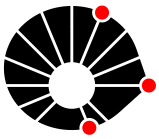
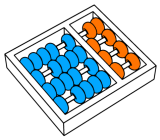




Dimensionamento de filas e fluxos de restaurantes universitários utilizando IoT na UNICAMP

Raphael Pontes Santana *Willian Tadeu Beltrão*
Juliana Freitag Borin

Relatório Técnico - IC-PFG-19-51
Projeto Final de Graduação
2019 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Dimensionamento de filas e fluxos de restaurantes universitários utilizando IoT na UNICAMP

Raphael Pontes Santana Willian Tadeu Beltrão Juliana Freitag Borin

11/12/2019

Abstract

This work proposes a low-cost Internet of Things solution to estimate the size of queues and flow of people at the entrance of university restaurants, especially the administrative restaurant at UNICAMP. For this purpose, sensors were installed at the entrance of the restaurant to determine the flow and images captured by a camera were analyzed to determine the queue size. The entire solution has been integrated into the Internet allowing easy access for the academic community. This work is within the scope of the Smart Campus UNICAMP project.

Resumo

Este trabalho tem como intuito o dimensionamento de filas e fluxo de pessoas nos restaurantes universitários, em especial o restaurante administrativo da UNICAMP, utilizando uma solução de baixo custo baseada em Internet das Coisas. Para isso, foi realizado sensoriamento em catracas para determinação do fluxo e análise de imagens capturadas por uma câmera para determinar o tamanho da fila. Toda a solução foi integrada a Internet possibilitando o fácil acesso para a comunidade acadêmica. Este trabalho está dentro do escopo do projeto do Smart Campus UNICAMP.

1 Introdução

O problema de contagem de pessoas é um problema antigo e que possui grande aplicação para as mais diferentes áreas. No contexto da universidade, existem os restaurantes universitários que possuem grandes fluxos diário de pessoas, sendo a contagem de pessoas em suas filas e fluxos um importante dado para comunidade acadêmica, seja para o planejamento dos alunos e funcionários para realização da sua refeição, seja para estimação de demanda para refeições.

Na Universidade Estadual de Campinas, UNICAMP, são servidas diariamente mais de 10 mil refeições [1], sendo que a comunidade acadêmica não possui informações sobre o tamanho da fila, o fluxo de pessoas do momento e a lotação do restaurante. Além disso, o controle de fluxo de pessoas que entram do restaurante é feito de maneira manual pelos funcionários dos restaurantes, assim como a quantidade de refeições feita por dia é estimada de maneira não automática com base em dados históricos coletados e armazenados em planilhas.

O controle de entrada no restaurante é realizado por um funcionário que observa dois fatores: se a fila interna para pegar a refeição está muito longa ou/e se não há mais lugares disponíveis para se sentar. Em ambos os casos, o funcionário controla o fluxo de entrada barrando a fila de pessoas que está na área externa ao restaurante e faz sua liberação quando há uma amenização da lotação.

Diante desse contexto, acredita-se que a coleta e disponibilização de informações sobre as filas e fluxos de entrada dos restaurantes universitários da UNICAMP poderia contribuir para o bem estar dos usuários bem como para um gerenciamento mais eficiente do ambiente e dos recursos. Pensando nisso, este trabalho procura soluções eficientes, de baixo custo, e que utilizam Internet das Coisas (IoT, do inglês, *Internet of Things*) para resolução deste problema. Ele também faz parte do projeto Smart Campus da UNICAMP, que tem como objetivo utilizar o conceito de Internet das Coisas na Unicamp de modo a obter informações para uma inteligência de controle mais eficiente e tomada de ações mais assertivas [2].

1.1 Objetivos

Este projeto tem como objetivo propor e desenvolver uma solução computacional para gerar informações sobre a fila e fluxos dos restaurantes universitários. Mais especificamente, espera-se:

1. desenvolver uma solução de baixo custo para dimensionar o tamanho da fila de entrada do restaurante administrativo da UNICAMP;
2. desenvolver uma solução de baixo custo para dimensionar o fluxo de entrada do restaurante administrativo da UNICAMP;
3. disponibilizar as soluções desenvolvidas para a administração da universidade.

2 Fundamentos Teóricos

Esta seção apresenta conceitos e tecnologias que foram necessários para o desenvolvimento deste projeto, a saber: princípio hall, protocolo MQTT e visão computacional.

2.1 Princípio Hall

O princípio do efeito hall é definido como surgimento de um diferencial de potencial ocasionado pela influência de um campo magnético. É melhor exemplificado na Figura 1, onde uma força eletromagnética é aplicada no sentido horizontal, criando uma força perpendicular magnética desviando a trajetória dos elétrons [3].

Baseado neste princípio, sensores magnéticos do tipo hall utilizam este conceito convertendo o sinal magnético em sinal elétrico, permitindo assim a leitura da variação de sinal magnético através de circuitos eletrônicos. A topologia do dispositivo hall é demonstrado na Figura 2.

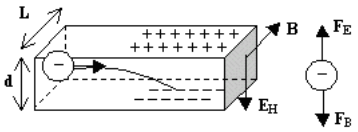


Figura 1: Efeito Hall

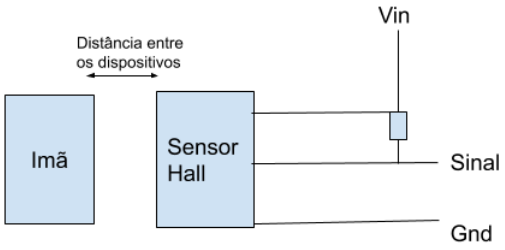


Figura 2: Topologia

A tensão de saída (ou sinal) do dispositivo é definida como:

$$V_h = K_h * I * B$$

Onde:

- K é a constante do dispositivo hall;
- I é a corrente; e
- B é a intensidade do campo magnético.

Portanto, a tensão de saída é proporcional ao campo magnético exposto ao sensor. Deste modo, sensores hall são excelentes para detecção de presença de imãs por causa do campo magnético. Além disso, são dispositivos pequenos, com baixo consumo de energia e possuem uma excelente precisão. Seus exemplos de uso são muitos, como por exemplo, contagem de rotação de motores, indicador de nível de combustível em automóveis e contagem de rotação de catraca. Este último exemplo será utilizado neste projeto para contagem de pessoas que entram em um restaurante através de catracas.

2.2 MQTT

O protocolo MQTT (*Message Queue Telemetry Transport*) foi criado pela IBM no final dos anos 90, e tinha como propósito vincular sensores em pipelines de petróleo e satélites [4].

