multi-scale binary patterns for texture analysis. In: IMAGE Analysis. [S.l.]: Springer, 2003. P. 885–892.

MESSIKOMMER, Nico et al. Event-based asynchronous sparse convolutional networks. In: SPRINGER. EUROPEAN Conference on Computer Vision. [S.l.: s.n.], 2020. P. 415–431.

MILFORD, Michael et al. Towards visual SLAM with event-based cameras. In: THE Problem of Mobile Sensors: Setting Future Goals and Indicators of Progress for SLAM Workshop in Conjunction with Robotics: Science and Systems (RSS). [S.l.: s.n.], 2015.

MILFORD, Michael J; WYETH, Gordon F. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In: IEEE. ROBOTICS and Automation (ICRA), 2012 IEEE International Conference on. [S.l.: s.n.], 2012. P. 1643–1649.

MIRUS, Florian. Event-based optical-flow for autonomous driving using synthetic sensory data from an open-source simulator, 2016.

MOEYS, Diederik Paul et al. Steering a predator robot using a mixed frame/event-driven convolutional neural network. In: IEEE. 2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP). [S.l.: s.n.], 2016. P. 1–8.

MOSCA, Rafael. **HATS**. [S.l.: s.n.], 2020. Disponível em: https://github.com/rfma23/HATS. Acesso em: em 20 abr. 2021.

MUEGGLER, Elias; BARTOLOZZI, Chiara; SCARAMUZZA, Davide. Fast Event-based Corner Detection. In: EPFL-CONF-232510. BRITISH Machine Vision Conference (BMVC). [S.l.: s.n.], 2017.

MUEGGLER, Elias; HUBER, Basil; SCARAMUZZA, Davide. Event-based, 6-DOF pose tracking for high-speed maneuvers. In: IEEE. INTELLIGENT Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on. [S.l.: s.n.], 2014. P. 2761–2768.

MUEGGLER, Elias et al. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 36, n. 2, p. 142–149, 2017.

NAN, Ying et al. An event-based hierarchy model for object recognition. In: IEEE. 2019 IEEE Symposium Series on Computational Intelligence (SSCI). [S.l.: s.n.], 2019. P. 2342–2347.

NANNI, Loris; LUMINI, Alessandra; BRAHNAM, Sheryl. Local binary patterns variants as texture descriptors for medical image analysis. **Artificial intelligence in medicine**, Elsevier, v. 49, n. 2, p. 117–125, 2010.

NEGRI, Pablo et al. Scene Context Classification with Event-Driven Spiking Deep Neural Networks. In: IEEE. 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS). [S.l.: s.n.], 2018. P. 569–572.

OJALA, Timo; PIETIKAINEN, Matti; HARWOOD, David. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In: 1. PATTERN Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on. [S.l.: s.n.], 1994. P. 582–585.

OJALA, Timo; PIETIKÄINEN, Matti; MÄENPÄÄ, Topi. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, IEEE, v. 24, n. 7, p. 971–987, 2002.

OJEDA, Fernando Cladera et al. On-Device Event Filtering with Binary Neural Networks for Pedestrian Detection Using Neuromorphic Vision Sensors. In: IEEE. 2020 IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], 2020. P. 3084–3088.

ORCHARD, Garrick. **event-Python: Python code for event based vision**. [S.l.: s.n.], 2018. Disponível em: https://github.com/gorchard/event-Python. Acesso em: em 23 mar. 2021.

ORCHARD, Garrick et al. Converting static image datasets to spiking neuromorphic datasets using saccades. **Frontiers in neuroscience**, Frontiers, v. 9, p. 437, 2015.

PADALA, Vandana; BASU, Arindam; ORCHARD, Garrick. A noise filtering algorithm for event-based asynchronous change detection image sensors on truenorth and its implementation on truenorth. **Frontiers in neuroscience**, Frontiers, v. 12, p. 118, 2018.

PEASON, K. On lines and planes of closest fit to systems of point in space. **Philosophical Magazine**, v. 2, p. 559–572, 1901.

PEROT, Etienne et al. Learning to Detect Objects with a 1 Megapixel Event Camera. **arXiv e-prints**, arxiv–2009, 2020.

PHILLIPS, P Jonathon et al. The FERET evaluation methodology for face-recognition algorithms. **IEEE Transactions on pattern analysis and machine intelligence**, IEEE, v. 22, n. 10, p. 1090–1104, 2000.

