



UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
CAMPUS LUIZ MENEGHEL

ROBSON ALVES NOGUEIRA

**ABORDAGEM PARA APOIAR O MONITORAMENTO DE
SEGURANÇA POR VÍDEO UTILIZANDO TÉCNICAS DE
PROCESSAMENTO DE IMAGEM**

Bandeirantes
2016

Robson Alves Nogueira

**ABORDAGEM PARA APOIAR O MONITORAMENTO DE
SEGURANÇA POR VÍDEO UTILIZANDO TÉCNICAS DE
PROCESSAMENTO DE IMAGEM**

Trabalho de Conclusão de Curso submetido à
Universidade Estadual do Norte do Paraná, como
requisito parcial para a obtenção do grau de
Bacharel em Sistemas de Informação.

Orientador: Prof. Carlos Henrique Machado

Co-orientador: Prof. Thiago Adriano Coleti

Bandeirantes

2016

Robson Alves Nogueira

**Abordagem para apoiar o monitoramento de segurança
por vídeo utilizando técnicas de processamento de
imagem**

Trabalho de Conclusão de Curso submetido à
Universidade Estadual do Norte do Paraná,
como requisito parcial para a obtenção do grau
de Bacharel em Sistemas de Informação.

COMISSÃO EXAMINADORA

Prof. Carlos Henrique Machado
UENP – *Campus* Luiz Meneghel

Prof. Thiago Adriano Coleti
UENP – *Campus* Luiz Meneghel

Prof. Ederson Marcos Sgarbi
UENP – *Campus* Luiz Meneghel

Bandeirantes, __ de _____ de 2016

Dedico este trabalho a minha família, pelo apoio e incentivo em cada momento de dificuldade que enfrentei nessa etapa da minha vida.

“O que sabemos é uma gota; o que ignoramos é um oceano.” (Isaac Newton)

AGRADECIMENTOS

Agradeço a todos aqueles que contribuíram em algum momento na realização deste trabalho, em especial aos professores Rodrigo Tomaz Pagno, Carlos Henrique Machado, Thiago Adriano Coleti e Ederson Marcos Sgarbi por todas suas valiosas orientações que tornaram possível a realização deste trabalho.

RESUMO

O presente trabalho visa propor uma metodologia para apoiar sistemas de monitoramento por vídeo na recuperação de conteúdo, sem grandes esforços, computacional ou humano através da criação de um sistema de indexação e pesquisa baseado em características extraídas através de um processamento inicial do vídeo que torna possível a pesquisa por regiões do ambiente monitorado e exclusão de áreas em que não haja interesse para pesquisa.

Palavras-chave: Processamento de imagens; Monitoramento por vídeo; Banco de dados de vídeo; Vídeo pesquisável; Detecção de movimentos.

ABSTRACT

This study aims to propose a methodology to support video monitoring systems in content retrieval, without great human or computer efforts, by creating an indexing system and search based on features extracted through an initial processing of the video which will make it possible to search for regions of the monitored environment and exclusion of areas where there is no interest for search.

Key-words: Image processing; Video monitoring; Video database system; Searchable video; Motion detection.

LISTA DE TABELAS

Tabela 1 – Exemplo de execução do algoritmo de detecção de movimentos....	41
Tabela 2 – Descrição dos campos da tabela de movimentos.....	45
Tabela 3 – Descrição dos campos da tabela de câmeras.....	45
Tabela 4 – Identificação das cores na classificação dos resultados da pesquisa.....	54

LISTA DE FIGURAS

Figura 1: Aplicando filtros para isolar objetos do fundo da imagem para facilitar contagem	20
Figura 2: Imagem com desfoque por movimento	21
Figura 3: Imagem restaurada	21
Figura 4: Resultado da execução de algoritmo de detecção facial	21
Figura 5: Área de interesse para algoritmo de reconhecimento facial	21
Figura 6: Ilustração de imagem aditiva	24
Figura 7: Representação da posição dos pixels	25
Figura 8: Representação de imagem em pixels	25
Figura 9: Ilustração da quantidade variável de frames em um segundo de vídeo	26
Figura 10: Janela principal do software	37
Figura 11: Funcionamento do algoritmo de detecção de movimentos	39
Figura 12: Coordenadas de um pixel e sua região correspondente	40
Figura 13: Modelo físico do banco de dados usado para salvar os movimentos e câmeras	46
Figura 14: Fluxograma do funcionamento da pesquisa	48
Figura 15: Fluxograma do funcionamento do algoritmo de captura, processamento inicial e salvamento no banco de dados e sistema de arquivos	50
Figura 16: Ilustração de movimento em um frame de vídeo indicando os setores onde existe maior quantidade de movimentos	51
Figura 17: Componente para selecionar área de interesse para pesquisa	52
Figura 18: Ilustração de apenas parte de um ambiente selecionada para pesquisa	53
Figura 19: Seleções criadas e possíveis de serem usadas como parâmetros de uma pesquisa	54
Figura 20: Exemplo de consulta básica usando áreas selecionadas no ambiente monitorado	54
Figura 21: Resultado de uma pesquisa classificado de acordo com sua relevância presumida	56
Figura 22: Sequência de ações realizadas em cada um dos testes	58
Figura 23: Exemplo de grupo de checkpoints retornados como resultado de uma pesquisa	59
Figura 24: Exemplo de histograma ilustrando a duração e o nível de movimentos do resultado de uma pesquisa	60
Figura 25: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno com iluminação natural	63
Figura 26: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a 3 metros de distância com iluminação artificial	64

Figura 27: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a vinte metros de distância com iluminação natural.....	66
Figura 28: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a vinte metros de distância com iluminação artificial.....	67
Figura 29: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a três metros de distância com iluminação natural.....	68
Figura 30: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a três metros de distância com iluminação artificial.....	69
Figura 31: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a vinte metros de distância com iluminação natural.....	71
Figura 32: Checkpoint e histograma do único teste a gerar resultados em ambiente externo a vinte metros de distância com iluminação artificial.....	71
Figura 33: Gráfico comparativo de desempenho relacionando a quantidade de testes realizados e a quantidade de interações que foram identificadas.....	72

LISTA DE SIGLAS

CLR	Common Language Runtime
C#	C Sharp
FPS	Frame por segundo
IL	Intermediate Language
IP	Internet Protocol
JPEG	Joint Photographics Experts Group
MPEG	Moving Picture Experts Group
RGB	Vermelho (Red), Verde (Green) e Azul (Blue)
SQL	Structured Query Language
WPF	Windows Presentations Foundation

SUMÁRIO

1. INTRODUÇÃO.....	14
1.1 Contextualização.....	15
1.2 Formulação e Escopo do Problema.....	16
1.3 Justificativa.....	16
1.4 Hipótese.....	17
1.5 Objetivos.....	17
1.5.1 Objetivo Geral.....	17
1.5.2 Objetivos Específicos.....	17
1.6 Organização do Trabalho.....	18
2. Fundamentação Teórica.....	19
2.1 Processamento de imagens digitais.....	19
2.1.1 Aspectos do Processamento de Imagens.....	19
2.1.1.1 Aprimoramento de imagens.....	20
2.1.1.2 Restauração de imagens.....	20
2.1.1.3 Segmentação de imagens.....	21
2.2 Detecção de movimento.....	22
2.3 Representação de imagens.....	23
2.3.1 Cores Aditivas.....	23
2.3.2 Modelo RGB.....	24
2.4 Video.....	24
2.4.1 Pixel.....	24
2.4.2 Frame.....	25
2.4.3 JPEG.....	26
2.4.4 MPEG-4.....	26
2.4.5 Armazenamento de vídeos.....	27
2.5 Implementação.....	27
2.5.1 Microsoft .Net Framework.....	28
2.5.2 Aforge.Net.....	28
2.5.3 C# (Sharp).....	28
2.5.4 Thread.....	29
2.5.5 WPF.....	30
2.5.6 SQL.....	30
2.6 Codificação em arquivos de vídeo.....	30
2.7 Sistemas de monitoramento por vídeo.....	31
2.8 Detecção de Pessoas em vídeo.....	32
2.8.1 Rastreamento.....	32
2.8.2 Estimativa de Silhueta.....	32

2.8.3 Reconhecimento.....	32
3. Metodologia.....	34
4. Desenvolvimento.....	35
4.1 Revisão Bibliográfica.....	35
4.2 Desenvolvimento do Protótipo.....	35
4.2.1 Etapas do desenvolvimento do protótipo.....	35
4.2.1.1 Captura de vídeo.....	35
4.3 Implementação.....	37
4.3.1 Detecção de Movimentos.....	37
4.3.1.1 Criação do algoritmo de Detecção de Movimentos.....	39
4.3.2 Banco de Dados.....	45
4.3.3 Arquivos de Imagem.....	48
4.3.4 Funcionamento do Algoritmo.....	48
4.3.5 Pesquisa.....	51
4.3.5.1 Criação de Componente de Representação de Setores.....	52
4.3.5.2 Classificação Automática de Relevância.....	55
4.4 Realização de Testes.....	56
4.4.1 Introdução aos Testes.....	57
4.4.2 Detalhes e Configurações.....	61
4.4.3 Testes em ambientes internos.....	61
4.4.4 Testes em ambientes externos.....	67
4.4.5 Desempenho do algoritmo.....	72
5. Trabalhos Futuros.....	74
6. Considerações Finais.....	75
7. Conclusão.....	76
Referências.....	77

1. INTRODUÇÃO

A necessidade da utilização de sistemas computacionais para administrar todo o conteúdo que um sistema de monitoramento por vídeo é capaz de gerar aumenta a medida que a demanda por sistemas de vigilância cresce e devido a inviabilidade da administração desse conteúdo sem o uso de um sistema computacional que automatize alguns processos, pois segundo Knight (2012), uma rede de câmeras de vigilância é tipicamente usada para monitorar locais de alto risco, mas isto pode gerar muitas imagens para o olho humano ser capaz de analisar.

A grande quantidade de variáveis e dados para processar tornam visível a incapacidade para um ser humano lidar com todos esses dados de forma bruta, sendo necessário um tratamento anterior a análise, um filtro para remover dados desnecessários ou com pouca utilidade.

Neste cenário, pode-se usar algoritmos para reduzir a quantidade de dados que merecem atenção e apresentar ao usuário do sistema apenas os fatos presumivelmente importantes e assim tornar o processo de análise dos dados de vídeo menos trabalhoso, pois segundo Tian (2008) dados gerados pelas câmeras de monitoramento demandam muitos recursos e lidar com a administração de todo este conteúdo tornando-o acessível tem sido um grande desafio.

Segundo Silberschatz, Korth e Sudarshan (2011), para obter acesso aleatório rápido a registros em um banco de dados, podemos usar uma estrutura de índice que é similar a um índice de um livro ou um catálogo de biblioteca. Assim um sistema de pesquisa baseado em índices seria capaz de fornecer a agilidade no acesso a partes específicas de um conteúdo e tornar possível a realização de pesquisas específicas para cada situação desejada.

O que define a importância em uma parte específica de um vídeo é o que foi capturado pelas câmeras: pessoas, carros, movimento, objetos, etc. A identificação dessas características exigem uma análise posterior nos dados capturados. Porém segundo Collins et al. (2000), manter o controle de pessoas, veículos e suas interações em um ambiente [...] é uma tarefa difícil. São muitas variáveis para

analisar, além de uma grande quantidade de imagens que necessitam ser processadas.

Segundo Webber (1992), algoritmos de processamento de imagens são tarefas numericamente intensivas tipificadas por grandes estruturas de dados que exigem muitas operações complexas a serem realizadas em alta velocidade.

De acordo com Silberschatz, Korth e Sudarshan (2011) arquivos de multimídia são formados por grandes estruturas de dados e por isso devem ser codificados de forma a reduzir a quantidade de utilização de recursos computacionais.

Devido ao grande número de bytes de dados necessários para representar dados de multimídia, é essencial que os dados de multimídia sejam armazenados e transmitidos de forma comprimida (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 1077).

Porém, como a dependência gerada pela codificação do vídeo aumenta o trabalho para recuperar partes do vídeo para análise posterior neste trabalho optou-se por gravar os dados de forma independente como arquivos de imagem, assim o processamento necessário para se recuperar uma determinada parte do vídeo é minimizado.

1.1 Contextualização

O aumento da utilização de sistemas de vigilância tem tornado evidente a necessidade de lidar com a grande quantidade de dados que um sistema de câmeras de vigilância pode gerar, pois é impraticável uma pessoa monitorar o tempo todo o que está sendo capturado por diversas câmeras, sendo necessário a ajuda de sistemas computacionais para reduzir o trabalho do operador do sistema.

Algoritmos podem direcionar a atenção do operador ao eventos importantes que estão ocorrendo ou já ocorreram através da análise dos dados capturados.

Ao focar no que presumivelmente é mais importante, o trabalho do operador é minimizado e a eficiência do sistema aprimorada a medida em que trabalhos desnecessários não são mais realizados.

1.2 Formulação e Escopo do Problema

A incapacidade humana de realizar uma análise adequada à grande quantidade de dados gerados por um sistema de monitoramento pode inviabilizar o próprio sistema, pois o objetivo do sistema de monitoramento, que é utilizar os dados capturados para realizar ações baseadas nas informações extraídas do vídeo, não é alcançado.

A confiança nos dados capturados e, conseqüentemente, no próprio sistema de monitoramento fica comprometida, uma vez que não se pode dizer exatamente se todo um montante de dados gerados pelo sistema em um determinado período de tempo tem importância presente ou futura para a organização.

1.3 Justificativa

A crescente demanda por sistemas de vigilância torna claro a necessidade das empresas em garantir a segurança de seu patrimônio através do monitoramento de áreas de risco.

Embora não haja dados disponíveis, estamos observando nos últimos anos uma forte disseminação das tecnologias de vigilância eletrônica no mundo inteiro. O Brasil [...] não foge à regra de um sistema de vigilância que confunde as esferas da segurança pública com a privada. Cada vez mais, na verdade, os gastos privados (de empresas e de particulares) vem se ampliando, fazendo face ao também crescente gasto público com segurança (ALLAIN, 2016).

O aumento da utilização de sistemas de vigilância deve ser acompanhado pelo aumento da capacidade dos sistemas que administram todo o conteúdo gerado por esses sistemas.

Como o ser humano não é capaz de analisar eficientemente todos os dados gerados por sistemas de monitoramento é necessário o desenvolvimento de sistemas para apoiar o operador na tarefa de análise dos dados.

Esses sistemas processam os dados gerados pelas câmeras a fim de extrair informações relevantes e tornar todo um montante de dados em algo útil e acessível ao operador do sistema computacional.

1.4 Hipótese

A importância de um banco de dados de vídeo coerente à realidade do local monitorado e acessível é imprescindível para a tomada de decisões em uma organização, assim pode-se dizer que a criação de um sistema de pesquisa baseado na indexação de características do vídeo pode tornar a recuperação de conteúdo mais rápida e com menos esforços devido a capacidade de acesso aleatório fornecida por uma estrutura de índices como citado por Silberschatz, Korth e Sudarshan (2011).

Pode-se constatar que a análise de cada parte do frame visando detectar movimentos ocorridos nesta área e posterior indexação em um banco de dados pode fornecer os benefícios para se pesquisar e permitir uma futura análise por um operador.

1.5 Objetivos

Neste tópico é abordado o objetivo geral e os objetivos específicos que pretende-se alcançar com a realização deste trabalho.

1.5.1 Objetivo Geral

Desenvolvimento de uma ferramenta de monitoramento que permita a busca de imagens relevantes de acordo com áreas de movimento, com isso pretende-se melhorar a capacidade de busca e análise em grande quantidade de imagens.

1.5.2 Objetivos Específicos

- Determinar o algoritmo para detecção de movimento no vídeo capaz de detectar a quantidade de movimento em partes de um frame.
- Desenvolver o sistema de gravação e pesquisa das imagens para testar os conceitos expostos.
- Determinar a eficiência da ferramenta com base em testes.

1.6 Organização do Trabalho

No Capítulo 2 é apresentada a fundamentação teórica necessária para o correto entendimento do trabalho, como: a definição de processamento de imagens digitais, detecção de movimentos, modelo mais comum usado para representação de imagens digitais, como funciona e como é formado um arquivo de vídeo, assim como conceitos relacionados ao desenvolvimento, como linguagens e frameworks usados.

No Capítulo 3 é abordado a metodologia usada na realização deste trabalho como: a abordagem a ser utilizada, a forma em que a viabilidade do sistema será atestada.

No Capítulo 4 será abordada a sequência de atividades realizadas para o desenvolvimento do software que implementa os conceitos expostos, buscando fornecer uma explicação detalhada de seu funcionamento.

No Capítulo 5 é abordado os trabalhos futuros que podem ser realizados a partir do presente trabalho.

O Capítulo 6 possui as considerações finais a respeito do trabalho.

No Capítulo 8 será abordado as conclusões que se formaram com base na criação do presente trabalho.

2. Fundamentação Teórica

Este tópico fornece a fundamentação teórica necessária para o entendimento de todos os fatores relacionados de forma direta ou indireta no desenvolvimento deste trabalho.

2.1 Processamento de imagens digitais

Uma imagem pode ser definida como uma função bi dimensional $f(x,y)$, em que tanto x quanto y representam coordenadas de uma posição específica na imagem e a amplitude de f de um par qualquer de coordenadas (x, y) é chamado de intensidade da imagem neste ponto, quando tanto os valores de (x, y) quanto a intensidade de $f(x,y)$ são todos finitos a imagem é considerada uma imagem digital. Processamento de imagens pode então ser definido como o processamento de imagens digitais com propósitos específicos de um computador digital (GONZALEZ; WOODS, 2007).

De acordo com Mcandrew (2004), O processamento de imagens tem uma enorme gama de aplicações, quase todas as áreas da ciência e da tecnologia pode fazer uso de métodos de processamento de imagem como: a Medicina, Agricultura, Indústria, Segurança.

2.1.1 Aspectos do Processamento de Imagens

As tarefas realizadas em imagens são em grande parte distintas, por isso é comum a divisão da area Processamento de Imagens em subclasses de acordo com as funções realizadas em cada algoritmo.

É conveniente subdividir diferentes algoritmos de processamento de imagem em subclasses. Existem diferentes algoritmos para diferentes tarefas e problemas, e muitas vezes gostaríamos de distinguir a natureza da tarefa em questão (MCANDREW, 2004).

Logo, a area de processamento de imagens é subdividida nas seguintes subclasses:

- Aprimoramento de imagens;

- Restauração de imagens;
- Segmentação de imagens.

2.1.1.1 Aprimoramento de imagens

A subclasse aprimoramento de imagens refere-se ao processamento de uma imagem de acordo com Mcandrew (2004), de modo que o resultado é mais apropriada para uma aplicação particular.