Atualmente, o MQTT é um protocolo de referência para dispositivos IoT, visto que possui características como leveza no tamanho de mensagens em rede, e também por ser baseado no protocolo TCP/IP. Suas vantagens em relação a outro protocolo comum como o HTTP além de ser mais leve são a bidirecionalidade entre dispositivos, não necessitar iniciar

uma conexão e assincronia de comunicação. As trocas de mensagens são baseadas no modelo de publicação e assinatura onde se tem uma entidade *broker* e inúmeras entidades cliente que podem ser *publishers* e/ou *subscribers*. A Figura 3 mostra a arquitetura de um sistema que usa MQTT. O *broker* é o elemento responsável por receber as mensagens do *publisher* e encaminhá-las aos dispositivos *subscribers*. O *publisher* é um dispositivo que geralmente possui uma informação relevante, como dados coletados por sensores, e que publica essa informação através de tópicos no *broker*.

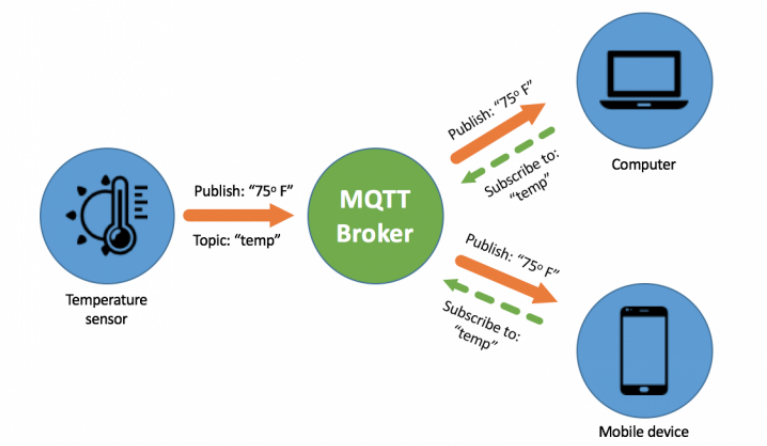


Figura 3: Protocolo MQTT

2.3 Visão computacional

A visão computacional é o processo da construção de um sistema visual artificial que tem como base o sistema visual humano. Ela passa pela aquisição, processamento e identificação de imagens e para isso utiliza conceitos de processamento de imagens, aprendizado de máquina, recuperação de informação e sistemas cognitivos de neurociência.

Através da visão computacional, é possível realizar algumas tarefas complexas, tais como:

- reconhecimento de padrões;
- reconstrução de cenários em 3D;
- análise de movimentos;
- identificação de objetos, como por exemplo, pessoas.

Para o reconhecimento de pessoas, existem técnicas de aprendizado de máquina, sendo um grande exemplo o uso de redes neurais como YOLO [5] e SSD [6].

You Only Look Once (YOLO) e *Single Shot Detector* (SSD) são modelos de deteção de objetos em tempo real que produzem informações a respeito da localização e tipo dos objetos presentes em uma imagem.

Esses modelos consistem basicamente de redes neurais CNNs (*Convolutional Neural Networks*) que a partir dos valores de pixel em uma imagem de entrada retornam como resultado valores preditivos acerca dos objetos. Estes informam a localização e dimensão de retângulos que contém os objetos detectados na imagem com a sua probabilidade de pertencerem às diferentes classes de objeto que podem ser detectadas pela rede neural.

Essas redes neurais possuem diversas versões, inclusive versões otimizadas para baixo consumo computacional. Para avaliação de modelos de redes neurais, utilizam diversas métricas, mas, em específico, para detecção de objetos em imagens, uma métrica importante a ser observada é a mAP.

A mAP (*mean Average Precision*) é uma métrica popular de acurácia utilizada para detectores de objetos como a YOLO e SSD. Esta medida utiliza em seu cálculo outras medidas como precisão, revocação e IoU (*Intersect over Union*), todas variando entre 0 e 1. A seguir está ilustrado um breve equacionamento dessas medidas [7].

$$Precisão = \frac{VP}{VP + FP}$$

$$Revocacão = \frac{VP}{VP + FN}$$

$$VP = verdadeiro\ positivo \quad FN = falso\ negativo$$

$$VN = verdadeiro\ negativo \quad FP = falso\ positivo$$

$$IoU = \frac{Área\ de\ União}{Área\ de\ Interseccão}$$

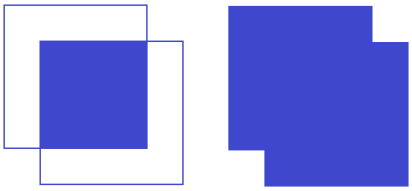


Figura 4: Área de intersecção e área de união

Em outras palavras, para as predições de retângulos na imagem, a precisão é o total de acertos sobre o total de predições, a revocação é o total de acertos sobre o total de objetos existentes e o IoU é a área de interseção sobre a área da união entre o retângulo predito e o verdadeiro que contém o objeto, como mostra a Figura 4. Dessa maneira, quanto mais próximas de 1 estiverem essas medidas, mais próximo de 1 estará o mAP, o que significa uma melhor acurácia. O IoU é utilizado para dizer se uma predição está correta ou não, ou seja, se atribuirmos que um acerto deve ter um IoU maior que 50%, teremos uma medida de mAP50. O cálculo completo do mAP envolve um procedimento um pouco mais rico em detalhes e será deixado como opção de leitura na referência [7].

3 Metodologia

3.1 Solução proposta

Para o problema apresentado, são propostas as seguintes abordagens:

1) Dimensionamento da fila

Utilização de uma câmera para a captura de imagens da fila e de um dispositivo embarcado contendo algoritmos de inteligência artificial para o processamento das imagens, obtendo-se assim dados sobre o tamanho da fila. Posteriormente, realizar o envio desses dados via rede WiFi para um servidor do qual poderá se ter a visualização dos dados, a sua análise e serem feitos outros processamentos.

Sabendo que algoritmos de inteligência artificial em um dispositivo embarcado possuirão uma certa imprecisão na contagem de pessoas, optou-se por definir a escala pequena, média e grande para o tamanho da fila a fim de que mensurá-la e facilitar a visualização da informação para o usuário.

2) Dimensionamento do fluxo

Utilização de sensores nas catracas do restaurante a fim de obter a quantidade de pessoas que entram em um determinado período de tempo. Tais sensores são acoplados a dispositivos como ESP8266 a fim de que possam tratar a informação adequadamente e enviar via rede WiFi. No servidor, as informações seriam tratadas e disponibilizadas para a comunidade acadêmica e o setor administrativo do restaurante universitário. Com o uso de tais tecnologias, a solução seria de baixo custo e alto desempenho, permitindo, ainda, cruzar as informações com o dimensionamento da fila para agregar mais valor ao usuário.

3.1.1 Topologia do projeto

A topologia da solução proposta é apresentada na Figura 5. Os sensores hall se comunicam com o ESP8266 e a câmera se comunica com a Raspberry Pi 3B+. Ambos os dispositivos enviam dados para a plataforma de IoT da Konker Labs [8] utilizando o protocolo MQTT. A partir da plataforma da Konker os dados são obtidos, tratados e disponibilizados para os usuários.

