

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS BLUMENAU**

**Monitoramento com Visão
Computacional para Lojas de Varejo**

Alunos	Eduardo Mafra Pereira (15102929) Jorge Lucas de Santana (14104055)
Disciplina	BLU3702 (20182) - Projeto Integrador
Professores	Luiz Antonio Maccari Junior, Adão Boava e Carlos Roberto Moratelli

Blumenau, Junho de 2018

Conteúdo

1	Descrição da problema	1
2	Identificação da proposta	1
3	Revisão bibliográfica	1
3.1	Visão Computacional	1
3.2	OpenCV	2
3.3	Pycharm	3
3.4	Redes neurais	3
3.5	Mapa de calor	3
4	Objetivos e limitações	3
5	Funcionamento	5
6	Resultados	12
7	Sugestões para melhorias futuras	13
8	Conclusão	13
9	Referências	14

1 Descrição da problema

Coletar dados do meio tangível aos seres humanos é uma tarefa ainda pouco explorada, existindo poucas empresas que oferecem a outros empreendimentos serviços de tratamento de dados através de visão computacional. E portanto, muitas empresas não imaginam que existam meios de tratar dados visuais sem que hajam grandes modificações em seus estabelecimentos.

2 Identificação da proposta

Este relatório apresenta a proposta do desenvolvimento de um projeto de monitoramento baseado nos conhecimentos de visão computacional. Propõe-se desenvolver um software capaz de monitorar clientes através de câmera, para assim analisar quais as seções de produtos que mais geram interesse de um cliente durante sua passagem por uma loja de varejo. O software recebe como entrada um vídeo da seção de produtos que se quer monitorar, e em seguida faz identificação das pessoas que aparecem na cena e armazena em um vetor os locais em que estas pessoas se encontram, retornando uma imagem de mapa de calor do ambiente.

Obter a demografia de lojas e o tempo que seus clientes observam cada modelo de produto, permite entender melhor as necessidades do cliente, possibilitando assim posicionar os produtos de forma estratégica e compreender o motivo pelo qual um produto pode ser amplamente observado porém pouco vendido. Fatores estes, que podem refletir positivamente na lucratividade dos estabelecimentos.

Na secção a seguir serão apresentados os conceitos básicos para compreensão do desenvolvimento do projeto.

3 Revisão bibliográfica

Mapas de calor são ferramentas muito utilizadas para analisar dados, podendo serem implementados para as mais diversas aplicações. Neste documento o mapa de calor indicará quais são as ar

3.1 Visão Computacional

Para o desenvolvimento do projeto serão utilizados conceitos de visão computacional. A visão computacional é o processo de modelagem e replicação da visão humana usando software e hardware. Pode ser também

compreendida como uma ciéncia que possibilita extrair de forma automáti-
ca informações de imagens.

Algumas das tarefas típicas de visão computacional são: reconhecimento,
identificação, detecção, movimento, reconstrução de cena e restauração de
imagens.

Além de ser capaz de identificação de objetos em imagens, pode ser uti-
lizada para identificar doenças médicas em raios-x, inspeção de controle de
qualidade em produtos na industria, verificação de anúncios dentro de ima-
gens editoriais, ou ainda veículos autônomos que também utilizam bastante
esta tecnologia. A tecnologia é complexa e, assim como todas as tarefas
mencionadas, requer mais do que apenas reconhecimento de imagem, mas
também análise semântica de grandes conjuntos de dados.

3.2 OpenCV

Desenvolvido pela Intel, o Opencv é uma biblioteca multiplataforma de
visão computacional, de código aberto tanto para uso acadêmico quanto co-
mercial. Com suporte para linguagens C++, Python e Java, foi projetado
para eficiência computacional e aplicações em tempo-real. Na figura 1 vemos
um exemplo de utilização dessa biblioteca, que está sendo executada junto
com outras bibliotecas para identificar e classificar as classes: dog, bicycle e
truck (cachorro, bicicleta e caminhonete).

