CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

João Vitor dos Santos Amorim

DETECÇÃO DE AÇÕES SUSPEITAS EM AMBIENTES COMERCIAIS UTILIZANDO VISÃO COMPUTACIONAL COM *YOLOV8*

Timóteo

2024

João Vitor dos Santos Amorim

DETECÇÃO DE AÇÕES SUSPEITAS EM AMBIENTES COMERCIAIS UTILIZANDO VISÃO COMPUTACIONAL COM *YOLOV8*

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Lucas Pantuza Amorim

Timóteo

2024

João Vitor dos Santos Amorim

Detecção de Ações Suspeitas em Ambientes Comerciais Utilizando Visão Computacional com YOLOv8

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, campus Timóteo, como requisito parcial para obtenção do título de Engenheiro de Computação.

Trabalho aprovado. Timóteo, 12 de fevereiro de 2025.

_____

Prof. Dr. Lucas Pantuza Amorim
Orientador

_____

Prof. Dr. Bruno Rodrigues Silva
Professor Convidado

_____

Prof. Me. Adilson Mendes Ricardo
Professor Convidado

Timóteo 2025

FOLHA DE APROVAÇÃO DE TCC Nº 5/2025 - DECOMTM (11.63.11)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 20/02/2025 14:44)
ADILSON MENDES RICARDO
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOMTM (11.63.11)
Matrícula: ###493#8

(Assinado digitalmente em 20/02/2025 19:54)
BRUNO RODRIGUES SILVA
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOMTM (11.63.11)
Matrícula: ###759#5

(Assinado digitalmente em 20/02/2025 15:57)
LUCAS PANTUZA AMORIM
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOMTM (11.63.11)
Matrícula: ###974#1

Visualize o documento original em https://sig.cefetmg.br/documentos/ informando seu número: ***5****, ano:* ***2025****, tipo:* ***FOLHA DE APROVAÇÃO DE TCC****, data de emissão:* ***20/02/2025*** *e o código de verificação:* ***76a443f06b***

Agradecimentos

Primeiramente, agradeço à minha família, em especial a meus pais Agnaldo e Denise, por sempre estarem me apoiando enquanto vou atrás das minhas pequenas e grandes conquistas. Agradeço ao meu professor orientador, Lucas Pantuza, não só como orientador, mais também como meu professor de Programação de computadores 1 que me ensinou admirar a programação desde o início do semestre.

A todos os outros professores que também passaram em minha história dentro do CEFET-MG. Que me ajudaram e ensinaram não só na formação como aluno, mas também com experiências de vida.

E gostaria finalmente de agradecer a todos os meus amigos, que me assistiram e estiveram comigo dentro desse tempo, seja dentro ou fora do CEFET. Me escutando me ajudando ou então tornando a experiência de passar pelo ensino superior, mais leve e menos desafiador. Uma menção honrosa a Karol Castelli que por mais de uma vez puxou minha orelha para que eu me dedicasse mais em minha formação nesses momentos finais.

Resumo

Este trabalho apresenta o desenvolvimento de um sistema de segurança baseado em visão computacional para detecção de furtos em ambientes comerciais. Utilizando o modelo YOLOv8 e a biblioteca OpenCV, o sistema foi projetado para identificar objetos e ações suspeitas, como o movimento de inserir itens em mochilas, em tempo real.

O método desenvolvido combina técnicas de aprendizado profundo e rastreamento de objetos para fornecer uma análise precisa, mesmo em cenários complexos. Durante os testes realizados, o sistema demonstrou uma taxa de acerto superior a 90% em condições controladas e boa robustez em ambientes reais, como supermercados. No entanto, limitações como iluminação inadequada e objetos fora do campo de visão foram observadas.

Como trabalhos futuros, propõe-se a integração do sistema em redes de câmeras e o aumento da base de dados para maior personalização e acurácia.

Palavras-chave: Visão computacional, YOLOv8, prevenção de furtos, deep learning.

Abstract

This paper presents the development of a computer vision-based security system for theft detection in commercial environments. Using the YOLOv8 model and the OpenCV library, the system is designed to identify objects and suspicious actions, such as inserting items into backpacks, in real time.

The proposed method combines deep learning techniques and object tracking to provide precise analysis, even in complex scenarios. During testing, the system achieved an accuracy rate exceeding 90% in controlled conditions and demonstrated robustness in real-world environments like supermarkets. However, limitations such as inadequate lighting and objects outside the camera's field of view were observed.