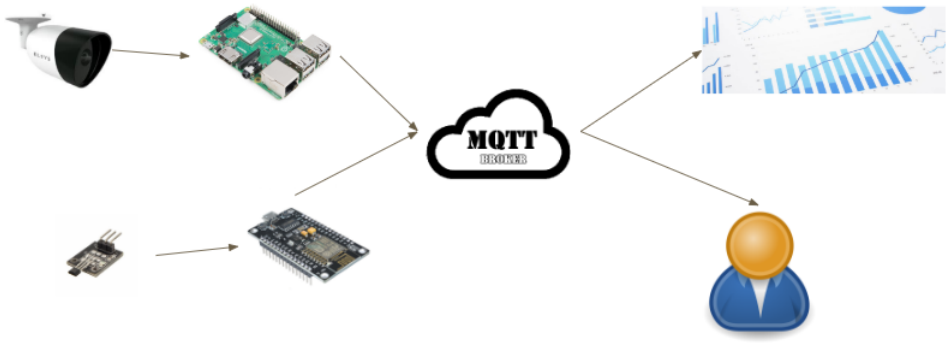


Figura 5: Topologia da solução proposta

3.2 Material

3.2.1 Software

O projeto utilizou como linguagens de programação C, C++ e Python no desenvolvimento das aplicações embarcadas para a estimativa do fluxo e tamanho da fila. Além disso, utilizou a YOLO como rede neural principal para a detecção de pessoas.

A ideia para fazer a estimativa do tamanho da fila externa do restaurante universitário se baseou na captação e processamento de imagens de câmera com o auxílio de um modelo de inteligência artificial para, então, gerar predições acerca das pessoas presentes na fila. Com base em um levantamento de diferentes modelos de inteligência artificial e levando em conta as limitações da Raspberry e a meta de uma detecção rápida, foram selecionados duas potenciais redes neurais, a YOLO e a SSD. Ambos os modelos possuem alguns pontos positivos como:

- estão entre os melhores da atualidade, contando com uma ótima acurácia para a detecção de pessoas;
- possuem uma versão leve, otimizada para baixo nível de processamento e menor uso de memória, ideal para nosso ambiente embarcado;
- não requerem o uso de bibliotecas mais complexas como Tensorflow ou Caffe, dispensando esforços com configurações de ambiente;
- compatíveis com a arquitetura ARM 32 bits utilizada em hardwares como a Raspberry Pi 3B+.

Visto isso, levando em conta que o foco do trabalho não era realizar comparações a fundo entre modelos de inteligência artificial e para não haver total duplicidade de experimentos, optou-se pelo uso de apenas um dos modelos, neste caso, a YOLO. Alguns fatores que levaram a essa escolha foram:

- seu desempenho se apresentava ser semelhante ao da SSD, logo ambos atenderiam bem ao nosso problema;
- havia prévio conhecimento do seu funcionamento por parte dos autores, facilitando assim seu uso e aumentando a produtividade no projeto em sua configuração e modificação;
- seu código fonte está em C e totalmente disponível, facilitando a realização de mudanças para atender às necessidades do trabalho;
- foi encontrada uma documentação maior de apoio para seu funcionamento e uso; e
- o tempo de processamento da Yolo Tiny foi mais que 2 vezes menor comparado com o da SSD na Raspberry Pi 3 B+.

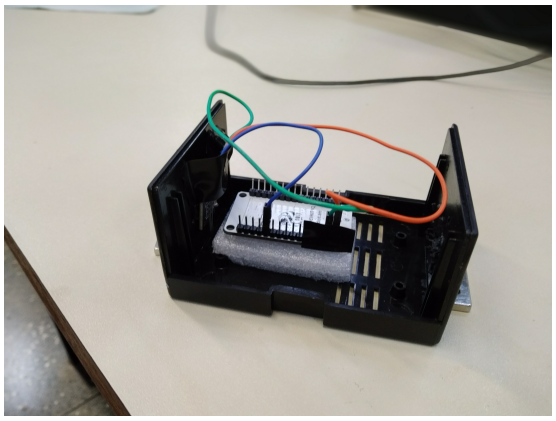
Apesar da escolha pela YOLO, ambas as redes foram instaladas e configuradas na Raspberry com o uso do sistema operacional Raspian. As versões utilizadas para os modelos foram a Yolov3 Tiny [9] e a SSD MobileNet v2 [10].

3.2.2 Hardware

Foram escolhidos os seguintes dispositivos para o projeto com base na restrição orçamentária e adequação para o perfeito funcionamento da solução:

- 3 módulos ESP8266 - módulo de alto desempenho com suporte WI-FI que permite coleta de dados de sensores e aciona atuadores.
- 4 módulos de sensor hall KY-003, que permitem distinguir a presença de um ímã de acordo com a distância;
- 6 ímãs de neodímio, que possuem grande força de atração e que podem ser utilizados para detecção na rotação de catracas;
- 1 RaspberryPi 3 B+, dispositivo que possui alto desempenho e permite processamento mais elevado como esperado em algoritmos de processamento de imagem e redes neurais;
- 1 Câmera WebCam genérica USB com resolução de 1280x960 que permite testes básicos com fotografias da fila; e
- 1 Câmera Elsys externa, câmera com resolução de 1920x1080 que foi instalada na fase final do projeto pela prefeitura do Campus no poste em frente ao restaurante para dimensionamento da fila.

A Figura 6 mostra alguns dos dispositivos utilizados.



(a) ESP8266 e Sensor Hall

(b) Ímas

(c) Raspberry Pi 3B+

(d) PowerBank

Figura 6: Imagens de alguns dispositivos utilizados

3.3 Método

A parte prática do trabalho foi dividida nas três seguintes etapas:

1) Coleta de dados e configurações

Inicialmente, foram realizadas coletas de imagens com câmeras de celular e uma câmera externa USB, a qual podia ser utilizada e conectada com a Raspberry. Essas imagens foram utilizadas como entrada para a construção do algoritmo de predição do tamanho da fila e sua implementação embarcada na Raspberry.

Com a utilização de um framework [11] e a partir de algumas imagens coletadas da fila do restaurante, foi realizada sua anotação manual marcando-se as pessoas presentes de modo a ser gerada uma base de dados experimental com 100% de acurácia. Comparando as anotações com as predições feitas pelos algoritmos, pode-se medir e comparar seus desempenhos.

Com o auxílio de um tutorial [12], também foi realizado um novo treino da rede neural para identificar apenas a classe de pessoas com o objetivo de melhorar a acurácia. A base de dados utilizada para o treinamento foi a COCO 2014 [13] da qual foram filtradas apenas as imagens que continham pessoas. Como a rede neural inicial identificava inúmeras classes desnecessárias para o problema da fila, com o mesmo tamanho da rede e um número menor de classes, esperava-se obter melhor desempenho na predição com a nova estrutura, o que de fato aconteceu após o novo treino como será mostrado na secção dos resultados.

Os dados resultantes das predições como quantidade de pessoas e tamanho da fila eram então enviados pela Raspberry para a plataforma da Konker Labs para armazenamento e pós processamento.