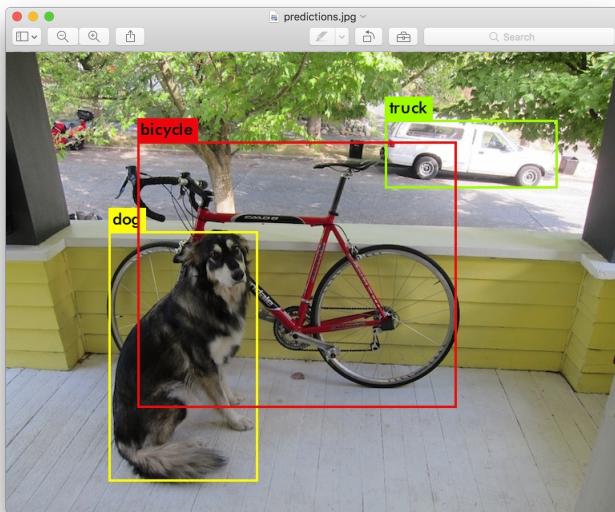


Figura 1: Exemplo de aplicação do OpenCV

3.3 Pycharm

O Pycharm é um ambiente de desenvolvimento integrado utilizado especificamente para linguagem Python. Disponibilizado pela empresa JetBrains de forma gratuita na sua versão Community e Student, esta ferramenta auxilia o programador realizando detecção de erros, completando funções, e disponibilizando um compilador próprio.

3.4 Redes neurais

Para fazer a detecção de objetos em imagens, existe toda uma teoria de Deep Learning (aprendizado profundo) em que são treinados os computadores com intuito de realizarem tarefas como os seres humanos. Na detecção e localização de objetos em cenas, existem as redes neurais pré-treinadas, que são diversos modelos já treinados para certo tipo de identificação. Tipicamente a saída dessas redes é uma caixa retangular (bounding box) que registra a região de maior confiança da presença do objeto detectado. Alguns modelos são: Mobilenet, CNN: R-CNN, Faster-R-CNN, YOLO, etc.

3.5 Mapa de calor

Mapa de calor (heat map) é uma ferramenta muito utilizada para analisar dados de ambientes multidimensionais (2 ou até 3 dimensões) através de imagens, e representa a densidade geográfica de elementos de pontos em um mapa utilizando áreas coloridas. Seu gradiente de cor varia conforme esta concentração, apresentando cores mais opacas para regiões com menor concentração de pontos e cores mais vibrantes para maiores concentrações.

Como toda a imagem é apresentada como uma matriz de duas dimensões, os pontos que geram o mapa de calor representam posições (em coordenadas x, y) na imagem utilizada. Sendo assim, a área mais visitada na imagem será a de maior concentração demográfica.

4 Objetivos e limitações

O objetivo geral deste projeto consiste na elaboração de um produto que utilize visão computacional direcionado a lojas de varejo, retornando um mapa de calor do estabelecimento baseado no fluxo de clientes. O produto irá recorrer as câmeras de segurança já presentes nas unidades de vendas a fim de diminuir os custos do produto. Para isto utilizará a biblioteca de código aberto OpenCV e a linguagem de programação Python em seu

desenvolvimento. Uma ilustração do mapa de calor esperado pode ser visto na figura 2.

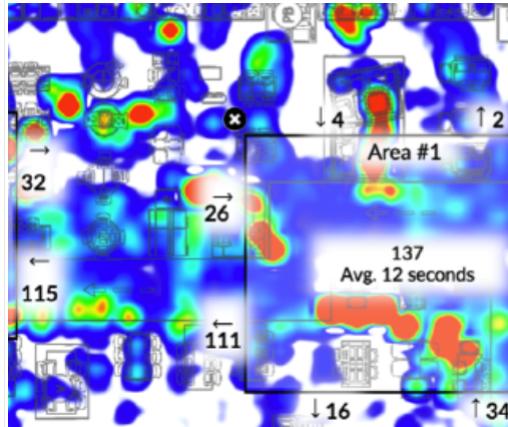


Figura 2: Mapa de calor

Para atingir os objetivos gerais, os seguintes objetivos específicos devem ser atendidos:

- Rastrear os clientes durante todo seu tempo no local.
- Capturar dados que retornam o período que o cliente ficou observando um secção de produtos.
- Realizar a construção do mapa de calor de vídeos já gravados e com duração a ser determinada pelo cliente. Ficando a solução em tempo real para projetos futuros, pois imagina-se fácil transição para esse modelo.
- Capturar os dados com base apenas no fluxo de pessoas pode ser um problema, pois a porta de entrada, por exemplo, é o local de maior fluxo mas não necessariamente apresenta dados relevantes sobre as observações dos produtos. Sendo assim, serão tratados estes possíveis problemas de implementação e especificado o posicionamento mais adequado das câmeras.
- Como o objetivo inicial é desenvolver apenas o mapa de calor, não será necessário um grande banco de dados para armazenar informações. Portanto inicialmente os mapas de calor serão armazenados em disco, possibilitando a empresa acessá-los a qualquer momento.
- Inicialmente será utilizada apenas uma câmera, pois já contemplam uma região razoável de uma loja de varejo.