For future work, integrating the system into camera networks and expanding the dataset for increased accuracy and customization are proposed.

Keywords: Computer vision, YOLOv8, theft prevention, deep learning.

Sumário

1 Introdução

Diversas áreas, incluindo computação, psicologia e segurança, têm utilizado tecnologias emergentes como visão computacional para resolver problemas cotidianos e complexos (BARELLI, 2018). Essas tecnologias podem ajudar na criação de soluções sofisticadas para uma variedade de usos, como vigilância, prevenção de crimes e proteção de propriedades. O desenvolvimento da inteligência artificial e dos sistemas de reconhecimento de imagens tem feito com que as câmeras de segurança equipadas com visão computacional se tornem cada vez mais comuns. Isso chamou a atenção dos pesquisadores para descobrir como usar essas tecnologias para ajudar na segurança (FERREIRA, 2020).

A visão computacional é uma das tecnologias que tem sido muito utilizada presentemente, tornando computadores capazes de capturar, reconhecer e interpretar imagens (FERREIRA, 2020). A utilização dessa tecnologia em câmeras de segurança pode melhorar significativamente a proteção do ambiente, permitindo que o sistema identifique atividades suspeitas, como colocar objetos dentro de bolsas de maneira estranha, ou até mesmo o reconhecimento de armas de fogo entre outras atividades suspeitas.

O uso de câmeras com visão computacional é especialmente útil em locais com grande circulação de pessoas, como aeroportos, shoppings e escolas. Enquanto os humanos não conseguem monitorar vários lugares ao mesmo tempo, e podem cometer erros ou se cansar com o tempo, essa tecnologia pode oferecer uma solução eficaz para garantir a segurança nesses ambientes. (RAHANGDALE; KOKATE, 2016).

Um problema crescente no Brasil é o aumento dos furtos no setor varejista, especialmente em supermercados. Em 2023, esses furtos atingiram um nível recorde desde 2019, resultando em perdas estimadas em R$ 11 bilhões. Esse montante representa aproximadamente 0,5% do movimento total do varejo brasileiro, que alcançou R$ 2,23 trilhões (MATTOS, 2024). A escalada desses crimes tem levado empresas a investirem cada vez mais em tecnologias de monitoramento, como câmeras e sistemas eletrônicos de proteção, para identificar e deter atividades suspeitas. No entanto, apesar desses esforços, o desafio persiste, indicando a necessidade de estratégias mais abrangentes que combinem segurança tecnológica com políticas de prevenção e combate à criminalidade no varejo.

A visão computacional tem ganhado destaque em diversas áreas, e sua aplicação em sistemas de segurança aparece como uma solução promissora ao ampliar significativamente a capacidade de monitoramento e prevenção de diversos tipos de

crimes. No contexto do varejo, essa tecnologia pode se tornar uma aliada crucial na proteção de propriedades, ajudando a detectar comportamentos suspeitos em tempo real e, assim, reduzir os índices de furtos. Ao integrar inteligência artificial e análise de imagens, a visão computacional não só potencializa a eficiência dos sistemas de segurança, mas também promove um ambiente mais seguro para consumidores e comerciantes, contribuindo para a sustentabilidade e a confiança no setor varejista.

1.1 Justificativa e problema

A evolução contínua da tecnologia oferece novas oportunidades para inovar em uma variedade de setores, incluindo a segurança. Atualmente, programadores e desenvolvedores podem usar tecnologias que antes eram restritas a grandes corporações. Isso lhes permite desenvolver e administrar soluções complexas para problemas cotidianos.

O uso da visão computacional em sistemas de segurança é um exemplo desse avanço. Câmeras com algoritmos de reconhecimento de imagem avançados podem monitorar ambientes em tempo real e detectar atividades suspeitas, como dissimular a colocação de objetos dentro de bolsas. A análise desses dados em tempo real pode permitir uma resposta rápida e eficaz para prevenir furtos.

Essa tecnologia muda a maneira como os ambientes são monitorados e protegidos. A utilização de câmeras inteligentes pode ser um meio eficaz de mitigar os danos causados por pequenos furtos, que são uma parte significativa das perdas no varejo. Um sistema de câmeras que detectam automaticamente furtos de bolsas poderia aumentar a prevenção de crimes e também servir como um dissuasor para os infratores.

