

UNIVERSIDADE TUIUTI DO PARANÁ

MATHEUS MENDES DA SILVA SANTOS

**ESTUDO DE TÉCNICAS PARA RASTREAMENTO DE PESSOAS EM
VÍDEO UTILIZANDO RASPBERRY PI**

CURITIBA

2017

MATHEUS MENDES DA SILVA SANTOS

**ESTUDO DE TÉCNICAS PARA RASTREAMENTO DE PESSOAS EM
VÍDEO UTILIZANDO RASPBERRY PI**

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Faculdade de Ciências Exatas e de Tecnologia da Universidade Tuiuti do Paraná, como requisito para obtenção do grau de Bacharel.

Orientador: Prof. MSc. Chauã Coluene Queirolo Barbosa da Silva

CURITIBA

2017

RESUMO

Com o surgimento e a viabilização econômica de sistemas e projetos embarcados, o ensino da computação se popularizou no meio educacional, fazendo o número de pesquisadores e usuários crescer rapidamente. Por isso, a área de miniaturização de computadores e adoção dos sistemas embarcados vem colocando-se no meio profissional de maneira gradativa. O presente trabalho trata dos conceitos de Processamento Digital de Imagens aplicados ao Raspberry Pi, módulo de computação embarcada, e sua câmera. Neste trabalho é apresentado um estudo de diferentes algoritmos para rastreamento de pessoas em vídeo utilizando o Raspberry Pi. Foram analisados os principais algoritmos disponíveis na biblioteca OpenCV para rastreamento: AdaBoost, Multiple Instance Learning (MIL), Median Flow e Tracking Learning Detection (TLD). Para validação dos algoritmos no rastreamento de pessoas, foi criada uma base de dados composta por 25 vídeos, separadas em 3 cenários de diferentes dificuldades: (1) uma pessoa, (2) duas pessoas andando juntas, e (3) três pessoas, duas na mesma direção e outra em direção contrária. Os resultados mostraram que os algoritmos se comportam de maneira consistente no Raspberry Pi, sendo o MIL o mais estável e preciso. Atavés da análise, foi possível determinar o método MIL como o mais eficiente para aplicar em sistemas de vigilância, segurança e rastreamento.

Palavras-chave: Sistemas Embarcados, Raspberry Pi, Processamento Digital de Imagens, Segmentação de Pessoas, Vídeo.

LISTA DE FIGURAS

FIGURA 1 – Aquisição por sensor.	11
FIGURA 2 – Digitalização de uma imagem.	13
FIGURA 3 – Passos fundamentais do Processamento Digital de Imagens.	14
FIGURA 4 – Exemplo de execução do algoritmo Adaboost.	16
FIGURA 5 – Exemplo de execução do algoritmo Multiple Instance Learning.	17
FIGURA 6 – Exemplo de delimitação para o treino do Median Flow.	18
FIGURA 7 – Exemplo de definição da área para o treino do Median Flow.	18
FIGURA 8 – Raspberry Pi Modelo B.	20
FIGURA 9 – Módulo da câmera Raspberry Pi.	21
FIGURA 10 – Fluxo de execução proposto por Shilpashree <i>et al.</i>	22
FIGURA 11 – Visão geral do fluxo proposto por Shilpashree <i>et al.</i>	23
FIGURA 12 – Suavização com ROF.	23
FIGURA 13 – Robô construído por Gaier e Silva para a identificação de padrões.	24
FIGURA 14 – Exemplo de aplicação da técnica SURF.	25
FIGURA 15 – Treino para identificação de padrões.	25
FIGURA 16 – Fluxograma do processamento.	26
FIGURA 17 – Aplicação da segmentação de superfície.	27
FIGURA 18 – Imagem em que a segmentação será aplicada.	27
FIGURA 19 – Segmentação do fundo da cena.	28
FIGURA 20 – Separação da pessoa através das bordas.	28
FIGURA 21 – Resultado da segmentação de uma pessoa em vídeo.	29
FIGURA 22 – Resultado da aplicação do algoritmo <i>k-means</i>	30
FIGURA 23 – Esquematização do classificador Adaboost.	33
FIGURA 24 – Esquematização de características do tipo Haar.	34
FIGURA 25 – Resultado da aplicação dos métodos Adaboost e Haar.	34
FIGURA 26 – Ambiente utilizado para criação da base de dados.	37
FIGURA 27 – Vídeo composto por uma pessoa.	38
FIGURA 28 – Vídeo composto por duas pessoas.	38
FIGURA 29 – Vídeo composto por três pessoas.	39
FIGURA 30 – Acertos do rastreamento aplicado aos vídeos capturados.	44
FIGURA 31 – Erros do rastreamento aplicado aos vídeos capturados.	44

LISTA DE GRÁFICOS

GRÁFICO 1 – Resultados da aplicação do algoritmo AdaBoost nos cenários construídos.	40
GRÁFICO 2 – Resultados da aplicação do algoritmo Multiple Instance Learning nos cenários construídos.	40
GRÁFICO 3 – Resultados da aplicação do algoritmo Median Flow nos cenários construídos.	41
GRÁFICO 4 – Resultados da aplicação do algoritmo Tracking Learning Detection nos cenários construídos.	42
GRÁFICO 5 – Resultados da aplicação do algoritmo Kernelized Correlation Filters nos cenários construídos.	43

LISTA DE QUADROS

QUADRO 1 – Comparativo dos resultados obtidos por Almeida.	31
QUADRO 2 – Comparativo dos trabalhos relacionados.	32
QUADRO 3 – Cenários utilizados na criação das bases.	37
QUADRO 4 – Resultados obtidos para o AdaBoost.	39
QUADRO 5 – Resultados obtidos para o Multiple Instance Learning.	41
QUADRO 6 – Resultados obtidos para o Median Flow.	41
QUADRO 7 – Resultados obtidos para o Tracking-Learning-Detection.	42
QUADRO 8 – Resultados obtidos para o Kernelized Correlation Filters.	42

LISTA DE CÓDIGOS

CÓDIGO 1 – Criação dos classificadores no OpenCV.	36
CÓDIGO 2 – Utilização dos classificadores no OpenCV.	36

LISTA DE ABREVIATURAS E SIGLAS

ARM	<i>Acorn/Advanced RISC Machine</i>
CSI	<i>Camera Serial Interface</i>
IoT	<i>Internet of Things</i>
KCF	<i>Kernelized Correlation Filters</i>
MIL	<i>Multiple Instance Learning</i>
OpenCV	<i>Open Source Computer Vision Library</i>
PDI	Processamento Digital de Imagens
ROF	<i>Rudin Osher Fatemi</i>
SURF	<i>Speeded Up Robust Features</i>
TLD	<i>Tracking Learning Detection</i>

SUMÁRIO

1	INTRODUÇÃO	10
2	FUNDAMENTAÇÃO TEÓRICA	11
2.1	PROCESSAMENTO DIGITAL DE IMAGENS	11
2.1.1	Fundamentos das Imagens Digitais	11
2.1.2	Imagens Bidimensionais	12
2.1.3	Vídeo	13
2.1.4	Passos Fundamentais do Processamento Digital de Imagens	13
2.1.4.1	Aquisição de Imagens	13
2.1.4.2	Pré-Processamento	14
2.1.4.3	Segmentação	14
2.1.4.4	Representação e Descrição	14
2.1.4.5	Reconhecimento e Interpretação	15
2.1.4.6	Base de Conhecimento	15
2.2	DETECÇÃO DE RASTREAMENTO EM VÍDEO	15
2.2.1	AdaBoost	15
2.2.2	Multiple Instance Learning	16
2.2.3	Median Flow	17
2.2.4	Tracking Learning Detection	18
2.2.5	Kernelized Correlation Filters	18
2.3	ARQUITETURA ARM	19
2.3.1	Raspberry Pi	19
2.3.2	Câmera Raspberry Pi	21
3	REVISÃO DA LITERATURA	22
3.1	PROCESSAMENTO DE IMAGENS COM RASPBERRY PI	22
3.2	INTEGRAÇÃO DO RASPBERRY PI, ARDUINO E OPENCV	23
3.3	SISTEMA EMBARCADO PARA CAPTURA DE IMAGENS	25
3.4	SEGMENTAÇÃO DE IMAGENS EM VÍDEO	26
3.5	DETECÇÃO E LOCALIZAÇÃO DE PESSOAS EM VÍDEO	26
3.6	SISTEMA DE CONTAGEM DE PESSOAS BASEADO EM VÍDEO	29
3.7	RASTREAMENTO DE PESSOAS EM VÍDEOS DE FUNDO DINÂMICO	30
3.8	QUADRO COMPARATIVO	32
4	MATERIAIS E MÉTODOS	35
4.1	MATERIAIS UTILIZADOS	35
4.2	METODOLOGIA	36
5	RESULTADOS EXPERIMENTAIS	37
5.1	BASE DE DADOS	37
5.2	EXPERIMENTOS	39

6 CONCLUSÕES	45
REFERÊNCIAS	46

1 INTRODUÇÃO

Diversos projetos de *software* embarcado têm surgido nos últimos anos, motivados principalmente pela evolução dos chips, processadores e aumento da capacidade computacional. Gordon Moore, Químico e co-fundador da *Intel Corporation*, em 1965, observou em seu artigo "*Cramming more components onto integrated circuits*" que cada vez mais haveriam componentes menores e com maior processamento (MOORE, 1998). Dois componentes populares são o Arduino¹ e Raspberry Pi². Essas placas, do tamanho de cartões de crédito, possuem o mesmo poder computacional de estações de trabalho convencionais, com um custo muito menor.

O estudo de Processamento Digital de Imagens (PDI) tem sido empregado em campos da área industrial, saúde, segurança e robótica. Esse último, mais recentemente tem ganhado maior vínculo com a área de PDI, pois tem buscado a automação e a execução de tarefas mais complexas com o auxílio de sensores.

O objetivo deste trabalho é analisar a viabilidade do uso do Raspberry Pi em sistemas de segurança para o rastreamento de pessoas, utilizando técnicas do Processamento Digital de Imagens. Para isso, foram estudados diferentes algoritmos fornecidos pela biblioteca OpenCV. Os algoritmos estudados foram: (1) AdaBoost, (2) Multiple Instance Learning (MIL), (3) Median Flow, (4) Tracking Learning Detection (TLD) e (5) Kernelized Correlation Filters (KCF).

O trabalho está dividido como segue. O Capítulo 2 apresenta a fundamentação teórica abordando os conceitos de processamento embarcado no Raspberry Pi, conceitos de PDI, e estratégias para segmentação de pessoas em vídeo. O Capítulo 3 apresenta uma análise dos trabalhos relacionados ao problema de segmentação de pessoas em vídeo. O Capítulo 4 apresenta a metodologia aplicada no desenvolvimento deste trabalho, seguido dos resultados experimentais apresentados no Capítulo 5. Finalmente, o Capítulo 6 apresenta as conclusões.

¹ <<https://www.arduino.cc/>>

² <<https://www.raspberrypi.org/>>

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os fundamentos necessários para a implementação deste projeto. Inicialmente são apresentados os conceitos relacionados ao Processamento Digital de Imagens. Em seguida são apresentados detalhes sobre o Raspberry Pi e, após, são descritas as características do problema de segmentação de pessoas em vídeo.

2.1 PROCESSAMENTO DIGITAL DE IMAGENS

O conceito de processamento de imagens refere-se ao conjunto de métodos utilizados em imagens adquiridas, que devem ser corrigidas e melhor definidas, isso para que a interpretação visual se torne mais nítida e pontos perdidos na aquisição, e também na transmissão, possam ser restaurados com a finalidade de se obter a qualidade mais próxima possível da realidade (WOODS, 2006).

2.1.1 Fundamentos das Imagens Digitais

Uma imagem pode ser definida como uma função de intensidade de luz sobre determinado ponto, tratada comumente de maneira bidimensional. Explorando de maneira minuciosa a definição de imagem, tem-se que para a obtenção de uma imagem, em um plano de coordenadas espaciais dado por (x, y) , deve-se estabelecer um ponto de observação, também denominado ponto focal, conforme mostra a Figura 1, medir os graus de incidência e reflexão da luz gerada e presente no plano observável em consideração ao ponto de observação e sua redondeza, denominada cena (WOODS, 2006).

FIGURA 1 – Aquisição por sensor.



FONTE: Garrot e Felgueiras (2008, p. 24).

A incidência ou iluminação de pontos na cena é obtida partindo da fonte de luz para os elementos presentes no plano. A resultante varia de acordo com a exposição e a luminosidade do instante no qual a imagem é capturada. Já a reflexão, ou refletância do ponto de observação, corresponde ao grau de absorção que determinado objeto tem da quantidade de luz emitida no ambiente. Tipicamente, o valor é definido através

de um intervalo entre 0 e 1, onde 1 é a reflexão total do comprimento de onda referente à luz incidida sobre o objeto (WOODS, 2006).