5 Funcionamento

O programa em python é executado via terminal através da função "python trabalho.py –input videos/sala1.AVI", com a obtenção do vídeo (sala1.AVI) salvo ou Webcam o programa salva seu frame inicial como uma imagem de formato desejado (.jpg, .png, ...) para utilização desta posteriormente, sendo armazenada dentro do computador na pasta escolhida. Com o vídeo em execução o programa inicia a detecção da classe desejada, esta classe é definida pelo programador dentro do código a partir de uma lista de classes na qual a rede MobileNet é capaz de detectar. Para a primeira simulação utilizou-se a classe de detecção "person" a fim de detectar apenas pessoas.

Cada objeto detectado irá gerar um ID diferente, este tem o intuito de diferenciar os objetos, nota-se que para a imagem 1 da figura 3 o objeto é detectado e associado ao ID 0.

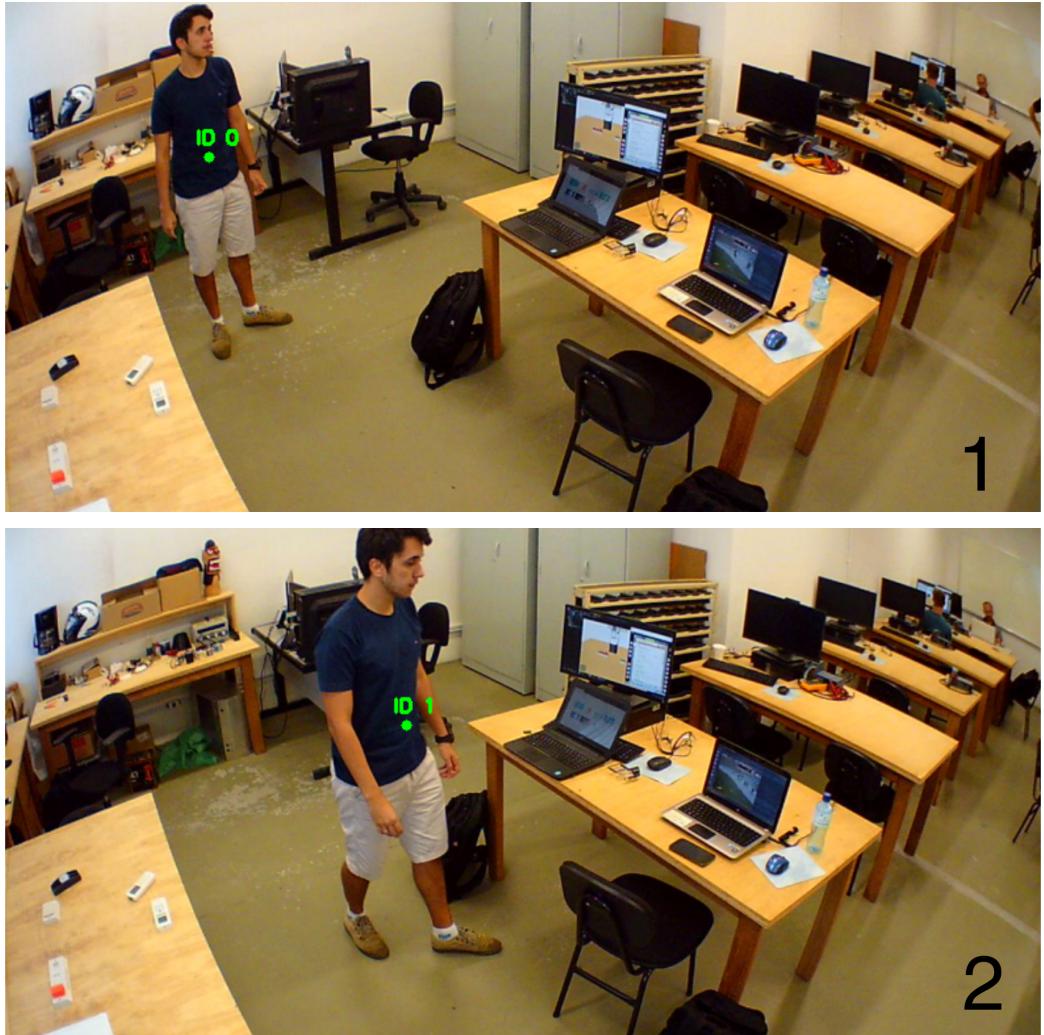


Figura 3: Simulação 1

Com o intuito de otimizar o processamento do programa realiza-se o rastreamento do objeto encontrado, não necessitando portanto que haja detecções frame a frame (solução bastante utilizada em sistemas de rastreamento utilizando OpenCV). Novas detecções serão feitas a cada 20 frames, a fim de identificar novos objetos e objetos perdidos durante o rastreamento. Sendo assim o monitoramento é dividido em duas etapas distintas: a detecção (detection) e o rastreamento (tracking).

Com o decorrer do rastreamento as posições do objeto são armazenadas em um vetor, este vetor será utilizado na geração o mapa de calor conforme figura 4.