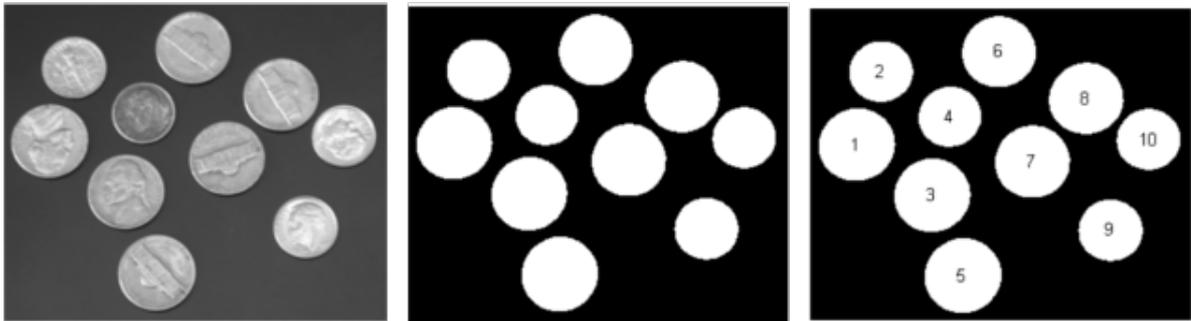


Figura 1: Aplicando filtros para isolar objetos do fundo da imagem para facilitar contagem

Adaptado de: Labeling labeled objects – www.blogs.mathworks.com

Aplicações que requerem muito processamento para realizar sua função executarão melhor em imagens com resolução baixa, a redução da resolução de uma imagem com a intenção de torná-la mais compatível com a execução de um determinado algoritmo é um exemplo de aprimoramento de imagem.

2.1.1.2 Restauração de imagens

As ações realizadas nessa subclasse podem ser segundo Mcandrew (2004), consideradas como ações de inversão dos danos causados a uma imagem por uma causa conhecida, por exemplo:

- remoção de desfoque,
- remoção de distorções ópticas,
- remoção de interferência periódica.



Figura 2: Imagem com desfoque por movimento



Figura 3: Imagem restaurada

Fonte: Instituto de Tecnologia de Massachusetts

Fonte: Instituto de Tecnologia de Massachusetts

A inversão dos danos pode ser total ou parcial dependendo das condições da imagem ou do tipo de dano ocorrido.

2.1.1.3 Segmentação de imagens

Conforme McAndrew (2004), a segmentação de imagens envolve subdividir uma imagem em partes constituintes, ou isolar certos aspectos de uma imagem com a intenção de, por exemplo:

- encontrar linhas, círculos ou formas particulares em uma imagem,
- identificar carros, árvores, edifícios ou estradas em uma imagem.

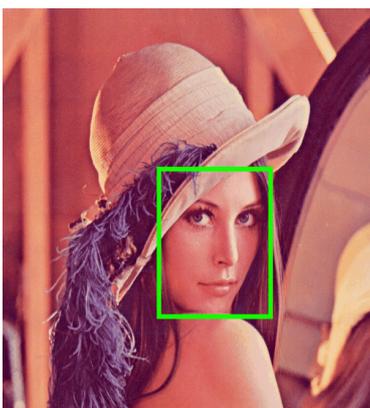


Figura 4: Resultado da execução de algoritmo de detecção facial



Figura 5: Área de interesse para algoritmo de reconhecimento facial

Adaptado de: The Lenna Story – www.lenna.org

Adaptado de: The Lenna Story – www.lenna.org

2.2 Detecção de movimento

A detecção de movimentos em um ambiente pode ser realizada de diversas formas, seja: usando sensores infravermelho, pois segundo Furnish (2007), eles são capazes de detectar a radiação emitida por outros objectos, tais como a energia térmica a partir de seres humanos; sensores que utilizam frequência de rádio que irradiam energia eletromagnética e detectam o eco devolvido de objectos reflectores (FURNISH, 2007); sensores sonoros, como microfones; sensores que camptam vibrações são capazes de medir a distância dos sinais produzidos pelos passos de uma pessoa ou por um objeto (TEIXEIRA; DUBLON, 2016); sensores magnéticos que relatam mudanças magnéticas significativas devido ao trânsito nas proximidades de materiais ferromagnéticos (ROWE; REED; FLORES, 2016) e sensores óticos que usam algoritmos para analisar imagens digitais, que será o foco deste trabalho e será explicado mais detalhadamente.

Detecção de movimentos usando equipamentos óticos consiste na capacidade de identificar variações nas imagens digitais provocadas pelo movimento de objetos em um ambiente monitorado.

Algoritmos de detecção de movimentos devem particionar os pixels de cada quadro da sequência de frames de um vídeo em dois tipos: o fundo, que correspondem a pixels pertencentes ao local estático, e o primeiro plano, correspondendo a pixels pertencentes a um objeto em movimento (LACASSAGNE; MANZANERA, 2009).

A comparação de um frame com seu antecessor fornece a quantidade e local do movimento, podendo posteriormente ser utilizado para fins específicos como criação de sistema de alerta, identificação de comportamento de pessoas, análise de situações de risco, etc.

Um algoritmo de detecção de movimento deve conforme citado por Lacassagne e Manzanera (2009), discriminar os objetos em movimento a partir do fundo de forma mais exata possível, sem ser demasiado sensível aos tamanhos e velocidades dos objetos, ou para as condições de mudança de uma cena estática.

Em um ambiente monitorado em busca de movimentos podem acontecer movimentos que não possuem relevância para o administrador do sistema. O algoritmo deve ser versátil a ponto de permitir que o usuário altere suas

configurações para que mudanças naturais de iluminação no ambiente não sejam consideradas como movimentos, assim como objetos pequenos demais para ter importância como: insetos ou pássaros.

Um algoritmo de detecção de movimentos deve então fornecer configurações ao usuário para fins de alteração de seu comportamento para que seja encontrado o equilíbrio entre a quantidade de dados a processar e a qualidade de detecção dos movimentos.

Para fins de longa autonomia, o sistema de monitoramento não deve consumir muitos recursos computacionais [...] já que módulos de processamento de imagens envolvem uma grande quantidade de dados (LACASSAGNE; MANZANERA, 2009).

A detecção de movimento é de acordo com Lacassagne e Manzanera (2009), certamente, a função que mais exige processamento em um sistema de vigilância por vídeo.

Então evitar que uma imagem que já foi processada em busca de movimentos seja processada novamente através da inserção em banco de dados dos movimentos identificados na imagem, pode otimizar um sistema de monitoramento por vídeo que usa detecção de movimentos.

2.3 Representação de imagens

A forma em que as imagens digitais são constituídas é abordada neste tópico de forma simplificada e com foco no conteúdo relacionado a este trabalho.

2.3.1 Cores Aditivas

A teoria das cores aditivas Thomas Young segundo Quoos (2016), é baseada no princípio de que diversos efeitos cromáticos são obtidos pela projeção da luz branca através dos filtros vermelho, verde e azul e pela superposição de círculos nas cores projetadas.

A teoria das cores aditivas tem como base o princípio de que todas as cores da luz podem ser misturadas opticamente através da combinação em proporções diferentes das três cores primárias do espectro: vermelho, verde e azul (RGB em

inglês). Apenas duas cores primárias já é suficiente para gerar diversas cores diferentes quando minsturadas [...] (HIRSCH, 2015)

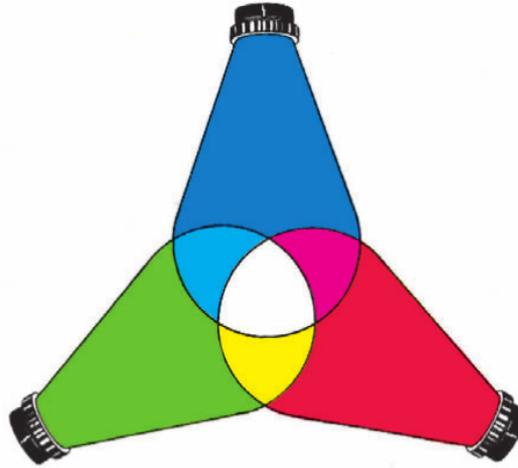


Figura 6: Ilustração de imagem aditiva

Fonte: Exploring Color Photography: From Film to Pixels 6th Edition p.19.

A imagem acima expressa graficamente como a sobreposição de cores é capaz de gerar novas cores.

2.3.2 Modelo RGB

O modelo de representação de cores RGB usa a combinação das cores vermelha, verde e azul para representar diversas cores e suas tonalidades.

Cada cor possui um valor numérico indicando sua intensidade do mais escuro ao mais claro, a escala de valores pode variar dependendo da combinação de software ou hardware usado sendo o mais comum as representações usando 0-256 e 0-65536 (BOURKE, 1995).

2.4 Video

Neste tópico os conceitos relacionados a arquivos de vídeo são apresentados com o objetivo de introduzir os conceitos necessários para o entendimento deste trabalho.

2.4.1 Pixel

Segundo Gonzalez e Woods (2007, p. 24) "pixel é o termo mais usado para denotar os elementos de uma imagem digital" é a unidade básica usada para representar as cores que constituem uma imagem.



Figura 7: Representação da posição dos pixels



Figura 8:
Representação de
imagem em pixels

Adaptado de: The Lenna Story - www.lenna.org

Adaptado de: The Lenna Story - www.lenna.org

Cada pixel é formado por valores numéricos a combinação de valores podem gerar diferentes cores que são usadas para representar cada parte específica de uma imagem.

2.4.2 Frame

Um frame é uma única imagem, que é mostrado como parte de um vídeo maior. Muitas frames individuais são executados em sucessão para produzir um vídeo (RUSSELL, 2016).

Um video é composto por frames e cada frame é uma imagem, quando os frames são exibidos em sequência ocorre o efeito visual do movimento, assim quanto mais frames forem exibidos por segundo melhor é a exatidão com o movimento real capturado pela câmera.

FPS ou frames por segundo é o número de quadros ou imagens que são projetadas ou exibidos por segundo. Taxas de quadros são utilizados na sincronização de áudio e imagens, seja em cinema, televisão ou arquivos de vídeo (ROUSE, 2016).

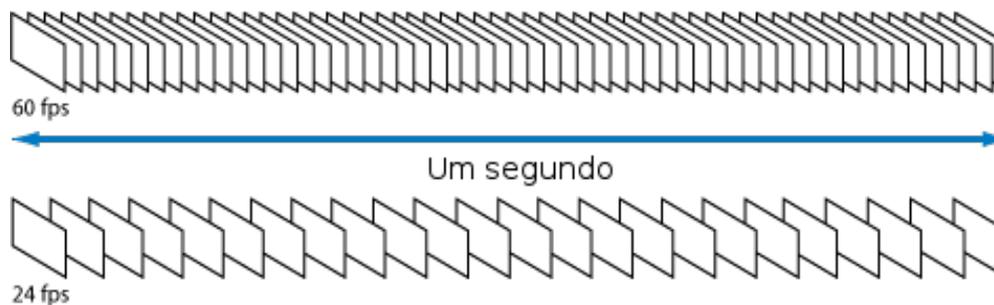


Figura 9: Ilustração da quantidade variável de frames em um segundo de vídeo

Adaptado de: Final Cut Pro 7, Manual do usuário, apêndice D

2.4.3 JPEG

Para dados de imagem, o formato mais utilizado é o JPEG, nomeado com base no nome do grupo que o criou, o Joint Picture Experts Group (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 1077).

Comumente usa-se o método de compressão de dados JPEG para compactar imagens permitindo especificar a taxa de compressão usada e assim reduzir o tamanho do arquivo gerado com perda de qualidade de forma controlada.

2.4.4 MPEG-4

O Moving Picture Experts Group desenvolveu a série de padrões MPEG para a codificação de dados de áudio e vídeo, estas codificações exploram semelhanças entre uma sequência de quadros para conseguir um maior grau de compressão (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 1077).

MPEG-4 é um formato de fluxo de bits digitais e associado a protocolos para representar conteúdo multimídia. MPEG-4 possibilita codificação e compressão para um grande grupo de funcionalidades incluindo manipulação dos dados de áudio e vídeo, interatividade, animações e texturas (Signès; Fisher; Eleftheriadis, 2000).

MPEG-4 fornece técnicas para uma maior compressão de vídeo, com largura de banda variável para apoiar a prestação de dados de vídeo através de redes com

uma ampla gama de larguras de banda (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 1078).

2.4.5 Armazenamento de vídeos

Os dados gerados pelas câmeras em um sistema de monitoramento por vídeo precisam ser armazenados em algum local para posterior análise.

O local onde esses dados são gravados dependem das características do próprio sistema de monitoramento, podendo ser no sistema de arquivo do servidor ou no próprio banco de dados, porém:

O banco de dados deve oferecer suporte a objetos grandes, pois os dados de multimídia, como vídeos podem ocupar até alguns gigabytes de armazenamento. Muitos sistemas de banco de dados não suportam objetos maiores do que alguns gigabytes. Objetos maiores poderiam ser divididos em partes menores e armazenados no banco de dados. Alternativamente, os objetos multimídia podem ser armazenados num sistema de arquivos e a base de dados pode conter um apontador para o arquivo; o ponteiro seria tipicamente um nome do arquivo (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 1077).

Como citado acima segundo Silberschatz, Korth e Sudarshan (2011), não é necessário guardar grandes objetos de multimídia em uma base de dados, sendo possível o simples armazenamento no sistema de arquivos e a utilização de um ponteiro apontado para o arquivo para fins de otimização de acesso. Neste trabalho a abordagem para lidar com os arquivos de vídeo será exatamente esta.

2.5 Implementação

Este tópico contém explicações resumidas sobre assuntos relacionados ao desenvolvimento do sistema proposto como: linguagem de programação usada, frameworks e bibliotecas.

2.5.1 Microsoft .Net Framework

O .Net Framework é um ambiente independente de linguagem criado pela Microsoft para desenvolvimento e gerenciamento de código. Assim qualquer código escrito em linguagens compatíveis com o .Net como Visual Basic, C#, J# podem ser compiladas para uma linguagem intermediária IL (Intermediate Language) que posteriormente será convertida pelo CLR (Common Language Runtime) para o código compatível com a plataforma alvo.

2.5.2 Aforge.Net

Aforge.Net é um framework open source criado em C# para tornar mais simples projetos nas áreas de computação visual, inteligência artificial, processamento de imagens, redes neurais, aprendizagem de máquina, etc.

Com as bibliotecas disponibilizadas pela Aforge.Net é possível de forma simples e rápida capturar o stream de vídeo de uma webcam ou câmera IP para processamento.

Nas bibliotecas da Aforge.Net existem diversos algoritmos para processamento de imagem, como detecção de movimentos, faces, objetos, etc. Porém neste trabalho foi usado apenas os recursos de administração de câmeras, detecção e captura de vídeo, os algoritmos de detecção de movimentos foram criados especificamente para este projeto e são explicados detalhadamente no Capítulo 4.

2.5.3 C# (Sharp)

Linguagem de programação orientada a objetos criada pela Microsoft com base nas linguagens C++ e Java.

O C# foi especialmente desenvolvido para ser usado no framework .net.

Usando C #, você pode, por exemplo, escrever uma página web dinâmica, um aplicativo do Windows Presentation Foundation, um serviço web XML, um componente de um aplicativo distribuído, um componente de acesso de banco de dados, um aplicativo de área de

trabalho clássico do Windows, ou mesmo uma nova aplicação cliente inteligente que permite a utilização de recursos online e offline (NAGEL; GLYNN; SKINNER, 2014, p. 25).

Sua sintaxe é muito similar a do Java, simples e de fácil aprendizagem. A linguagem C# foi influenciada por diversas linguagens, porém C++ e Java são as principais. C# é uma linguagem moderna que possui os melhores recursos do C++ e do Java com melhorias em diversas funções e recursos.

2.5.4 Thread

No presente trabalho threads foram usadas com o objetivo de tornar a extração de características do vídeo e sua gravação em ações que ocorrem de forma praticamente simultânea.

O conceito de thread, que consiste de acordo com Tanenbaum e Bos (2014), na execução intercalada de código, é muito utilizado atualmente devido à capacidade de otimização de recursos e tarefas e pode ser entendido da seguinte forma.

Em sistemas operacionais tradicionais cada processo possui um endereço e uma thread, [...] porém é muito comum ser necessário em diversas aplicações a utilização de múltiplas threads de controle em um mesmo endereço executando de forma quase paralela como se fossem processos diferentes exceto pelo fato de compartilharem o mesmo endereço (TANENBAUM; BOS, 2014, p. 97).

Na prática, cada aplicativo precisa executar diversas tarefas, mas como o processador só é capaz de executar uma tarefa por vez cada vez que algo está sendo executado a próxima tarefa é colocada em uma fila para ser executada em seguida.

Se uma determinada tarefa que estiver sendo executada for algo demorado como a conversão de um grande arquivo de vídeo não é possível ver o progresso da conversão até que a tarefa tenha sido concluída, pois para mostrar na interface do programa alguma informação ao usuário também é necessário executar uma tarefa.

Usando threads podemos fazer as duas coisas não ao mesmo tempo, mas de forma paralela, otimizando recursos e garantindo uma experiência mais agradável aos usuários do sistema.

2.5.5 WPF

WPF ou Windows Presentation Foundation é um moderno sistema gráfico para o Windows que trouxe diversas inovações em aceleração por hardware e independência de resolução (MACDONALD, 2012).

Antes do WPF desenvolvedores Windows usaram por cerca de 15 anos as mesmas tecnologias para desenvolver seus aplicativos, User32 que provê a tradicional aparência dos aplicativos windows e GDI/GDI+ com mais recursos gráficos (MACDONALD, 2012).

O WPF fornece uma extrema versatilidade para desenvolver aplicativos com interface rica e intuitiva.

A facilidade na criação de novos controles e componentes torna o WPF útil em aplicações que precisam abstrair situações complexas de forma simples.

2.5.6 SQL

Desenvolvida em meados dos anos 70 pela IBM o SQL é uma linguagem extremamente versátil usada para pesquisar, manipular ou controlar banco de dados relacionais.

A facilidade para realizar ações como criação, exclusão, atualização e pesquisas em tabelas tornou a linguagem SQL em algo extremamente popular.

A linguagem SQL está diretamente relacionada com bancos de dados relacionais, pois o resultado de uma pesquisa em SQL é uma tabela ou result set. Assim uma nova tabela pode ser criada em um banco de dados relacional simplesmente pelo armazenamento em memória do resultado de uma pesquisa. Da mesma forma uma nova consulta pode ser criada usando os resultados da consulta anterior para fornecer os resultados esperados (BEAULIEU, 2009).