Em resumo, uma imagem é o conjunto de pontos convergentes que constrói um todo. Para o estudo e processamento, imagem é o conjunto de informações coletadas para a representação mais nítida de um ponto de observação. Os pontos focais, no mundo real, são definidos como tridimensionais, por serem traduzidos em três coordenadas, x , y , z , respectivamente altura, largura e profundidade. Quando há captura através de um sistema, seja com câmeras ou sensores, e transcrição para uma imagem, existe também perda da informação espacial, correspondente à coordenada z . Logo, a profundidade da cena na qual o objeto, ponto focal, está é omitida (WOODS, 2006).

2.1.2 Imagens Bidimensionais

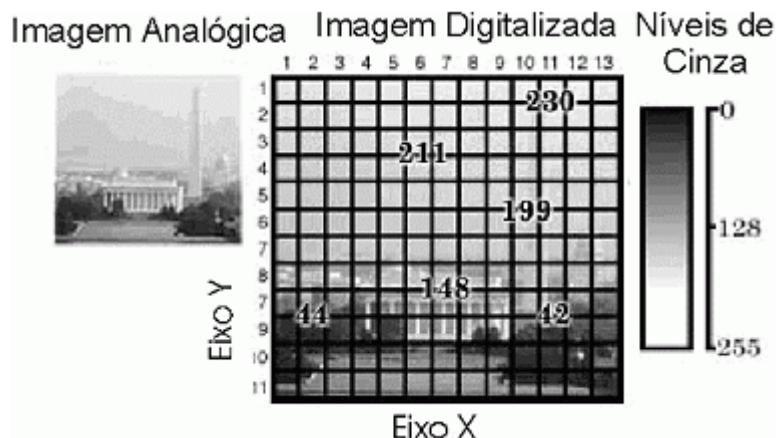
O conceito de imagem bidimensional, corriqueiramente chamada de 2D, refere-se à imagem obtida puramente através de câmeras, onde ocorre a perda da coordenada de profundidade do espaço-objeto. A imagem digital é constituída através de uma função, $f(x, y)$, descrita na Equação 2.1, como coordenadas espaciais transcritas por meio de uma matriz, onde índices referem-se à posição de um pixel na imagem.

$$f(x, y) = \begin{bmatrix} f(0, 0) & f(0, 1) & \cdots & f(0, n - 1) \\ f(1, 0) & f(1, 1) & \cdots & f(1, n - 1) \\ \vdots & \vdots & \ddots & \vdots \\ f(m - 1, 0) & f(m - 1, 1) & \cdots & f(m - 1, n - 1) \end{bmatrix} \quad (2.1)$$

A cor ou intensidade de uma imagem é obtida através do valor de cada tupla existente na matriz em escala de cinza (medida que varia de 0 a 255), como apresentado na Figura 2. Para a aquisição de uma imagem bidimensional é necessário um equipamento capaz de identificar níveis de ondas eletromagnéticas (raio-X, infravermelho, ultravioleta, entre outros) e produzir um sinal elétrico correspondente para que um segundo equipamento possa transcrever o sinal físico para o meio digital.

Os sinais identificados e transcritos passam pela aplicação de modelos e técnicas de processamento para eliminar ou sobrepor o eixo z , eixo da profundidade. Chamada de transformação de perspectiva, ou imageamento, essa é uma técnica que busca a aproximação da imagem identificada tridimensionalmente pelo olho. Após a transcrição do sinal recebido, é necessário armazenar a matriz para que, posteriormente, aplique-se à mesma o processamento.

FIGURA 2 – Digitalização de uma imagem.



FONTE: Garrot e Felgueiras (2008, p. 39).

O armazenamento pode ocorrer por curto prazo, no qual as ondas são armazenadas em *buffers* para que, dependendo do dispositivo, possa ocorrer nele próprio, por exemplo, um processamento inicial, sendo apenas a tradução para pixels ou até a montagem de uma imagem digital. Outro tipo de armazenamento típico é o massivo, onde os dados coletados são retidos para que, posteriormente, sejam analisados por programas para construir as imagens (WOODS, 2006).

2.1.3 Vídeo

Vídeo digital é o resultado de uma sequência de quadros (*frames*) de imagens digitais, representando uma determinada cena. A taxa de atualização de cada quadro se dá pela razão de quadros de imagens digitais por tempo contínuo, contado em segundos, e sua exibição é dada progressivamente. O armazenamento de vídeos leva em consideração a qualidade e o tamanho. Para isso, vários formatos (*codecs*) foram criados e são utilizados conforme a necessidade de recuperação das informações do vídeo e o suporte do hardware onde será reproduzido (OLIVEIRA, 2013, p.6).

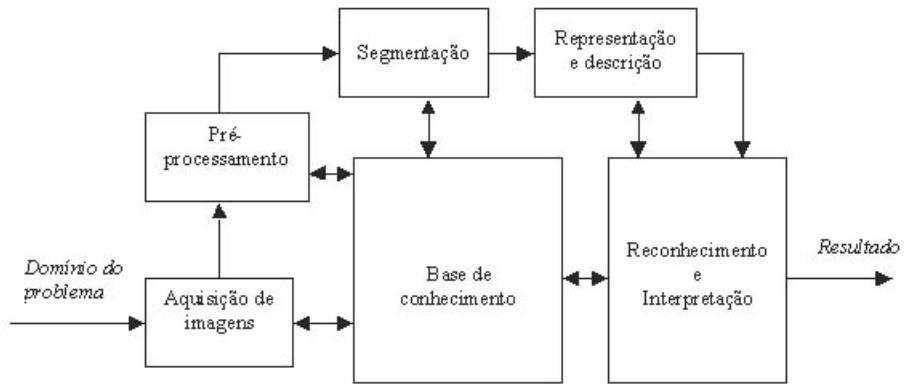
2.1.4 Passos Fundamentais do Processamento Digital de Imagens

Nesta seção são apresentados e definidos os passos fundamentais do PDI, mostrados na Figura 3.

2.1.4.1 Aquisição de Imagens

A primeira etapa do PDI é definida por traduzir os aspectos visíveis do mundo real por meio de sensores capazes de capturar uma banda eletromagnética e traduzir em sinais elétricos perceptíveis, bem como converter tais sinais em elementos digitais.

FIGURA 3 – Passos fundamentais do Processamento Digital de Imagens.



FONTE: Woods (2006).

Como exemplos de sensores tem-se satélites, aparelhos de ressonância magnética e câmeras fotográficas (WOODS, 2006, p.7–10).

2.1.4.2 Pré-Pocessamento

A etapa de pré-processamento é responsável pela melhoria da imagem para aumento da qualidade, de modo algorítmico, para os processos seguintes. O pré-processamento consiste na aplicação das técnicas de melhoria de contraste, realce de características e correção de defeitos de captura, como por exemplo, ruídos e foco (WOODS, 2006, p.6).

2.1.4.3 Segmentação

Segundo Woods (2006, p.6), a etapa de segmentação é responsável por separar a imagem em partes, regiões ou objetos de interesse e destacá-los em primeiro plano. Para a segmentação a imagem é tratada através dos valores reais dos *pixels* existentes, ou da sua presença ou ausência (modo binário). Tal processo baseia-se em três propriedades de uma imagem(SOLOMON; BRECKON, 2000, p.236-237):

1. Cor: Diferença do espaço de cores para definir uma região;
2. Textura: Diferenciação de intensidade espacial em uma imagem;
3. Movimento: Subtração de quadros de imagens em fundo estacionário que pode definir com precisão a movimentação de um objeto.

2.1.4.4 Representação e Descrição

Conforme Woods (2006, p.6), a representação é a etapa de definição e separação adequada dos objetos segmentados conforme a necessidade da análise posterior,

reconhecimento e interpretação. Já a descrição é a etapa quantitativa das características dos objetos segmentados e representados, com a finalidade de resultar em informações de interesse ou classificação.

2.1.4.5 Reconhecimento e Interpretação

A etapa de reconhecimento e interpretação, segundo Woods (2006, p.7–10), é a rotulação baseada na descrição de cada objeto presente na imagem e a atribuição de um significado, conforme a base de conhecimento, aos resultados adquiridos.

2.1.4.6 Base de Conhecimento

Dado como o objeto ou o conjunto de imagens a ser estudado pelo PDI, a base de conhecimento é responsável por guiar cada etapa e definir a interação entre as mesmas (WOODS, 2006, p.6). Para que os objetos de processamento sejam acessados, o armazenamento é realizado em três categorias:

1. De curto tempo: Armazenamento da imagem enquanto processada, em memória computacional (*frame buffers*), onde a velocidade de acesso é alta, isto é, cerca de 30 imagens por segundo;
2. Em massa ou *on-line*: Armazenamento em unidades com capacidade para, no mínimo, algumas centenas de imagens contabilizadas em MB. Para que as informações estejam disponíveis e possam ser acessadas de maneira rápida, essas são comprimidas juntamente com suas informações (tamanho, número de cores, entre outras) em diferentes formatos;
3. Arquivamento: Armazenamento em massa sem a necessidade corrente de acesso, onde agrupamentos de imagens são estocados.

2.2 DETECÇÃO DE RASTREAMENTO EM VÍDEO

Para detecção de movimento em vídeo, diversas técnicas podem ser empregadas. Dentre elas pode-se destacar: (1) AdaBoost, (2) Multiple Instance Learning (MIL), (3) Median Flow, (4) TLD, (5) KCF.

2.2.1 AdaBoost

O AdaBoost (GRABNER; BISCHOF, 2006) é um algoritmo de aprendizado de máquina. Assim como outros algoritmos dessa natureza, por exemplo, as Redes Neurais Artificiais, o AdaBoost possui duas etapas: treinamento e teste. Na etapa de treinamento o algoritmo é alimentando com diversos exemplos de imagens e qual

classe aquela imagem pertence. Dessa maneira, várias imagens com movimento são usadas para ensinar o algoritmo a diferenciar onde ocorre ou não movimento, como na Figura 4. Nessa etapa também é gerado um classificador. O classificador é então utilizado na etapa de teste, onde o algoritmo recebe uma imagem e gera como saída onde ocorreu o movimento, com o menor índice de erro, conforme equação 2.2.

$$h^{strong}(X) = \sum_{n=1}^n \alpha_n - h_n^{sel}(x) \quad (2.2)$$

Desta maneira, o sucesso do algoritmo está diretamente ligada à base de treino utilizada. Quanto maior a base, maior é o tempo de processamento e melhor é o classificador produzido, como descrito na equação 2.3. Neste trabalho, será avaliada a implementação incorporada na biblioteca OpenCV, não sendo necessário realizar o treinamento do classificador.

$$e_{n,m} = \frac{\lambda_{n,m}^{wrong}}{\lambda_{n,m}^{corr} + \lambda_{n,m}^{wrong}} \quad (2.3)$$

FIGURA 4 – Exemplo de execução do algoritmo Adaboost.



FONTE: Lu et al. (2009).

2.2.2 Multiple Instance Learning

O Multiple Instance Learning (MIL) (BABENKO; YANG; BELONGIE, 2009) é um algoritmo de aprendizado de máquina para treinamento supervisionado. Desta maneira, os dados não precisam ser classificados para etapa de treino, sendo as

classes identificadas pela próprio algoritmo. Tais classes são rotuladas, conforme equação 2.4

$$y_i = \max_j(y_{ij}) \quad (2.4)$$

Na medida em que são rotuladas e computadas as classes, o algoritmo compara a fim de maximizar a probabilidade delas serem o mesmo conjunto da imagem, segundo a equação 2.5, demonstrada na Figura 5.

$$\log L = \sum_i (\log p(y_i | X_i)) \quad (2.5)$$

No contexto de detecção de movimento, o classificador é alimentado com diversas imagens, e assim ele consegue identificar onde há movimento.

FIGURA 5 – Exemplo de execução do algoritmo Multiple Instance Learning.



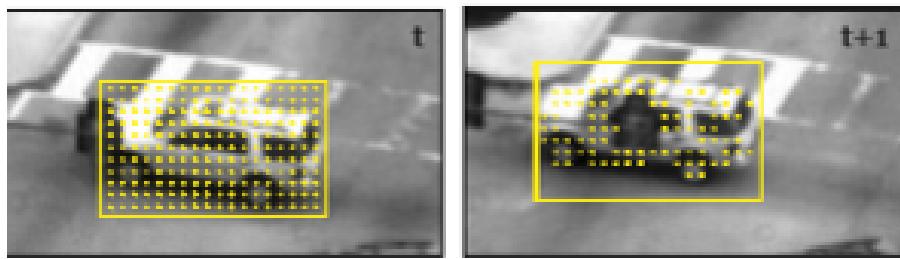
FONTE: Babenko, Yang e Belongie (2009).

2.2.3 Median Flow

O Median Flow (KALAL; MIKOŁAJCZYK; MATAS, 2010) é um algoritmo de rastreamento baseado no *Backward Error*. O algoritmo realiza análises sucessivas na trajetória de movimento entre várias imagens no decorrer do tempo. O conjunto de referência para a análise é dado sempre por uma região delimitada em par subsequentes de *frames*, conforme exemplificado na Figura 6.