Na parte dos sensores, foi selecionado o sensor hall para testes in loco e configurado para o uso nos dispositivos ESP8266 e ESP32. Também foi testado e configurado seu uso para transmissão de dados via WiFi e comunicação com a plataforma Konker Labs. Para alimentação utilizou-se um PowerBank ou uma fonte DC 5V.

2) Testes e validação do sistema

O sistema implementado em laboratório foi então levado para realizar testes reais em campo. Com a câmera externa USB ligada a Raspberry realizou-se a coleta de imagens da fila do restaurante, seu processamento embarcado em tempo real e envio dos resultados para a plataforma na nuvem.

Da mesma forma, os dispositivos ESP8266 com sensor hall foram levados para testes nas catracas utilizando um PowerBank para manter seu nível de energia. O envio de dados por parte do ESP8266 era feito a cada 30 segundos via rede WiFi local. Os dados posteriormente puderam ser extraídos da plataforma para verificação do funcionamento do sistema.

Foram instalados sensores hall nas catracas e 3 imãs de neodímio em três posições estratégicas diferentes de cada catraca que permitiam o acionamento do sensor sempre que houvesse a mudança de posição na rotação da catraca, contabilizando assim a entrada de uma nova pessoa, como é possível ver na Figura 7. Os dispositivos e imãs foram fixados com o auxílio de fita dupla face.

Figura 7: Sensor instalado na catraca

3) Novas coletas e otimização do sistema.

Após a validação do sistema na prática e com a instalação de uma câmera fixa no local do restaurante ligada a internet, puderam ser feitas coletas adicionais de imagens. A partir destas foram realizadas melhorias no algoritmo de predição como a definição da região de interesse para a fila nas imagens e os critérios para a classificação da fila entre pequena, média e grande.

Além disso, dois experimentos foram feitos na prática utilizando sensores em catracas, com a montagem já mencionado nas etapas anteriores, para coleta de informações a respeito do fluxo do restaurante administrativo em dois dias distintos no período do almoço.

4 Resultados e discussão

4.1 Dimensionamento da fila

Na elaboração dos resultados para o problema de dimensionamento da fila, foram escolhidas 4 imagens de uma câmera de posição fixa contemplando diferentes cenários de fila, como pequena, média, grande e noturna. O uso da câmera fixa diminui a variação ambiente das imagens o que facilita sua análise e extração de informações úteis. Os diferentes cenários trazem uma riqueza maior de dados, permitindo a elaboração de modelos computacionais mais robustos o que por fim resulta em melhores conclusões. A seguir encontram-se as imagens citadas já processadas, ou seja, com a marcação da localização de pessoas detectadas pela rede neural para a realização da estimativa do tamanho da fila.

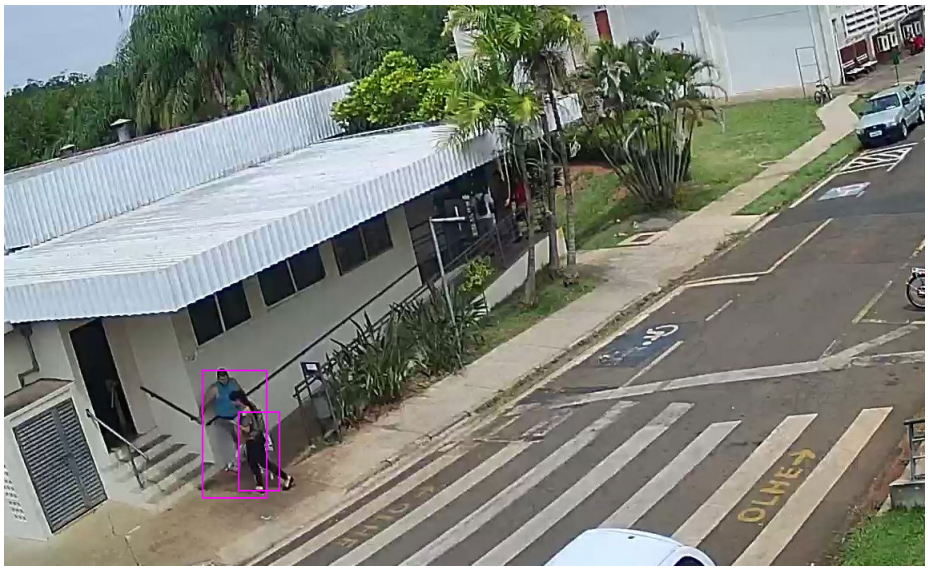


Figura 8: Fila pequena

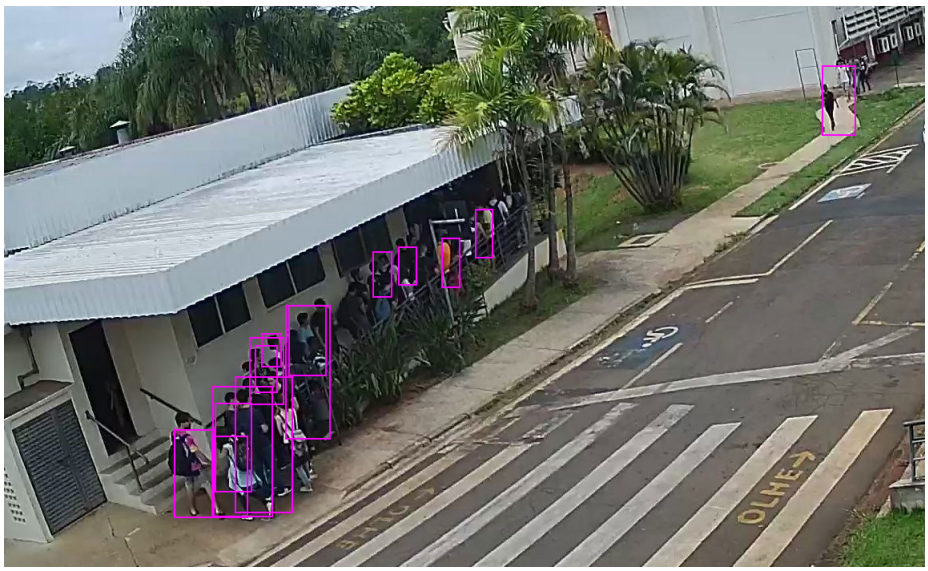


Figura 9: Fila média

Figura 10: Fila grande

Figura 11: Fila noturna

A partir da imagens, pode-se observar que a detecção de pessoas realizada pela rede neural não é perfeita, pois há muitas pessoas que não foram localizadas. No entanto, também pode ser visto que as detecções existentes estão praticamente corretas, bem distribuídas espacialmente e aumentam proporcionalmente com número real de pessoas. Desse modo, ainda é possível realizar a inferência da localização e tamanho da fila. A Tabela 1 apresenta a predição do número de pessoas nas imagens pela rede neural da YOLO em suas versões tiny, tiny com x iterações de retreino, original, a rede neural SSD MobileNet e o número real de pessoas.