2.6 Codificação em arquivos de vídeo

Como grande quantidade dos frames em um vídeo são idênticos os arquivos de vídeo são geralmente codificados para reduzir essa redundância (SILBERSCHATZ; KORTH; SUDARSHAN, 2011).

Os algoritmos de codificação reduzem o tamanho final do arquivo ao evitar esses dados idênticos, porém essa codificação torna os frames do vídeo muito dependentes um do outro.

Arquivos de imagem também passam por um processo de codificação com a intenção de reduzir o tamanho final do arquivo, também reduzindo os dados repetidos, porém em arquivos de vídeo a verificação de repetição ocorre também entre os frames o que resulta em uma maior redução no tamanho do arquivo final, logo, na prática as mesmas informações codificadas em formato de vídeo tendem a ocupar menos recursos por conseguirem uma maior compactação.

2.7 Sistemas de monitoramento por vídeo

Basicamente sistemas de monitoramento por vídeo são usados para monitorar uma determinada área utilizando câmeras, cabeamento, sensores, software de gestão de câmeras, sistema de armazenamento de dados e Molina, Garcia e Perez (2005) estabelecem a seguinte estrutura para sistemas de monitoramento por vídeo.

Um sistema de vigilância de vídeo é composto por vários processos: (1) Um processo de previsão de o fundo da imagem, em geral modelos gaussianos são aplicados para estimar a variação no fundo; (2) um processo detector de objetos em movimento, esse processo funciona sobre os quadros adquiridos anteriormente e atuais; (3) um processo agrupamento de pixel; (4) um processo de associação; e (5) um sistema que mantém o rastreamento de cada objeto detectado (MOLINA; GARCIA; PEREZ, 2005).

A função de um sistema de monitoramento por vídeo é a captura de eventos importantes ocorridos em uma área monitorada por câmeras para posterior revisão, verificação ou auxílio em determinadas situações em que a utilização de imagens de determinada situação se faz necessária como: identificação do autor de um crime, identificação de uma placa de carro, conhecer o comportamento de clientes

ou simplesmente manter um registro do que aconteceu no local monitorado durante um determinado tempo.

2.8 Detecção de Pessoas em vídeo

A detecção de pessoas em vídeo possuem de acordo com Moeslund e Granum (1999) três aplicações principais: em sistemas de vigilância, controle e análise e no presente tópico será abordado fatores importantes relativos a identificação, reconhecimento e rastreamento.

2.8.1 Rastreamento

Rastreamento é definido como a relação de coerência estabelecida entre algo em diferentes frames independentes. Na realização do rastreamento de um elemento três aspectos podem ser identificados: a segmentação da figura humana do resto da imagem, processamento da imagem segmentada para atender um algoritmo específico e como o assunto deve ser monitorado de quadro a quadro (MOESLUND; GRANUM, 1999)

2.8.2 Estimativa de Silhueta

A estimativa da silhueta humana pode ser definido como o processo de identificar um corpo humano em uma determinada cena e pode ser realizada após a execução do algoritmo de rastreamento ou pode funcionar conjuntamente com o algoritmo de rastreamento (MOESLUND; GRANUM, 1999).

2.8.3 Reconhecimento

O processo de reconhecimento de uma pessoa em uma cena pode ser realizado após a captura e a execução dos algoritmos de rastreamento e reconhecimento.

Basicamente seu funcionamento consiste em classificar os movimentos capturados em vídeo nos diversos tipos de ações possíveis, essas ações pode ser simples como andar, correr ou mais complexas como o comportamento de uma pessoa (MOESLUND; GRANUM, 1999).

2.8.3.1 Reconhecimento Estático

Reconhecimento realizado através da comparação de um frame por vez, geralmente o reconhecimento estático realizada a comparação de um frame com informações já salvas em uma banco de dados (MOESLUND; GRANUM, 1999).

2.8.3.2 Reconhecimento Dinâmico

Este método usa características identificadas de forma temporal através da análise de uma sequência de frames no reconhecimento e é comumente usada no reconhecimento de atividades mais simples como andar (MOESLUND; GRANUM, 1999).

3. Metodologia

Para a realização do presente trabalho foram realizadas pesquisas em fontes da internet, livros e artigos para fundamentar o desenvolvimento do software que aplicará os conceitos expostos neste trabalho.

Foi aplicada uma abordagem quantitativa através de testes para avaliar a viabilidade do sistema de monitoramento com base na recuperação de dados, pois:

Tipicamente banco de dados requerem um grande quantidade de espaço de armazenamento. [...] Como a memória principal do computador não é capaz de armazenar essa quantidade de informação, a informação é armazenada em discos. Como a movimentação dos dados para o disco e vice versa é relativamente lenta comparada à velocidade da unidade central de processamento, é imperativo que bancos de dados estruturam os dados para minimizar a movimentação dos dados entre o disco e a memória principal. (SILBERSCHATZ; KORTH; SUDARSHAN, 2011, p. 20)

Foi avaliada a capacidade do algoritmo de retornar conteúdo com base em pesquisa em diversas situações, como: distância da câmera, quantidade de iluminação e ambientes internos e externos.

Uma série de ações ocorreram na área monitorada pela câmera de vídeo e o objetivo do algoritmo foi identificar cada interação como uma ação independente.

Foram realizados ao todo 40 testes que visaram identificar o desempenho do algoritmo de detecção de movimentos em diversas situações.

Ao final foi gerado um gráfico comparativo do desempenho geral do algoritmo referente a quantidade de ações identificadas.

4. Desenvolvimento

4.1 Revisão Bibliográfica

A crescente necessidade de sistemas de vigilância sofisticados e a mudança para a infraestrutura digital de vigilância transformou o monitoramento por vídeo em um grande desafio de analisar e administrar dados (HAMPAPUR, 2004).

Segundo Hampapur (2004), enquanto a maioria das pesquisas foca na extração de aspectos presentes em imagens, muito pouco é explorado sobre o contexto da extração de informação em uma pesquisa, recuperação, administração e investigação de dados presentes em imagens.

4.2 Desenvolvimento do Protótipo

O desenvolvimento do protótipo assim como citado por Paula Filho (2000), tem o objetivo de explorar aspectos críticos dos requisitos do produto final, implementando algumas das funcionalidades principais que estarão contidas nele e ajudar a decidir sobre questões que sejam vitais para o sucesso final.

O protótipo desenvolvido visou testar as funcionalidades básicas do sistema assim como verificar o comportamento das bibliotecas usadas no desenvolvimento em um caso próximo do final e assim determinar de forma mais clara possível a viabilidade para adição das funcionalidades finais que o software deverá conter.

4.2.1 Etapas do desenvolvimento do protótipo

Neste tópico será explicado de forma resumida as etapas realizadas durante o processo de desenvolvimento do protótipo.

O objetivo não é detalhar todos os processos e sim expor o funcionamento e a sequência lógica de desenvolvimento de alguns dos principais elementos que estarão presentes no sistema final.

4.2.1.1 Captura de vídeo

A primeira função ao ser implementada de forma lógica no protótipo foi a captura de vídeo, pois assim é possível trabalhar com dados próximos aos de um caso real, lidando com problemas relacionados a resolução do vídeo, armazenagem de arquivos, recursos computacionais necessários para captura e processamento, etc.

Foi usado para facilitar o processo de captura de vídeo a biblioteca Aforge.NET que dentre suas diversas características relacionadas com processamento de imagens existe suporte à captura de vídeo.

4.2.1.2 Detecção de movimento

Um algoritmo específico para detecção de movimentos em partes de um frame foi criado baseado na identificação das diferenças entre um frame e o anterior.

O algoritmo é capaz de analisar os setores de um frame e indicar a quantidade de modificações que ocorreram em uma parte específica de um determinado frame.

O seguinte pseudocódigo representa o funcionamento do algoritmo de detecção de movimentos.

```
Setores DetectarMovimento(frameAtual, frameAnterior, nivelMovimento, areasInteresse){  
  
    setoresComMovimento ← []  
    para cada X até largura_frameAtual faça  
        para cada Y até altura_frameAtual faça  
            se areaDeInteresse(X,Y) então  
                se qtdMov(frameAtual[X][Y],frameAnterior[X][Y]) > nivelMovimento então  
                    setor ← identificarSetor(frameAtual,X,Y)  
                    setoresComMovimento[setor] += 1;  
                fim_se;  
            fim_se;  
        fim_para;  
    fim_para;  
    retornar setoresComMovimento;  
}
```

4.2.1.3 Indexação de movimento

Após a criação de um mapeamento completo dos movimentos presentes em um frame realizado por um processamento inicial do vídeo, as informações referentes aos movimentos identificados são salvas em um banco de dados para permitir a pesquisa.

4.2.1.4 Pesquisa por movimento

Através da pesquisa por movimento identificado no vídeo é possível informar tanto os setores de interesse quanto o nível de movimento em cada setor.

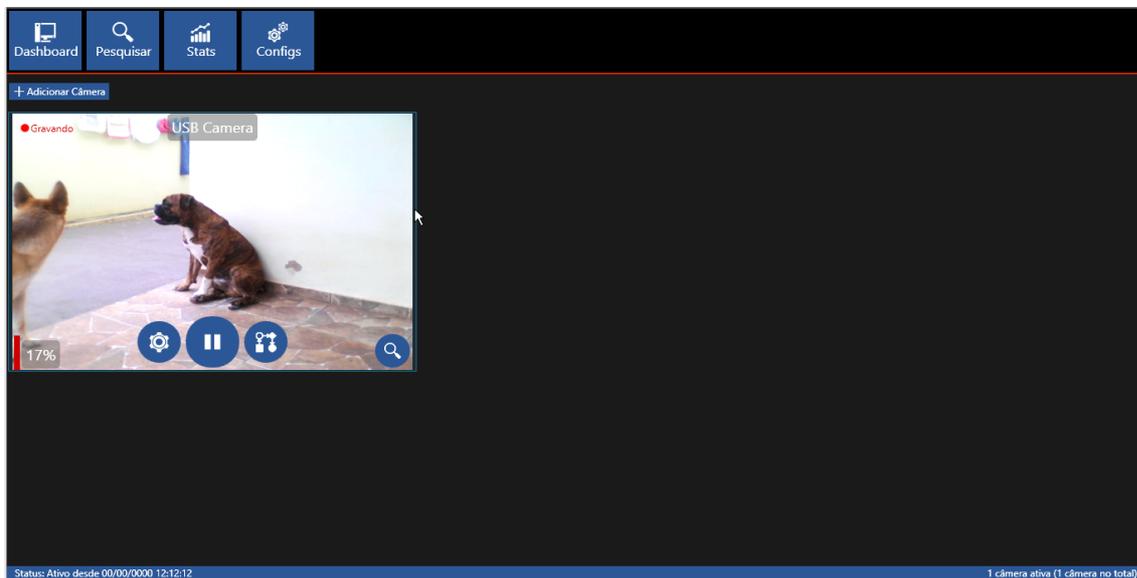


Figura 10: Janela principal do software

Fonte: próprio autor

Através da janela do protótipo ilustrada pela Figura 10 é possível verificar qual a aparência do software final. Também é possível notar a quantidade de movimentos identificada na imagem representada pelo indicador de cor vermelha no canto inferior esquerdo da janela informando que 17% (dezesete por cento) da imagem possui movimento.

4.3 Implementação do Software Final

O objetivo deste tópico é explicar como o software final foi criado, suas etapas e funcionamento de cada um de seus elementos que compõem o sistema de gravação, processamento e pesquisa em vídeo proposto.

4.3.1 Detecção de Movimentos

Diversas formas de detecção de movimentos existem como citado na sessão 2.2 do capítulo de Fundamentação Teórica, neste trabalho será usado a detecção ótica de movimentos.

Uma imagem é formada por frames e seus frames por pixels, os pixels são formados por uma série valores numéricos que representam as cores que em conjunto formam uma imagem (BOURKE, 1995).

Com base nessas informações uma forma de detectar movimentos de forma que seja possível identificar a quantidade de movimentos em cada parte de um frame pode ser realizada através da subtração do frame atual, correspondente ao exato instante que está sendo capturado e um frame anterior (SINGLA, 2014).

Essa comparação pode ser realizada com um frame mestre, o primeiro frame gerado que representa o ambiente em seu estado original (ALEX; WAHI, 2014), e qualquer objeto que entre no ambiente monitorado seria considerado uma mudança após realizada a subtração dos frames, essa abordagem realmente torna possível a comparação, identificação do local e a quantificação dos movimentos ocorridos em um frame.

Porém usando um frame mestre qualquer mudança como a simples mudança de iluminação natural causada pelo passar do tempo, ou uma nuvem tornar o dia mais escuro seria considerada após a comparação do frame atual com o mestre uma modificação no ambiente e necessitaria de tratamento adicional para remover a variável grau de iluminação do cálculo de movimento como citado por Singla (2014).

Outra forma de comparação é o método de diferença entre frames também conhecido como método de diferença de frames adjacentes, que refere-se a comparação entre frames de pequenos intervalos de tempo (ALEX; WAHI, 2014).

Ao realizar a comparação com o antecessor, mudanças geradas pela iluminação somente seriam percebidas se fossem mudanças bruscas como o farol de um carro iluminando parte do ambiente monitorado.

Porém ao comparar um determinado frame com seu antecessor pode ocorrer que um objeto entre na área monitorada e seja considerado como movimento, pois o frame anterior não possuía esse objeto capturado e se esse objeto permanecesse imóvel ele seria incorporado ao ambiente e não mais seria considerado como um objeto em movimento.

Esse comportamento pode não ser desejado em determinadas situações, porém como o objetivo do presente trabalho é auxiliar na recuperação de conteúdo com base nos movimentos capturados o importante é saber quando um movimento se iniciou e quando concluiu, pois qualquer informação entre esses pontos poderia ser recuperada com bases em regras específicas que são possíveis de serem estabelecidas quando se possui um modelo pesquisável de vídeo.

Durante ou antes da comparação não é realizado nenhum tipo de tratamento na imagem, pois o objetivo nessa etapa é única e exclusivamente identificar onde na imagem houve modificações e quantificar essa mudança para que então tanto os dados gerados através da comparação como o próprio vídeo possam ser salvos de forma relacionada.

4.3.1.1 Criação do algoritmo de Detecção de Movimentos

Neste tópico será abordado a criação do algoritmo de detecção de movimentos que servirá de base para o sistema de pesquisa em vídeo.

Existem muitos algoritmos de detecção de movimentos prontos usando a biblioteca Aforge.NET¹ de acordo com a documentação², porém neste trabalho optou-se pela criação de um algoritmo específico capaz de ser adaptável as peculiaridades do sistema proposto.

O Figura 11 ilustra como é o funcionamento do algoritmo de detecção de movimentos.

1 <http://www.aforgenet.com/>

2 http://www.aforgenet.com/framework/features/motion_detection_2.0.html

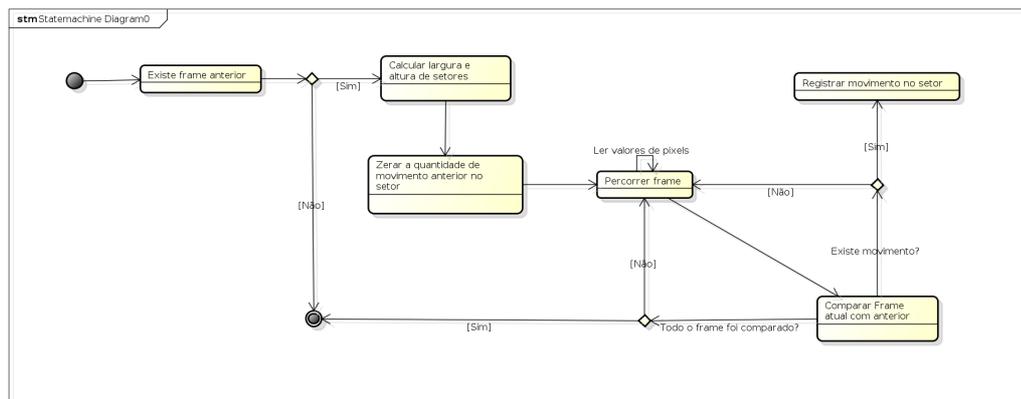


Figura 11: Funcionamento do algoritmo de detecção de movimentos

Fonte: próprio autor

Como é necessário realizar uma comparação com um frame anterior para determinar a quantidade de movimento presente na imagem, caso não haja nenhum frame anterior não há motivo para verificação de movimentos.

Para saber a qual setor da imagem um determinada coordenada (x,y) do frame pertence é necessário saber a largura e altura do setor, ou seja, quantos pixels estão presentes na largura e altura de um setor e conseqüentemente em que posições iniciam e terminam os limites de um setor.

A ilustração abaixo mostra de forma mais detalhada o motivo desse cálculo na determinação do local do movimento.

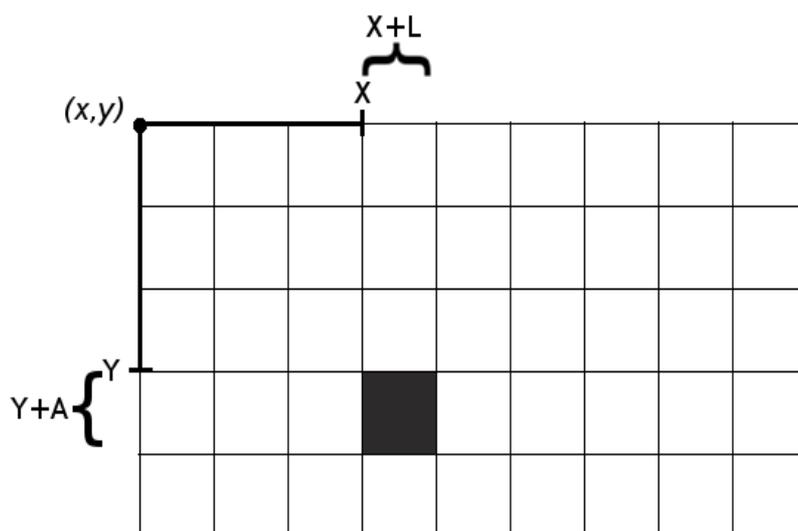


Figura 12: Coordenadas de um pixel e sua região correspondente

Fonte: próprio autor

A ilustração acima visa demonstrar que a posição de um setor em um frame tem início exatamente nas coordenadas (x,y) do frame e se estende até as dimensões do setor representadas pelas letras “A” altura e “L” largura.

Logo, todos os pixels entre essas coordenadas pertencem ao setor em destaque, assim como qualquer movimento que venha a ser identificado nesses pixels.

