UFPI - CCN - DC
CIÊNCIA DA COMPUTAÇÃO
VISÃO COMPUTACIONAL- Prof. Dr. Kelson Romulo Teixeira Aires

Relatório Técnico

Detecção de Movimento com Intersecção de Histogramas

Lucas Jesus Santos Silva

Teresina
2024

1. Introdução

O desenvolvimento de algoritmos para detecção de movimento em vídeos é uma área fundamental em visão computacional e processamento de imagens. Este relatório descreve o processo de implementação de uma técnica de detecção de movimento baseada na intersecção de histogramas, que visa identificar regiões com alterações significativas entre frames consecutivos de um vídeo.

Sendo assim, como será discutido, o algoritmo apresenta algumas limitações, especialmente em cenários com mudanças rápidas de iluminação ou onde o contraste entre objetos em movimento e o fundo é baixo. Essas situações desafiam a precisão do método, o que indica a necessidade de integração com técnicas mais avançadas, como modelagem de fundo.

O relatório documenta as várias etapas de evolução do algoritmo, destacando os principais ajustes realizados para alcançar um equilíbrio entre a simplicidade da implementação e a eficácia na detecção de movimento, incluindo a aplicação de filtros de suavização.

2. Metodologia e Implementação Técnica

A detecção de movimento em vídeos envolve uma sequência de etapas que integram técnicas de processamento de imagem, reconhecimento de padrões e ajuste contínuo de parâmetros. A escolha das ferramentas adequadas, o planejamento da abordagem metodológica e a implementação de ajustes específicos para maximizar a precisão foram essenciais no desenvolvimento do sistema.

2.1 Metodologia e Implementação Técnica

Para a implementação deste sistema de detecção de movimento, foi necessário utilizar ferramentas de manipulação de vídeo e imagem, e bibliotecas de programação. O projeto fez uso das seguintes ferramentas:
Python: A linguagem escolhida por sua clara sintaxe e suas bibliotecas para manipulação de vídeo e imagem.
OpenCV: Biblioteca essencial para o processamento e análise de imagens. Ela facilita a captura de frames, o cálculo de histogramas e a aplicação de filtros.
Sistema de Arquivos (OS): A biblioteca OS foi usada para gerenciar diretórios e salvar os frames resultantes da detecção de movimento.

2.2 Implementação da Metodologia

A metodologia implementada foi baseada na comparação de histogramas para detecção de movimento, considerando as seguintes etapas:

Divisão dos Frames em Células: Cada frame de vídeo é dividido em uma grade dinâmica que muda de acordo com o tamanho da largura e altura da imagem. O tamanho da célula é determinado automaticamente, levando em consideração a altura e a largura da imagem e o tamanho da grade, a fim de garantir uma cobertura uniforme.

Comparação de Histogramas: Para cada célula, os histogramas de intensidade de dois frames consecutivos são calculados. A comparação é feita por meio da interseção dos histogramas, uma métrica menos sensível a pequenas variações de iluminação e ruídos. Isso garante que apenas mudanças significativas entre os frames sejam detectadas como movimento real.

Suavização de Imagens: Antes da geração dos histogramas, os frames são suavizados usando um filtro Gaussiano. Esse passo serve para eliminar variações pontuais, como pixels com ruído ou pequenas oscilações na iluminação, o que reduz a incidência de falsos positivos.

Marcação e Salvamento de Frames com Movimento: Quando o sistema detecta movimento, as áreas em movimento são destacadas com uma cor sólida (vermelho, neste caso), e o frame resultante é salvo em uma pasta separada, além de ser incorporado no vídeo final processado.

2.3 Ajustes Finais e Otimização

Ao longo do desenvolvimento, foram necessários diversos ajustes para garantir que o sistema conseguisse captar os movimentos. Os principais ajustes incluíram:

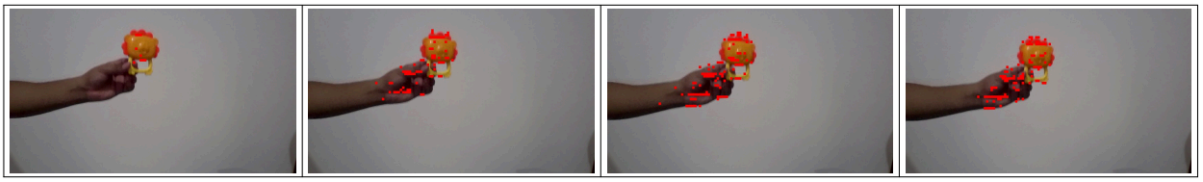
Filtro Gaussiano: A aplicação de filtros nos frames foi ajustada para equilibrar a suavização necessária para eliminar ruídos sem comprometer a precisão na detecção de objetos em movimento, o sistema estava captando muitos falsos positivos.

Threshold Dinâmico: O limite para a detecção de movimento foi sendo ajustado até um valor que detectasse bem os movimentos, para garantir a flexibilidade em vídeos com diferentes tipos de conteúdo.