Em seguida, o deslocamento desse conjunto é estimado dentro da área esparsa do vídeo, como na Figura 7 . Para que o algoritmo identifique com maior precisão o

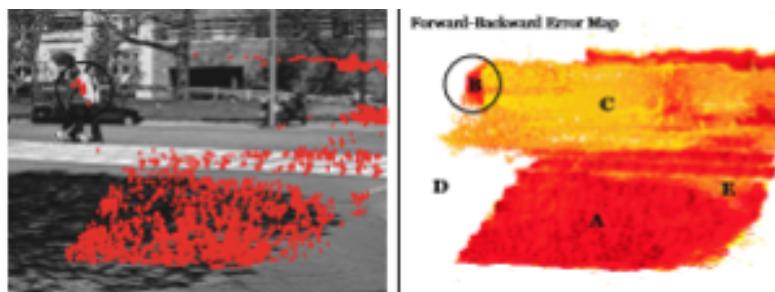
FIGURA 6 – Exemplo de delimitação para o treino do Median Flow.



FONTE: Kalal, Mikolajczyk e Matas (2010).

deslocamento do movimento, mesmo em casos de oclusão do objeto, é calculada a razão entre os deslocamentos estimados e a mediana dessa razão.

FIGURA 7 – Exemplo de definição da área para o treino do Median Flow.



FONTE: Kalal, Mikolajczyk e Matas (2010).

2.2.4 Tracking Learning Detection

Definido por (KALAL; MIKOŁAJCZYK; MATAS, 2012), Tracking-Learning-Detection (TLD) é um *framework* composto pelas etapas de rastreamento, detecção e aprendizado. Esses três componentes tratam, respectivamente, da estimativa de movimentação de um objeto visível em *frames* consecutivos, do tratamento independente e busca completa em cada *frame* e da estimativa de erro gerada pelos fatores falso positivo e falso negativo das etapas anteriores.

2.2.5 Kernelized Correlation Filters

Kernelized Correlation Filters, KCF, é um algoritmo de rastreamento apresentado por Henriques et al. (2015) o qual trata do rastreamento em vídeo. O algoritmo traduz em dado analítico o vídeo, a saber uma matriz de valores para *pixel*, correlacionando cada *frame*. A correlação é tratada a partir da minimização de diferença entre pontos, conforme a equação 2.6

$$\min_w \sum_i (f(x_i) - y_i)^2 + \lambda |w|^2 \quad (2.6)$$

Após a correlação, é possível tratar como valores binários de cada *pixel*, assim minimizando o espaço de armazenamento e execução quando tata-se de uma grande quantidade de informações. Também foi verificado por Henriques et al. (2015) que a matriz resultante da correlação de máscara será uma matriz circulante, ou seja, uma matriz onde os valores de cada linha se repetem, apenas deslocando-se por i posições, dadas pela correlação realizada, conforme a equação 2.7

$$C(x) = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ x_n & x_1 & x_2 & \cdots & x_{n-1} \\ x_{n-1} & x_n & x_1 & \cdots & x_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_2 & x_3 & x_4 & \cdots & x_1 \end{bmatrix} \quad (2.7)$$

2.3 ARQUITETURA ARM

A arquitetura ARM (*Acorn/Advanced RISC Machine*) é um modelo computacional com conjunto reduzido de instruções (*Reduced Instruction Set Computer*), desenvolvido pela companhia britânica *Acorn Computers (Cambridge)* para funcionalidades integradas de alto desempenho para aplicações embarcadas e para mercados emergentes, reduzindo a quantidade e o tamanho dos transistores, o que reduz custo e consumo de energia (GRISENTHWAITE, 2011).

A implementação da microarquitetura ARM é dada em *System-on-a-Chip (SoC)* ou *Single-Board Computer*, que compreende em uma única placa micro-controladores (microprocessadores integrados a circuitos e memória), GPU e co-processadores (ARM Processor Architecture). A exemplo de um *Single-Board-Computer*, tem-se: BeagleBone, Raspberry Pi, HiKey 960 e Tinker Board.

2.3.1 Raspberry Pi

O *Raspberry Pi* é um computador com dimensões de um cartão de crédito (85,6mm x 56,5mm), com 45g, anunciado em 2011 (ver Figura 8) sendo seu primeiro modelo lançado em fevereiro de 2012. Com a finalidade de ser o computador mais barato em circulação, estimular e expandir o ensino de computação e programação, foi idealizado por Pete Lomas no Reino Unido, tendo seu valor em torno de 35 dólares e código aberto (FOUNDATION, 2017c).

O computador é baseado na arquitetura ARM, modelo computacional com conjunto reduzido de instruções criado para funcionalidades integradas de alto desempenho

FIGURA 8 – Raspberry Pi Modelo B.



FONTE: O próprio autor, 2017.

para aplicações embarcadas e para mercados emergentes.

Tal arquitetura possibilita a criação de técnicas de micro-arquitetura tendo o foco no tamanho da implementação, desempenho e baixo consumo de energia. Em sua placa, o *Raspberry Pi* apresenta um chip *Broadcom BCM2835* (modelo A/Zero) de 32 bits, *BCM2835* (modelo A) ou *BCM2836/BCM2837* (modelo B) de 32/64 bits, com processador *ARM1176JZF-S*, *ARM Cortex-A7* e *ARM Cortex-A53*, respectivamente, com velocidades de 700 MHz a 1.2 GHz. Sua GPU é a *Videocore IV* de 250 MHz, com capacidade de reprodução em alta definição (1080p).

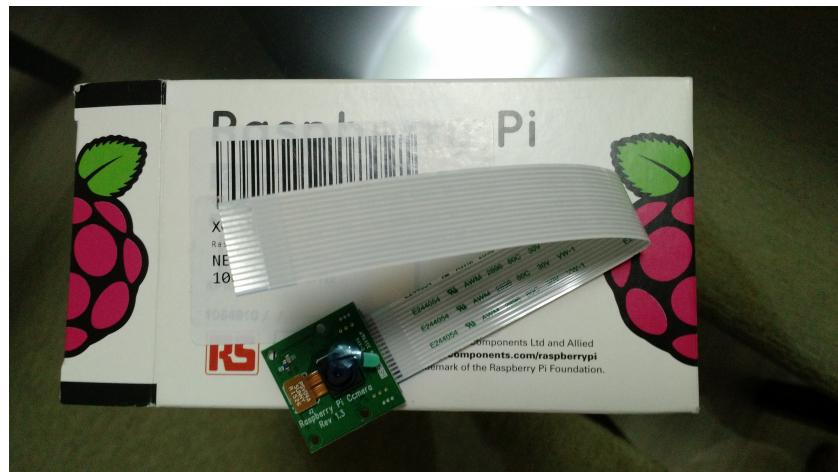
A memória (SDRAM) é de 256 MB, 512 MB e 1 GB, para os modelos A, Zero e B, respectivamente. Também compõem a placa portas USB 2.0, interface de entrada de vídeo CSI (*Camera Serial Interface*) e saída de vídeo digital HDMI e analógico através do componente GPIO (*General-Purpose Input/Output*), o qual também possibilita a conexão de periféricos. A fonte de energia, via USB, é de 5 V.

Seu armazenamento se dá através de cartões de memória SD (modelo A/B), MicroSDHC (modelo B) e MMC (modelo Zero) com capacidade a ser definida pelo utilizador, onde serão instalados o sistema operacional e demais aplicações. Os sistemas operacionais adotados para a *Raspberry Pi* variam conforme a sua finalidade. Em sua grande maioria são sistemas baseados em *Linux* e otimizados para o *Raspberry*, como o popular *Raspbian* e o *PiDora*. Recentemente, com o avanço da Internet das Coisas (*IoT*), foi possível incluir o *Windows 10* como sistema operacional base (FOUNDATION, 2017b).

2.3.2 Câmera Raspberry Pi

Módulo adicional ao *Raspberry*, a *Raspberry Pi Câmera* (ver Figura 9) é ligada ao socket CSI (*Camera Serial Interface*), interface projetada especificamente para a conexão com a câmera. O primeiro modelo vem com um sensor *OmniVision OV5647* de 5 *Mega Pixels* para imagens e suporte a vídeos em 1080p 30fps, 720p 60fps e VGA90. O segundo modelo, v2, traz um sensor Sony IMX219 de 8 *Mega Pixels*, com maior desempenho em baixa-luminosidade e fidelidade a cores. A conexão com a porta CSI se dá através de um cabo do tipo *ribbon* invertido, com 15 vias. (FOUNDATION, 2017a)

FIGURA 9 – Módulo da câmera Raspberry Pi.



FONTE: O próprio autor, 2017.

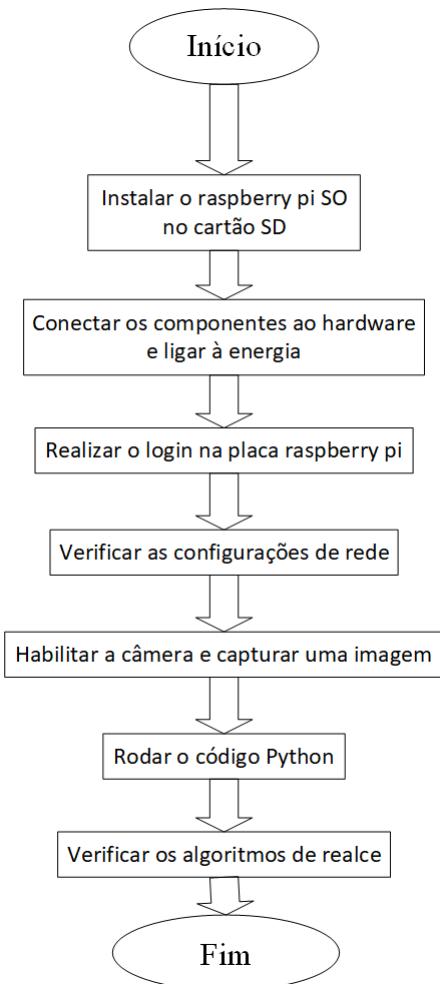
3 REVISÃO DA LITERATURA

Este capítulo apresenta uma revisão dos trabalhos relacionados a detecção de pessoas em vídeo e Processamento Digital de Imagens, por meio de diferentes técnicas, aplicando-se também ao Raspberry Pi.

3.1 PROCESSAMENTO DE IMAGENS COM RASPBERRY PI

Shilpashree, Lokesh e Shivkumar (2015) apresentam o processamento de imagens através do computador *Raspberry Pi*, com seus componentes, como, por exemplo, a câmera *Pi*. Os autores apresentam a implementação de metodologias e algoritmos com seu fluxo de ação para a captação de imagens sem ruídos. O conceito de processamento de imagens é explanado de maneira sucinta, como a captura por câmera, processamento através de um sistema e amostragem da imagem processada, conforme fluxos representados nas Figuras 6 e 7.

FIGURA 10 – Fluxo de execução proposto por Shilpashree *et al.*



FONTE: Shilpashree, Lokesh e Shivkumar (2015).

FIGURA 11 – Visão geral do fluxo proposto por Shilpashree *et al.*



FONTE: Shilpashree, Lokesha e Shivkumar (2015).

O artigo apresenta o dispositivo Raspberry Pi, uma placa com *chip Broadcom 700MHz*, com CPU ARM 32 bits, rodando um sistema operacional Linux a partir de um cartão SD. Os autores utilizaram um módulo de câmera de 15 pinos, com resolução de 5MP, o que traz como resultado uma imagem de até 1920x1080 pixels e um vídeo com captura de aproximadamente 30 quadros por segundo. O módulo utiliza conectores *Camera Serial Interface* (CSI) para serem inseridos diretamente na interface desenvolvida para câmeras do dispositivo, sendo sua lente capaz de entregar um vídeo de alta qualidade.

A metodologia aplicada traz a instalação do sistema operacional a ser utilizado na placa, bem como a inserção do módulo de câmera. Os algoritmos foram desenvolvidos usando a linguagem de programação *Python*. O algoritmo utilizado para demonstrar o processamento de imagem realizado na placa Raspberry Pi é *Rudin-Osher-Fatemi* (ROF). Esse algoritmo aplica uma suavização na imagem, retirando os ruídos da imagem capturada, minimizando as diferenças para a imagem desejada e preservando as bordas e estruturas reais. Como resultado da aplicação, o algoritmo apresentou imagens mais nítidas, tendo sido executado com sucesso no dispositivo, como mostra a Figura 12.

FIGURA 12 – Suavização com ROF.