4.3.1.1.1 Nível de Sensibilidade da Detecção de Movimentos

Durante a execução do protótipo foi detectado que ao comparar imagens de momentos diferentes é comum encontrar diferenças pequenas, mesmo quando visualmente não seja possível identificar nenhuma e de acordo com Singla (2014) qualquer sistema de detecção de movimento com base na subtração de fundo precisa lidar com os ruídos presentes na imagem devido a uma baixa qualidade da fonte da imagem.

Nesse sentido torna-se necessária a criação de um filtro para definir qual o nível de movimento deverá ser considerado relevante, esse nível deve ser acessível

ao usuário do sistema de forma que seja possível ajustar seu valor com base no modelo de câmera usado, resolução do vídeo capturado e especificidades do ambiente monitorado.

O nível de comparação visando encontrar diferenças de um pixel com outro é com base nos valores numéricos das camadas de cores (RGB) e para remover pequenas mudanças ocasionadas por diferenças pequenas de luminosidade foi criado o seguinte algoritmo:

```
public unsafe bool comparePixels(byte* p1, byte* p2, int precisao){
    try
    {
        if (p1[0] < p2[0] - precisao ||
            p1[0] > p2[0] + precisao ||
            p1[1] < p2[1] - precisao ||
            p1[1] > p2[1] + precisao ||
            p1[2] < p2[2] - precisao ||
            p1[2] > p2[2] + precisao ||
            p1[3] < p2[3] - precisao ||
            p1[3] > p2[3] + precisao)
            return true;
        return false;
    }
    catch (AccessViolationException){
        return false;
    }
}
```

A variável “precisao” presente no algoritmo representa uma margem em que pequenas mudanças não sejam consideradas ao se comparar pixels de imagens diferentes.

Para essa variável, durante testes realizados no protótipo, foi identificado através da execução do protótipo como padrão uma margem de aproximadamente 20 para cada canal de cor referente as imperfeições e ruídos presentes na captura do vídeo.

Assim, após comparar pixels de diferentes frames o algoritmo teria o seguinte comportamento ilustrado pela Tabela 1.

Tabela 1 – Exemplo de execução do algoritmo de detecção de movimentos			
	Pixel 1	Pixel 2	Resultado
Caso1	rgb(125,78,156)	rgb(125,78,156)	Não
Caso2	rgb(22,13,28)	rgb(15,19,32)	Não
Caso3	rgb(190,30,70)	rgb(138,30,70)	Sim
Caso4	rgb(0,0,0)	rgb(0,0,0)	Não
Caso5	rgb(0,0,0)	rgb(255,255,255)	Sim

Fonte: Próprio autor

No “Caso1” não houve modificações, ou seja, não houve movimento nesse pixel, no “Caso2” todos os valores possuem diferenças mas suas diferenças então dentro da margem de precisão.

Quando a diferença entre apenas um dos valores supera a margem de precisão essa diferença é considerada como movimento, como ocorrido no “Caso3”.

No “Caso4” não houve modificações logo não há movimento já no “Caso5” houve alterações em todos os valores e em todos a diferença supera a margem de precisão.

A comparação de valores pode parecer simples e levar presumir que mudanças ocorridas em poucos pixels sejam consideradas como movimento, porém o que é armazenado no banco de dados não são as diferenças entre os pixels.

A comparação exemplificada na tabela 1 tem a única função de identificar se um determinado pixel mudou comparado com o pixel de mesma posição de um frame diferente.

O que é armazenado no banco de dados é a quantidade de pixels que mudaram em um determinado setor e quando o resultado de uma comparação entre dois pixels resulta em movimento, a quantidade de movimento do setor correspondente as coordenadas do pixel comparado é incrementada.

Após a análise de todos os pixels de cada setor do frame, os dados referentes a quantidade de pixels alterados em cada setor é salvo no banco de dados para servir de base para a pesquisa.

4.3.1.1.2 Divisão do Frame em Setores Lógicos

A principal característica do modelo pesquisável de vídeo proposto neste trabalho não está na capacidade de determinar quando ocorreu algum movimento e sim na capacidade de determinar em que local do frame ocorreu algum movimento em um dado intervalo de tempo informado.

Para realizar este tipo de pesquisa é necessário guardar o local em que houve movimento e a quantidade de movimento, ou seja, uma divisão do frame em partes menores se faz necessária para identificar e salvar valores específicos de cada região.

Usando o modelo de divisão de frame proposto por este trabalho é possível criar diversos setores dependendo da resolução do vídeo capturado, a quantidade é definida pelo usuário e como valor padrão foi definido uma divisão de 24 setores na horizontal por 16 na vertical totalizando 384 setores possíveis de serem selecionados por um usuário na realização de uma pesquisa.

Vídeos com divisões diferentes podem possuir quantidades diferentes de setores e cada setor pode possuir tamanho diferente, porém é possível realizar pesquisas mesmo em banco de dados com setores de tamanhos e quantidades diferentes bastando a criação de um algoritmo de adaptação de um modelo para o outro, porém a criação dessa funcionalidade foge ao escopo deste trabalho então sempre será usado o modelo padrão de 384 setores em todos os testes realizados seja na gravação, processamento ou pesquisa.

4.3.1.1.3 Localização dos Movimentos nos Setores

Uma imagem é uma função bi-dimensional composta pelos valores correspondentes as coordenadas em uma imagem (GONZALEZ; WOODS, 2007). Ao repartir uma imagem em setores torna-se necessário identificar o setor correspondente com base nas coordenadas (x,y) da imagem onde houve movimento.

A seguinte formula visa identificar o setor a que pertence uma determinada coordenada (x,y) de uma imagem:

Fórmula 1 - Identificação do setor dado um par de coordenadas em um frame

$$f(x, y) = \left\lfloor \frac{(x+1)}{\left(\frac{L}{h}\right)} \right\rfloor * h + \left\lfloor \frac{y}{\left(\frac{A}{v}\right)} \right\rfloor$$

Onde:

“x” é a posição horizontal no frame

“y” é a posição vertical no frame

“L” é o tamanho da imagem na horizontal

“A” é o tamanho da imagem na vertical

“h” é a quantidade de setores na horizontal que o frame possui

“v” é a quantidade de setores na vertical que o frame possui

As duas funções identificadas pelos caracteres “[]” e “[]” representam respectivamente o inteiro anterior a um dado número racional e o inteiro posterior a um dado número racional.

Assim:

Formula 2 – Função menor inteiro

$$\lfloor x \rfloor = \{ n \in \mathbb{Z}_+ \mid n \leq x \wedge x \leq n+1 \}$$

Exemplo:

X	$\lfloor x \rfloor$
3,8	3
0	0
3	3
2,1	2
2,9	2

Formula 3 – Função maior inteiro

$$\lceil x \rceil = \{ n \in \mathbb{Z}_+ \mid n \geq x \wedge n < x+1 \}$$

Exemplo:

X	$\lceil x \rceil$
3,8	4
0	0
3	3
2,1	3
2,9	3

4.3.2 Banco de Dados

Os dados identificados pelo processamento do vídeo devem ser armazenados em um banco de dados para posterior pesquisa. Neste trabalho optou-se por utilizar o banco de dados MySQL para salvar esses dados.

4.3.2.1 Modelo

O banco de dados usado para criar o sistema é simplesmente formado por duas tabelas como pode-se ver na Figura 13.

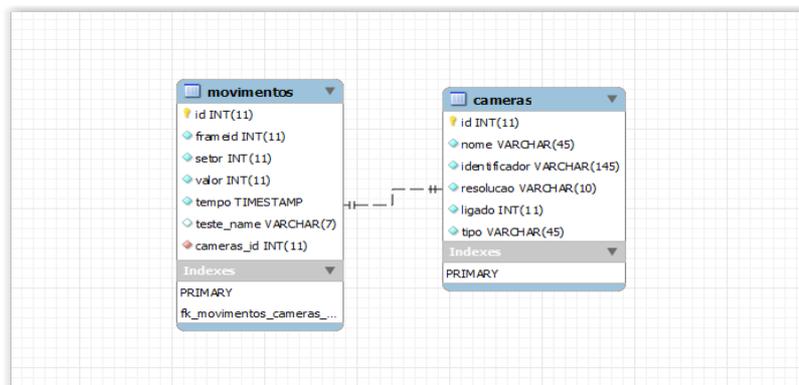


Figura 13: Modelo físico do banco de dados usado para salvar os movimentos e câmeras

Fonte: próprio autor

Cada tupla da tabela de movimentos representa um setor e sua respectiva quantidade de movimento.

A descrição da função de cada campo é detalhada na Tabela 2 e Tabela 3 informadas abaixo.

Tabela 2 – Descrição dos campos da tabela de movimentos	
Campo	Descrição
frameid	Identificador do frame processado relacionado com o nome do arquivo de imagem gerado
setor	Setor correspondente
valor	Quantidade de movimentos identificada no setor
tempo	Quando a captura do vídeo ocorreu
teste_name	Nome do teste executado “relevante apenas para testes”
camera_id	Identificação da câmera de origem

Fonte: próprio autor

Tabela 3 – Descrição dos campos da tabela de câmeras	
Campo	Descrição
nome	Nome da câmera
identificador	Identificador da câmera “Identificador do hardware ou endereço IP”
resolucao	Resolução definida para gravação
ligado	Identifica se a câmera está ligada ou não
tipo	Tipo de câmera USB ou IP

Fonte: próprio autor

4.3.2.2 Salvando dados no Banco

Como as informações relativas aos movimentos de cada setor precisam ser salvas no banco de dados é essencial que uma forma de otimizar o processo de inserção seja realizada, pois nos testes realizados neste trabalho cada frame possui 384 setores, ou seja, a cada segundo 384 inserções são realizadas, pois a taxa de análise do vídeo escolhida foi de 1 frame por segundo.

Segundo a documentação do MySQL³, o tempo necessário para a inserção de uma linha no banco de dados é determinada pelos seguintes factores, onde os números indicam proporções aproximadas:

- Conexão: (3)
- Enviar consulta ao servidor: (2)
- Análise da consulta: (2)
- Inserções consecutivas: (1 × tamanho da linha)
- Inserção de índice: (1 × número de índices)
- Encerramento: (1)

Ainda de acordo com a documentação² para otimizar a velocidade de inserção, pode-se combinar muitas pequenas operações em uma única operação. O ideal é que isso permita uma única ligação, enviar muitos dados de uma vez, e atrasar todas as atualizações de índice e verificação de consistência até a conclusão da operação.

Dessa forma os dados identificados no processamento inicial do vídeo devem ser armazenados temporariamente na memória RAM até acumular uma certa quantidade, aproximadamente equivalente a 5 segundos de vídeo processado, e então todos estes dados são enviados para serem salvos no banco de dados de uma vez.

O processo de captura e geração dos arquivos de imagem acontece independentemente do banco de dados, logo, não precisam ser mantidos na memória RAM até a conclusão do processo de salvamento dos dados.

4.3.3 Arquivos de Imagem

Os vídeos são salvos em formato de imagem para facilitar a recuperação e seu nome está relacionado com os dados relativos aos movimentos identificados salvos no banco de dados.

O nome de cada imagem segue o formato ano-mês-dia-hora-minuto-segundo-milésimos, por exemplo: 2016-06-17-10-21-23-6720.

3 <http://dev.mysql.com/doc/refman/5.7/en/insert-speed.html>

O formato de imagem definido como padrão usado nos testes foi o jpeg.

4.3.4 Funcionamento do Algoritmo

Nesse tópico é apresentado o fluxograma que representa o funcionamento do algoritmo em relação à pesquisa, captura e processamento do vídeo.

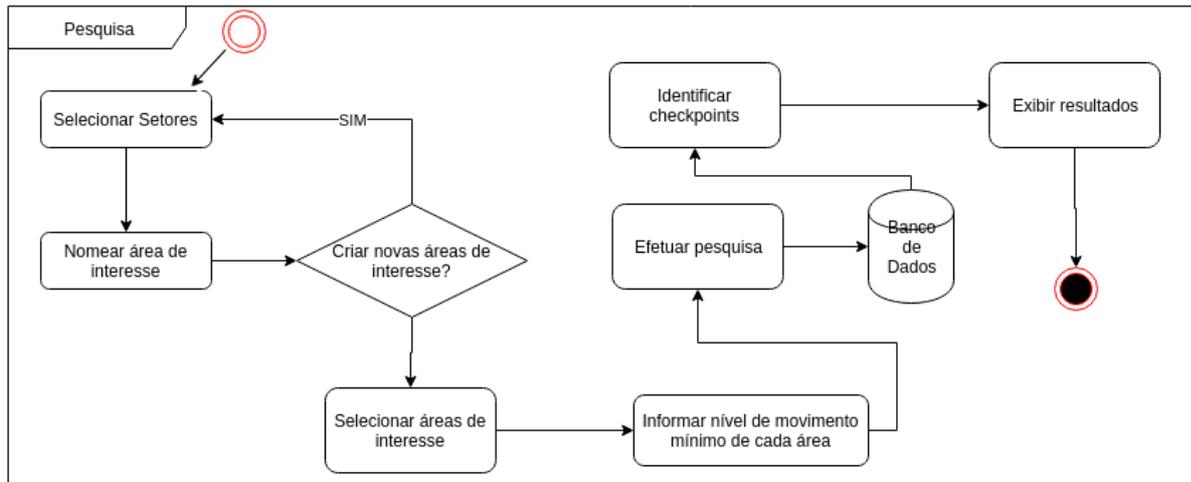


Figura 14: Fluxograma do funcionamento da pesquisa

Fonte: próprio autor

O usuário deve informar ao menos uma região de interesse para a pesquisa, essa região deve ter sido criada previamente após selecionar todos os setores de interesse e definir um nome para essa área.

Diversas regiões de interesse podem ser criadas e usadas em conjunto ou individualmente em uma pesquisa.

A identificação de checkpoints visa identificar momentos independentes capturados com base em movimentos ocorridos na área de interesse, dessa forma se uma pessoa passou pela área de interesse e após 5 segundos retornou a passar por essa mesma área, após uma pesquisa o resultado seria composto por dois checkpoints.

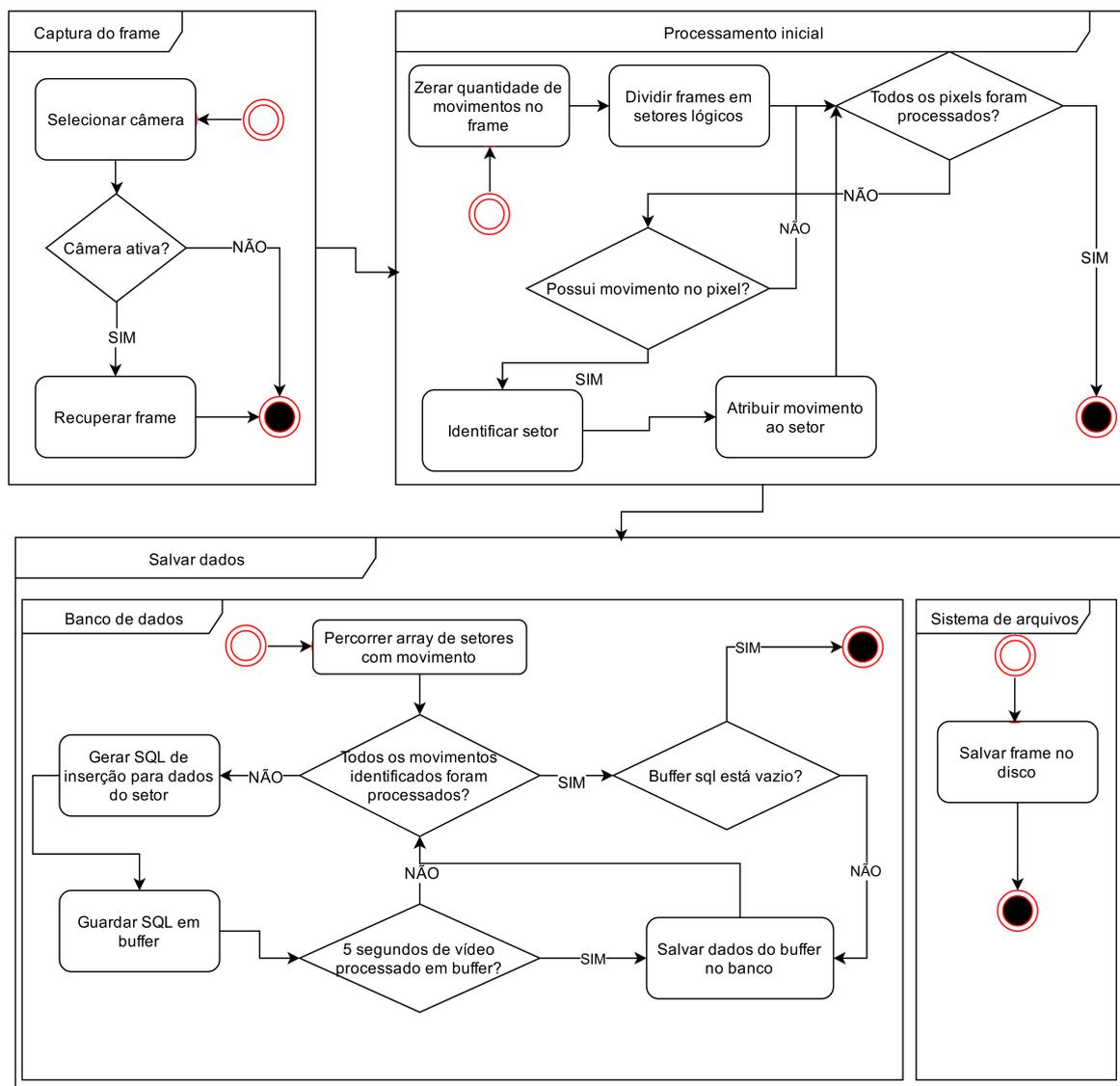


Figura 15: Fluxograma do funcionamento do algoritmo de captura, processamento inicial e salvamento no banco de dados e sistema de arquivos

Fonte: próprio autor

No Figura 15 é ilustrado como o algoritmo trabalha para criar o modelo pesquisável do vídeo, partindo da captura dos dados do vídeo e do processamento inicial visando identificar os movimentos assim como o local onde cada movimento ocorreu e o salvamento tanto das informações extraídas do vídeo quanto o próprio arquivo.

4.3.5 Pesquisa

Neste tópico é abordado de forma explicativa como a pesquisa em vídeo funciona com a intenção abstrair de maneira simples o funcionamento básico do software final.

Em um vídeo podem ter sido capturados diversos movimentos, mas nem todo movimento pode ser considerado importante para fazer parte do resultado de uma pesquisa, assim uma forma de selecionar regiões em um ambiente monitorado é necessária.

Como os movimentos são identificados em relação ao seu respectivo setor é representado na Figura 16, presente abaixo.