Tabela 1: Número de Pessoas

Modelo	Fila Pequena	Fila Média	Fila Grande	Fila Noite
Yolov3 Tiny	3	7	30	8
Yolov3 Tiny 10000	1	0	1	1
Yolov3 Tiny 20000	6	20	41	7
Yolov3 Tiny 41000	2	15	32	17
SSD MobileNet v2	3	12	43	13
Yolov3	3	26	39	16
Real	5	34	62	17

A tabela mostra que, de fato, o número de pessoas detectadas é quase sempre menor que o real e que ocorre o crescimento de detecções proporcionalmente ao número real. Vejamos isso também através de gráficos de barras para melhor comparação nas Figuras 12 e 13.

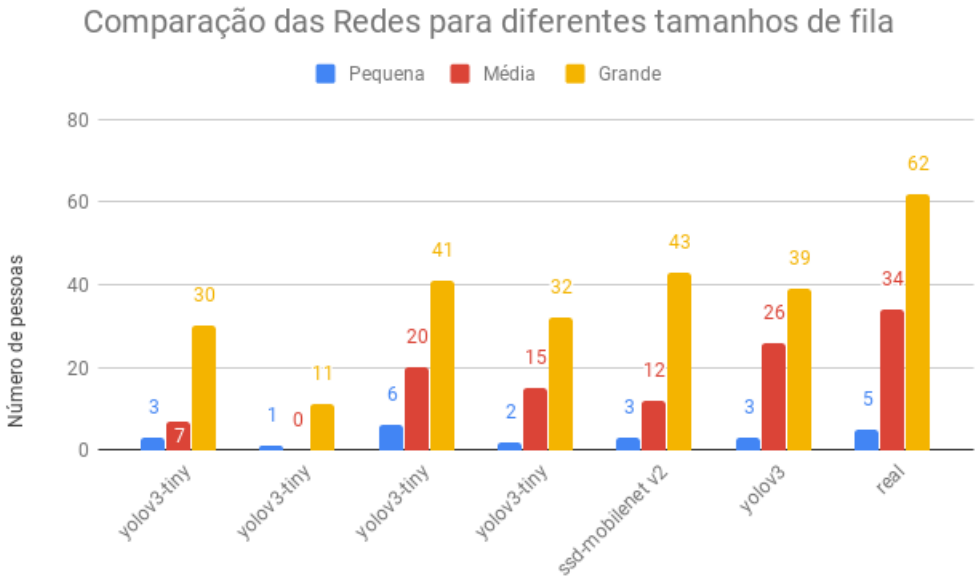


Figura 12: Comparação das redes para diferentes tamanhos de fila

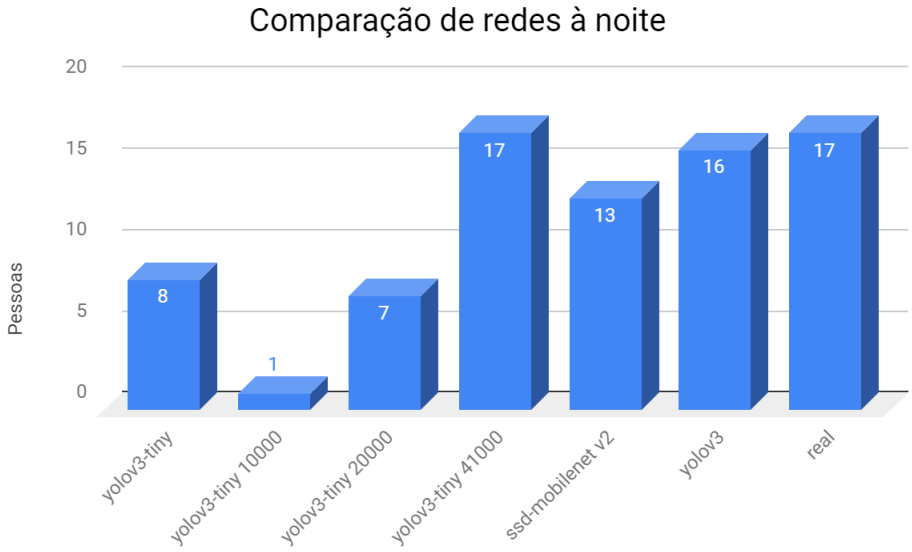


Figura 13: Comparação das redes à noite

Para uma análise mais detalhada das predições, a seguir está a Tabela 2 com as medidas de acurácia mAP relacionadas aos dados da Tabela 1 anterior, que foram calculadas com um script em Python [14].

Tabela 2: Mean Average Precision

Modelo	mAP25	mAP50	mAP75	Avg. Time (s)
Yolov3 Tiny	8,98%	0,00%	0,00%	1,117
Yolov3 Tiny 10000	8,44%	3,15%	0,00%	1,117
Yolov3 Tiny 20000	39,07%	9,57%	0,06%	1,117
Yolov3 Tiny 41000	37,84%	10,28%	0,00%	1,117
SSD MobileNet v2	38,03%	11,56%	0,11%	2,743
Yolov3	58,49%	44,76%	11,54%	———
Real	100%	100%	100%	———

A partir da tabela, valem ser ressaltadas algumas conclusões. A rede original tiny possui uma acurácia ruim, podendo ser levado em conta que esta não está otimizada para o problema da localização de pessoas. A rede tiny em seu retreino melhorou sua acurácia com o aumento das iterações até atingir uma ponto de estabilização com acurácia bem acima da tiny original. A rede SSD demonstra uma boa acurácia semelhante a rede retreinada, porém possui um tempo de predição cerca de 2.5 vezes maior. A rede original completa possui a melhor acurácia como esperado, porém não tem viabilidade para uso com a Raspberry Pi 3.

Em um dos experimentos, com uma série de imagens coletadas em um horário de almoço em intervalos de um minuto, realizou-se um comparativo do número de pessoas e tamanho da fila preditos pelo modelo e seus valores reais. Os resultados foram colocados para visualização no gráfico da Figura 14.