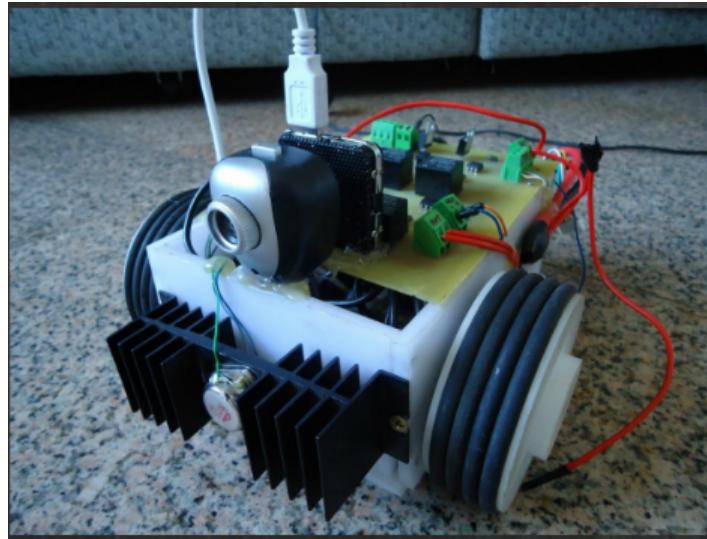


FONTE: Shilpashree, Lokesha e Shivkumar (2015).

3.2 INTEGRAÇÃO DO RASPBERRY PI, ARDUINO E OPENCV

Gaier e Silva (2013) apresentam o processamento de imagens por meio da biblioteca OpenCV implementada no computador *Raspberry Pi*, atrelado a um robô construído a partir dos componentes da plataforma Arduino (ver Figura 13) com o propósito da identificação de padrões de maneira inteligente.

FIGURA 13 – Robô construído por Gaier e Silva para a identificação de padrões.



FONTE: Gaier e Silva (2013).

Para a identificação foi utilizada a biblioteca OpenCV, que tem como seus diferenciais ser multiplataforma e focada em ambientes móveis, por ser rápida. O controle de movimentos e resposta do robô ao ambiente foram configurados dentro do Arduino Uno, plataforma específica para elaboração de projetos eletrônicos, com maior popularidade na robótica, onde sua configuração e instalação é fácil e prática, além das respostas serem ágeis, pela maneira como é instalada, diretamente nos circuitos do robô.

As técnicas utilizadas para a identificação de padrões incluem a detecção de bordas por meio do algoritmo de *Canny*, que estabelece como premissas a não perda de nenhuma borda presente na imagem, a maior precisão nas bordas com relação ao mundo real e a completa eliminação de ruído, isso para que uma borda seja estabelecida apenas uma vez; e também o *Speeded Up Robust Features* (SURF), técnica que traduz a imagem a um conjunto de coordenadas cartesianas para que seja criado um padrão de escala (ver Figura 14) a ser aplicado em outro cenário onde deverá ser marcado e exibido um objeto correspondente. Ambas as técnicas aplicam transformações gaussianas e escalas de cinza para obter maior precisão sem perda de informação.

A implementação foi testada no robô montado, passando por cada ponto de tratamento e passo dos algoritmos. Com o robô treinado a partir de uma imagem (Figura 15), obteve-se êxito na busca pelo contorno dentro de um ambiente com certo atraso na visualização das respostas, porém não afetando no seu resultado, e com comunicação estável entre as diferentes plataformas.

FIGURA 14 – Exemplo de aplicação da técnica SURF.



FONTE: Gaier e Silva (2013).

FIGURA 15 – Treino para identificação de padrões.



FONTE: Gaier e Silva (2013).

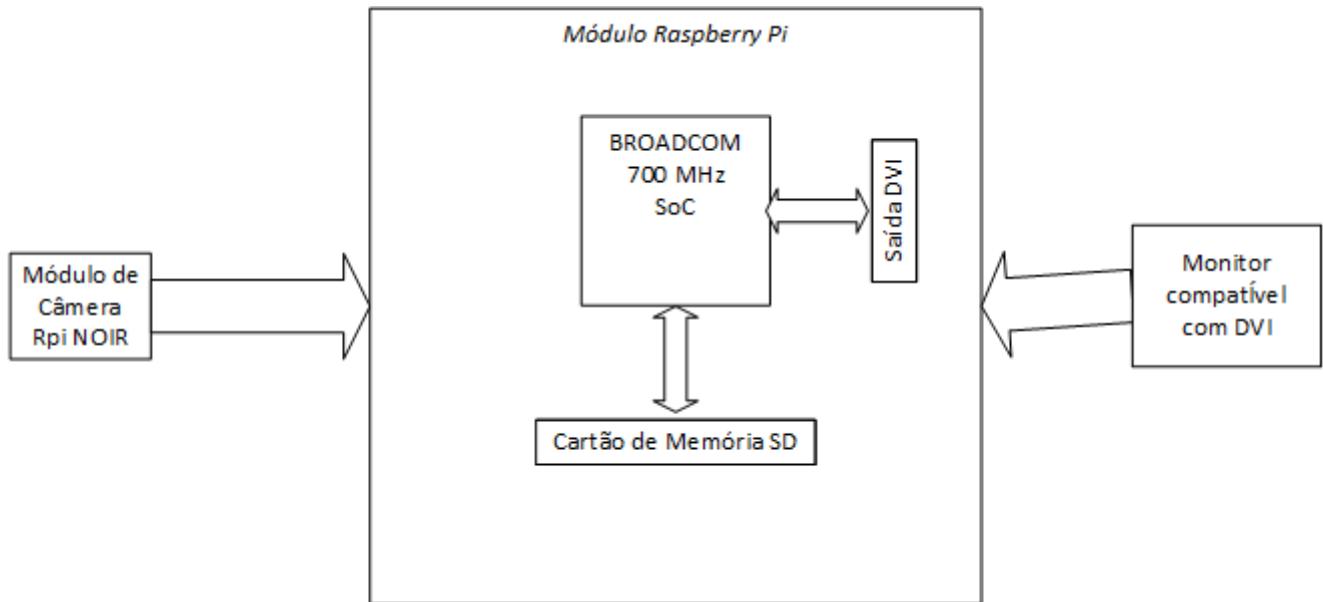
3.3 SISTEMA EMBARCADO PARA CAPTURA DE IMAGENS

Senthilkumar, Gopalakrishnan e Kumar (2014) apresentam as técnicas de obtenção de imagens a partir do computador Raspberry Pi e as possíveis aplicações do abordado, como câmeras em automóveis e elevadores inteligentes. O artigo trata com detalhes cada componente do computador e como esses podem ser aproveitados para a obtenção de imagens a partir da câmera projetada especificamente para o computador *Raspberry Pi Camera Board*.

Para a aplicação é especificado o sistema embarcado utilizando uma placa Raspberry Pi, um módulo de câmera MIPI CSI e monitores. O método utilizado para exemplificar o uso de sistemas embarcados é a aquisição de imagens pela câmera e o seu armazenamento em um dispositivo, segundo fluxograma apresentado na Figura 16.

Após a aquisição das informações, cada imagem é analisada conforme a metodologia de reconhecimento facial *Eigenfaces*. Tal método define a região de uma face a partir de imagens previamente adquiridas, classifica como reconhecida ou desconhecida a imagem atualmente adquirida, se é ou não é uma face e estabelece relações entre as informações atuais e a base. Os autores concluíram que o sistema proposto é menor e mais ágil do que as aplicações de PC. Também seus resultados foram satisfatórios para o ambiente imposto.

FIGURA 16 – Fluxograma do processamento.



FONTE: Senthilkumar, Gopalakrishnan e Kumar (2014).

3.4 SEGMENTAÇÃO DE IMAGENS EM VÍDEO

Li et al. (2013) propõem uma segmentação de múltiplas imagens abordada por um algoritmo que trata a superfície adquirida da segmentação inicial dos quadros do vídeo. Tal segmentação de vídeo é realizada a partir da sobreposição de faixas do vídeo que é reconstruída a partir de algoritmos não supervisionados.

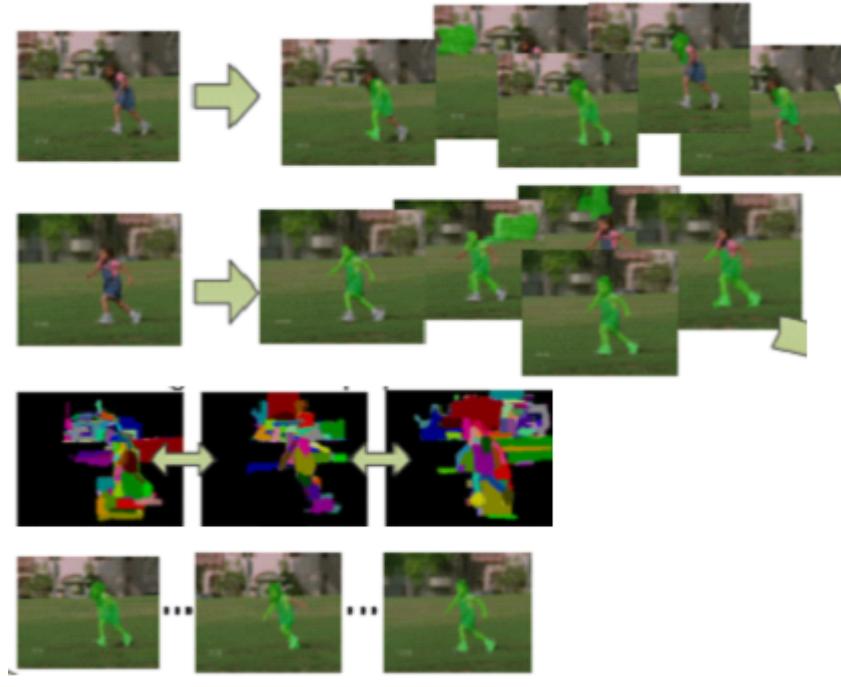
O artigo propõe resolver o problema de segmentação de vídeo não normalizada, separando todos os quadros simultaneamente para gerar uma única imagem disposta quadro-a-quadro. Consequentemente, gerar um modelo global de trilha para buscar e validar em cada quadro se o padrão é mantido, atualizando assim sempre os modelos dentro da base. A aplicação do método construído é exemplificada na Figura 17.

Para cada quadro foi utilizado um algoritmo de separação por múltiplas imagens e estabelecida a configuração de aparência. Logo após a diferenciação dos quadros, foi aplicada simultaneamente a aprendizagem dos modelos para todos os quadros da imagem, uma abordagem gulosa para os seguintes quadros, repetindo o algoritmo. Neste artigo foi demonstrado que a múltipla secção para segmentação é possível e sua adoção melhora a interpretação de sequências de vídeos.

3.5 DETECÇÃO E LOCALIZAÇÃO DE PESSOAS EM VÍDEO

Jabri et al. (2000) apresentam um método de extração e segmentação de pessoas baseado na coloração predominante e na área de borda, focando na área mais recentemente modificada. Tendo como premissa que a parte a ser segmentada

FIGURA 17 – Aplicação da segmentação de superfície.



FONTE: Li et al. (2013).

será uma pessoa e que, em um vídeo, ela será a parte mais recente a ser inserida na imagem, ver Figura 18, o método analisa o contorno e as bordas das regiões de acordo com a frequência e canais de cores definidas na imagem.

FIGURA 18 – Imagem em que a segmentação será aplicada.



FONTE: Jabri et al. (2000).

O método é dividido em três partes para segmentar a região onde estará a pessoa a ser identificada:

1. Separar e manter o fundo da imagem;
2. Subtrair da imagem original o fundo;
3. Selecionar a parte frontal da imagem.

A separação é realizada através da medida de peso de cada pixel em duas imagens subsequentes, onde cada vez mais ao fundo, mais leve e estático o pixel será. A resultante do método é apresentada na Figura 19.

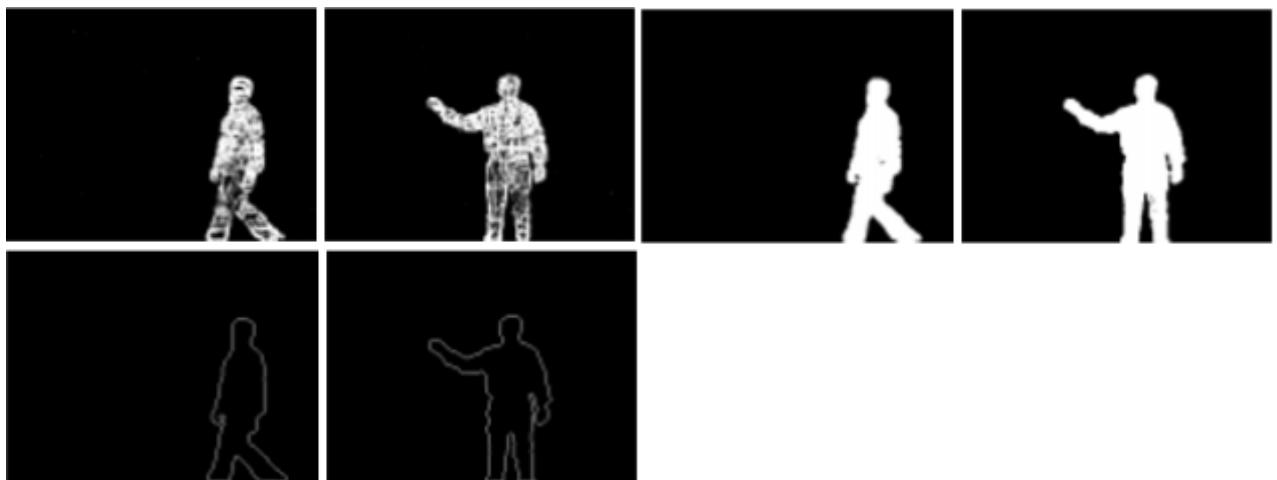
FIGURA 19 – Segmentação do fundo da cena.