PIETIKÄINEN, Matti; OJALA, Timo; XU, Zelin. Rotation-invariant texture classification using feature distributions. **Pattern Recognition**, Elsevier, v. 33, n. 1, p. 43–52, 2000.

PIETIKÄINEN, Matti; ZHAO, Guoying. Two decades of local binary patterns: A survey. In: ADVANCES in independent component analysis and learning machines. [S.l.]: Elsevier, 2015. P. 175–210.

QIAN, Xueming et al. PLBP: An effective local binary patterns texture descriptor with pyramid representation. **Pattern Recognition**, Elsevier, v. 44, n. 10-11, p. 2502–2515, 2011.

RAMESH, Bharath; YANG, Hong. Boosted kernelized correlation filters for event-based face detection. In: PROCEEDINGS of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops. [S.l.: s.n.], 2020. P. 155–159.

RAMESH, Bharath et al. DART: Distribution Aware Retinal Transform for Event-based Cameras. **arXiv preprint arXiv:1710.10800**, 2017.

RAMESH, Bharath et al. PCA-RECT: An energy-efficient object detection approach for event cameras. In: SPRINGER. ASIAN Conference on Computer Vision. [S.l.: s.n.], 2018. P. 434–449.

REBECQ, Henri; GALLEGO, Guillermo; SCARAMUZZA, Davide. EMVS: Event-based multi-view stereo. In: EPFL-CONF-221504. BRITISH Machine Vision Conference (BMVC). [S.l.: s.n.], 2016.

REBECQ, Henri; HORSTSCHAEFER, Timo; SCARAMUZZA, Davide. Real-time visualinertial odometry for event cameras using keyframe-based nonlinear optimization. In: BRITISH Machine Vis. Conf.(BMVC). [S.l.: s.n.], 2017. v. 3.

REBECQ, Henri et al. EMVS: Event-Based Multi-View Stereo—3D Reconstruction with an Event Camera in Real-Time. **International Journal of Computer Vision**, p. 1–21, 2017.

REBECQ, Henri et al. EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. **IEEE Robotics and Automation Letters**, IEEE, v. 2, n. 2, p. 593–600, 2017.

RIDWAN, Iffatur; CHENG, Howard. An Event-Based Optical Flow Algorithm for Dynamic Vision Sensors. In: SPRINGER. INTERNATIONAL Conference Image Analysis and Recognition. [S.l.: s.n.], 2017. P. 182–189.

RUBLEE, Ethan et al. ORB: An efficient alternative to SIFT or SURF. In: IEEE. COMPUTER Vision (ICCV), 2011 IEEE international conference on. [S.l.: s.n.], 2011. P. 2564–2571.

RUECKAUER, Bodo; DELBRUCK, Tobi. Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor. **Frontiers in neuroscience**, Frontiers, v. 10, p. 176, 2016.

SCHÖLKOPF, Bernhard et al. Support vector methods in learning and feature extraction. Citeseer, 1998.

SCIKIT. **Principal Component Regression vs Partial Least Squares Regression**. [S.l.: s.n.], 2021. Disponível em: https://scikit-learn.org/stable/auto_examples/cross_decomposition/plot_pcr_vs_pls.html. Acesso em: em 22 mar. 2021.

SCIKIT. **SVM-Kernels**. [S.l.: s.n.], 2021. Disponível em: https://scikit-learn.org/stable/auto_examples/svm/plot_svm_kernels.html. Acesso em: em 23 mar. 2021.

SCIKIT. **SVM: Maximum margin separating hyperplane**. [S.l.: s.n.], 2021. Disponível em: https://scikit-learn.org/stable/auto_examples/svm/plot_separating_hyperplane.html. Acesso em: em 23 mar. 2021.

SERRANO-GOTARREDONA, Rafael et al. CAVIAR: A 45k neuron, 5M synapse, 12G connects/s AER hardware sensory–processing–learning–actuating system for high-speed visual object recognition and tracking. **IEEE Transactions on Neural networks**, IEEE, v. 20, n. 9, p. 1417–1438, 2009.

SERRANO-GOTARREDONA, Teresa et al. ConvNets experiments on SpiNNaker. In: IEEE. 2015 IEEE International Symposium on Circuits and Systems (ISCAS). [S.l.: s.n.], 2015. P. 2405–2408.