Uma solução desse tipo seria capaz de proteger os clientes e os funcionários ao mesmo tempo, em que diminuía o número de furtos em estabelecimentos comerciais? Um passo importante para resolver esse problema e melhorar a segurança em ambientes de alto risco é usar essa tecnologia como uma ferramenta preventiva.

1.2 Objetivos

Entende-se como objetivo deste trabalho criar um sistema de segurança baseado em visão computacional que auxilie na prevenção de furtos em ambientes comerciais. Utilizando tecnologias de detecção de objetos e reconhecimento de padrões, o sistema será capaz de identificar automaticamente ações suspeitas, como a colocação de itens em bolsas de forma suspeita, alertando imediatamente para uma possível tentativa de furto.

Os objetivos específicos desse trabalho são:

- Utilizar a tecnologia de visão computacional para detectar e identificar furtos em tempo real;
- Desenvolver um sistema de monitoramento que integre câmeras com algoritmos de reconhecimento de imagem;
- Implementar uma alerta que seja ativado automaticamente ao detectar comportamentos suspeitos, permitindo uma resposta rápida e eficaz;

1.3 Estrutura do trabalho

- O 1.º Capítulo contém a contextualização do trabalho de conclusão de curso;
- O 2.º Capítulo contém a fundamentação teórica dos temas apresentados no trabalho e trabalhos correlatos;
- O 3.º Capítulo contém a metodologia aplicada para o desenvolvimento do trabalho;

2 Fundamentos teóricos

Neste trabalho, será desenvolvido um sistema de visão computacional para detecção de furtos em ambientes comerciais, utilizando câmeras para identificar ações suspeitas em tempo real. O objetivo do sistema é auxiliar na prevenção de perdas, permitindo a identificação automática de objetos e a análise de interações entre clientes e itens no ambiente. Neste capítulo, são descritas as definições fundamentais para a composição deste trabalho.

Nessa seção será apresentado os conceitos de: Visão computacional, captura de movimento, *YOLO*, *COCO*, *Python*, furtos em super mercado.

2.1 Visão computacional

A visão computacional é um campo da ciência da computação que busca replicar a complexidade do sistema de visão humana para permitir que os computadores identifiquem e processem objetos em imagens e vídeos de forma semelhante aos seres humanos. No entanto, até recentemente, a capacidade da visão computacional era limitada (TRINDADE; PEREIRA; HOUNSELL, 2022).

Embora a visão computacional siga princípios semelhantes aos da visão humana, os humanos possuem uma vantagem inicial. A visão humana é enriquecida com anos de contexto para treinar a distinção de objetos, determinar distâncias, detectar movimentos e identificar anomalias em uma imagem (SOUZA; SUZANO, 2021).

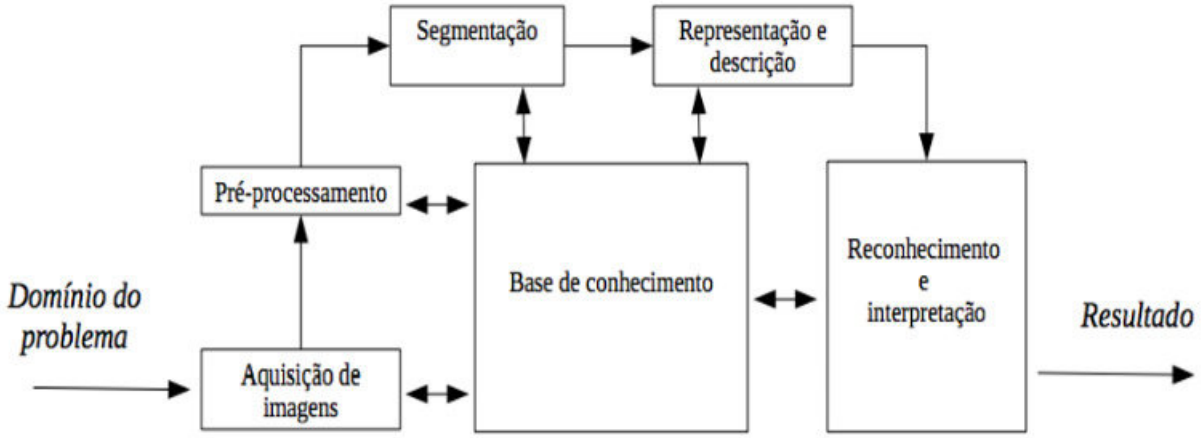
Com o intuito de obter aparência e formas tridimensionais dos objetos a partir de imagens, os desenvolvedores utilizam técnicas matemáticas na visão computacional. Essas técnicas estão sendo aprimoradas para permitir a obtenção dessas características. No entanto, mesmo com esses avanços, ainda não foi possível projetar um computador capaz de interpretar imagens e suas informações de maneira equivalente aos seres humanos (SZELISKI, 2022).