FONTE: Jabri et al. (2000).

A separação através de bordas, Figura 20, é calculada pela alteração dos canais de cor, utilizando o fator Sobel, o qual realça os contornos através da intensidade dos pixels e consegue definir bordas horizontais e verticais, bem como o sentido da opacidade ocorrida, quando há.

FIGURA 20 – Separação da pessoa através das bordas.



FONTE: Jabri et al. (2000).

A imagem frontal é obtida através da diferença de escala de cinza atribuída na imagem retirada. Para que se extraia com maior definição, é aplicada a extração de contornos, conforme mostra a Figura 21.

Como conclusão, os autores verificaram que a extração de ruídos é favorecida com a aplicação de borda e cor, se adaptando bem a baixas frequências, podendo ser a porta de entrada para cenários mais específicos em realidade virtual, interação e reconhecimento de gestos.

FIGURA 21 – Resultado da segmentação de uma pessoa em vídeo.



FONTE: Jabri et al. (2000).

3.6 SISTEMA DE CONTAGEM DE PESSOAS BASEADO EM VÍDEO

Almeida *et al.* (2014) apresentam uma abordagem de segmentação, rastreamento e contagem de pessoas segundo estratégias de clusterização, separação e associação de blocos através do algoritmo *k-means*. A utilização de câmeras posicionadas de modo zenital é escolhida por ser o modo de sensoriamento mais preciso e com maior detalhamento de informações, superando os sensores infravermelhos ou mecânicos.

As técnicas de Processamento Digital de Imagens escolhidas pelos autores para a abordagem são a segmentação do plano de fundo, detecção de objetos em movimento através da diferença de frequência de escala de cinza na imagem (*template matching*) e a escolha do posicionamento zenital para a captura, pois, segundo os autores, define uma constante para o tamanho dos objetos, a melhor visão no cenário, mantém privacidade não levando em consideração rostos e não tem necessidade de calibração.

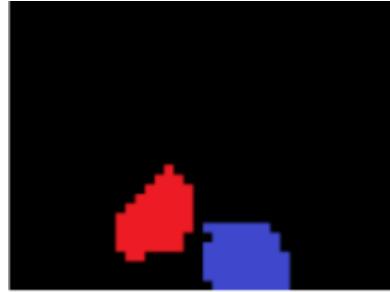
A abordagem segue o seguinte fluxo:

- a) Captura de vídeo;
- b) Remoção do plano de fundo;
- c) Segmentação por *k-means*;
- d) Rastreamento;
- e) Validação;
- f) Detecção;
- g) Contagem.

Utilizando a técnica *k-means* após a diferença entre os canais da imagem e definição do que faz parte do fundo do vídeo, é estimado o número de centróides, que

correspondem ao número de pessoas na cena, bem como a área de cada centróide, correspondendo ao tamanho médio de uma pessoa, ilustrado na Figura 22.

FIGURA 22 – Resultado da aplicação do algoritmo *k-means*.



FONTE: Almeida et al. (2014).

Após a definição de cada *cluster* (centróide), correspondente a uma pessoa, o algoritmo proposto calcula a menor distância Euclidiana entre os centróides pela diferença de quadros consecutivos, marcando-os como a mesma pessoa. A relação de *cluster* por quadro é armazenada dentro de uma matriz, onde a posição t corresponderá a 1, caso o mesmo *cluster* estiver no correspondente quadro $t + 1$. A contagem de pessoas, passo final do algoritmo, se dá através da análise da matriz resultante onde, caso exista a alteração dos valores de 0 para 1, um contador é somado.

Como experimento, o algoritmo foi treinado em dois ambientes, um instável e um controlado, com a mesma duração e o mesmo número máximo de pessoas. Observou-se que a distância da câmera ao chão afeta consideravelmente a distância entre os *clusters*, prejudicando a contagem. Sendo assim, foi necessária a utilização de métodos de correção: precisão e *recall*. A precisão é dada pela razão entre o ponto verdadeiro positivo e a soma verdadeiro positivo e falso positivo. O *recall* é dado pela razão entre o ponto verdadeiro positivo e a soma verdadeiro positivo e falso negativo. O F-Score, relação média ponderada entre precisão e *recall*, resulta em um valor mais real da diferença entre os *clusters*.

De acordo com o Quadro 1, foi visto que, com o melhor ajuste de parâmetros e filtro eliminando ruídos, os resultados seriam mais precisos.

3.7 RASTREAMENTO DE PESSOAS EM VÍDEOS DE FUNDO DINÂMICO

Siqueira e Machado (2015) apresentam uma minimização no número de quadros a serem analisados para a detecção e rastreamento de pessoas em vídeo. Utilizando como método principal a segmentação, o autor baseou-se no classificador *Adaboost*, esquematizado na Figura 23. O classificador consiste na combinação linear de características identificadas por meio do método *Mean Shift*, que define um objeto segundo suas medidas ou cor e delimitando-as segundo o Filtro de *Kalman*.

QUADRO 1 – Comparativo dos resultados obtidos por Almeida.

	Método Original
Pessoas	20
TP	20
FP+FN	0+1
Precisão	1.00
<i>Recall</i>	0.95
F-score	0.97
Questão	Alternativa

FONTE: (ALMEIDA et al., 2014).

Seguindo a classificação, é aplicada uma função do tipo *Haar* (Figura 24), definida pela transformada de *Haar*, que consiste na subtração da média dos valores da região mais escura pelos valores da região mais clara da imagem. A classificação dita forte se dá pela taxa de acerto nos quadros selecionados, ou seja, onde existe o objeto a ser rastreado.

Seguindo a estratégia de diminuição de quadros, os autorer utilizaram-se de um sensor de captura de 16 quadros por segundo. Após a aplicação dos métodos descritos, percebe-se, conforme mostra a Figura 25, que a detecção não se baseia diretamente na quantidade de elementos analisados, mas sim nas características definidas para o objeto a ser detectado ou rastreado, podendo assim diminuir a quantidade de quadros coletados para análise.

3.8 QUADRO COMPARATIVO

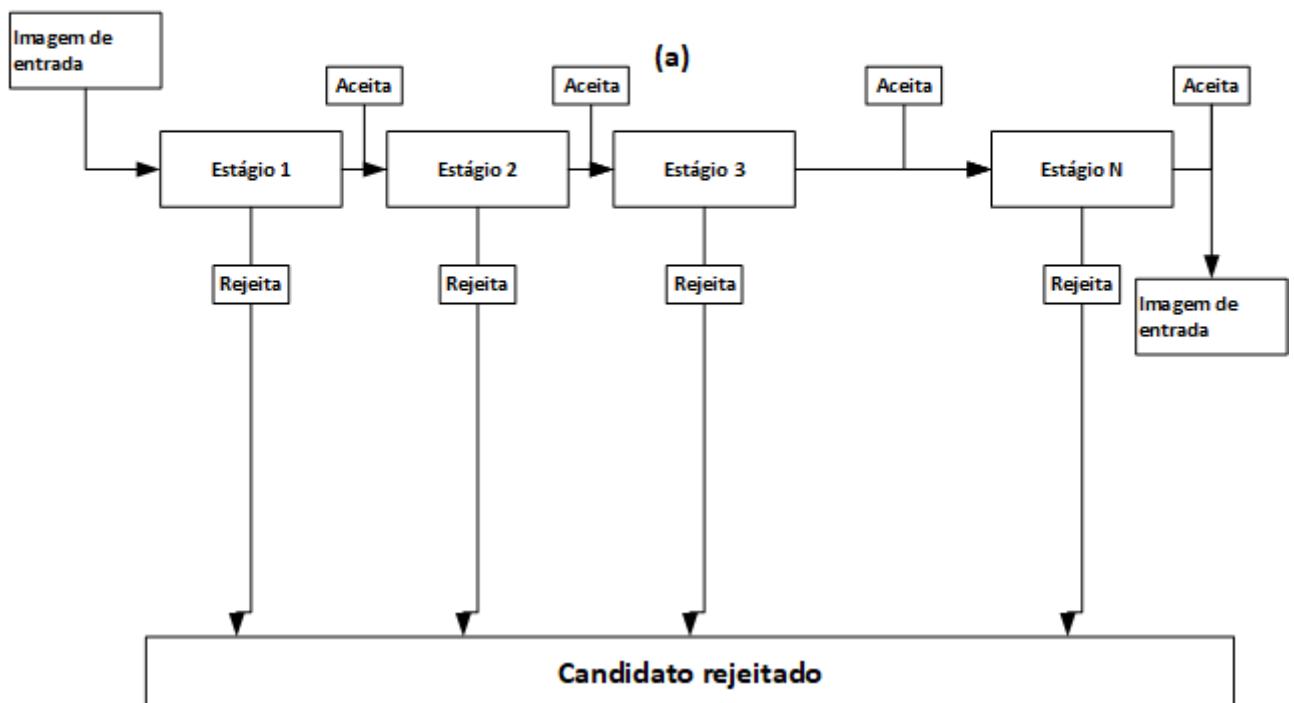
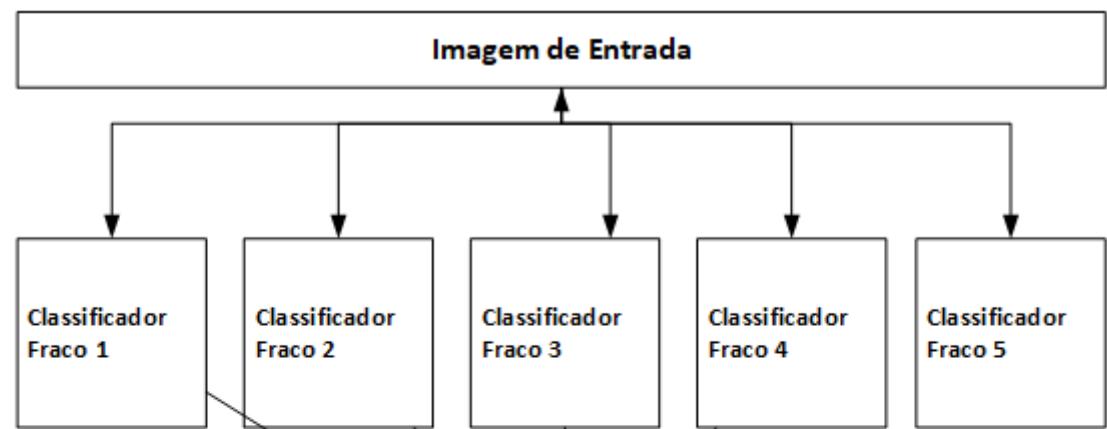
O Quadro 2 apresenta um comparativo entre os trabalhos apresentados e suas abordagens.

QUADRO 2 – Comparativo dos trabalhos relacionados.

Trabalho	Raspberry Pi	PDI	Abordagem
Shilpashree <i>et al.</i> (2015)	Sim	Sim	Algoritmo de redução de ruído ROF.
Gaier <i>et al.</i> (2013)	Sim	Sim	Integração Arduino, <i>Raspberry Pi</i> e OpenCV para reconhecimento de padrões.
Senthilkumar <i>et al.</i> (2014)	Sim	Sim	Algoritmo <i>Eigenfaces</i> .
Li <i>et al.</i> (2013)	Não	Sim	Algoritmo de modelo global para a segmentação de vídeo baseado na sobreposição de frames.
Jabri <i>et al.</i> (2000)	Não	Sim	Método de segmentação utilizando a separação de planos do vídeo.
Almeida <i>et al.</i> (2014)	Não	Sim	Detecção e contagem de pessoas com base no algoritmo <i>k-means</i> .
Siqueira e Machado (2015)	Não	Sim	Minimização dos frames utilizados para o rastreamento em vídeo.

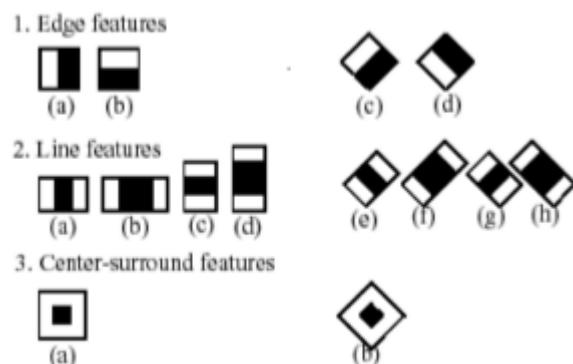
FONTE: O próprio autor, 2017.

FIGURA 23 – Esquematização do classificador Adaboost.