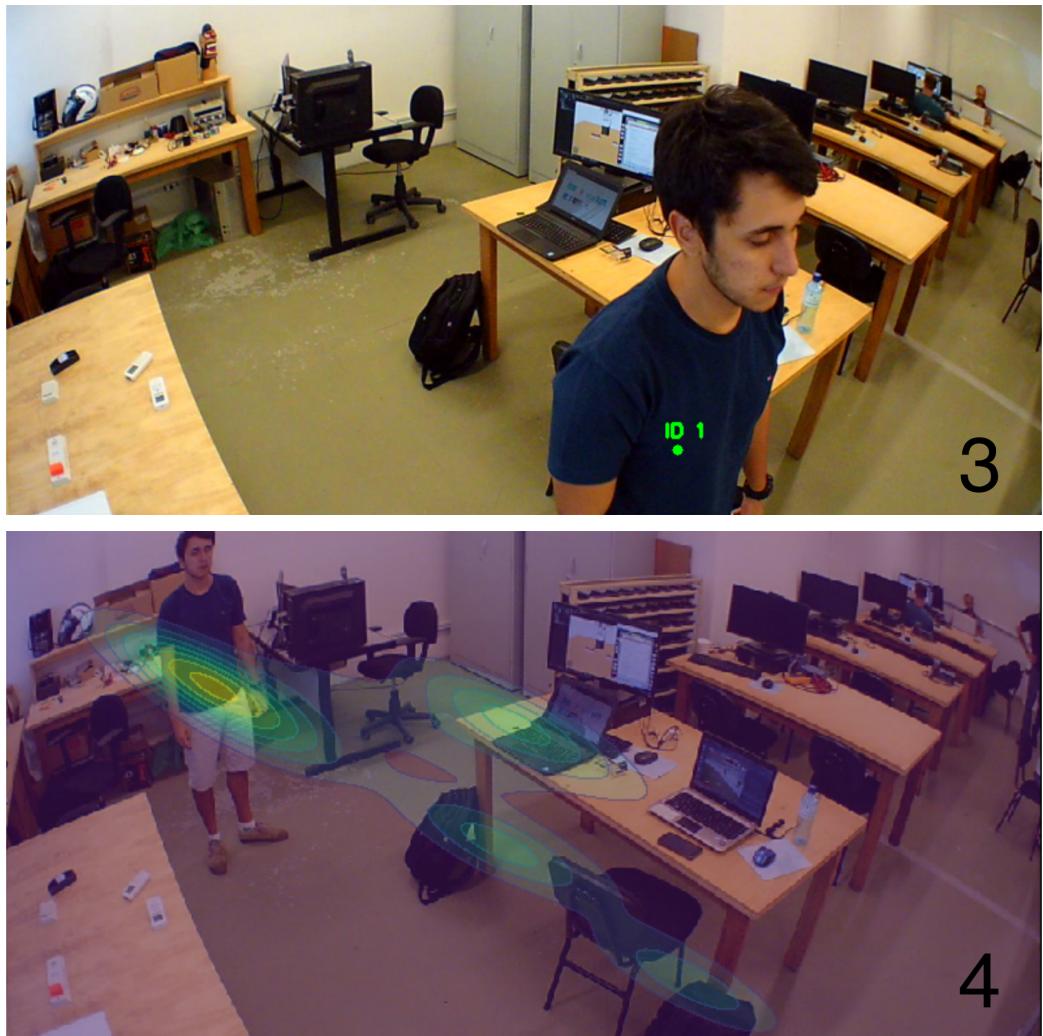


Figura 4: Simulação 1

Após processar o vídeo de interesse e armazenar as posições do objeto no vetor o mapa de calor é processado, este processo é realizado calculando a media gaussiana gerada através das coordenadas x,y armazenadas no vetor. Estas médias geram um plot matemático que sobreposto a imagem capturada pelo primeiro frame, gera o mapa de calor (imagem 4 da Figura 4).

Uma segunda simulação foi realizada em ângulo e posição diferentes para comprovar e eficácia do projeto desenvolvido, esta simulação pode ser verificada pelas figuras 5 e 6.