A área de estudo da visão computacional tem como objetivo principal capturar imagens por meio de equipamentos e sensores, realizar a análise e o processamento dos dados contidos nessas imagens. Esses dados processados são utilizados em tomadas de decisões ou em outros processos, como no controle de um braço robótico, por exemplo (SZELISKI, 2022).

2.1.1 Etapas da visão computacional

A visão computacional pode ser dividida em várias etapas para permitir que os computadores interpretem informações visuais como descrito a seguir:

Figura 1 – Etapas visão computacional



Fonte: (GALLON, 2014)

A visão computacional tem início na etapa de aquisição, onde uma imagem é convertida em representação numérica para o processamento pelo computador. A aquisição é composta por um dispositivo sensível ao espectro eletromagnético que produz um sinal analógico, e um digitalizador que converte esse sinal analógico em um sinal digital. Esse processo é geralmente realizado por meio de uma câmera ou outros tipos de sensores capazes de capturar imagens (FILHO; NETO, 1999).

O pré-processamento tem como objetivo melhorar a qualidade da imagem para obter melhores resultados nas etapas subsequentes. Isso pode incluir a aplicação de técnicas de contraste ou o isolamento de regiões de interesse (GONZALEZ; WOODS, 2000).

Na etapa de segmentação, o objetivo é dividir a imagem em diferentes regiões, de modo que cada região possa ser analisada por algoritmos que extrairão informações relevantes. Em imagens monocromáticas, por exemplo, a segmentação pode levar em consideração os níveis de cinza e suas diferenças (GONZALEZ; WOODS, 2000).

A representação e descrição são os resultados da segmentação, adaptados para serem processados por algoritmos específicos. Nessa etapa, ocorre a extração

das informações de interesse (FILHO; NETO, 1999).

A etapa de reconhecimento e interpretação é responsável por classificar os itens da imagem com base nas informações extraídas, utilizando uma base de conhecimento previamente definida como referência. Além disso, essa etapa atribui um significado aos objetos identificados (GONZALEZ; WOODS, 2000).

A base de conhecimento desempenha um papel fundamental, pois auxilia com o conhecimento existente sobre o problema a ser resolvido, orientando o funcionamento das etapas. Idealmente, a base de conhecimento também deve fornecer feedback para as etapas anteriores caso ocorram falhas ou inadequações (FILHO; NETO, 1999).

2.2 OpenCv

O *OpenCV* é uma plataforma composta por um conjunto de bibliotecas de código aberto projetadas para desenvolvimento de software em visão computacional e aprendizado de máquina. Inicialmente desenvolvido pela Intel, o *OpenCV* foi criado com o objetivo de fornecer uma infraestrutura comum para pesquisas e aplicações comerciais em visão computacional. Suas bibliotecas abrangem mais de 2700 algoritmos otimizados para diversas tarefas, como detecção e reconhecimento de faces, identificação de objetos, classificação de ações humanas em vídeos, rastreamento de movimentos de câmeras e objetos, modelagem 3D e visão estéreo, entre outras aplicações (BARELLI, 2018). Embora tenha sido inicialmente desenvolvido em C++, o *OpenCV* agora oferece suporte para várias linguagens de programação, incluindo Python e Java, além de interfaces para MATLAB, e pode ser executado nos sistemas operacionais Windows, Linux, Android e Mac.

2.2.1 DNN

A *DNN (Deep Neural Network)* do *OpenCV* é um componente essencial da biblioteca *OpenCV* e é frequentemente utilizada no processamento de imagens e computação de visão. A *DNN* permite que as redes neurais profundas funcionem para funções como detectar objetos, reconhecer rostos e segmentar imagens.

A *DNN* do *OpenCV* é fácil de usar e eficiente, o que permite que os desenvolvedores usem modelos treinados em plataformas como *TensorFlow*, *Caffe* e *PyTorch*. A biblioteca suporta a importação de modelos pré-treinados, o que facilita o desenvolvimento de soluções complexas sem a necessidade de treinamento inicial (SHALINI et al., 2021).