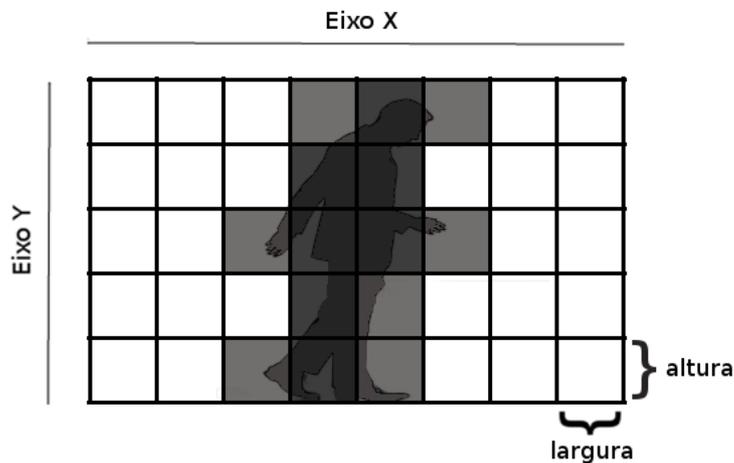


Figura 16: Ilustração de movimento em um frame de vídeo indicando os setores onde existe maior quantidade de movimentos

Fonte: próprio autor

Na ilustração acima é possível perceber que ao dividir um frame em regiões é possível identificar em quais regiões houve movimentos assim como o nível de movimento presente em cada parte do frame.

Dessa forma, um usuário poderia selecionar suas regiões de interesse assim como excluir as regiões que mesmo possuindo movimentos não possuem importância para sua pesquisa.

O resultado desse modelo seria um mecanismo de pesquisa capaz recuperar conteúdo com base em regras dinâmicas que se comportaria da forma que o usuário precisa em cada situação.

4.3.5.1 Criação de Componente de Representação de Setores

O usuário deve ser capaz de criar regras complexas de forma simples, então foi criado um componente WPF, uma modificação do componente Grid comumente usado para exibir dados em forma de lista.

Esse componente cria uma grade sobre a área de vídeo como ilustrado na Figura 17 e permite ao usuário selecionar os setores que deseja e dar um nome para esse conjunto.

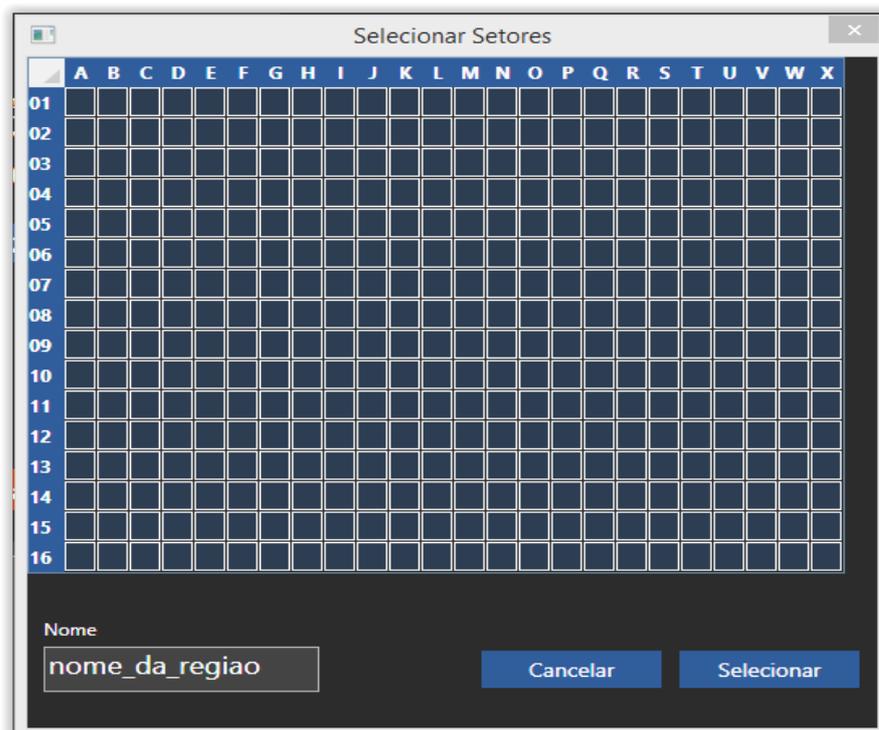


Figura 17: Componente para selecionar área de interesse para pesquisa

Fonte: próprio autor

Com todos os conjuntos de setores do interesse da pesquisa selecionados e devidamente nomeados o usuário pode então criar sua pesquisa usando os conjuntos como parâmetros para inserir ou excluir áreas de interesse assim como especificar o nível de movimento em cada conjunto.

Abaixo é mostrado na Figura 18, como o usuário poderia criar uma pesquisa com base em apenas algumas regiões que são de seu interesse.

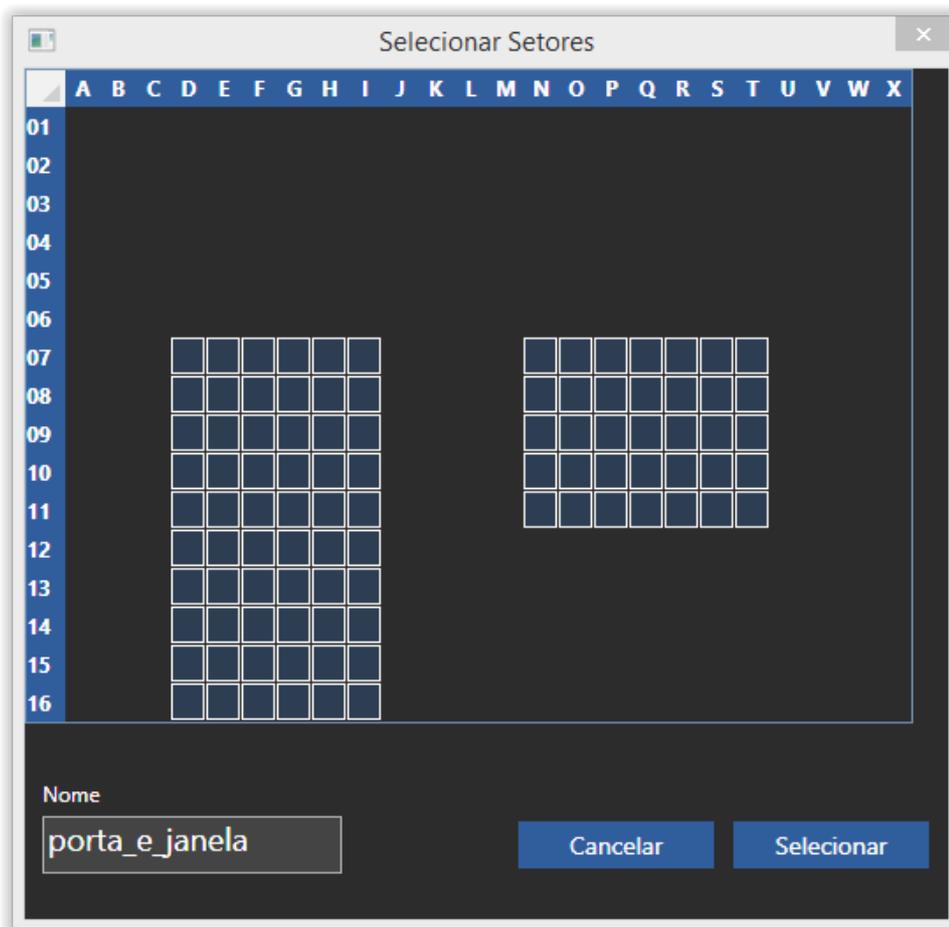


Figura 18: Ilustração de apenas parte de um ambiente selecionada para pesquisa

Fonte: próprio autor

Na ilustração acima, apenas movimentos que ocorreram na porta ou na janela possuem interesse para o usuário.

Várias seleções podem ser definidas como ilustrado na Figura 19.

Cada uma das seleções como exemplificado na Figura 19, recebem um nome e podem ser usadas para a realização da pesquisa.

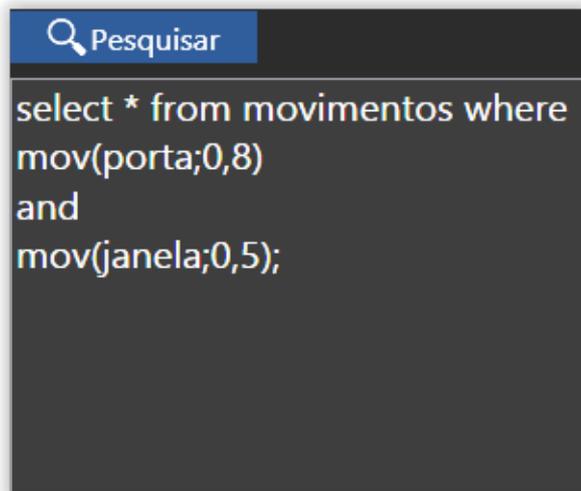


Figura 19: Seleções criadas e possíveis de serem usadas como parâmetros de uma pesquisa

Fonte: próprio autor

Para a realização da pesquisa foi definida a linguagem SQL como base. O usuário então deve escrever sua consulta usando as seleções como parâmetros e informando o nível de movimento mínimo desejado.

A Figura 20 ilustra o código necessário para realizar uma pesquisa em duas áreas de interesse.

A screenshot of a search interface. At the top, there is a blue button with a magnifying glass icon and the word "Pesquisar". Below it, a dark grey box contains a SQL query in white text: "select * from movimentos where mov(porta;0,8) and mov(janela;0,5);".

```
select * from movimentos where
mov(porta;0,8)
and
mov(janela;0,5);
```

Figura 20: Exemplo de consulta básica usando áreas selecionadas no ambiente monitorado

Fonte: próprio autor

Qualquer outra condição poderia ser adicionada à pesquisa como: tempo de gravação dos frames, exclusão de áreas sem interesse, etc.

4.3.5.2 Classificação Automática de Relevância

O resultado de uma pesquisa pode conter diversas informações e apresentar essas informações de forma amigável visualmente para o usuário é algo essencial, pois segundo Vieira e Correa (2011), uma interface baseada na visualização possibilita a compreensão de como está organizada a informação no sistema, oferecendo uma visão geral dos assuntos tratados, contextualizando o resultado da busca e permitindo o descobrimento de novos conceitos e relações entre eles.

Ainda segundo Vieira e Correa (2011), as informações representadas por meio de recursos visuais auxiliam os Sistemas de Recuperação de Informação (SRIs) a exercerem seu objetivo principal de recuperar informações relevantes que atendam a necessidade informacional de seus usuários.

O usuário, então, deve ser capaz de facilmente abstrair o resultado de uma pesquisa e analisar tanto o resultado mais detalhadamente como refinar sua pesquisa com a intenção de alcançar um resultado mais favorável através de uma nova pesquisa.

Com base na importância da organização na apresentação das informações aos usuários foi criado uma forma automática de classificar os conteúdos que retornam de uma pesquisa com base no tempo de duração dos movimentos.

Essa forma de classificação ilustrada na Figura 21, visa colocar em destaque movimentos que ocorreram por um período maior de duração, pois altos níveis de movimentos em apenas alguns milésimos de segundo pode ser menos relevante do que um movimento mais sutil porém que durou alguns segundos ou minutos.

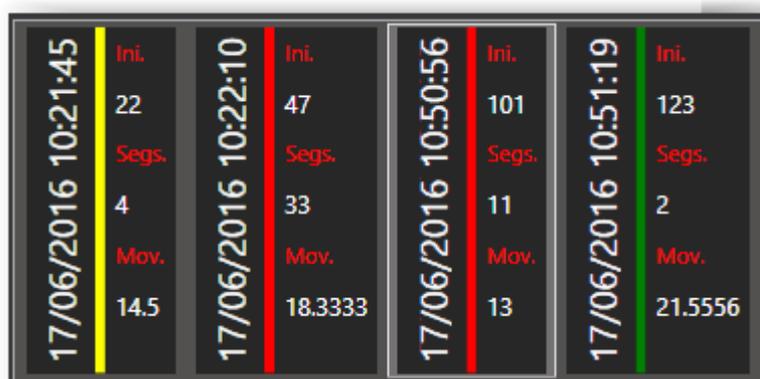


Figura 21: Resultado de uma pesquisa classificado de acordo com sua relevância presumida

Fonte: próprio autor

Após realizada uma pesquisa o resultado é agrupado por duração de movimento de acordo com a seguinte tabela:

Tabela 4 – Identificação das cores na classificação dos resultados da pesquisa	
Duração do movimento	Cor descritiva
Até 3 segundos	Verde
De 4 a 10 segundos	Amarelo
Acima de 10 segundos	Vermelho

Fonte: próprio autor

A classificação de movimentos não visa colocar em destaque parte da pesquisa que possui mais relevância e sim identificar momentos recuperados no

resultado da pesquisa de forma que seja mais simples a compreensão desse resultado pelo usuário.

4.4 Realização de Testes

Neste tópico será abordado como a viabilidade do algoritmo proposto será atestada.

4.4.1 Introdução aos Testes

Os testes se baseiam na principal característica do algoritmo, ou seja, na recuperação de conteúdo de vídeo com base em movimento ocorrido em determinadas áreas que são relevantes ao usuário.

Dessa forma, em cada teste uma pessoa repete uma determinada sequência de ações e o algoritmo terá a função de identificar esses comportamentos como ações independentes.

Um objeto será colocado na área monitorada e uma pessoa repetirá as seguintes ações interagindo com esse objeto:

- 1º – Uma pessoa entra na área monitorada e se dirige até o objeto e retorna por onde veio;
- 2º – Uma pessoa entra na área monitorada, se dirige ao objeto e sai da área monitorada em direção oposta de onde veio;
- 3º – Uma pessoa entra na área monitorada vindo do local onde saiu de cena na interação anterior, se dirige ao objeto e sai da área monitorada em direção oposta de onde veio;
- 4º – Uma pessoa entra na área monitorada, se dirige ao objeto pega o objeto e sai da área monitorada por onde veio.

Assim, após selecionar apenas a área próxima do objeto que está no ambiente monitorado, após uma pesquisa em cada teste quatro checkpoints

deveriam ser retornados representando cada uma das interações que esse objeto sofreu.

Os testes foram realizados em dias e horários diferentes assim como em ambientes internos e externos, com iluminação natural e artificial.

Através dessa sequência de ações é possível testar a capacidade do algoritmo de identificar movimentos que ocorreram em momentos diferentes, identificar movimentos apenas de uma região específica desprezando movimentos que ocorreram em áreas fora da região de interesse e medir o comportamento do algoritmo em diferentes ambientes com graus de iluminação diversos.

A Figura 22 informada abaixo tem a intenção de exemplificar melhor como a sequência de ações realizadas em cada um dos testes, as setas indicam as direções que a pessoa deverá seguir ao entrar e sair da área de interesse.

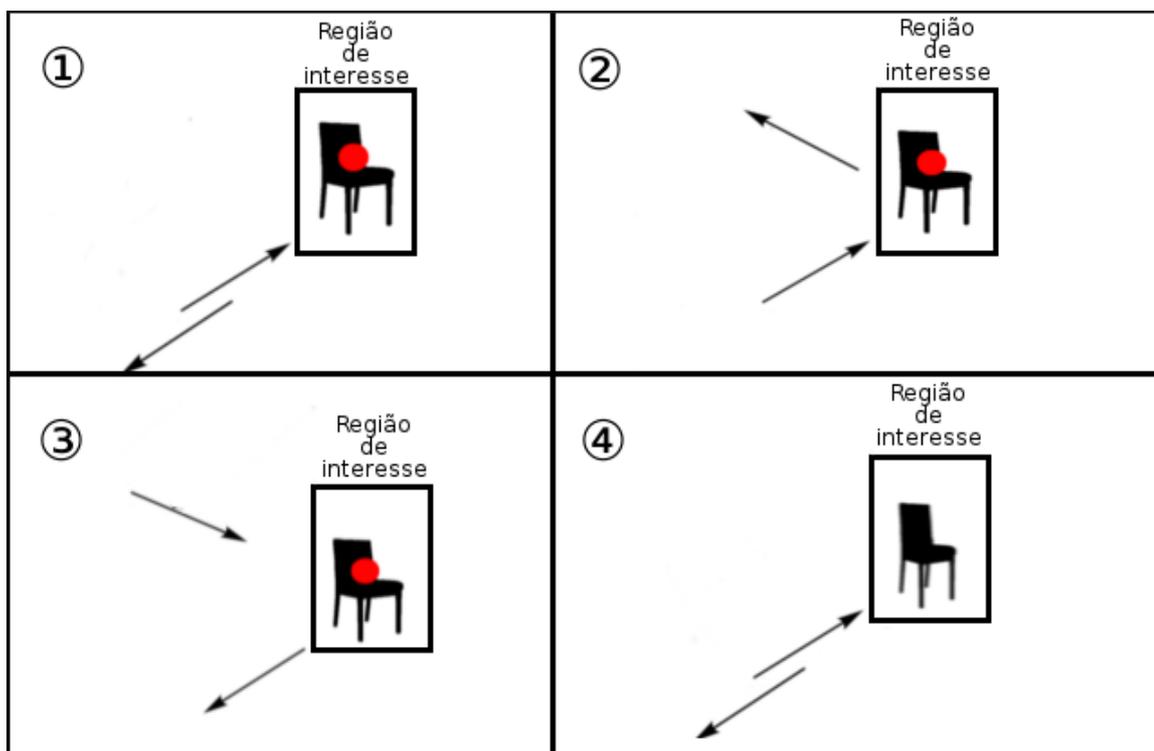


Figura 22: Sequência de ações realizadas em cada um dos testes

Fonte: próprio autor

Serão realizados testes para os seguintes ambientes:

Ambiente Interno

- iluminação natural e interações acontecendo 3 metros da câmera;
- iluminação artificial e interações acontecendo a 3 metros da câmera;
- iluminação natural e interações acontecendo a 20 metros da câmera;
- iluminação artificial e interações acontecendo a 20 metros da câmera;

Ambiente Externo

- iluminação natural e interações acontecendo 3 metros da câmera;
- iluminação artificial e interações acontecendo a 3 metros da câmera;
- iluminação natural e interações acontecendo a 20 metros da câmera;
- iluminação artificial e interações acontecendo a 20 metros da câmera;

Em cada caso 5 testes serão realizados com a intenção de aumentar a confiabilidade dos dados identificados.

Cada teste é representado pelo quadro de checkpoints gerado como resultado de uma pesquisa e pelo histograma que representa visualmente a quantidade de movimentos na área de interesse em relação ao tempo também retornado como resultado para cada pesquisa.

4.4.1.1 Interpretação dos Resultados

Este tópico é necessário para o correto entendimento da forma com que cada resultado é representado.