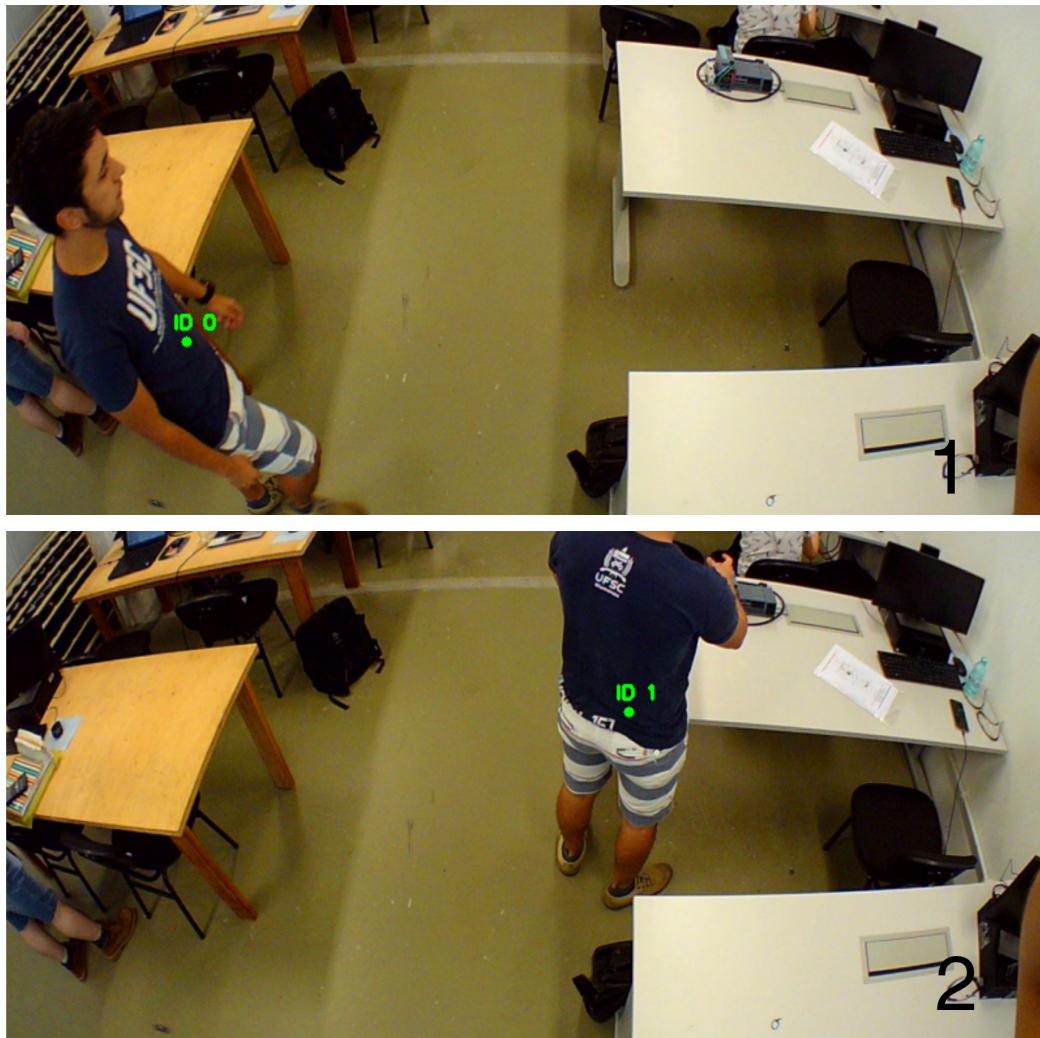


Figura 5: Simulação 2

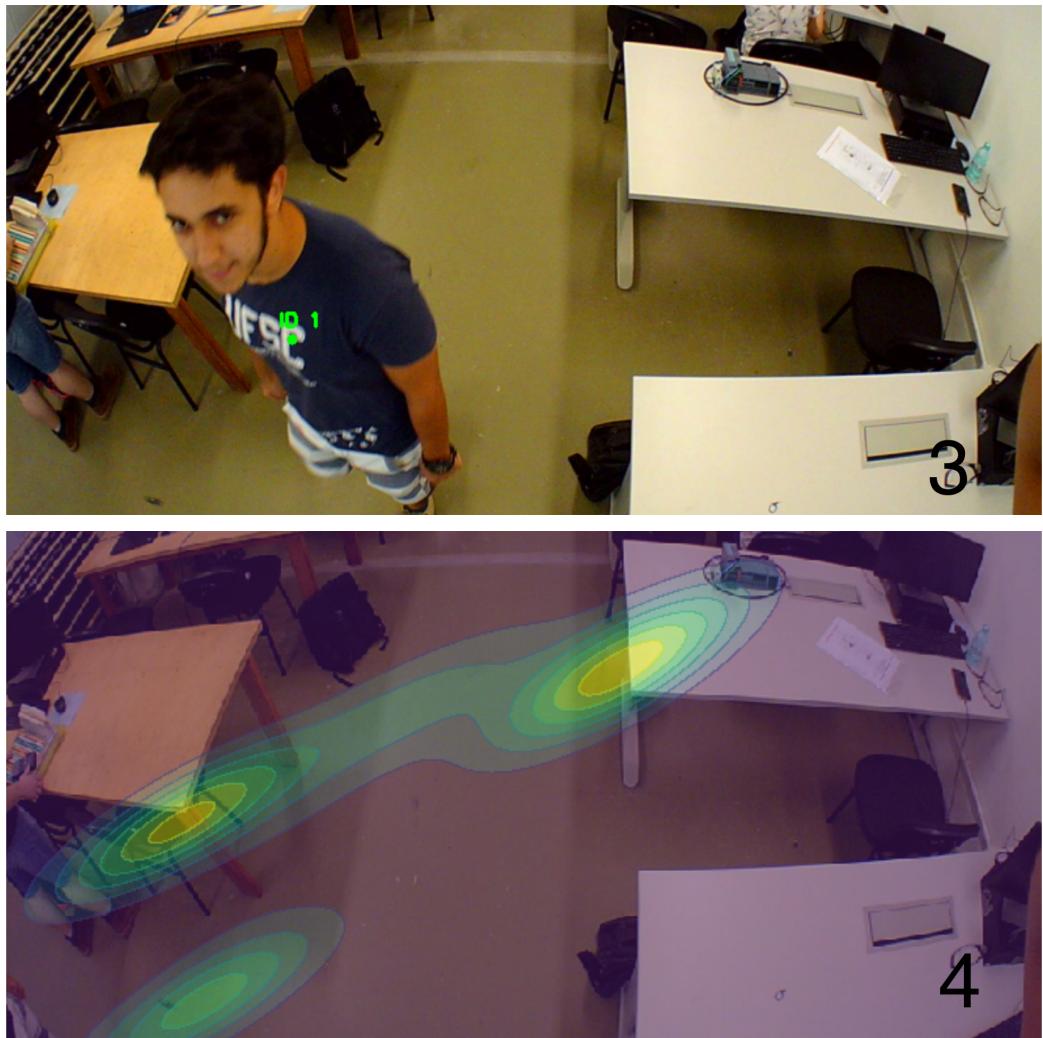


Figura 6: Simulação 2

Sendo o mapa de calor gerado através das posições do objeto capturadas durante a execução do vídeo, as regiões mais quentes serão as que contêm as posições capturadas com maior frequência pelo vetor posição. Estas regiões referentes a simulação 2 podem ser observadas na imagem 2 da Figura 7.