3. Resultados

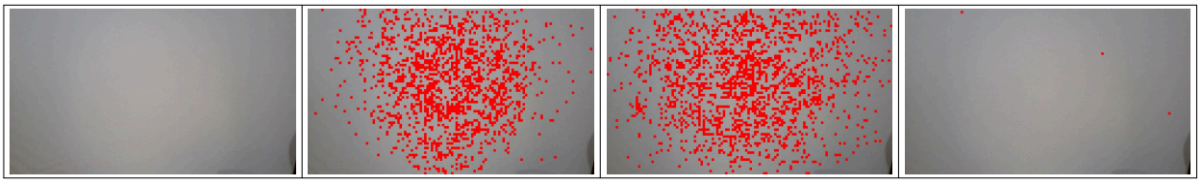
O código final mostrou uma capacidade razoável de detecção de movimento em vídeos, como pode ser visto na **Figura 1**, onde o movimento é destacado em vermelho nas áreas detectadas. No entanto, em cenários com mudanças bruscas de iluminação ou em superfícies de baixa

textura, ocorrem **falsos positivos** (áreas sem movimento sendo marcadas), como mostrado na **Figura 2**. Este tipo de erro é comum em vídeos onde há reflexos de luz ou oscilações rápidas na iluminação ambiente.



Frame 190 Frame 191 Frame 192 Frame 193

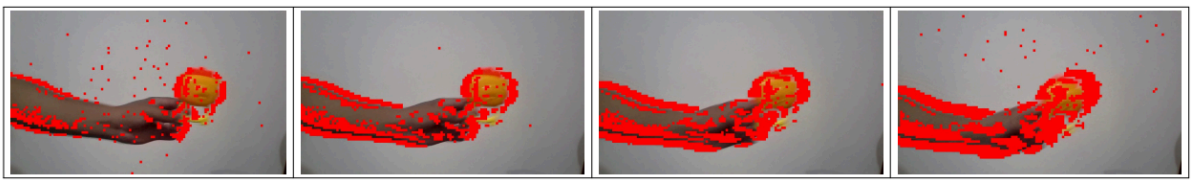
Figura 1: Positivo, a imagem está em movimento.



Frame 505 Frame 506 Frame 507 Frame 508

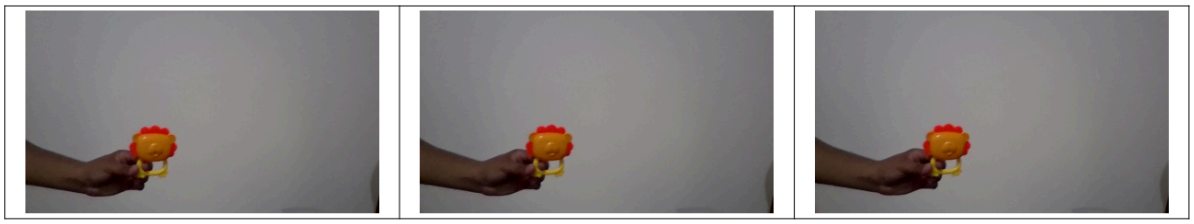
Figura 2: Falso positivo, a imagem não tem movimento, mas ocorre uma mudança na iluminação.

Nos casos de movimentações de maior contraste e controle de iluminação, como ilustrado na **Figura 3**, o algoritmo conseguiu identificar corretamente as áreas de movimento. No entanto, em situações de movimentos sutis, onde o contraste entre frames consecutivos é baixo, o algoritmo falha em detectar com precisão, como mostrado na **Figura 4**.



Frame 493 Frame 494 Frame 495 Frame 496

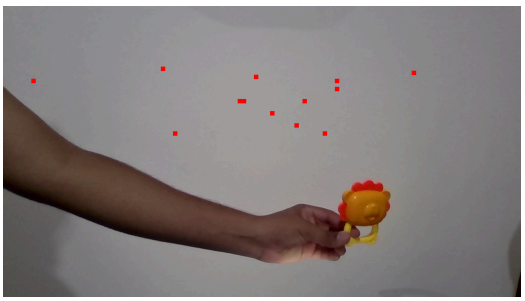
Figura 3: Positivo, a imagem está em movimento, com grande contraste e iluminação controlada.



Frame 70 Frame 71 Frame 72

Figura 4: Falso negativo, a imagem está em movimento, porém com pouco contraste.

As limitações foram particularmente evidentes quando o vídeo apresentava ruído ou pequenas variações de iluminação, onde o sistema gerou áreas de movimento onde, na verdade, não havia. Esses cenários são ilustrados na **Figura 5**, que mostra um falso positivo devido à iluminação variável.



Frame 288

Figura 5: Falso positivo, a imagem não está em movimento, porém com ruído.

6. Conclusão

A abordagem baseada na intersecção de histogramas demonstrou ser eficiente em cenários com contraste moderado e iluminação controlada (**Figura 1** e **Figura 3**). Entretanto, como evidenciado pelos exemplos nas **Figuras 2**, **4** e **5**, o algoritmo ainda apresenta limitações quando há mudanças rápidas de luz ou em superfícies homogêneas.

Essas limitações sugerem a necessidade de técnicas mais robustas, como a aplicação de filtros temporais ou o uso de aprendizado de máquina, para uma detecção mais precisa. A inclusão de métodos complementares, como modelagem de fundo e comparação estrutural, pode melhorar a robustez em uma gama mais ampla de situações.