2.3 YOLO *(You Only Look Once)*

A evolução das técnicas de detecção de objetos em imagens e vídeos trouxe avanços significativos no campo da visão computacional, sendo o algoritmo *YOLO (You Only Look Once)* um dos principais marcos nesse desenvolvimento. O *YOLO* adota uma abordagem inovadora ao realizar a detecção e classificação de objetos em uma única passagem pela rede neural convolucional, diferentemente de métodos tradicionais que exigem múltiplas etapas de processamento para obter resultados precisos (ALVES, 2020).

Essa característica confere ao *YOLO* uma velocidade notável, permitindo a operação em tempo real com taxas de processamento de até 30 frames por segundo. A agilidade proporcionada por essa metodologia é particularmente relevante para aplicações em que a rápida identificação de objetos é essencial, como em veículos autônomos, onde é necessário identificar obstáculos e elementos do ambiente de maneira instantânea para garantir a segurança e a eficiência do sistema (ROMANI, 2020).

O YOLO tem sido amplamente adotado em diversas áreas que envolvem visão computacional, como (ROMANI, 2020):

- Detecção de objetos para carros autônomos;
- Robótica e interação humano-robô;
- Sistemas de segurança e vigilância;
- Auxílio a pessoas com deficiência visual;

2.3.1 *YOLO V8*

O algoritmo *YOLOv8* representa um avanço significativo na detecção de objetos, combinando rapidez e precisão em um nível superior em comparação com suas versões anteriores. Quando avaliado em termos de *MAP (mean Average Precision)* com *IOU (Intersection over Union)* de 0.5, o YOLOv8 não apenas supera o YOLOv5, mas também se destaca entre outros algoritmos contemporâneos, oferecendo melhorias substanciais na velocidade de inferência e na acurácia das detecções (RESEARCHGATE, 2024).

Uma das inovações do *YOLOv8* é a sua arquitetura otimizada, que permite uma maior eficiência no processamento de imagens. O modelo é capaz de realizar detecções em tempo real, alcançando taxas de até 60 quadros por segundo (FPS), o que o torna ideal para aplicações que exigem resposta rápida, como vigilância e veículos autônomos (DATACAMP, 2024). Além disso, o YOLOv8 incorpora técnicas

avançadas de aprendizado profundo que melhoram a robustez do modelo em diversas condições de iluminação e cenários complexos.

Diferentemente dos métodos tradicionais que utilizam classificadores ou localizadores separados para detectar objetos, o *YOLOv8* aplica uma única rede neural sobre a imagem inteira. Essa abordagem permite que o modelo divida a imagem em regiões e preveja caixas delimitadoras (bounding boxes) juntamente com as probabilidades associadas a cada região simultaneamente. As caixas são então ponderadas com base nas probabilidades previstas, resultando em uma detecção mais precisa e eficiente (MQL5, 2024).

Além disso, o YOLOv8 oferece uma flexibilidade notável na configuração do modelo, permitindo ajustes entre velocidade e precisão através da alteração do tamanho do modelo sem a necessidade de re-treinamento. Isso proporciona aos desenvolvedores uma ferramenta poderosa para otimizar aplicações específicas conforme suas necessidades (ULTRALYTICS, 2024).

Em termos de desempenho, o *YOLOv8* foi avaliado usando métricas como *MAP (mean Average Precision)* e *IOU (Intersection over Union)*, mostrando resultados superiores em comparação com suas versões anteriores. Essa melhoria é especialmente evidente em cenários complexos onde a precisão na detecção é crítica 56. A combinação dessas características torna o YOLOv8 uma escolha preferencial para aplicações que exigem detecção rápida e precisa, consolidando sua posição como uma das melhores opções disponíveis no campo da visão computacional.

2.4 COCO *(Common Objects in Context)*

Common Objects in Context (COCO) é um conjunto de dados amplamente utilizado na área de visão computacional, principalmente para tarefas de detecção de objetos, segmentação e legendagem de imagens. *COCO* se destaca pela diversidade e complexidade, oferecendo uma base rica para o treinamento e avaliação de algoritmos de aprendizado de máquina e criando um ambiente de compra mais seguro.

O conjunto de dados de *COCO* possui 2.500.000 instâncias rotuladas em mais de 300.000 imagens de 91 categorias de objetos comuns. COCO tem menos categorias que outros conjuntos de dados, porem mais instancias pra cada categoria. O que pode ajudar no aprendizado de modelos detalhados de objetos capazes de uma localização 2D precisa (LIN et al., 2014).