Cada teste gera um grupo de checkpoints como resultado de uma pesquisa assim como um histograma representando a quantidade, o nível e os intervalos entre cada movimento identificado, abaixo, na Figura 23, é mostrado um exemplo de um grupo de checkpoints gerado por uma pesquisa.

02/07/2016 15:02:30	Ini.	141	02/07/2016 15:03:04	Ini.	150	02/07/2016 15:03:40	Ini.	161	02/07/2016 15:04:15	Ini.	169
	Segs.	5		Segs.	3		Segs.	5		Segs.	4
	Mov.	20.6		Mov.	31.6429		Mov.	26.4286		Mov.	33.416

Figura 23: Exemplo de grupo de checkpoints retornados como resultado de uma pesquisa

Fonte: próprio autor

Na ilustração acima 4 checkpoints estão representando um resultado para uma pesquisa qualquer, cada checkpoint possui uma data na vertical informando quando exatamente o movimento iniciou e três números representando respectivamente o identificador do frame onde foi identificado movimento, a quantidade em segundos que o movimento durou e o nível de movimento médio para o intervalo identificado.

Cada pesquisa também gera também um histograma como resultado como ilustrado na figura 24.

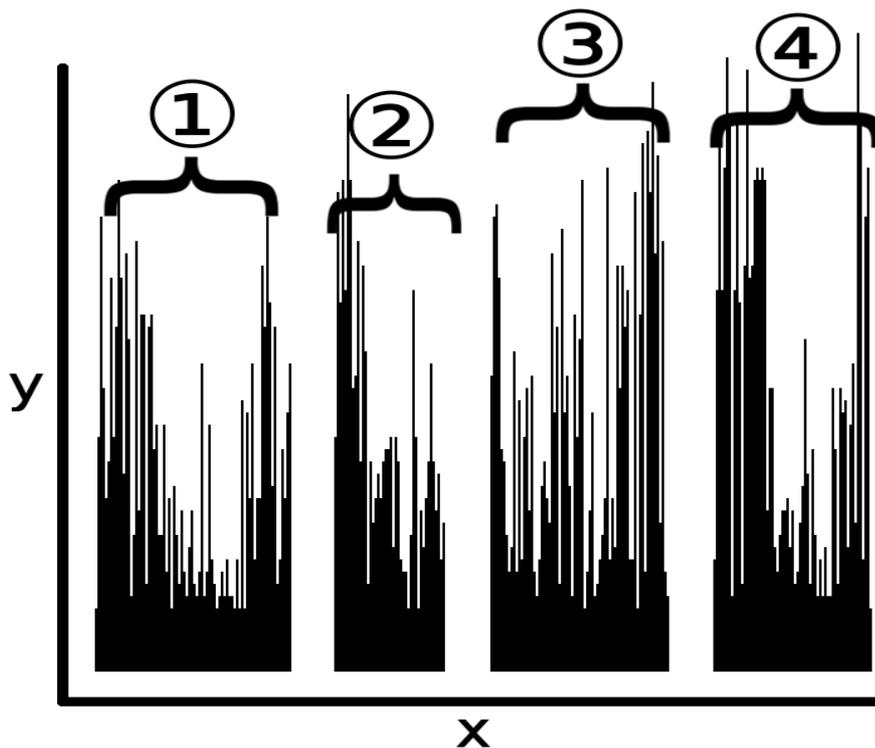


Figura 24: Exemplo de histograma ilustrando a duração e o nível de movimentos do resultado de uma pesquisa

Fonte: próprio autor

O histograma acima representa todos os movimentos retornados como resultado de uma pesquisa.

Cada agrupamento, identificado pelos números 1, 2, 3, 4, são referentes as quatro interações que ocorreram no ambiente.

O eixo “y” representa o nível do movimento identificado já o eixo “x” representa a duração completa de todos os movimentos identificados na área de interesse.

As lacunas entre cada agrupamento é referente ao momento de inatividade que ocorreu entre uma ação e outra.

4.4.2 Detalhes e Configurações

Na realização dos testes foi usado uma câmera IP capturando vídeos na resolução 640x480 a 30 frames por segundo.

A câmera em todos os testes ficou fixa a 3 metros de altura, apontando para o objeto principal dos testes.

Mesmo a câmera capturando 30 frames por segundo, apenas 1 frame por segundo é analisado pelo sistema em busca de movimento, porém todos os frames são salvos como arquivo independente de possuírem ou não movimento e de terem sido processados ou não.

A iluminação natural representa um ambiente claro naturalmente, geralmente os testes nesses casos são realizados as três horas da tarde, a informação da hora em que cada teste foi realizado está presente em cada checkpoint referente a cada teste executado.

4.4.3 Testes em ambientes internos

Neste tópico é abordada a realização dos testes em ambientes internos com interações ocorrendo a diferentes distâncias e em diferentes condições de iluminação.

Ao todo 20 (vinte) testes foram realizados em ambientes internos, diferentes graus de iluminação e distância da câmera foram usados para compreender o comportamento do algoritmo em diferentes condições em ambientes internos.

4.4.3.1 Testes realizados a três metros de distância da câmera

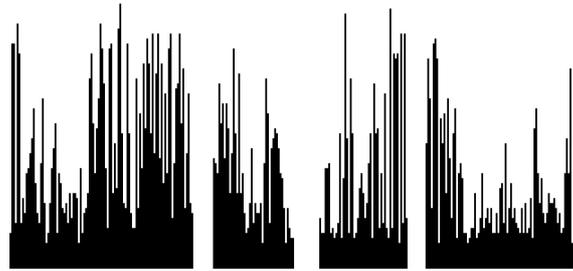
O objetivo da realização de testes a três metros de distância da câmera é identificar como o algoritmo de identificação de movimento se comportará com uma quantidade maior da imagem sendo alterada a cada ação executada nos testes.

4.4.3.1.1 Iluminação natural

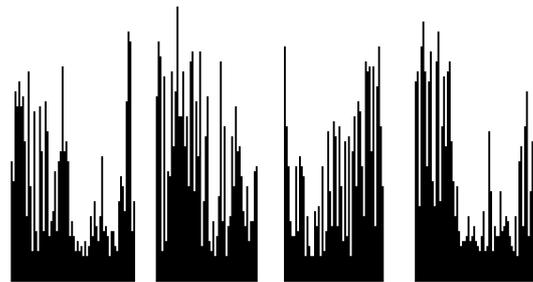
O seguinte teste foi realizado com a intenção de avaliar o comportamento do algoritmo em um ambiente interno durante o dia com movimentos ocorrendo próximos à câmera.

A figura abaixo representa tanto o grupo de checkpoints como o histograma gerado pela pesquisa em cada um dos 5 (cinco) testes realizados neste ambiente.

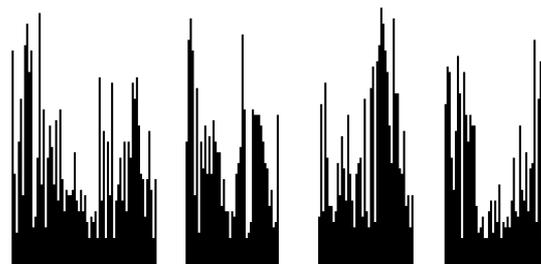
02/07/2016 14:52:25	Ini. 2	02/07/2016 14:52:47	Ini. 10	02/07/2016 14:53:15	Ini. 21	02/07/2016 14:53:35	Ini. 27
	Segs. 5		Segs. 3		Segs. 3		Segs. 6
	Mov. 26.7778		Mov. 12		Mov. 13.25		Mov. 11



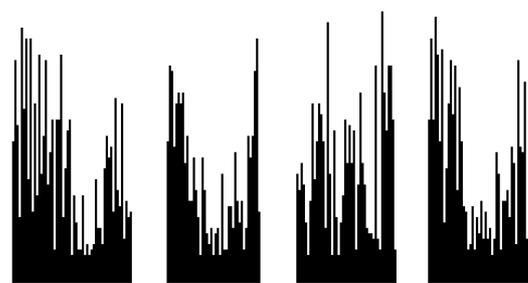
02/07/2016 14:55:50	Ini. 37	02/07/2016 14:56:12	Ini. 45	02/07/2016 14:56:40	Ini. 57	02/07/2016 14:57:13	Ini. 66
	Segs. 4		Segs. 4		Segs. 4		Segs. 4
	Mov. 28.2143		Mov. 28.875		Mov. 25.1667		Mov. 33.3333



02/07/2016 15:02:30	Ini. 141	02/07/2016 15:03:04	Ini. 150	02/07/2016 15:03:40	Ini. 161	02/07/2016 15:04:15	Ini. 169
	Segs. 5		Segs. 3		Segs. 5		Segs. 4
	Mov. 20.6		Mov. 31.6429		Mov. 26.4286		Mov. 33.416



02/07/2016 15:00:13	Ini. 107	02/07/2016 15:00:45	Ini. 115	02/07/2016 15:01:18	Ini. 127	02/07/2016 15:01:47	Ini. 134
	Segs. 4		Segs. 5		Segs. 4		Segs. 3
	Mov. 32.0909		Mov. 30.3333		Mov. 16.1429		Mov. 29



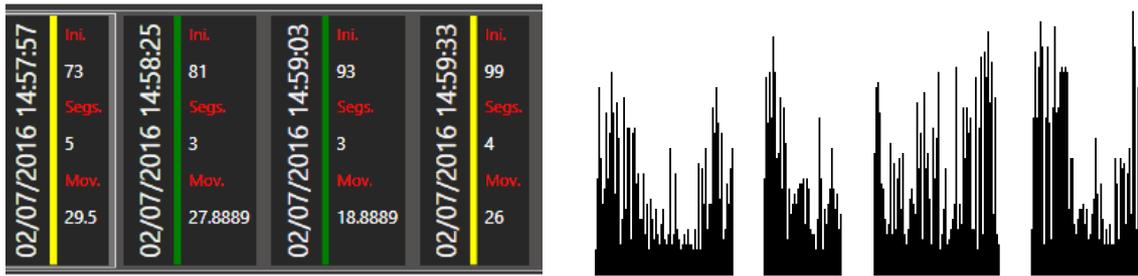


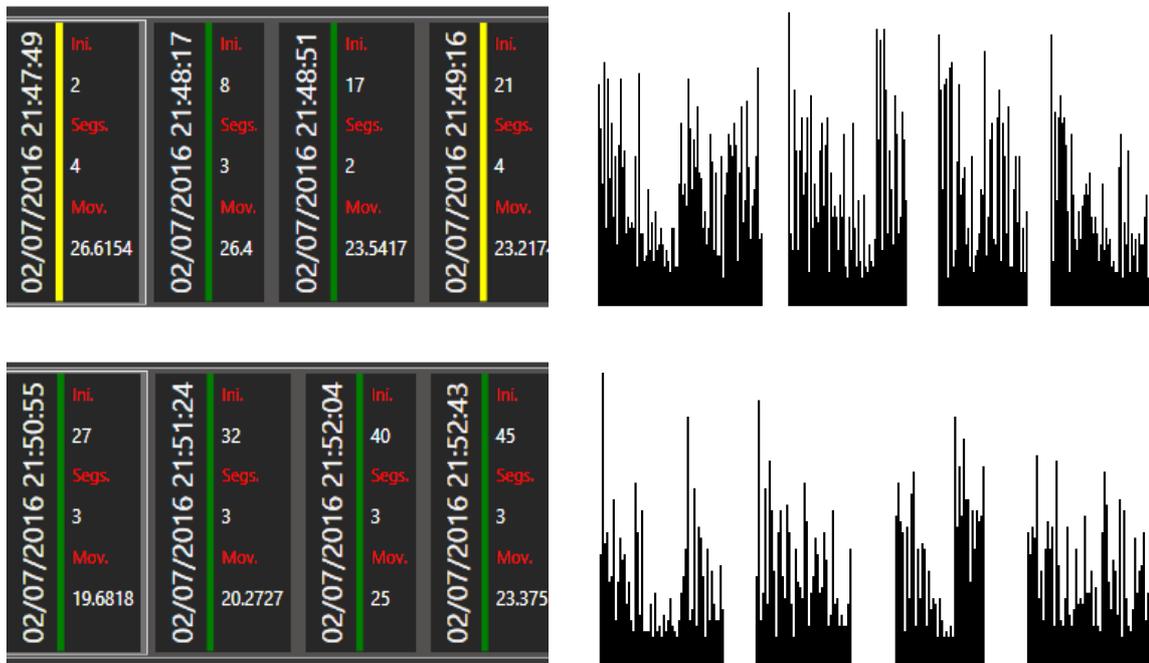
Figura 25: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno com iluminação natural

Fonte: próprio autor

4.4.3.1.2 Iluminação artificial

Este teste foi realizado com a intenção de avaliar o comportamento do algoritmo em um ambiente interno durante a noite com movimentos ocorrendo próximos à câmera.

A figura abaixo representa tanto o grupo de checkpoints como o histograma gerado pela pesquisa em cada um dos 5 (cinco) testes realizados neste ambiente.



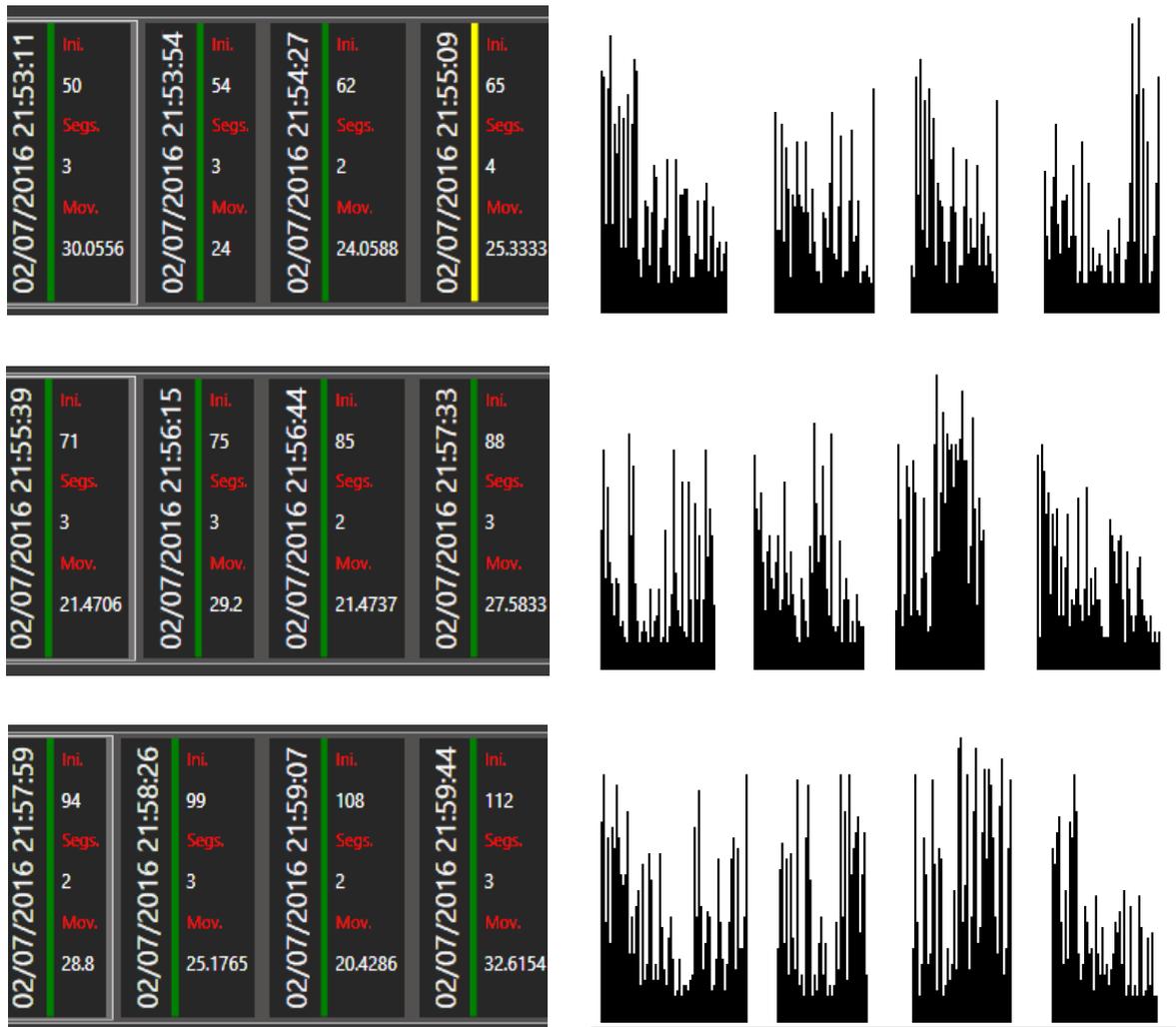


Figura 26: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a 3 metros de distância com iluminação artificial

Fonte: próprio autor

4.4.3.2 Testes realizados a vinte metros de distância da câmera

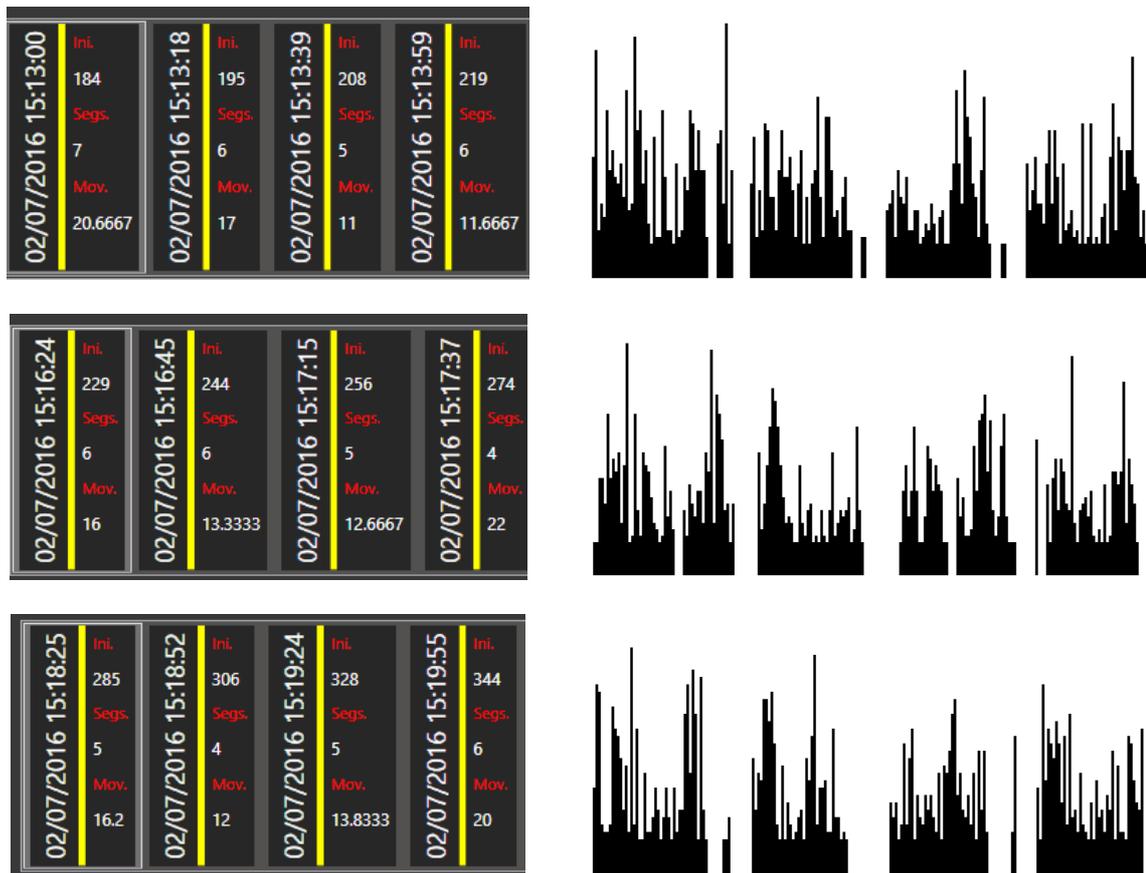
Os testes realizados a vinte metros da câmera visam verificar o comportamento do algoritmo quando os movimentos ocorrem longe da câmera.