SILVA, Caroline; BOUWMANS, Thierry; FRÉLICOT, Carl. An extended center-symmetric local binary pattern for background modeling and subtraction in videos. In: INTERNATIONAL Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISAPP 2015. [S.l.: s.n.], 2015.

SIRONI, Amos et al. Hats: Histograms of averaged time surfaces for robust event-based object classification. In: PROCEEDINGS of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2018. P. 1731–1740.

STEWART, Terrence C. A technical overview of the neural engineering framework. **University of Waterloo**, 2012.

SULLIVAN, Keith; LAWSON, Wallace. Representing motion information from event-based cameras. In: IEEE. 2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN). [S.l.: s.n.], 2017. P. 1465–1470.

TAN, Xiaoyang; TRIGGS, Bill. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In: SPRINGER. INTERNATIONAL workshop on analysis and modeling of faces and gestures. [S.l.: s.n.], 2007. P. 168–182.

THOMPSON, Neil; SPANUTH, Svenja. The decline of computers as a general purpose technology: why deep learning and the end of Moore's Law are fragmenting computing. **Available at SSRN 3287769**, 2018.

VAPNIK, Vladimir. Pattern recognition using generalized portrait method. **Automation and remote control**, v. 24, p. 774–780, 1963.

VAPNIK, Vladimir Naumovich; KOTZ, Samuel. **Estimation of dependences based on empirical data**. [S.l.]: Springer-verlag New York, 1982. v. 40.

VAPNIK, Vladimir Naumovich; VAPNIK, Vlamimir. **Statistical learning theory**. [S.l.]: Wiley New York, 1998. v. 1.

VASCO, Valentina; GLOVER, Arren; BARTOLOZZI, Chiara. Fast event-based Harris corner detection exploiting the advantages of event-driven cameras. In: IEEE. INTELLIGENT Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. [S.l.: s.n.], 2016. P. 4144–4149.

VERMA, Manisha; RAMAN, Balasubramanian. Center symmetric local binary co-occurrence pattern for texture, face and bio-medical image retrieval. **Journal of Visual Communication and Image Representation**, Elsevier, v. 32, p. 224–236, 2015.

VIDAL, Antoni Rosinol et al. Hybrid, Frame and Event based Visual Inertial Odometry for Robust, Autonomous Navigation of Quadrotors. **arXiv preprint arXiv:1709.06310**, 2017.

VISHNYAKOV, B et al. Fast moving objects detection using ilbp background model. **The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences**, Copernicus GmbH, v. 40, n. 3, p. 347, 2014.

XIAO, Rong et al. An event-driven categorization model for aer image sensors using multispike encoding and learning. **IEEE transactions on neural networks and learning systems**, IEEE, v. 31, n. 9, p. 3649–3657, 2019.

YAO, Cheng-Hao; CHEN, Shu-Yuan. Retrieval of translated, rotated and scaled color textures. **Pattern Recognition**, Elsevier, v. 36, n. 4, p. 913–929, 2003.

YOUSEFZADEH, Amirreza et al. Hardware implementation of convolutional STDP for on-line visual feature learning. In: IEEE. 2017 IEEE International Symposium on Circuits and Systems (ISCAS). [S.l.: s.n.], 2017. P. 1–4.

ZHANG, Wenchao et al. Local gabor binary pattern histogram sequence (lgbphs): A novel non-statistical model for face representation and recognition. In: IEEE. TENTH IEEE International Conference on Computer Vision (ICCV'05) Volume 1. [S.l.: s.n.], 2005. v. 1, p. 786–791.

ZHAO, Bo et al. Feedforward categorization on AER motion events using cortex-like features in a spiking neural network. **IEEE transactions on neural networks and learning systems**, IEEE, v. 26, n. 9, p. 1963–1978, 2014.

ZHAO, Guoying; PIETIKAINEN, Matti. Dynamic texture recognition using local binary patterns with an application to facial expressions. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 29, n. 6, p. 915–928, 2007.

ZHU, A; ATANASOV, Nikolay; DANIILIDIS, Kostas. Event-based visual inertial odometry. In: PROC. IEEE Int. Conf. Comput. Vis. Pattern Recog. [S.l.: s.n.], 2017. v. 3.