Figura 7: Simulação 3

Como já comentado anteriormente existem as classes de objetos que podem ser detectados pela rede MobileNet, são elas: "background", "airplane", "bicycle", "bird", "boat", "bottle", "bus", "car", "cat", "chair", "cow", "diningtable", "dog", "horse", "motorbike", "person", "pottedplant", "sheep", "sofa", "train", "tvmonitor".

Foi feito um teste com o programa desenvolvido para gerar um mapa de calor utilizando a classe "car", que pode ser visualizado nas figuras 8 e 9. O mapa de calor obtido para esse caso pode ser visto na imagem 4 da Figura 9 e demonstra as regiões onde por mais tempo identificou carros.



Figura 8: Simulação para carros

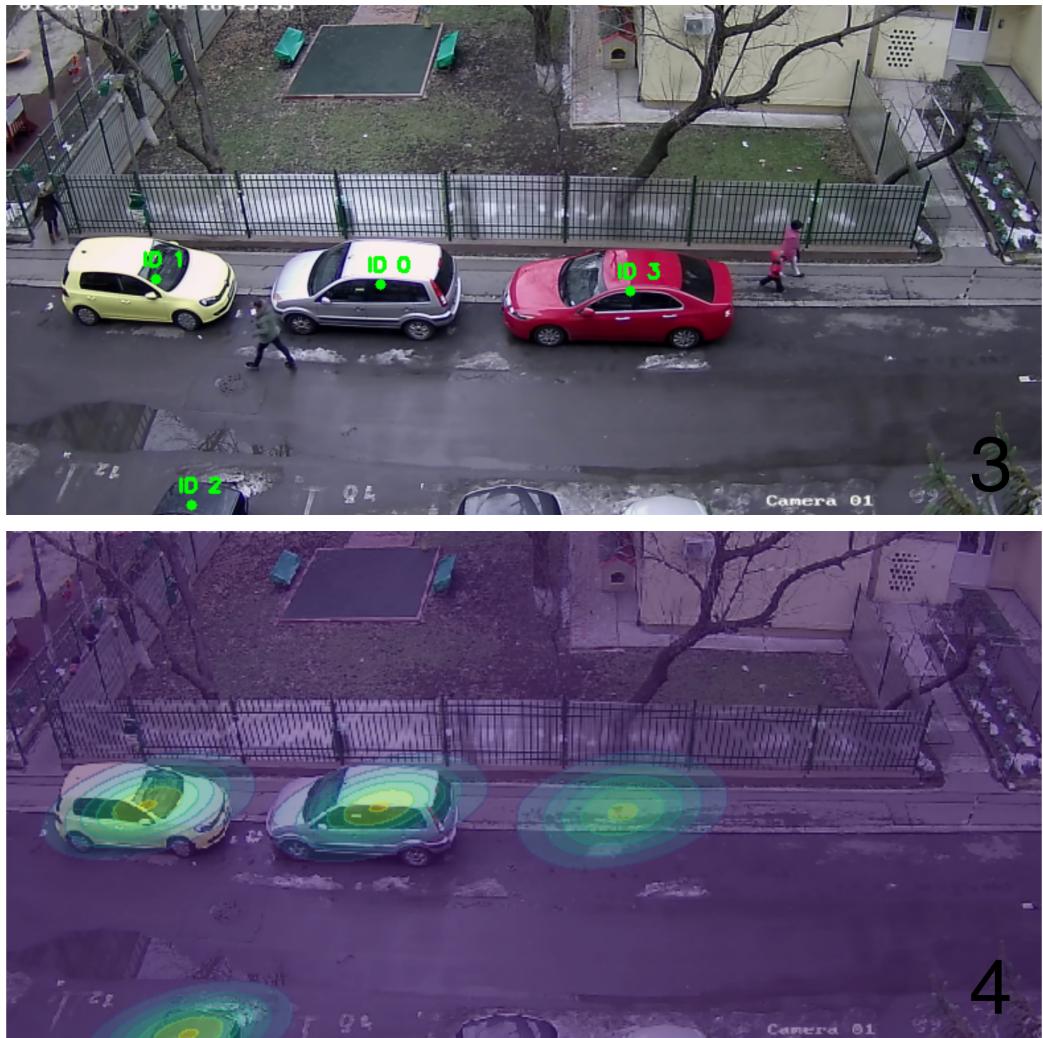


Figura 9: Simulação para carros

6 Resultados

Após o desenvolvimento do projeto verificou-se que este atende seus resultados esperados, gerando de forma satisfatória um mapa de calor de diversos ambientes (não se restringindo apenas a lojas de varejo). O projeto não necessita de modificações significativas nos ambientes, pois processa gravações que podem ser fornecidas através do sistema de segurança do estabelecimento. Vale ressaltar que a angulação da câmera pode variar a qualidade do mapa de calor gerado, portanto recomenda-se uma análise do ângulo de gravação.