Testes em ambientes com diferentes níveis de iluminação foram realizados com a intenção de identificar o comportamento do algoritmo com interações ocorrendo distante da câmera com diferentes graus de iluminação.

4.4.3.2.1 Iluminação natural

Com o objetivo de avaliar o comportamento do algoritmo em um ambiente interno durante o dia com movimentos ocorrendo longe da câmera foram realizados testes que geraram os seguintes resultados.

A figura abaixo representa tanto o grupo de checkpoints como o histograma gerado pela pesquisa em cada um dos 5 (cinco) testes realizados neste ambiente.



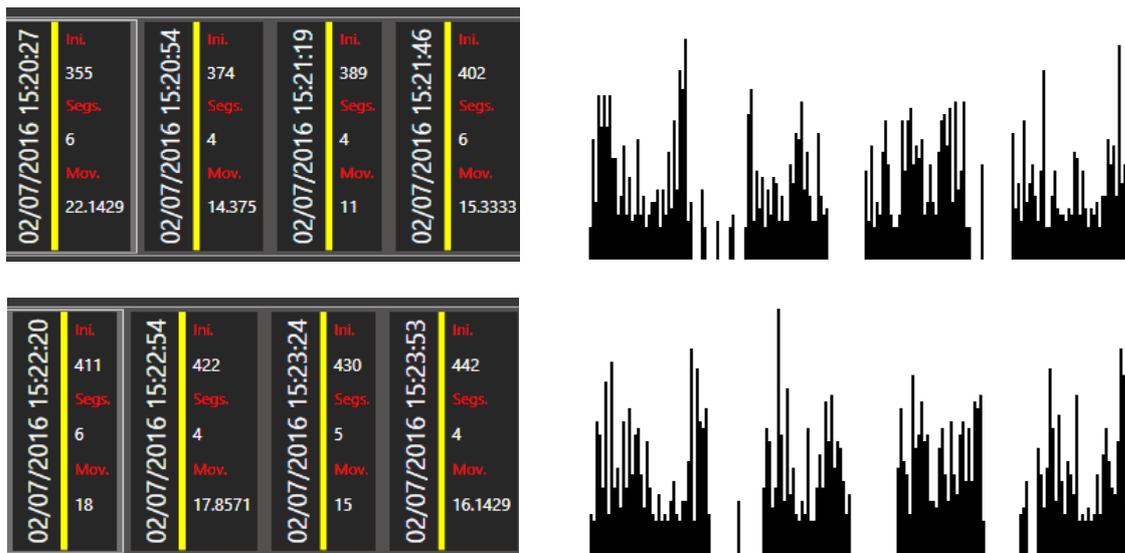


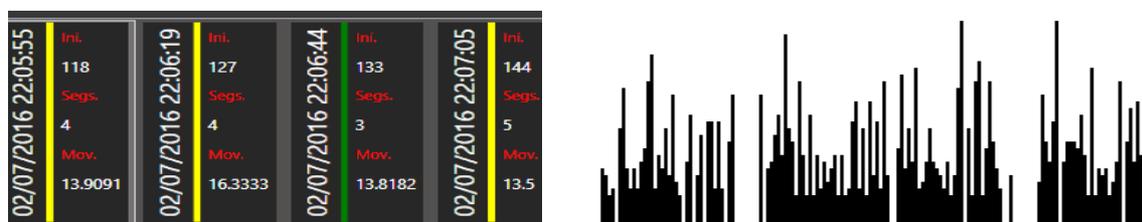
Figura 27: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a vinte metros de distância com iluminação natural

Fonte: próprio autor

4.4.3.2.2 Iluminação artificial

O seguintes resultados são referentes aos testes realizados com a intenção de avaliar o comportamento do algoritmo em um ambiente interno durante a noite com movimentos ocorrendo longe da câmera.

A figura abaixo representa então, tanto o grupo de checkpoints como o histograma gerado pela pesquisa em cada um dos 5 (cinco) testes realizados neste ambiente.



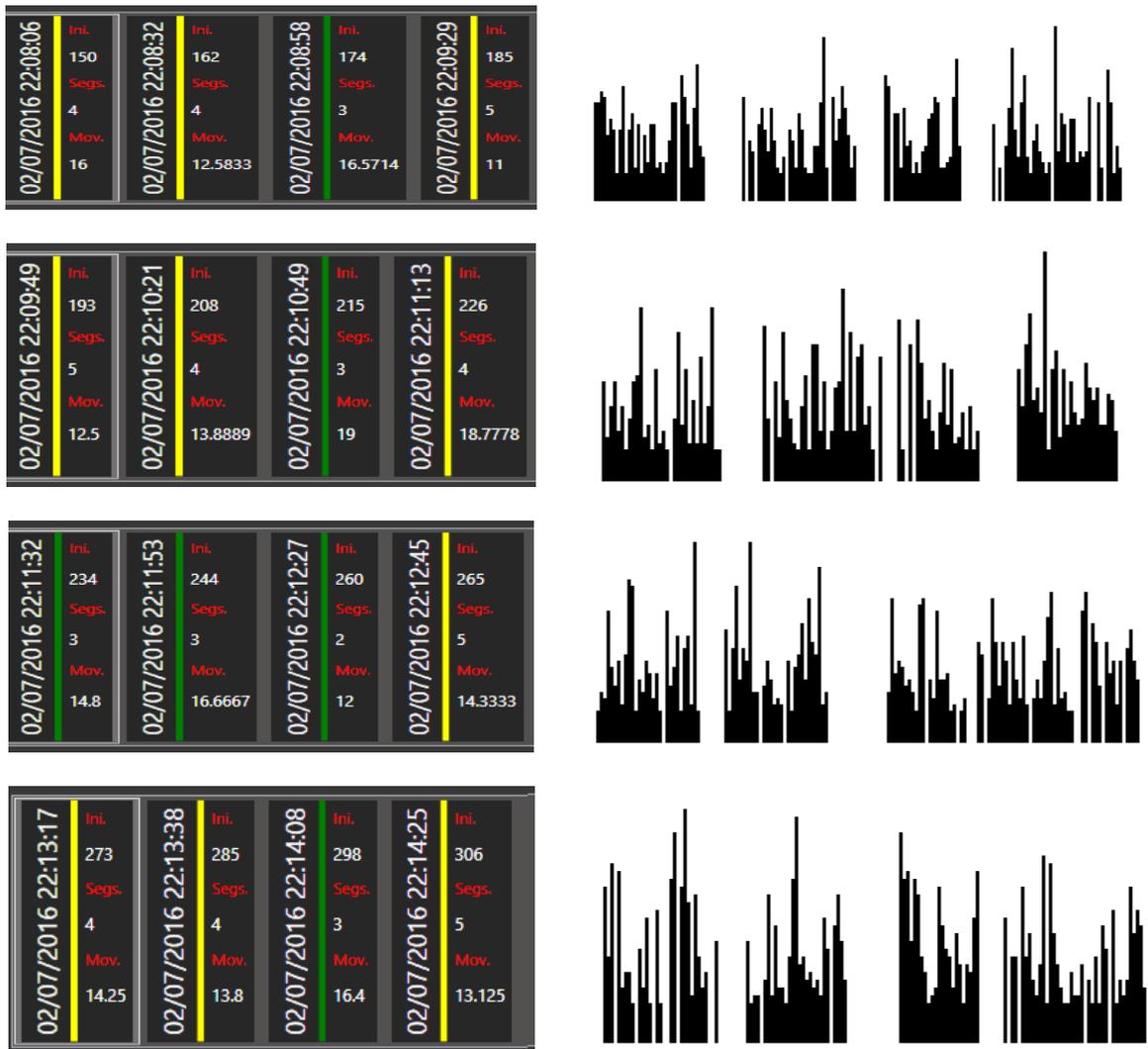


Figura 28: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente interno a vinte metros de distância com iluminação artificial

Fonte: próprio autor

4.4.4 Testes em ambientes externos

Neste tópico será abordada a realização dos testes em ambientes externos com interações ocorrendo a diferentes distâncias e em diferentes condições de iluminação.

Em cada caso figuras referentes a cada teste são informadas com a intenção de identificar o comportamento do algoritmos em cada ambiente testado.

4.4.4.1 Testes realizados a três metros de distância da câmera

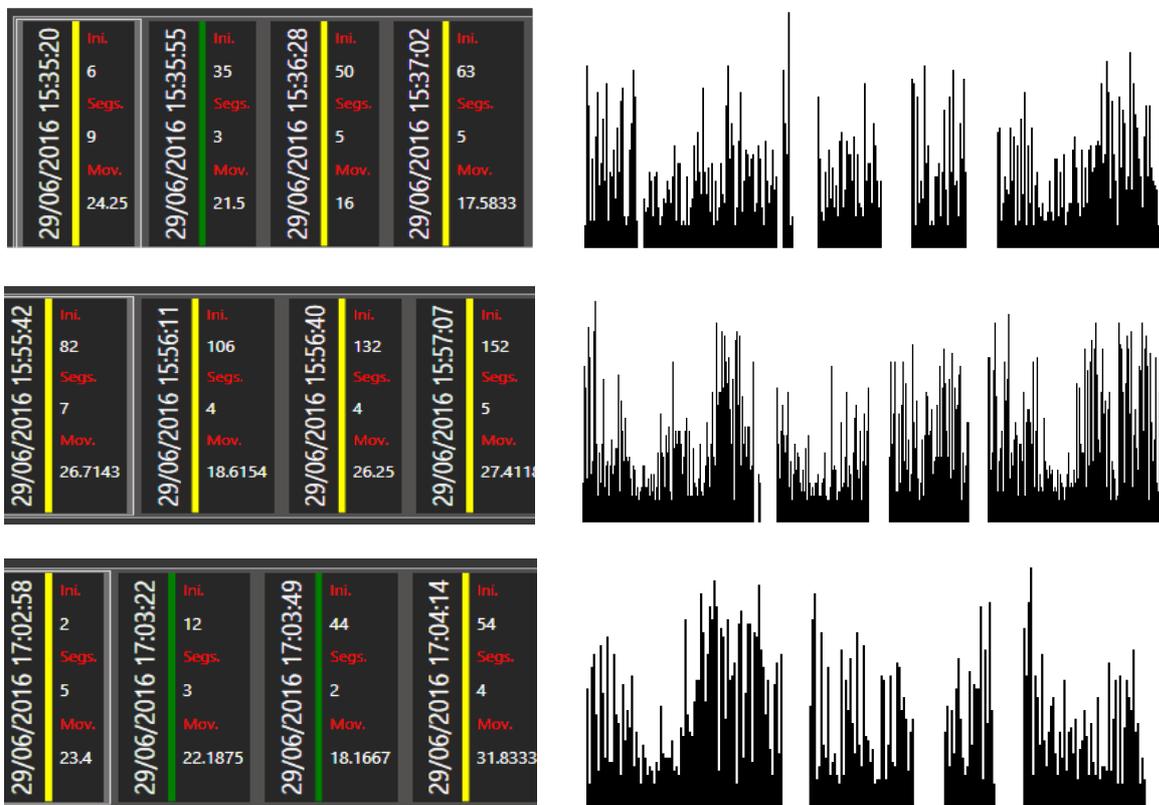
Os testes realizados a três metros da câmera visam verificar o comportamento do algoritmo quando os movimentos ocorrem próximo à câmera.

Aliado com diferentes graus de iluminação o objetivo deste grupo de testes é avaliar o desempenho do algoritmo em ambientes com bastante ou pouca iluminação com interações ocorrendo próximas da câmera.

4.4.4.1.1 Iluminação natural

Os seguintes resultados representam o comportamento do algoritmo na realização de pesquisas em ambientes externos durante o dia com movimentos ocorrendo próximos à câmera.

Cada dupla de imagens ilustradas na Figura 29 representam cada um dos 5 (cinco) testes realizados em iluminação natural com interações ocorrendo próximas à câmera em ambiente externo.



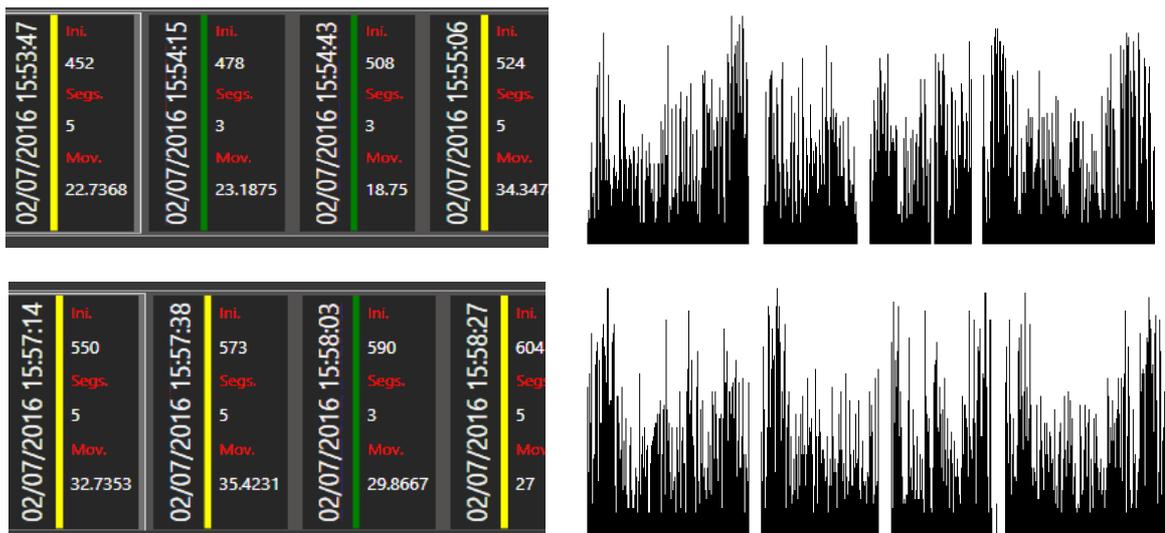


Figura 29: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a três metros de distância com iluminação natural

Fonte: próprio autor

4.4.4.1.2 Iluminação artificial

Os seguintes resultados representam o comportamento do algoritmo na realização de pesquisas em ambientes externos durante a noite com movimentos ocorrendo próximos à câmera.

A Figura 30 representa os testes realizados em ambiente externos com iluminação artificial com interações ocorrendo próximas à câmera.



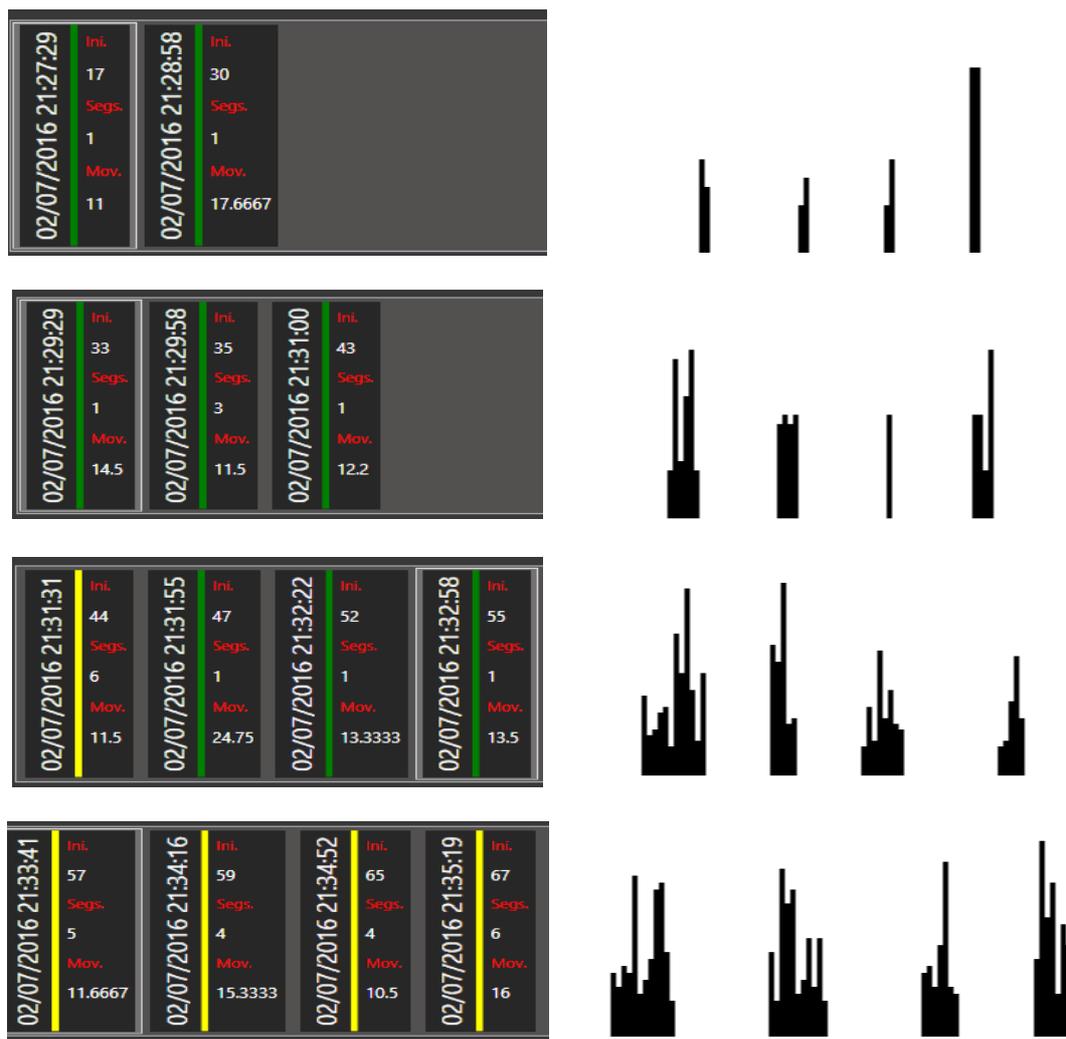


Figura 30: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a três metros de distância com iluminação artificial