2.5 Python

Python é uma linguagem de programação de alto nível, amplamente adotada em diversos setores tecnológicos devido à sua versatilidade e simplicidade. Sua aplicação abrange áreas como o desenvolvimento web, ciência de dados, automação, inteligência artificial e aprendizado de máquina. O sucesso de *Python* se deve, a algumas características, sendo suas principais:

- Sintaxe Fácil de Ler: A sintaxe do *Python* é simples e fácil de entender, tornando o código mais fácil de ler e escrever. Novos programadores podem aprender rapidamente e programadores experientes podem facilmente manter e alterar o código (AWARI, 2023);
- Tipagem Dinâmica: No Python, as variáveis não precisam especificar seus tipos. Durante a execução, o tipo de uma variável é inferido automaticamente, o que aumenta a flexibilidade e diminui o número de código necessário para declarações de tipo;
- Bibliotecas e Módulos: O *Python* tem uma biblioteca padrão extensa que oferece uma variedade de funcionalidades prontas para uso. Além disso, há uma comunidade ativa que contribui com bibliotecas de terceiros para uma variedade de aplicações;
- Orientação a Objetos: A programação orientada a objetos (POO) é suportada pelo *Python*, o que significa que você pode criar classes e objetos. Através da herança e do polimorfismo, promove a reutilização de código e facilita a modelagem de problemas do mundo real;

2.6 Furtos em comércios

Os furtos em supermercados são um problema importante que tem um impacto significativo na lucratividade e na operação dos estabelecimentos varejistas. Fatores sociais, econômicos e comportamentais são envolvidos neste fenômeno complexo e sua mitigação requer uma abordagem estratégica.

Os furtos em supermercados correspondem a entre 25% e 30% das quebras operacionais e constituem uma fração significativa das perdas no varejo (ASSERJ, 2023). Isso resulta em perdas financeiras significativas no Brasil: dados mostram que os furtos podem causar perdas anuais de até R$ 11 bilhões, ou uma porcentagem preocupante do faturamento total do setor (MATTOS, 2024). A crise econômica e a diminuição do poder de compra dos consumidores têm sido associadas ao aumento

das taxas de furto. Isso resulta em um fenômeno conhecido como “furto famélico”, em que as pessoas furtam para comprar coisas básicas (ASSERJ, 2023).

2.6.1 Tipos de furtos

Os furtos podem ser classificados em duas categorias principais:

- Furtos externos. Os clientes fazem isso, geralmente por necessidade ou oportunidade. Produtos de alto valor, como alimentos e produtos de higiene pessoal, são frequentemente alvos.
- Furtos internos ocorrem quando os funcionários têm acesso a informações sobre produtos e estoque. Para reduzir esses problemas, as empresas devem ter controles rigorosos, como listas de produtos de alto risco e procedimentos internos de movimentação (GARDIN, 2022).

2.6.2 Medidas de prevenção

Para combater esse problema, os supermercados podem adotar algumas estratégias como:

- Gestão de Gôndolas: Colocar os produtos de maneira estratégica e usar equipamentos de segurança, como alarmes e gancheiras, pode ajudar a evitar furtos. Produtos de alto risco devem ser colocados em áreas de controle e visibilidade altas;
- Treinamento de Funcionários: Para evitar furtos, é fundamental treinar os funcionários e as equipes de segurança. A conscientização sobre comportamentos suspeitos e a implementação de protocolos de segurança estão entre essas coisas (ASSERJ, 2023);
- Tecnologia de Vigilância: Sistemas de monitoramento por vídeo e reconhecimento facial podem aumentar a segurança nas compras e ajudar na identificação e prevenção de furtos (CHIARA, 2022);

3 Trabalhos correlatos

Neste capítulo serão apresentados trabalhos que utilizam a mesma tecnologia que utilizada durante o desenvolvimento do projeto, a captura de movimento, *OpenCV* sendo utilizada voltada para segurança. A busca foi feita com foco em pesquisas cientificas na área nos últimos 10 anos. E os que poderiam melhor auxiliar durando o desenvolvimento foram os projetos citados abaixo.