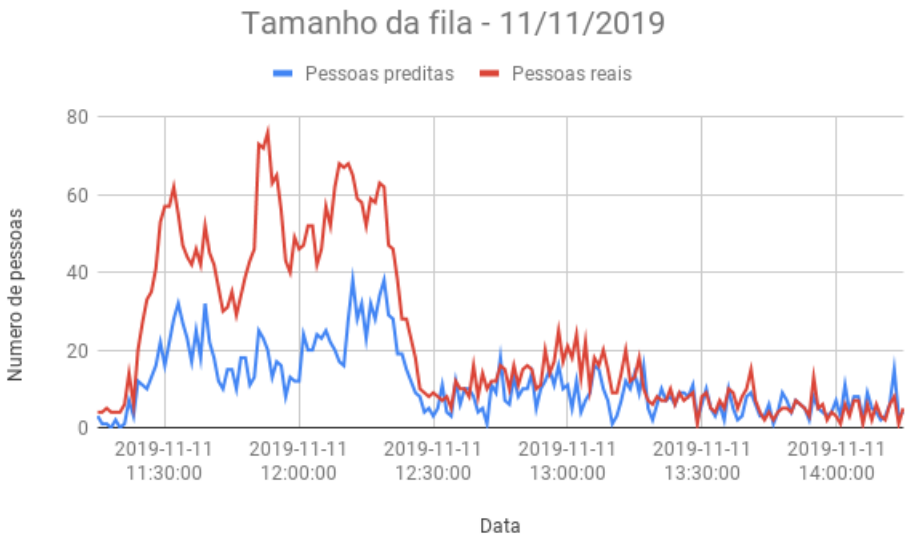


Figura 14: Tamanho da fila predito

Coerentemente com o que foi visto até agora e de acordo com as escalas das curvas e suas formas semelhantes, os números gerados pelo modelo são inferiores e proporcionais ao número real. Um fenômeno comum que se nota pela curva é um tamanho maior para a fila nos horários mais usuais de almoço, próximos ao meio dia, seguido por uma diminuição posterior da fila. Outro fenômeno comum que pode ser observado na prática e se reflete no gráfico é o pico no horário de abertura do restaurante às 11:30, pois normalmente as pessoas que chegam antes deste horário acabam por se acumular na fila. Como os números dos gráficos acima não possuem um significado de valor prático para um usuário que queira saber como está a fila do restaurante, foi definido empiricamente um critério simples para a divisão dos números em fila pequena, média e grande. Os resultados estão a seguir.

$$1 - Fila\ pequena: \ 0 < numero\ de\ pessoas <= 10$$

$$2 - Fila\ media: 10 \ < numero\ de\ pessoas <= 20$$

$$3 - Fila\ grande: \ 20 < numero\ de\ pessoas$$

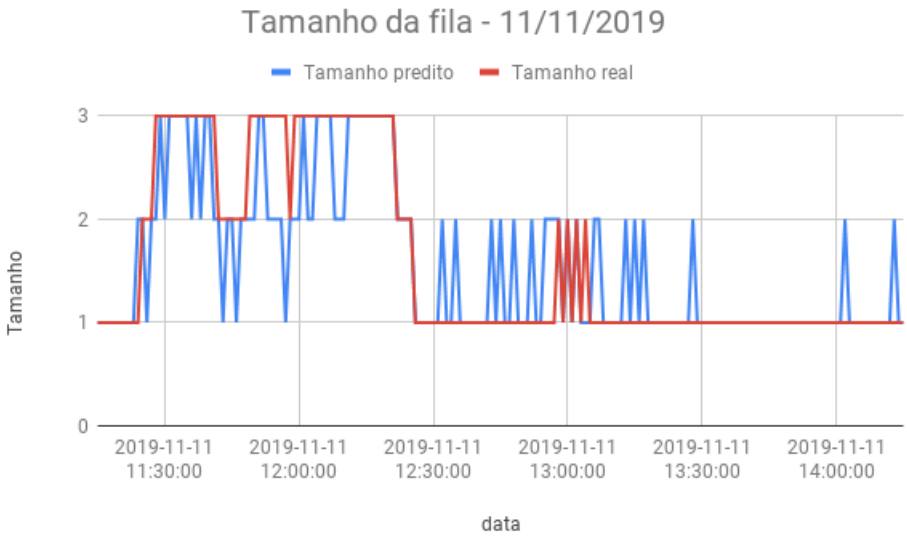


Figura 15: Tamanho da fila (1 - pequena, 2 - média, 3 - grande)

Por mais simples que seja o critério, pode-se ver no gráfico que este conseguiu fazer uma boa separação qualitativa inicial para o tamanho da fila, pois a curva apresenta uma boa distribuição entre os 3 estados e é condizente com os gráficos quantitativos anteriores.

4.2 Fluxo

Os resultados apresentados no fluxo das catracas foram coletados a partir dos sensores hall em que mediam a mudança da posição de rotação da catraca durante determinado período. Assim, baseado no número de rotações é possível saber quantas pessoas entraram, assim como a quantidade de pessoas que entram a cada determinado período de tempo, obtendo portanto o fluxo de pessoas de entrada na catraca. Os dados eram obtidos no ESP8266 e enviados via protocolo MQTT através da rede WiFi para o *broker* da plataforma Konker Labs. Os dados foram dispostos da seguinte maneira:

sensor	time	people	difference

- Sensor: A identificação do sensor utilizado, visto que é possível utilizar um sensor para cada catraca.
- Time: Horário em que o dado foi enviado para a plataforma
- People: Quantidade de pessoas contadas até o momento
- Difference: A diferença de pessoas para o último envio de dados

A utilização dos sensores hall se mostrou extremamente efetivo, visto que sua precisão foi praticamente de 100% na contagem de pessoas através da catraca, que foi atestada comparando com contagem manual da entrada de pessoas. A partir dos dados coletados, construiu-se gráfico para dois dias diferentes para o período do almoço como é possível ver abaixo:

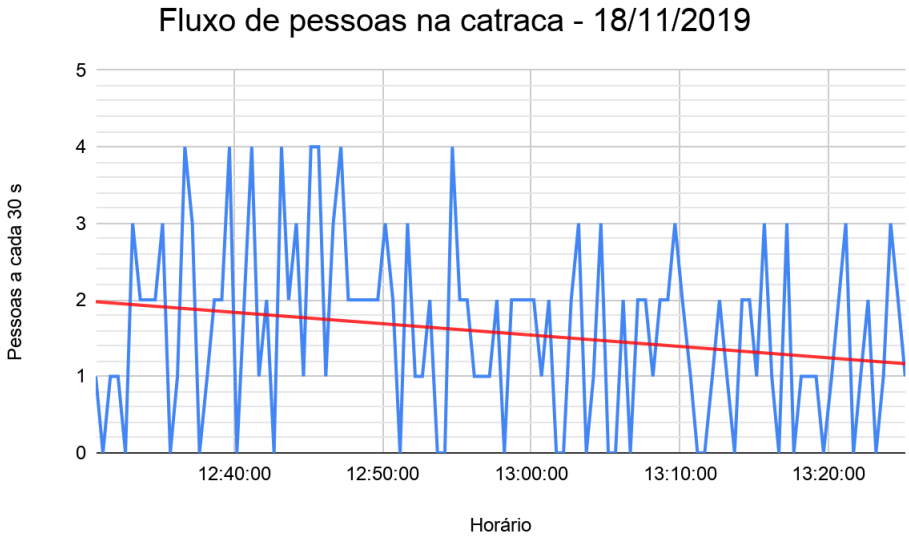


Figura 16: Fluxo de pessoas na catraca - 18/11/2019

Figura 17: Fluxo de pessoas na catraca - 25/11/2019

É possível notar que os padrões se repetem apesar dos dias serem diferentes e possuírem cardápio diferente, para o período das 12:30 às 13:25 o fluxo máximo de pessoas a cada 30 segundos é no valor de 4 pessoas. É possível notar apenas uma pequena variação no segundo dia, 25/11/19, que é refletida no fluxo médio devido a maior demanda no período das 13h.