Fonte: próprio autor

4.4.4.2 Testes realizados a vinte metros de distância da câmera

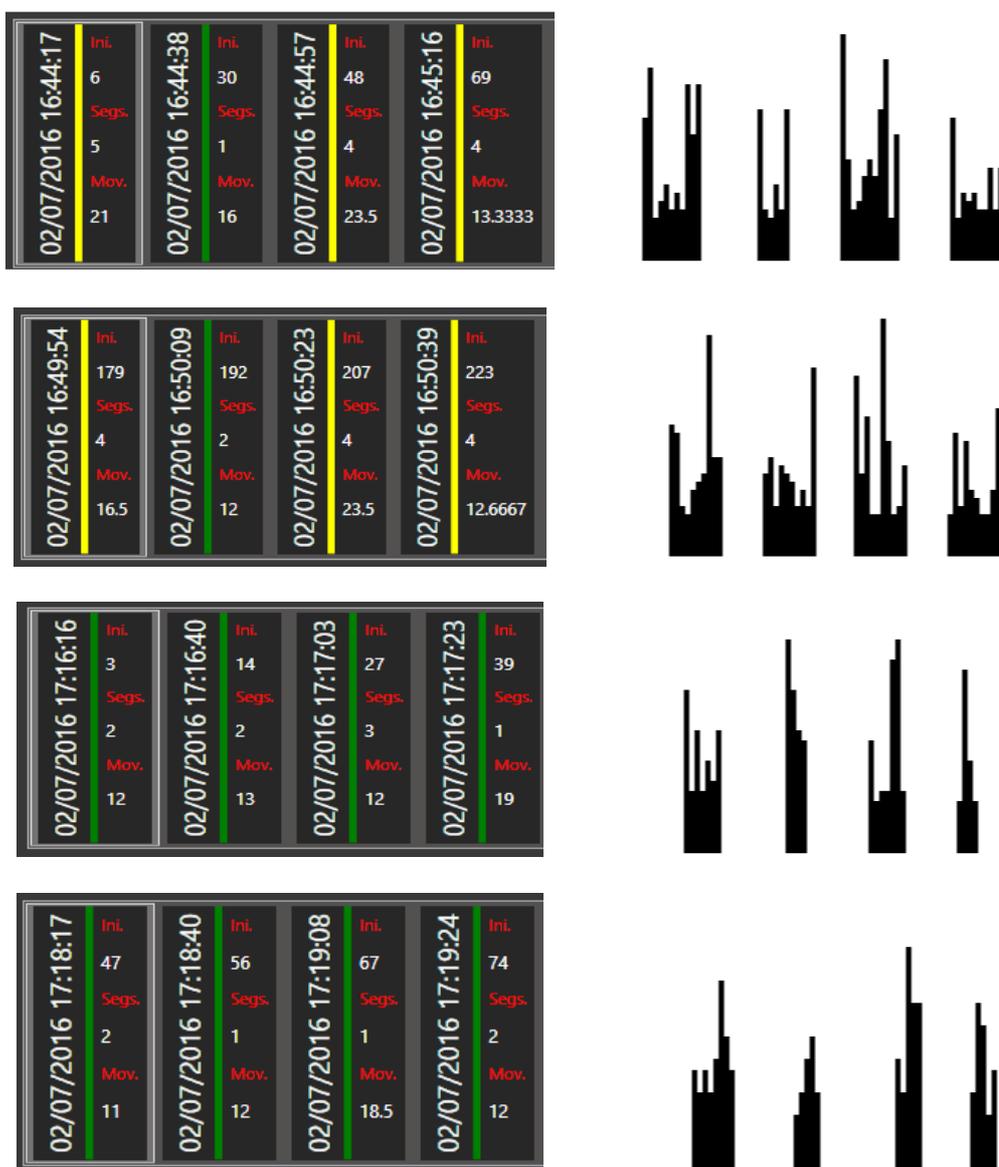
Os testes realizados a vinte metros da câmera visam verificar o comportamento do algoritmo quando os movimentos ocorrem longe da câmera.

Este grupo de testes foram realizados em ambientes externos e com diferentes graus de iluminação.

4.4.4.2.1 Iluminação natural

Os seguintes resultados representam o comportamento do algoritmo na realização de pesquisas em ambientes externos durante o dia com movimentos ocorrendo longe da câmera.

Ao todo 5 (cinco) testes foram realizados e o resultado de cada um é representado pela Figura 31 presente abaixo.



02/07/2016 17:20:17	Ini. 82	02/07/2016 17:20:48	Ini. 101	02/07/2016 17:21:13	Ini. 113	02/07/2016 17:21:33	Ini. 121
	Segs. 3		Segs. 2		Segs. 3		Segs. 1
	Mov. 12.5		Mov. 15.5		Mov. 11		Mov. 12



Figura 31: Checkpoints e histogramas dos resultados de 5 testes realizados em ambiente externo a vinte metros de distância com iluminação natural

Fonte: próprio autor

4.4.4.2 Iluminação artificial

Assim como nos testes anteriores para avaliar o comportamento do algoritmo, novamente foram realizados cinco testes, porém apenas 1 (um) dos cinco testes realizados retornou resultados para a pesquisa em um ambiente externo com pouca iluminação quando os movimentos ocorrem longe da câmera.

02/07/2016 20:54:58	Ini. 1
	Segs. 1
	Mov. 11



Figura 32: Checkpoint e histograma do único teste a gerar resultados em ambiente externo a vinte metros de distância com iluminação artificial

Fonte: próprio autor

Através da interpretação do histograma é possível identificar que devido a baixa luminosidade do ambiente testado a quantidade de movimentos identificados reduz drasticamente.

4.4.5 Desempenho do algoritmo

Ao todo quarenta testes foram realizados em ambientes com diferentes níveis de luminosidade e distância da câmera assim como ambientes internos e externos visando identificar quatro interações com um determinado objeto realizadas em sequência.

O desempenho do algoritmo referentes aos testes executados é representado pelo gráfico presente na Figura 33 informada abaixo.

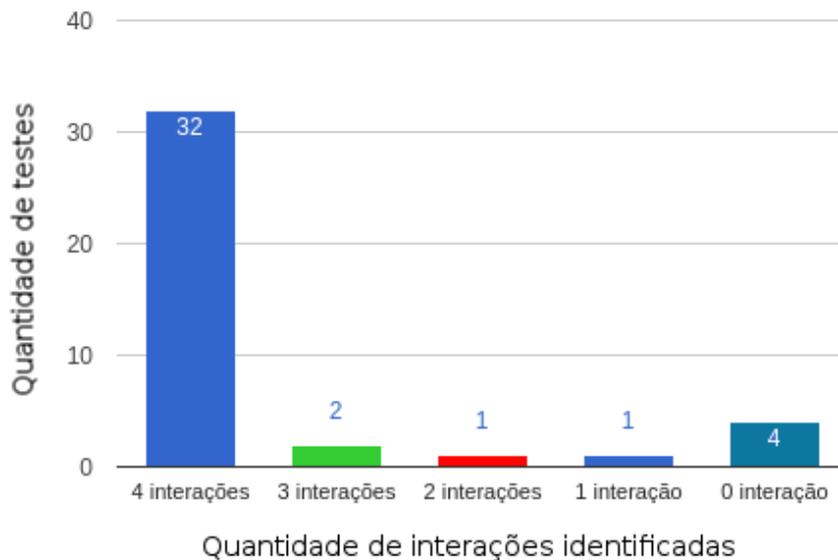


Figura 33: Gráfico comparativo de desempenho relacionando a quantidade de testes realizados e a quantidade de interações que foram identificadas

Fonte: próprio autor

Como ilustrado no gráfico acima, a maioria dos testes foi bem sucedido na recuperação das quatro interações, porém em ambientes com pouca luminosidade o desempenho das buscas foi muito prejudicado.

Isto pode ser observado pelos histogramas dos testes realizados em ambientes externos com pouca luminosidade, a quantidade de movimentos indentificados são menores se comparados com ambientes mais iluminados.

A duração dos movimentos identificados também é reduzida em ambientes com pouca luminosidade assim como o nível de movimento.

5. Trabalhos Futuros

Para futuros estudos e trabalhos relacionados podem ser realizados trabalhos principalmente sobre melhoramento do algoritmo e utilização da pesquisa como base para a execução direcionada de algoritmos.

A continuação do estudo realizado visando a extração de mais dados do vídeo para servir de base para pesquisa poderá tornar a pesquisa mais versátil.

A criação de um modelo de testes com base em algoritmos de processamento imagens baseado na execução direcionada de algoritmos visando reduzir o trabalho computacional executando apenas nos resultados de pesquisas pode demonstrar o comportamento do algoritmo de recuperação de conteúdo em um caso real.

A criação de um sistema distribuído para recuperação e processamento de partes de vídeos seria capaz de demonstrar como o algoritmo se comportaria em ambientes distribuídos.

6. Considerações Finais

O objetivo deste trabalho foi realizar um estudo sobre a criação e as funcionalidades de um mecanismo de busca para vídeo visando tornar dados capturados em vídeos mais acessíveis ao usuário na tomada de decisões.

Essa acessibilidade foi citada por Tian (2008) como sendo algo necessário devido a quantidade de dados que um sistema de vigilância por câmeras é capaz de gerar.

Segundo Collins et al. (2000), manter controle das interações de diversos elementos em um ambiente monitorado por câmeras é complicado devido a quantidade de variáveis que devem ser controladas, neste trabalho foi explorado apenas uma dessas variáveis, a quantidade de movimento, com a intenção de justamente identificar as interações que ocorrem em um ambiente. A extração de mais informações logicamente poderia tornar possível um maior entendimento do que foi capturado.

A extração de mais informações do vídeo poderia tornar a pesquisa mais versátil na diferenciação de situações que ocorreram na mesma região de interesse, por exemplo, através da identificação da cor predominante em um determinado setor poderia ser possível identificar objetos com maior exatidão, seja um carro, ônibus ou uma pessoa usando uma determinada cor de camisa.

7. Conclusão

O algoritmo de pesquisa em vídeo proposto se mostrou possível de realizar a recuperação de pontos específicos do vídeo quando se têm informações básicas como o local de interesse para pesquisa.

O método usado para detectar movimentos se mostrou ineficiente em ambientes com pouca iluminação, a criação de filtros para remover distorções causadas em ambientes escuros devido a iluminação artificial poderia melhorar o desempenho do algoritmo nesses ambientes.

Os resultados obtidos nos testes se mostraram promissores para a criação de um sistema de pesquisa em vídeo para apoiar sistemas de vigilância, a capacidade de adição de funcionalidades que necessitam de detecção de movimentos como a criação de sistemas de alarmes, automação de tarefas, etc, tornam visível a utilidade da realização de pesquisa em vídeo nesses sistemas.

Esse trabalho buscou esclarecer um pouco a área de extração de informações em vídeo que segundo Hampapur (2004) carece de mais pesquisas, com isso não pretendo resolver todos os problemas relativos a esta área e sim apenas mostrar as possibilidades trazidas pela criação de uma modelo pesquisável de vídeo na recuperação de informações de forma menos trabalhosa para o usuário e com isso facilitar a administração de sistemas de vigilância.

Referências

- ALLAIN, Clara. **A era da vigilância**. Disponível em: <<http://observatoriodeseguranca.org/seguranca/cameras>>. Acesso em: 3 jan. 2016.
- BEAULIEU, Alan. **Learning SQL**. 2. ed. California: O'reilly Media, 2009. 338 p.
- BOURKE, Paul. **Colour Spaces**. 1995. Disponível em: <http://paulbourke.net/texture_colour/colourspace/>. Acesso em: 29 ago. 1995.
- COLLINS, Robert T. et al. **A System for Video Surveillance and Monitoring**. 2000. Disponível em: <<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.63.1129&rep=rep1&type=pdf>>. Acesso em: 25 out. 2015.
- DE LIRA VIEIRA, Jessica Monique; CORREA, Renato Fernandes. Visualização da Informação na construção de interfaces amigáveis para Sistemas de Recuperação de Informação. **Encontros Bibli: revista eletrônica de biblioteconomia e ciência da informação**, v. 16, n. 32, p. 73-93, 2011.
- FURNISH, Tim. **Motion Sensors**. 2007. Disponível em: <<http://illumin.usc.edu/165/motion-sensors/>>. Acesso em: 13 fev. 2016.
- GONZALEZ, Rafael C.; WOODS, Richard E. **Digital Image Processing**. 3. ed. United States: Prentice Hall, 2007. 976 p.
- HAMPAPUR, Arun. **S3-R1: The IBM Smart Surveillance System- Release 1**. 2004. Disponível em: <<http://researcher.watson.ibm.com/researcher/files/us-smiyaza/IBMS3-R1Overview.pdf>>. Acesso em: 8 jun. 2016.
- HIRSCH, Robert. **Exploring Color Photography: From Film to Pixels**. 6. ed. New York: Focal Press, 2015. 408 p.
- KHALVATI, Farzad. **Computational Redundancy in Image Processing**. 2008. Disponível em: <https://uwspace.uwaterloo.ca/bitstream/handle/10012/4151/Khalvati_Farzad.pdf?sequence=1>. Acesso em: 11 jan. 2016.
- KNIGHT, Helen. **System improves automated monitoring of security cameras: New approach uses mathematics to reach a compromise between accuracy, speed**. 2012. Disponível em: <<http://news.mit.edu/2012/auto-video-surveillance-algorithm-0605>>. Acesso em: 26 dez. 2015.

- LACASSAGNE, L.; MANZANERA, A.. **MOTION DETECTION: FAST AND ROBUST ALGORITHMS FOR EMBEDDED SYSTEMS**. Disponível em: <<http://perso.ensta-paristech.fr/~manzaner/Publis/icip09.pdf>>. Acesso em: 13 fev. 2016.
- LIDIN, Serge. **.NET IL Assembler**. New York: Apress, 2014. 476 p.
- LUDKE, Menga & ANDRÉ, Marli E.D.A. **Pesquisa em educação: abordagens qualitativas**. São Paulo, Editora Pedagógica e Universitária, 1986. 99p.
- MACDONALD, Matthew. **Pro WPF 4.5 in C#**. 4. ed. New York: Apress, 2012. 1095 p.
- MCANDREW, Alasdair. **An Introduction to Digital Image Processing with Matlab**. Victoria: Victoria University Of Technology, 2004. Disponível em: <http://share.its.ac.id/pluginfile.php/371/mod_resource/content/1/An_Introduction_To_Digital_Image_Processing_With_Matlab.pdf>. Acesso em: 6 jan. 2016.
- MOESLUND, Thomas B.; GRANUM, Erik. **A Survey of Computer Vision-Based Human Motion Capture**. 1999. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S107731420090897X>>. Acesso em: 07 jun. 2016.
- MOLINA, J.m.; GARCIA, J.; PEREZ, O.. **Applying Fuzzy Logic in Video Surveillance Systems**. 2005. Disponível em: <<http://upcommons.upc.edu/bitstream/handle/2099/1831/4-molina.pdf>>. Acesso em: 20 fev. 2016.
- NAGEL, Christian; GLYNN, Jay; SKINNER, Morgan. **Professional C# 5.0 and .NET 4.5.1**. 2. ed. Canada: Wrox, 2014. 1563 p.
- PAULA FILHO, Wilson de Pádua. **Engenharia de Software: fundamentos, métodos e padrões**. 2000. Disponível em: <http://aulasprof.6te.net/Arquivos_Aulas/07-Proces_Desen_Soft/Livro_Eng_Soft_Fund_Met_Padroses.pdf>. Acesso em: 8 jun. 2016.
- QUOOS, João Henrique. **RGB - Cores Aditivas**. Disponível em: <http://coral.ufsm.br/cartografia/index.php?option=com_content&view=article&id=47:rgb-cores-aditivas-simulacao-verde-azul-vermelho&catid=18:material-didatico&Itemid=41>. Acesso em: 18 fev. 2016.
- ROUSE, Margaret. **Frame rate**. Disponível em: <<http://searchnetworking.techtarget.com/definition/frame-rate>>. Acesso em: 13 fev. 2016.

- ROWE, Neil C.; REED, Ahren A.; FLORES, Jose J.. **Detecting Suspicious Motion with Nonimaging Sensors**. Disponível em:
<http://faculty.nps.edu/ncrowe/rowe_bicom10.pdf>. Acesso em: 13 fev. 2016.
- RUSSELL, Wendy. **Video or Movie Frame**. Disponível em:
<<http://presentationsoft.about.com/od/uvw/g/95video-frame-definition.htm>>. Acesso em: 13 fev. 2016.
- SIGNÈS, Julien; FISHER, Yuval; ELEFTHERIADIS, Alexandros. **MPEG-4's binary format for scene description**. 2000. Disponível em:
<<http://www.ee.columbia.edu/ln/mmsp/papers/icj00-js.pdf>>. Acesso em: 29 ago. 2015.
- SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S.. **Database System Concepts**. 6. ed. New York: Mcgraw-hill, 2011. 1376 p.
- SINGLA, Nishu. Motion detection based on frame difference method. **International Journal of Information & Computation Technology**, v. 4, n. 15, p. 1559-1565, 2014.
- TANENBAUM, Andrew S.; BOS, Herbert. **Modern Operating Systems**. 4. ed. New Jersey: Pearson, 2014. 1137 p.
- TEIXEIRA, Thiago; DUBLON, Gershon. **A Survey of Human-Sensing: Methods for Detecting Presence, Count, Location, Track, and Identity**. Disponível em:
<http://www.eng.yale.edu/enalab/publications/human_sensing_enalabWIP.pdf>. Acesso em: 13 fev. 2016.
- TIAN, Ying-li et al. **IBM smart surveillance system (S3): event based video surveillance system with an open and extensible framework**. *Machine Vision and Applications*, v. 19, n. 5-6, p. 315-327, 2008.
- WEBBER, Hugh C.. **Image Processing and Transputers**. Amsterdam: los Press, 1992. 196 p.