3.1 Visão computacional para segurança pública

O trabalho de Ferreira (2020), o autor levantou uma questão bastante similar à que este projeto busca explorar: a viabilidade de utilizar a visão computacional como uma ferramenta eficaz no campo da segurança. Em sua pesquisa, Lucas desenvolveu um sistema de reconhecimento funcional, cujo objetivo era desenvolver um sistema de reconhecimento de faces com um custo reduzidos para órgãos públicos e analisar se câmeras equipadas com essa tecnologia poderiam contribuir para a prevenção de crimes. Embora o sistema criado por Lucas não tenha exatamente o mesmo foco ou aplicação específica que este trabalho propõe, seus resultados fornecem uma importante base teórica e prática para o uso da visão computacional no combate à criminalidade.

A pesquisa de Lucas demonstrou que a implementação de algoritmos de reconhecimento em sistemas de câmeras é não apenas viável, mas também altamente promissora no contexto de segurança pública e privada. Mesmo com as diferenças nas abordagens dos dois trabalhos, o estudo de Lucas validou que as câmeras, quando integradas com técnicas de visão computacional, são uma ferramenta bem-vinda na luta contra atividades criminosas, potencializando a capacidade de monitoramento e resposta a situações de risco.

Figura 2 – Testes realizados com as câmeras de segurança da UFMA.

Fonte: (FERREIRA, 2020)

3.2 Detecção de roubo de computadores em laboratório usando visão computacional

O trabalho de Cordeiro et al. (2020), o autor abordou um problema recorrente em laboratórios de informática: o furto de equipamentos, especialmente de gabinetes de computadores. A partir dessa questão, sua equipe propôs uma solução utilizando visão computacional, similar à abordagem deste projeto. Utilizando a linguagem *Python* em conjunto com o *framework OpenCV*, eles desenvolveram um sistema capaz de monitorar a movimentação de objetos dentro do ambiente.

Durante os testes, Gabriel e sua equipe obtiveram resultados positivos ao detectar quando indivíduos saíam da sala carregando os gabinetes, provando que a aplicação da visão computacional para esse fim é não apenas viável, mas funcional. A implementação conseguiu identificar esses eventos, reforçando o potencial da tecnologia como uma ferramenta preventiva em ambientes vulneráveis a furtos. Um exemplo dessa funcionalidade pode ser visto em uma das imagens do estudo, onde o sistema capturou o momento exato em que uma pessoa estava deixando a sala com um dos equipamentos.

Figura 3 – Pessoa saindo do laboratório carregando um gabinete. O Gabinete destacado em verde

Fonte: (CORDEIRO et al., 2020)

Esses resultados servem como um exemplo prático e bem-sucedido do uso de visão computacional para segurança, validando o uso de ferramentas como *OpenCV* e *Python* para monitoramento e detecção de situações suspeitas, o que fortalece ainda mais a proposta deste TCC, que também busca utilizar a visão computacional para prevenir crimes, especialmente furtos em ambientes comerciais.

3.3 Detecção de furtos utilizando aprendizado profundo

Nesse trabalho, Shirole (2023) apresentou dados que mostram como furtos e roubos são um problema global e levantou a questão de como o *machine learning* e a visão computacional podem contribuir para reduzir custos, diminuir a necessidade de intervenção humana, além de minimizar falsos alarmes em situações como essas.

O objetivo principal da pesquisa era detectar movimentos suspeitos, reconhecer expressões faciais e identificar pessoas usando máscaras, além de monitorar atividades suspeitas e o porte de armas. Para isso, o autor utilizou diversas metodologias,

como *machine learning*, o algoritmo *YOLO* e o framework *OpenCV*, entre outras ferramentas, para desenvolver o sistema.

Com o uso dessas tecnologias, o autor conseguiu identificar elementos críticos, como armas nas mãos de ladrões, como mostra na figura a seguir e outros comportamentos criminosos. A pesquisa concluiu que, para enfrentar problemas de segurança complexos como esses, é necessário adotar diferentes abordagens e metodologias conforme a tarefa, garantindo uma maior precisão na detecção e prevenção de crimes.