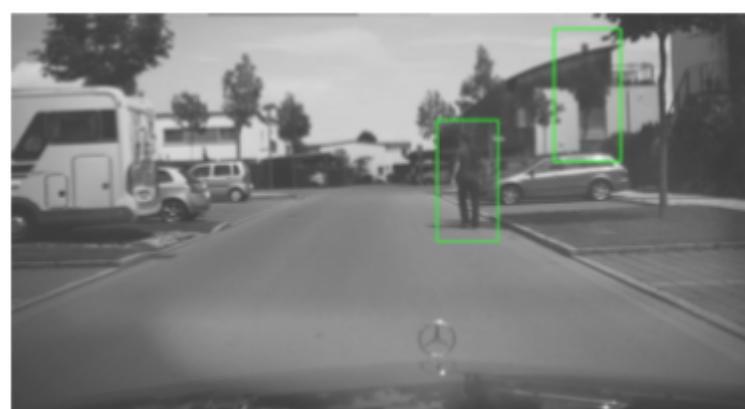
FONTE: Siqueira e Machado (2015).

FIGURA 24 – Esquematização de características do tipo Haar.



FONTE: Siqueira e Machado (2015).

FIGURA 25 – Resultado da aplicação dos métodos Adaboost e Haar.



FONTE: Siqueira e Machado (2015).

4 MATERIAIS E MÉTODOS

Este capítulo apresenta os materiais e metodologias aplicados no desenvolvimento do sistema de detecção de pessoas em vídeo. Para a implementação do trabalho proposto foi utilizada a placa Raspberry Pi, 3^a geração do seu modelo B, com o sistema operacional *Rasbian* em sua versão *Pixel*, de setembro de 2016, onde é realizado o processamento do vídeo capturado pelo módulo de câmera sob a biblioteca OpenCV, em sua versão 3.3.0, os quais são detalhados nas seções seguintes.

4.1 MATERIAIS UTILIZADOS

Neste trabalho foi utilizado o *Raspberry Pi* 3, modelo B com 1GB de memória e processador ARMv8 64-bit, quad-core de 1.2GHz. O sistema operacional instalado no dispositivo é o *Raspbian Jessie*, versão 8. Para a captura de imagens foi utilizado o módulo de câmera V2 com resolução de 8MP. Detalhes sobre o *Raspberry Pi* e sua arquitetura são discutidos na subseção 2.3.1.

Open Source Computer Vision Library (OpenCV) é uma biblioteca focada para desenvolvimento de aplicações voltadas ao PDI. Desenvolvida em âmbito acadêmico pela Intel, em 1999, a biblioteca se expandiu para o campo comercial visando o crescimento da necessidade de sistemas focados em visão computacional. As principais características do OpenCV são:

- a) Biblioteca multiplataforma com implementação a diversas linguagens de programação, como por exemplo, C++, Java, MatLab, Python, Perl e Ruby;
- b) Dividida em módulos: *cv* (funções principais), *highgui* (desenvolvimento de interface gráfica) e *cxcore* (estruturas de dados e funções de álgebra linear);
- c) Focada no desenvolvimento de sistemas de Processamento Digital de Imagens, análise estrutural e de movimento, rastreamento, entre outros;
- d) Suas funções tratam de imagens após a etapa de pré-processamento, onde essas imagens podem ser redimensionadas, padronizadas e filtradas para diminuição de ruídos;
- e) As funções implementam: detecção mais detalhada de ruídos, extração de informações, dentre outras.

4.2 METODOLOGIA

Neste trabalho foram analisados os métodos descritos na seção 2.1. Os métodos são: (1) AdaBoost, (2) MIL, (3) Median Flow, (4) TLD e (5) KCF. Estes métodos estão disponíveis na biblioteca OpenCV. As funções utilizadas para a criação dos classificadores são apresentadas no Código 1 e a utilização do classificador é apresentada no Código 2¹.

CÓDIGO 1 – Criação dos classificadores no OpenCV.

```

1   if tracker_type == 'BOOSTING':
2       tracker = cv2.TrackerBoosting_create()
3   if tracker_type == 'MIL':
4       tracker = cv2.TrackerMIL_create()
5   if tracker_type == 'KCF':
6       tracker = cv2.TrackerKCF_create()
7   if tracker_type == 'TLD':
8       tracker = cv2.TrackerTLD_create()
9   if tracker_type == 'MEDIANFLOW':
10      tracker = cv2.TrackerMedianFlow_create()
11

```

CÓDIGO 2 – Utilização dos classificadores no OpenCV.

```

1   ok, bbox = tracker.update(frame)
2

```

¹ <<https://github.com/spmallick/learnopencv/>>

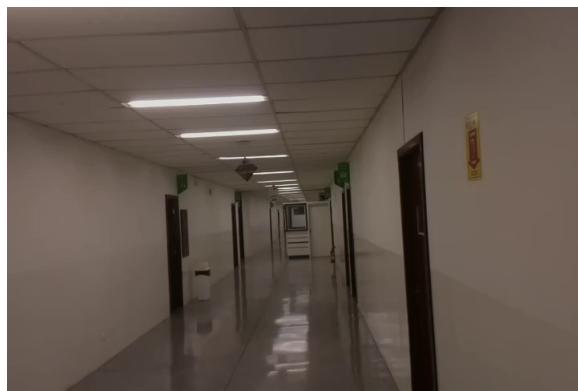
5 RESULTADOS EXPERIMENTAIS

Este capítulo apresenta os resultados obtidos a partir da metodologia construída na seção 4.2 aplicada à base de teste.

5.1 BASE DE DADOS

Para validação da metodologia proposta foi criada uma base de imagens simulando um ambiente real de aplicação de segurança. A base é composta por 25 vídeos, capturados através do *Raspberry Pi*, do corredor de acesso aos laboratórios de informática da Universidade Tuiuti do Paraná. A Figura 26 apresenta o ambiente utilizado para os testes.

FIGURA 26 – Ambiente utilizado para criação da base de dados.



FONTE: O próprio autor, 2017.

Ao todo foram definidos três cenários para validação do algoritmo de segmentação de pessoas em vídeo. O Quadro 3 apresenta os cenários definidos e a quantidade de vídeos capturados.

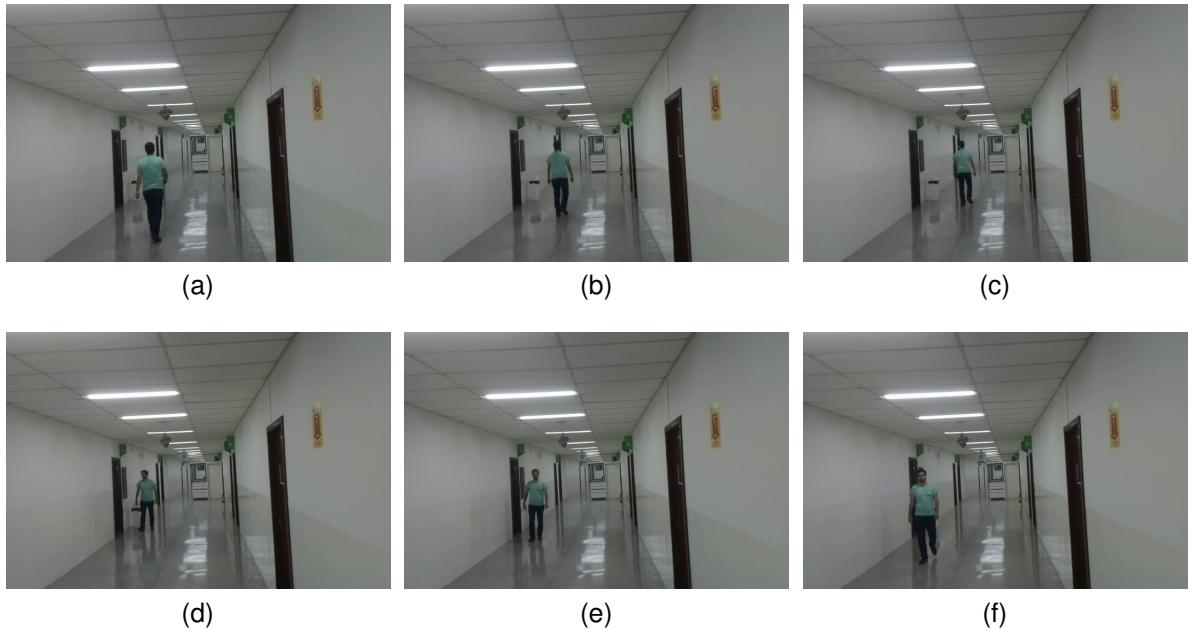
QUADRO 3 – Cenários utilizados na criação das bases.

Cenário	Quantidade de Pessoas	Quantidade de Vídeos
C1	1 pessoa	11
C2	2 pessoas	9
C3	3 pessoas	5

FONTE: O próprio autor, 2017.

Cada vídeo é composto por pessoas se aproximando ou se afastando da câmera. A Figura 27 apresenta um exemplo de vídeo composto por uma pessoa. As Figuras 27a-27c mostram quadros da pessoa se afastando do vídeo, e as Figuras 27d-27f mostram quadros da pessoa se aproximando do vídeo.

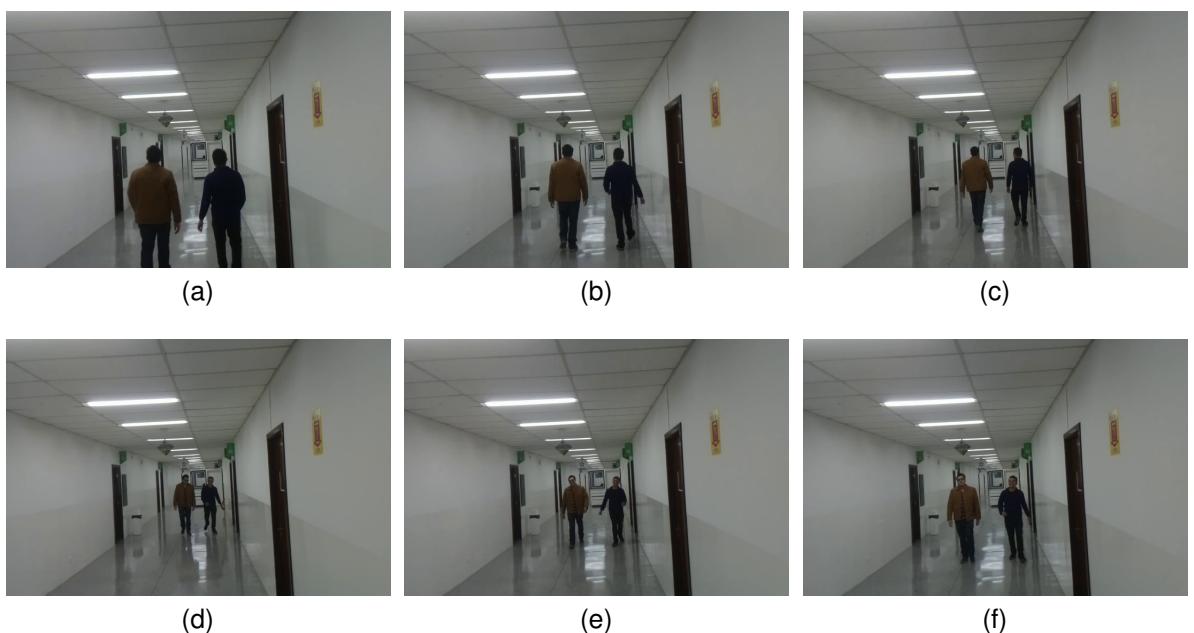
FIGURA 27 – Vídeo composto por uma pessoa.



FONTE: O próprio autor, 2017.

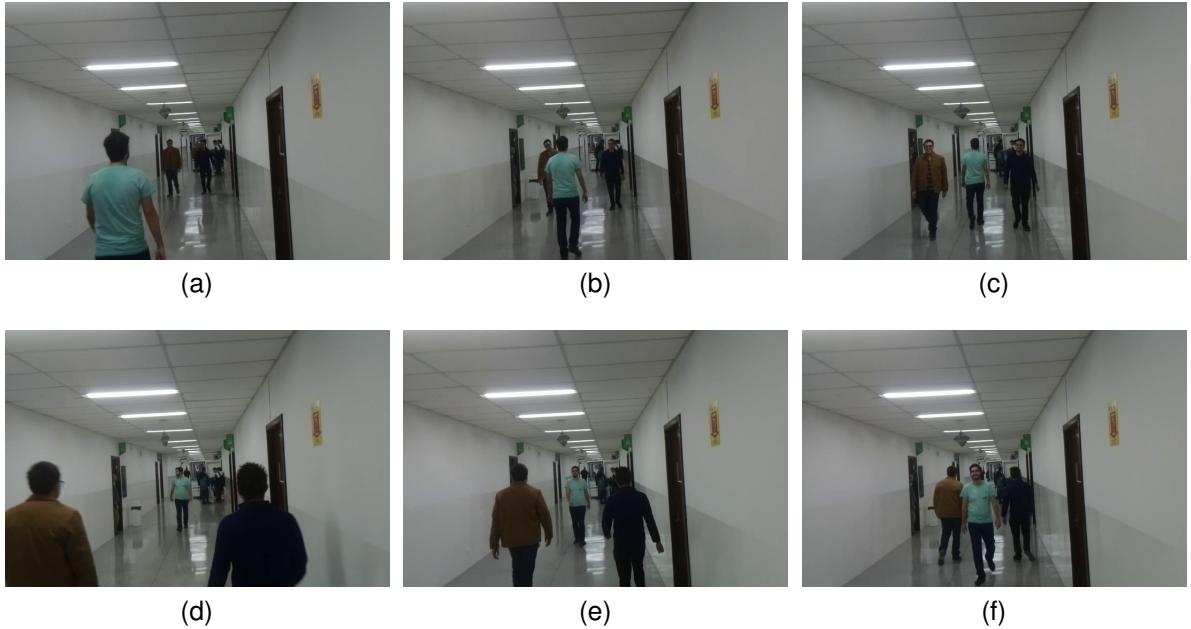
As Figuras 24 e 25 apresentam exemplos de vídeos capturados com duas e três pessoas, respectivamente. Uma das dificuldades encontradas na criação da base de dados foi manter a área isolada, isso por se tratar de um local de grande circulação de pessoas. Desta forma, em alguns vídeos apresentam pessoas circulando ao fundo, como pode ser visto na Figura 29.