7 Sugestões para melhorias futuras

Após observar as dificuldades enfrentadas ao desenvolver o projeto, os aperfeiçoamentos futuros considerados possíveis de serem implementados são listados a seguir:

- Propõe-se desenvolver um algoritmo utilizando a biblioteca de reconhecimento facial do OpenCV para identificar as pessoas dentro do ambiente, a fim de distinguir funcionários de clientes.
- Implementação com mais câmeras resolveria o problema de oclusão, evitando a perda de dados quando uma pessoa passa muito próxima a outra.
- O armazenamento dos mapas de calor obtidos em nuvem de dados poderia ser implementado, facilitando assim o acesso aos resultados na maioria dos dispositivos eletrônicos e em qualquer local, visto que o resultado gerado é uma imagem.

8 Conclusão

Com a elaboração do projeto pode-se concluir que este atendeu os objetivos propostos na secção 4. Rastreando objetos em vídeo, capturando a frequência de posições de cada objeto, e por fim fornecendo um mapa de calor ambiente apresentado. Constatou-se ainda que o produto desenvolvido mostrou-se bastante flexível, podendo portanto utilizar filmagens já obtidas de ambientes físicos, isto dependerá exclusivamente da qualidade de gravação dos vídeos e do posicionamento das câmeras.

Com os testes observamos que o posicionamento da câmera deve ser de uma vista superior, semelhante ao posicionamento mais comum encontrado em câmeras de segurança, assim evita-se problemas de oclusão. Deve-se ter a filmagem somente com regiões equivalentes em importância, pois se uma região como entrada da loja ou caixa de pagamentos estiver no vídeo de análise de seções de produtos, irá interferir no mapa de calor pois são regiões de grande fluxo de pessoas.

O teste realizado com outras classes como de carros ("car"), evidênciava várias possibilidades de aplicação desse gerador de mapas de calor.

9 Referências

DATASCIENCEACADEMY. Blog. **O que é visão computacional.** Disponível/ em:<samehttp://datascienceacademy.com.br/blog/o-que-e-visao-computacional/> Acesso em: 20 ago. 2018.

OPENCV (OPEN SOURCE COMPUTER VISION LIBRARY). **OPEN SOURCE COMPUTER VISION LIBRARY.** Disponível em:<samehttps://opencv.org> Acesso em: 20 ago. 2018.

PYCHARM. **PYTHON IDE FOR PROFISSIONAL DEVELOPERS.** Disponível em:<labelhttps://www.jetbrains.com/pycharm/> Acesso em: 20 ago. 2018.

LAPIX. Image Processing and Computer Graphics Lab. **Deep Learning::Detecção de Objetos em Imagens.** Disponível em: <samehttp://www.lapix.ufsc.br/ensino/visao-computacional/visao-computacionaldeep-learning/deteccao-de-objetos-em-imagens> Acesso em: 25 ago. 2018.

OPENCV (OPEN SOURCE COMPUTER VISION LIBRARY). **OPEN SOURCE COMPUTER VISION LIBRARY.** Disponível em:<samehttps://opencv.org> Acesso em: 20 ago. 2018.

BEHAVIOR ANALYTICSRETAIL. **People Tracking: 15 Technologies in 2018.** Disponível em:<samehttps://behavioranalyticsretail.com/technologies-tracking-people/> Acesso em: 01 set. 2018.

PYCHARM (OPENCV PEOPLE COUNTER). **OpenCV People Counter.** Disponível em:<samehttps://www.pyimagesearch.com/2018/08/13/opencv-people-counter/> Acesso em: 15 set. 2018.

STACK OVERFLOW. **Density map (heatmaps) in matplotlib.** Disponível em:<samehttps://stackoverflow.com/questions/36957149/density-map-heatmaps-in-matplotlib> Acesso em: 15 set. 2018.