Dia	Fluxo médio por catraca(12:30-13:25)
18/11/19	1,57 pessoas a cada 30s
25/11/19	1,95 pessoas a cada 30s

Como é um conjunto discreto, é interessante também obter o valor da moda:

Dia	Fluxo médio por catraca(12:30-13:25)
18/11/19	2 pessoas a cada 30s
25/11/19	2 pessoas a cada 30s

Chegando a valores congruentes não importando o dia para o mesmo horário. Também foi possível construir um gráfico com período maior de uma refeição do almoço, verificando que a maior demanda se dá na abertura do restaurante às 11:30 como é possível ver abaixo:

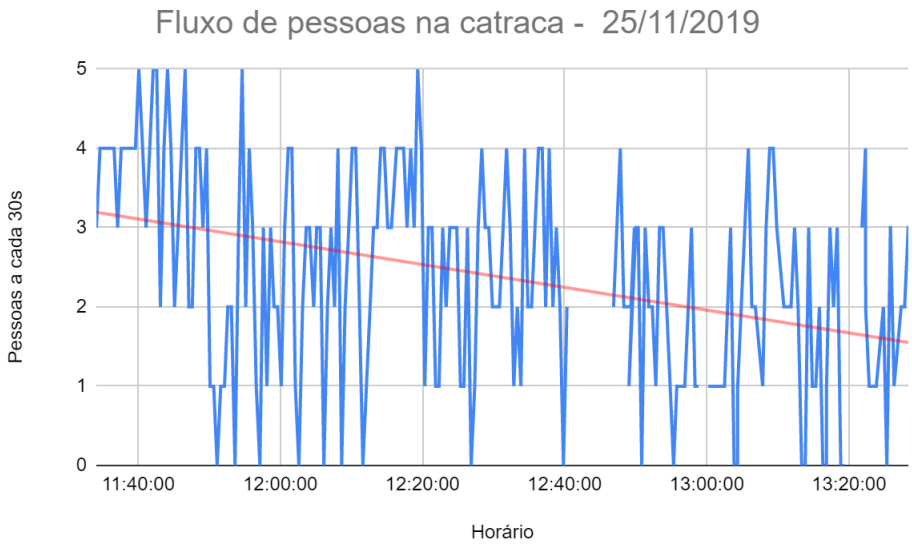


Figura 18: Fluxo de pessoas na catraca - 25/11/2019

O fluxo máximo de pessoas durante o período de 11:30 às 13:30 foi de 5 pessoas a cada 30 segundos. O valor médio para o período foi de 2,41 pessoas a cada 30 segundos. Esses dados quando utilizados juntos com o tamanho da fila, é possível estimar o tempo de espera da fila.

Vejamos agora o número de pessoas totais acumulada em uma catraca :

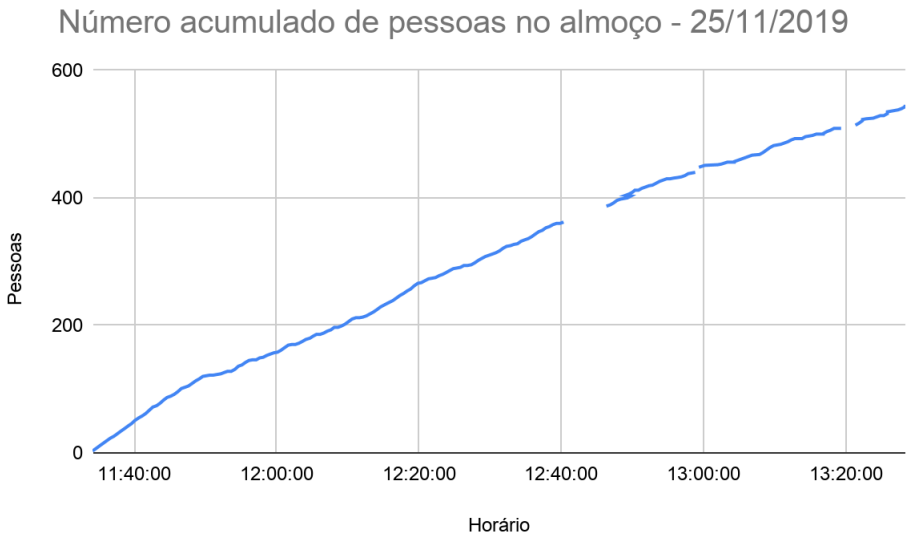


Figura 19: Número acumulado de pessoas no almoço - 25/11/2019

Assim como no gráfico do fluxo de pessoas, é possível ver que a entrada de pessoas é maior no período inicial de abertura do restaurante, visto que o coeficiente angular da curva é maior nesta região. Também é interessante notar que o número total de pessoas que entram por uma catraca chega a quase 600 pessoas, e como são duas catracas no restaurante com fluxos similares, mostra que a demanda na refeição do almoço ultrapassa a 1000 pessoas. Há pequenos buracos no gráfico devido a falha na rede para transmissão de dados, que pode ser sanado em trabalho futuro através da *bufferização* dos dados com seu respectivo horário.

5 Conclusão

Neste trabalho foi possível dimensionar e analisar filas e fluxos de restaurante universitários, em específico do restaurante administrativo da UNICAMP. Mostrou-se que utilizando dispositivos embarcados e IoT é possível agregar valor para comunidade acadêmica com informações sobre o fluxo e a fila em cada horário para o efetivo planejamento dos frequentadores do restaurante.

Além disso, foi possível estabelecer algumas métricas qualitativas para o tamanho da fila que poderão ser úteis para o usuário, como a escala pequeno, médio e grande. Assim como foi possível estabelecer o fluxo, os picos característicos no fluxo e na fila, e a possibilidade de contagem de pessoas que entram no restaurante universitário. É de se destacar que o projeto teve grande exatidão para o fluxo de pessoas na catraca, e possuindo bons resultados para o tamanho da fila que permitem que projeto possar ser colocado em prática nos restaurantes universitários da UNICAMP.

O repositório de software do projeto se encontra em [16].

6 Trabalhos Futuros

6.1 Visão Computacional

Para a parte de visão computacional foram realizados alguns experimentos que não foram explorados a fundo devido a limitações de hardware da Raspberry Pi 3 e não estar exatamente no escopo do projeto, no entanto estes podem ser uma ideias de trabalho para outros equipamentos como a Raspberry pi 4. A seguir estão algumas imagens dos experimentos processadas em um computador pessoal, que possui maior capacidade de processamento e memória que a Raspberry utilizada no projeto.