FIGURA 28 – Vídeo composto por duas pessoas.



FONTE: O próprio autor, 2017.

FIGURA 29 – Vídeo composto por três pessoas.



FONTE: O próprio autor, 2017.

5.2 EXPERIMENTOS

Os experimentos foram realizados através do processamento de cada vídeo da base de dados utilizando os métodos propostos no Capítulo 4. O processamento foi realizado no próprio Raspberry Pi, com o intuito de verificar se o dispositivo consegue processar os vídeos de maneira satisfatória. Durante a execução dos vídeos, foi realizada uma análise visual do resultado da segmentação das pessoas em vídeo.

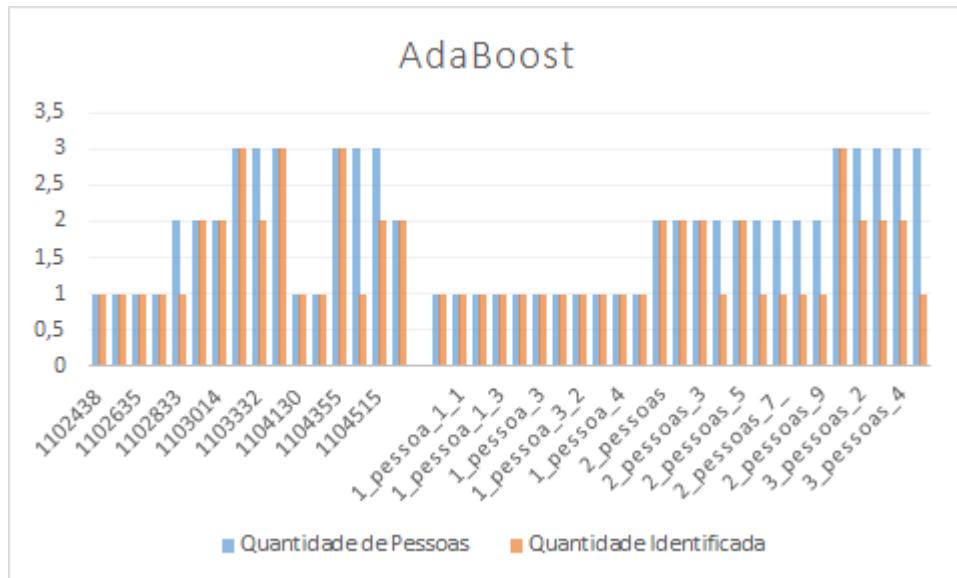
Os quadros a seguir apresentam os resultados obtidos das execuções dos algoritmos AdaBoost, MIL, Median Flow, TLD e KCF, respectivamente. Em cada Quadro, a primeira coluna apresenta o cenário analisado, descritos no Quadro 3. As demais colunas apresentam a taxa de acerto para a detecção de uma, duas ou três pessoas em cena, respectivamente. A definição *n/a* refere-se à relação onde não consta o número de pessoas esperado no cenário apresentado. A taxa de acerto define-se pela razão da quantidade de pessoas detectada pelo total de pessoas alocadas em todos os cenários.

QUADRO 4 – Resultados obtidos para o AdaBoost.

Cenário	1 pessoa	2 pessoas	3 pessoas
C1	100%	n/a	n/a
C2	100%	44%	n/a
C3	100%	60%	20%

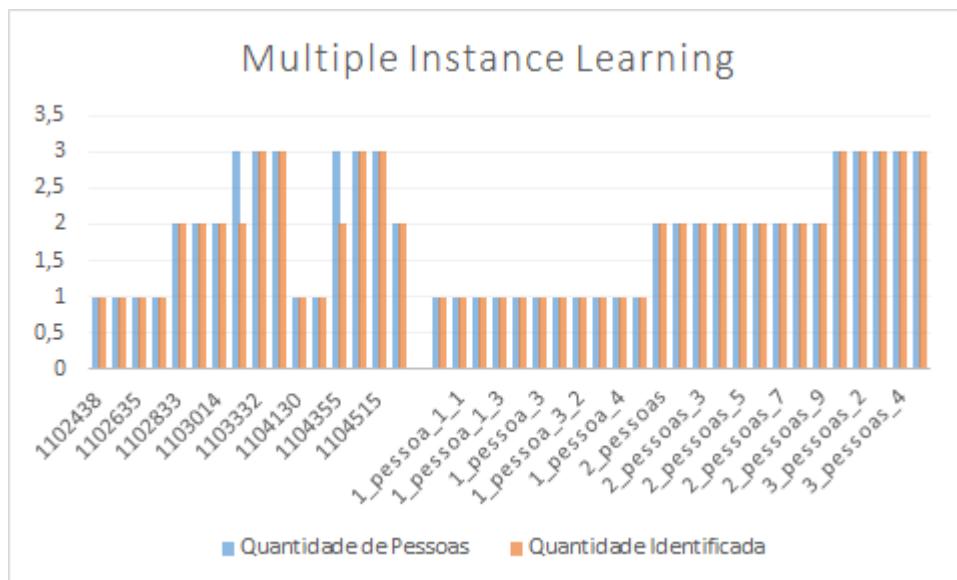
FONTE: O próprio autor, 2017.

GRÁFICO 1 – Resultados da aplicação do algoritmo AdaBoost nos cenários construídos.



FONTE: O próprio autor, 2017.

GRÁFICO 2 – Resultados da aplicação do algoritmo Multiple Instance Learning nos cenários construídos.



FONTE: O próprio autor, 2017.

Como pode ser observado nos resultados, o algoritmo Adaboost obteve uma taxa de acerto de aproximadamente 70,6%, resultado da grande variação de pesos por conta da variação do número de pessoas nos cenários propostos, causando um maior número de falsos positivos.

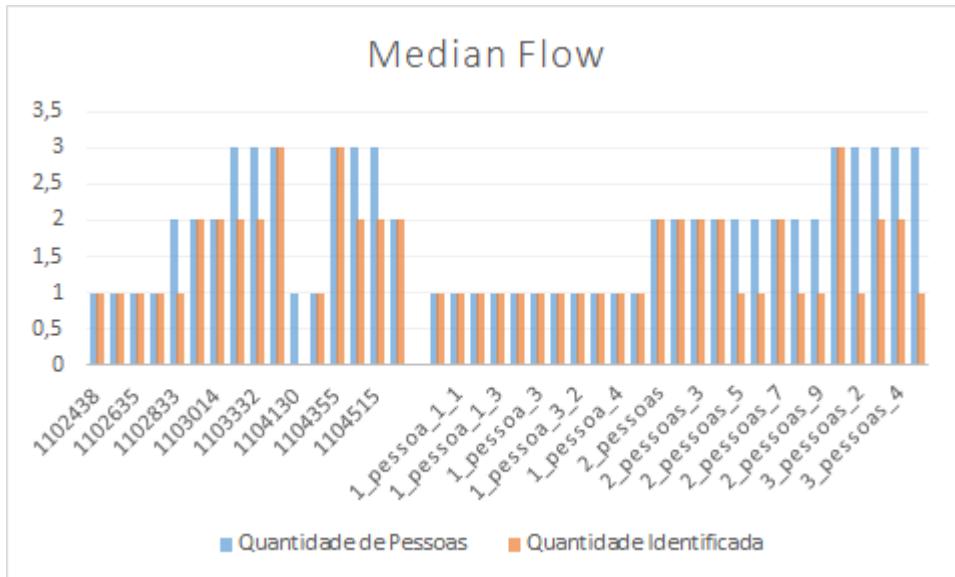
O algoritmo MIL apresentou melhor resultado em todos os cenários, identificando todas as pessoas presentes nas cenas. Esse algoritmo mostrou ser robusto para o rastreamento de pessoas se afastando e se aproximando da câmera por analisar um

QUADRO 5 – Resultados obtidos para o Multiple Instance Learning.

Cenário	1 pessoa	2 pessoas	3 pessoas
C1	100%	n/a	n/a
C2	100%	100%	n/a
C3	100%	100%	100%

FONTE: O próprio autor, 2017.

GRÁFICO 3 – Resultados da aplicação do algoritmo Median Flow nos cenários construídos.



FONTE: O próprio autor, 2017.

QUADRO 6 – Resultados obtidos para o Median Flow.

Cenário	1 pessoa	2 pessoas	3 pessoas
C1	100%	n/a	n/a
C2	88%	56%	n/a
C3	100%	60%	20%

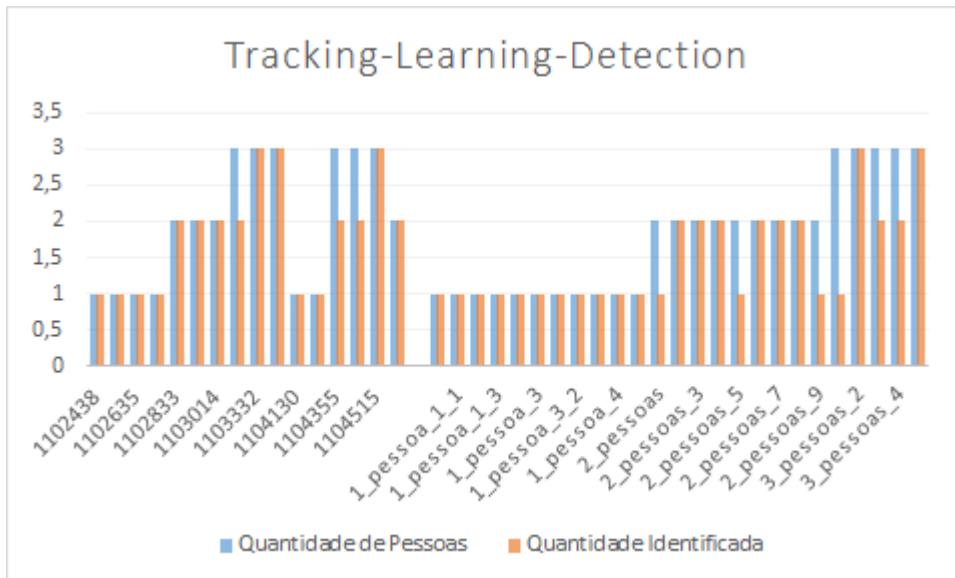
FONTE: O próprio autor, 2017.

conjunto de múltiplas instâncias do vídeo e rotulá-las com a finalidade de comparar com a delimitação criada da possível localização de uma pessoa no vídeo.

Com o método de rastreamento TLD, a taxa de acerto foi próxima a 79,2%, mostrando que, para as cenas onde existam mais de uma pessoa em movimento, o algoritmo temporariamente descarta um ponto dado como positivo para avaliar a trajetória da nova pessoa, afetando assim o resultado.

O método Median Flow obteve uma taxa de acerto de cerca de 70,6%, por

GRÁFICO 4 – Resultados da aplicação do algoritmo Tracking Learning Detection nos cenários construídos.



FONTE: O próprio autor, 2017.

QUADRO 7 – Resultados obtidos para o Tracking-Learning-Detection.

Cenário	1 pessoa	2 pessoas	3 pessoas
C1	100%	n/a	n/a
C2	88%	67%	n/a
C3	100%	80%	40%

FONTE: O próprio autor, 2017.

QUADRO 8 – Resultados obtidos para o Kernelized Correlation Filters.