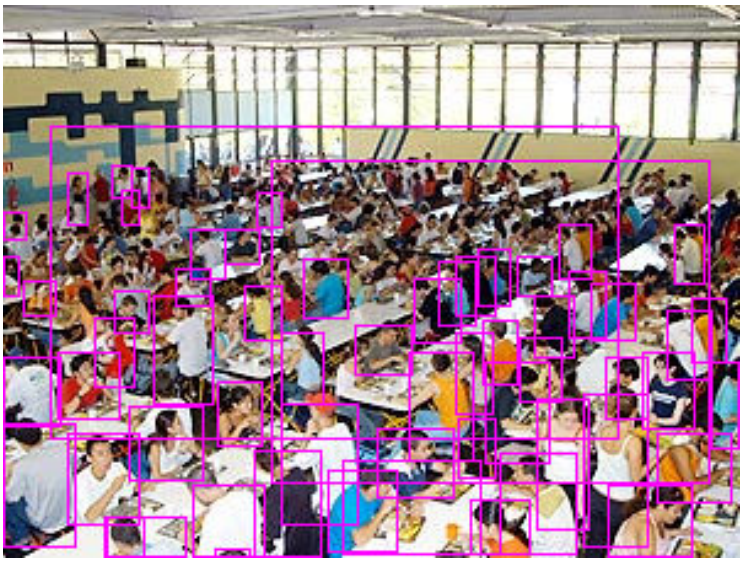


Figura 20: Lotação interna do restaurante

Figura 21: Fluxo da fila externa do restaurante

Figura 22: Fluxo de pessoas na catraca

A Figura 20 mostra o interior de um restaurante universitário com a detecção de pessoas feita pela rede neural. A ideia para este experimento seria realizar uma predição semelhante ao problema da fila trabalhado só que para o caso da lotação interna do restaurante, classificando-a em estados de vazio até cheio. Esta informação poderia ser útil para o controle de entrada do restaurante. As Figuras 21 e 22 mostram também a detecção de pessoas pela rede neural porém com o diferencial do adicional de um algoritmo, o Deep Sort [15], que atribui um ID para cada pessoa detectada e pode realizar o seu rastreamento entre imagens. No caso da Figura 21, isto poderia ser utilizado como uma maneira alternativa de realizar a contagem de pessoas que entram no restaurante, o que poderia ser útil por exemplo em uma situação onde não se tem a presença de catracas, e dificultando assim a implementação de sensores de alta precisão. No caso da Figura 22, isto poderia ser utilizado por exemplo para estimar o deslocamento temporal da fila e predizer seu tempo total.

6.2 Sensores

Em outra ideia um pouco mais elaborada, que poderia ter sido implementada com os sensores hall, estes poderiam ser posicionados em todas as catracas de entrada e saída do restaurante. Desse modo, por meio da intercomunicação entre os sensores ou do envio de seus dados para um servidor comum poderia ser calculado com boa precisão o número de pessoas no interior do restaurante e consequentemente sua lotação. Isso permitiria o controle de fluxo nas catracas de modo até mesmo automático. Além disso, esse dado também poderia ser disponibilizado para o usuário, seja através de painéis eletrônicos com a quantidade de pessoas presentes no restaurante no momento, como por meio de aplicativos, que além de poderem indicar a quantidade de fluxo e tamanho de fila em tempo real, poderia indicar a lotação do restaurante no momento, para que o planejamento do usuário fosse feita de maneira mais proveitosa possível.

O grande diferencial desta ideia em comparação com a utilização de câmeras e inteligência artificial, é o seu custo reduzido e algoritmos de menor complexidade.

Figura 23: Contador de pessoas

7 Agradecimentos

Agradecemos a cada uma das pessoas que nos ajudou de maneira direta e indiretamente para conclusão deste projeto. Algumas pessoas foram fundamentais para a condução do projeto como a professora Juliana Freitag Borin que não mediu esforços nos orientando e disponibilizando material para o projeto.

O apoio das equipes da Prefeitura da UNICAMP, do Restaurante Administrativo da UNICAMP e da Konker Labs, e da Equipe Phoenix de Robótica da UNICAMP foram essenciais para a implementação prática do projeto, seja através do fornecimento de materiais, seja pelo compartilhamento de conhecimento e orientações.

Referências

[1] Carla Cusihuallpa, Diogo Moreira, Lahis de Almeida, Raphael Santana. Analisando e Predizendo a Demanda Diária em Restaurantes Universitários. Disponível em: https://smartcampus.prefeitura.unicamp.br/pub/artigos_relatorios/Lahis-Trabalho_final_de_Aprendizado_de_Maquina.pdf. Acesso em: 11 de dezembro de 2019.

[2] SMARTCAMPUS. Disponível em: https://smartcampus.prefeitura.unicamp.br. Acesso em: 11 de dezembro de 2019.

[3] UFRGS. O Efeito Hall. Disponível em: https://www.if.ufrgs.br/tex/fis142/mod08/m_s03.html. Acesso em: 11 de dezembro de 2019.

[4] YUAN, Michael. Por que o MQTT é um dos melhores protocolos de rede para a Internet das Coisas?. Disponível em: https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html. Acesso em: 11 de dezembro de 2019.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. Disponível em: https://arxiv.org/abs/1506.02640. Acesso em: 11 de dezembro de 2019.

[6] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. SSD: Single Shot MultiBox Detector. Disponível em: https://arxiv.org/abs/1512.02325. Acesso em: 11 de dezembro de 2019.

[7] HUI, Jonathan. mAP (mean Average Precision) for Object Detection. Disponível em: https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173. Acesso em: 11 de dezembro de 2019.

[8] KONKERLABS. Disponível em: https://www.konkerlabs.com/. Acesso em: 11 de dezembro de 2019.

[9] ZHAI, Xiang. Deep Learning with Raspberry Pi – Real-time object detection with YOLO v3 Tiny!. Disponível em: http://funofdiy.blogspot.com/2018/08/deep-learning-with-raspberry-pi-real.html. Acesso em: 11 de dezembro de 2019.

[10] ROY, Saumya. Real-Time Object Detection on Raspberry Pi Using OpenCV DNN and MobileNet-SSD. Disponível em: https://heartbeat.fritz.ai/real-time-object-detection-on-raspberry-pi-using-opencv-dnn-98827255fa60. Acesso em: 11 de dezembro de 2019.

[11] GitHub. Ybat - YOLO BBox Annotation Tool. Disponível em: https://github.com/drainingsun/ybat. Acesso em: 11 de dezembro de 2019.

[12] REDMON, Joseph; FARHADI, Ali. YOLOv3: An Incremental Improvement. Disponível em: https://pjreddie.com/darknet/yolo/. Acesso em: 11 de dezembro de 2019.

[13] COCODATASET. Common Objects in Context. Disponível em: http://cocodataset.org/#home. Acesso em: 11 de dezembro de 2019.

[14] CARTUCHO, João. mAP (mean Average Precision). Disponível em: https://github.com/Cartucho/mAP#create-the-ground-truth-files. Acesso em: 11 de dezembro de 2019.

[15] SAGAR, Abhinav. Pedestrian Tracking in Real-Time Using Yolov3. Disponível em: https://towardsdatascience.com/pedestrian-tracking-in-real-time-using-yolov3-33439125efdf. Acesso em: 11 de dezembro de 2019.

[16] Repositório Restaurant Flow Analysis - https://github.com/Raphpontes/restaurant-flow-analysis Acesso em: 12 de dezembro de 2019.