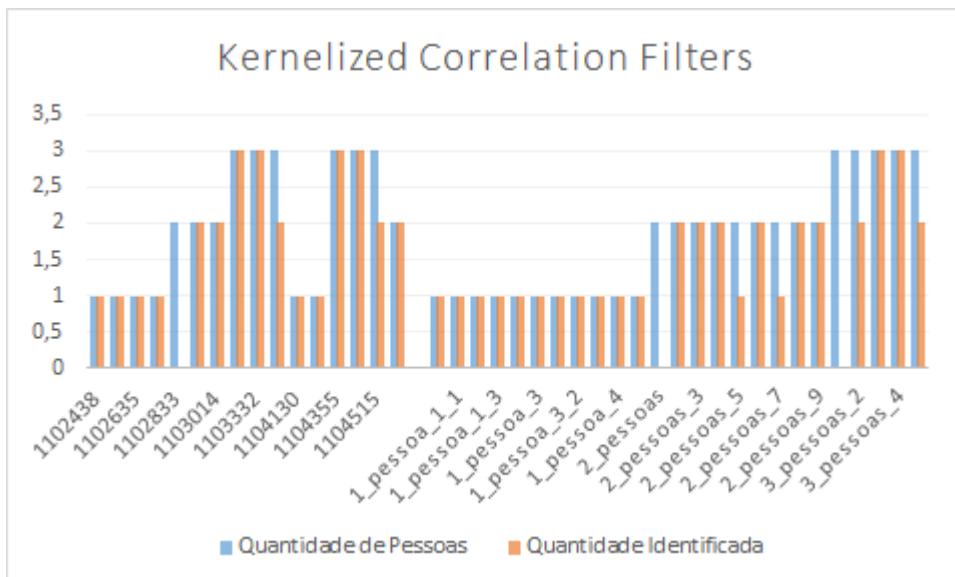
Cenário	1 pessoa	2 pessoas	3 pessoas
C1	100%	n/a	n/a
C2	96%	67%	n/a
C3	100%	60%	40%

FONTE: O próprio autor, 2017.

se tratar de uma base onde a orientação e posição das pessoas rastreadas varia consideravelmente, o método calcula a mediana dos pontos centrais do objeto rastreado, dessa maneira, quando há mais de uma pessoa em cena, existe a perda desse ponto central ou a mudança para outra pessoa que tenha seu ponto central em maior movimento no *frame*, afetando assim o cálculo.

Já o algoritmo KCF apresentou uma taxa de acerto de 77,16% para a amostragem construída, onde se mostrou eficiente nos cenários onde é identificada apenas

GRÁFICO 5 – Resultados da aplicação do algoritmo Kernelized Correlation Filters nos cenários construídos.



FONTE: O próprio autor, 2017.

uma pessoa.

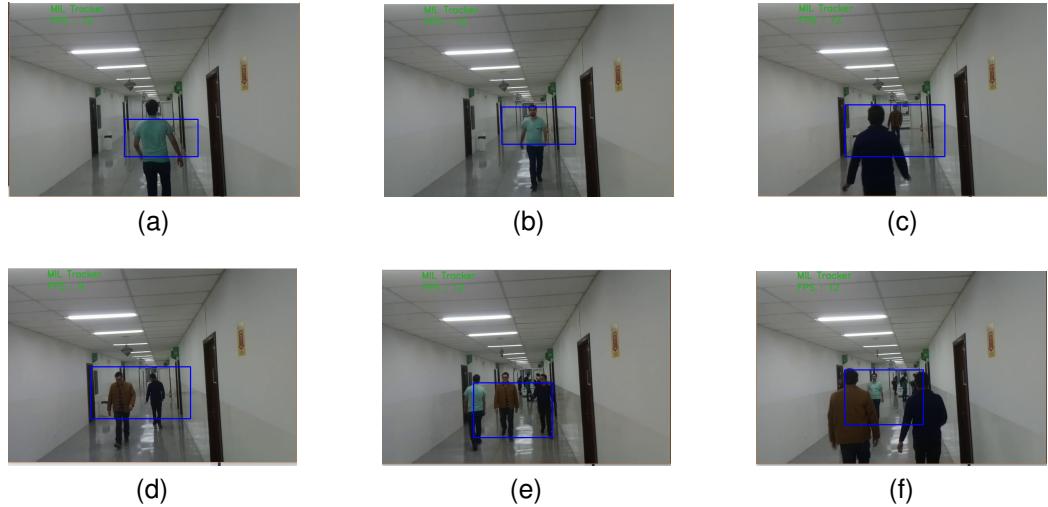
Com a análise dos resultados, pode-se afirmar que o uso do algoritmo MIL em sistemas de segurança e vigilância com um alto nível de precisão em imagens capturadas pelo Raspberry

A Figura 30 demonstra exemplos onde o rastreamento foi bem sucedido e puderam-se observar as pessoas destacadas. A Figura 31 demonstra *frames* onde as técnicas de rastreamento não obtiveram resultados satisfatórios.

Os resultados apresentados na Figura 30 são satisfatórios pois em todos os *frames* do vídeo, foi possível rastrear todas as pessoas das cenas.

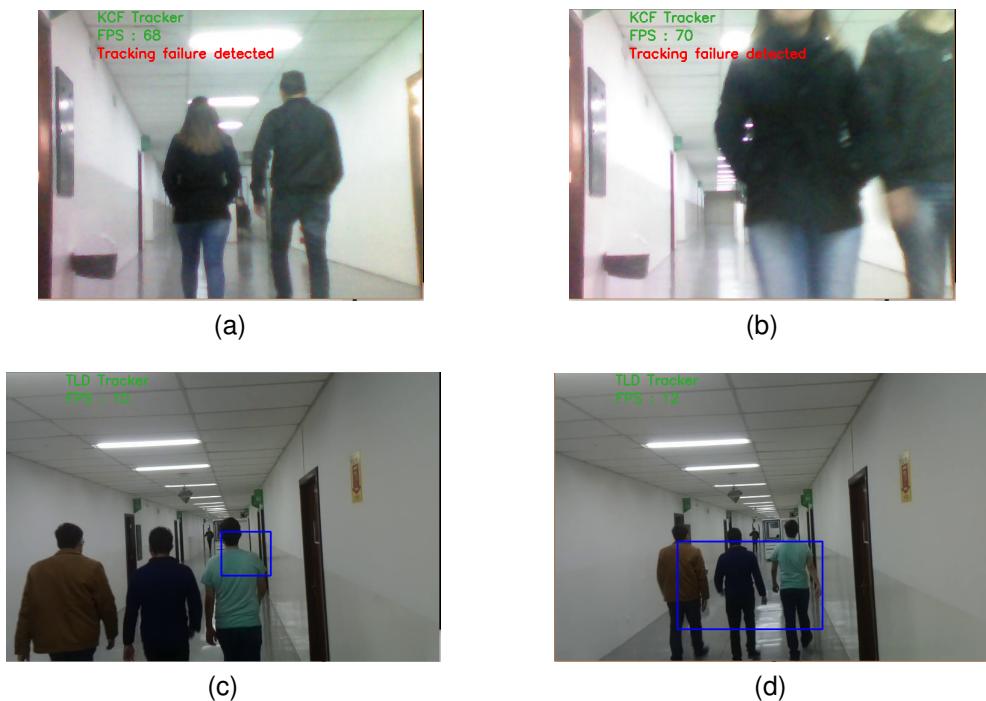
Os resultados apresentados nas Figuras 27 (a) e (b) demonstram a saída de erro onde não foi possível localizar ao menos uma pessoa na cena em todos os *frames* da figura. Já as Figuras 27 (c) e (d) apresentam os frames da saída onde foram detectadas pessoas, porém não todas as presentes no vídeo ou em *frames* do mesmo, causando a perda de informações no rastreamento

FIGURA 30 – Acertos do rastreamento aplicado aos vídeos capturados.



FONTE: O próprio autor, 2017.

FIGURA 31 – Erros do rastreamento aplicado aos vídeos capturados.



FONTE: O próprio autor, 2017.

6 CONCLUSÕES

Neste trabalho foram abordados métodos para o rastreamento de pessoas, implementados dentro da biblioteca *OpenCV* para vídeos capturados com o módulo de câmera do *Raspberry Pi*, sendo seu processamento realizado no mesmo dispositivo.

Como base para os testes foram determinados cenários de captura, os quais foram adquiridos dentro da Universidade Tuiuti do Paraná, em seu corredor de acesso aos laboratórios de informática. Foram montados cenários de uma pessoa, duas pessoas e três pessoas circulando pelo corredor.

Para cada cenário as imagens de vídeo capturadas simulam a circulação em um ambiente onde pessoas podem ser detectadas tanto movimentando-se em direção à câmera quanto afastando-se do local onde a câmera foi instalada.

Os vídeos capturados foram submetidos à abordagem de quatro diferentes algoritmos de rastreamento, esses implementados dentro da biblioteca *OpenCV*, para verificar a taxa de acerto e confiabilidade desses algoritmos aplicados à base montada.

Após suas validações, pode-se verificar que o método de rastreamento MIL foi o que obteve melhor taxa de identificação, em aproximadamente 100% da base construída nos três cenários.

Como conclusão do estudo das técnicas de rastreamento implementadas dentro da *OpenCV*, observou-se que os resultados dentro do dispositivo *Raspberry Pi* mostraram a eficiência de técnicas de PDI em sua arquitetura, uma vez que a biblioteca foi compilada diretamente nela.

Como sugestão de trabalhos futuros indica-se o uso do sistema proposto em outro ambiente, como dentro de salas de aulas ou em bibliotecas, para validar a movimentação de pessoas. Outra melhoria possível para o sistema proposto é a definição no algoritmo e separação automática das pessoas detectadas, possibilitando assim a contagem e definição de possíveis falsos positivos.

REFERÊNCIAS

- ALMEIDA, S. S. de et al. Uma implementação de um sistema de contagem de pessoas baseado em vídeo processamento. UFOP, MG, 2014.
- BABENKO, B.; YANG, M. hsuan; BELONGIE, S. *Visual Tracking with Online Multiple Instance Learning*. 2009.
- FOUNDATION, R. P. *Módulo Câmera Raspberry Pi*. 2017. Disponível em: <<https://www.raspberrypi.org/documentation/hardware/camera/README.md>>.
- FOUNDATION, R. P. *Raspberry Pi* -. 2017. Disponível em: <<https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2837/README.md>>.
- FOUNDATION, R. P. *Raspberry Pi Foundation*. 2017. Disponível em: <<https://www.raspberrypi.org/about>>.
- GAIER, M. B.; SILVA, R. S. *Configuração: uma perspectiva de Arquitetura da Informação da Escola de Brasília*. Dissertação (Mestrado) — Instituto Federal de Educação, Ciência e Tecnologia - DAEE, Mato Grosso, 2013.
- GARROT, J.; FELGUEIRAS, C. *Introdução ao Processamento Digital de Imagem*. [S.I.]: FCA, 2008.
- GRABNER, H.; BISCHOF, H. On-line boosting and vision. In: IEEE. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. [S.I.], 2006. v. 1, p. 260–267.
- GRISENTHWAITE, R. *ARMv8 Technology Preview*. 2011. Disponível em: <https://www.arm.com/files/downloads/ARMv8_Architecture.pdf>.
- HENRIQUES, J. F. et al. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 37, n. 3, p. 583–596, 2015.
- JABRI, S. et al. Detection and location of people in video images using adaptive fusion of color and edge information. In: IEEE. *Pattern Recognition, 2000. Proceedings. 15th International Conference on*. [S.I.], 2000. v. 4, p. 627–630.
- KALAL, Z.; MIKOLAJCZYK, K.; MATAS, J. Forward-backward error: Automatic detection of tracking failures. In: *In Proceedings of the 2010 20th International Conference on Pattern Recognition, ICPR '10*. [S.I.]: IEEE Computer Society, 2010. p. 2756–2759.
- KALAL, Z.; MIKOLAJCZYK, K.; MATAS, J. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 34, n. 7, p. 1409–1422, 2012.
- LI, F. et al. Video segmentation by tracking many figure-ground segments. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.I.: s.n.], 2013. p. 2192–2199.

- LU, H. et al. Robust tracking based on pso and on-line adaboost. In: IEEE. *Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP'09. Fifth International Conference on.* [S.I.], 2009. p. 690–693.
- MOORE, G. E. Cramming more components onto integrated circuits. *Proceedings of the IEEE*, IEEE, v. 86, n. 1, p. 82–85, 1998.
- OLIVEIRA, J. B. Vianna S. de M. *Arquitetura Distribuída para Transcodificação de Vídeo com Alto Desempenho.* Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2013.
- SENTHILKUMAR, G.; GOPALAKRISHNAN, K.; KUMAR, V. S. Embedded image capturing system using raspberry pi system. *International Journal of Emerging Trends & Technology in Computer Science*, v. 3, n. 2, p. 213–215, 2014.
- SHILPASHREE, K.; LOKESHA, H.; SHIVKUMAR, H. Implementation of image processing on raspberry pi. *International Journal of Advanced Research in Computer and Communication Engineering*, IJARCCE, v. 4, n. 5, p. 199–202, 2015.
- SIQUEIRA, D. L.; MACHADO, A. M. C. Avaliação da amostragem temporal na detecção e no rastreamento de pessoas em vídeos de fundo dinâmico. PUC-MG, 2015.
- SOLOMON, C.; BRECKON, T. *Fundamentos de processamento digital de imagens: uma abordagem prática com exemplos em Matlab.* [S.I.]: Grupo Gen-LTC, 2000.
- WOODS, R. C. G. R. E. *Digital Image Processing.* [S.I.]: Prentice-Hall, Inc., 2006.