Figura 4 – Detecção de arma

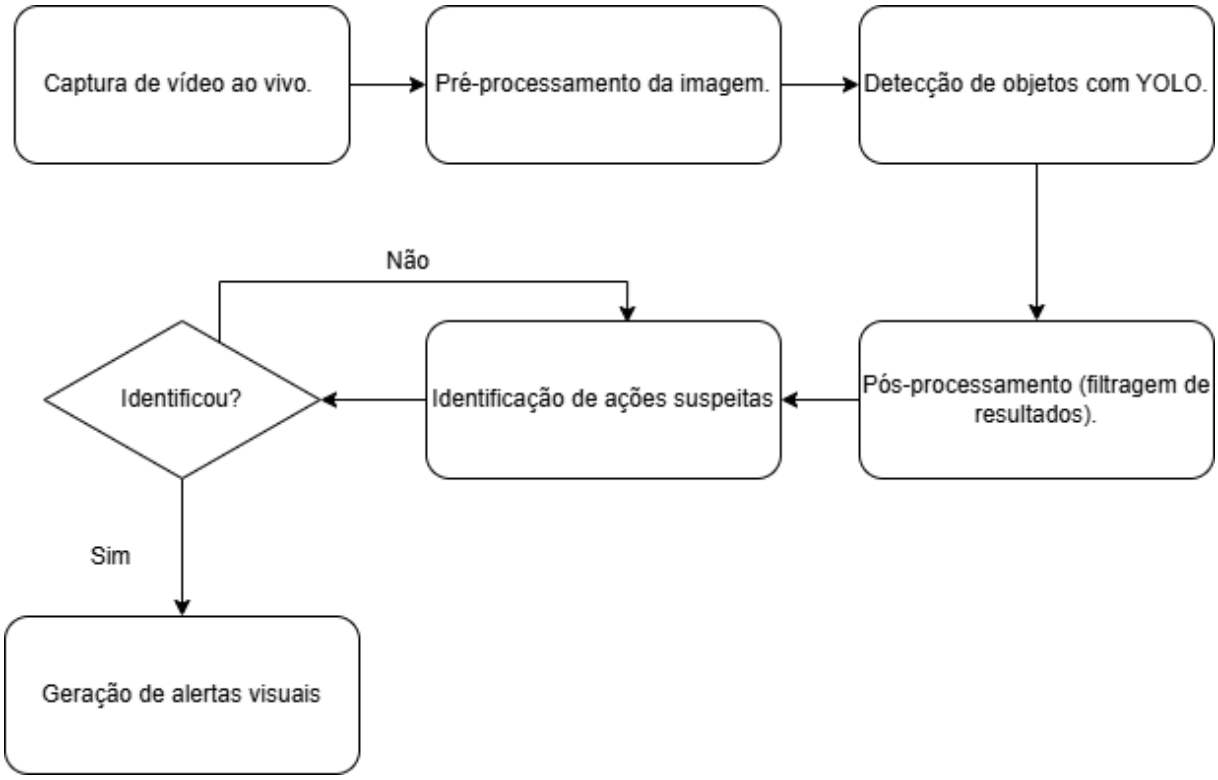
Fonte: (SHIROLE, 2023)

4 Desenvolvimento

Com base no referencial teórico e nas tecnologias já aplicadas em sistemas de visão computacional, foi projetado um método capaz de identificar ações suspeitas em ambientes comerciais, como o ato de colocar itens dentro de bolsas ou mochilas. Este método utiliza como base o modelo *YOLOv8* para detecção de objetos, treinado com a base de dados *COCO*, uma coleção que abrange mais de 80 classes de objetos com ampla aplicabilidade. Além disso, o algoritmo implementado se apoia na biblioteca OpenCV para manipulação de imagens e extração de características, possibilitando maior eficiência na análise em tempo real.

Uma visão geral do fluxo do algoritmo pode ser observada na Figura 5.

Figura 5 – Fluxograma do algoritmo



Fonte: Elaborado pelo autor

4.1 Preparação

4.1.1 Planejamento do desenvolvimento

A primeira etapa foi definir os conjuntos de objetos e ações a serem detectados. Para isso, foi utilizado o modelo pré-treinado *YOLOv8*, que se destaca por seu equilíbrio entre precisão e velocidade, ideal para aplicações em tempo real em ambientes comerciais (ULTRALYTICS, 2024). Este modelo foi treinado com a base de dados COCO, que fornece imagens anotadas para objetos comuns, como bolsas, mochilas, e itens de supermercado, relevantes para o contexto de prevenção a furtos.

Como a etapa de treinamento é computacionalmente custosa (ZHAO et al., 2019), utilizou-se o modelo pré-treinado, adaptando-o para reconhecer padrões específicos, como o movimento de colocar objetos em mochilas.

Com base na revisão das técnicas disponíveis e nos requisitos do projeto, foram definidas as ferramentas e o ambiente de desenvolvimento necessários para a implementação do sistema.

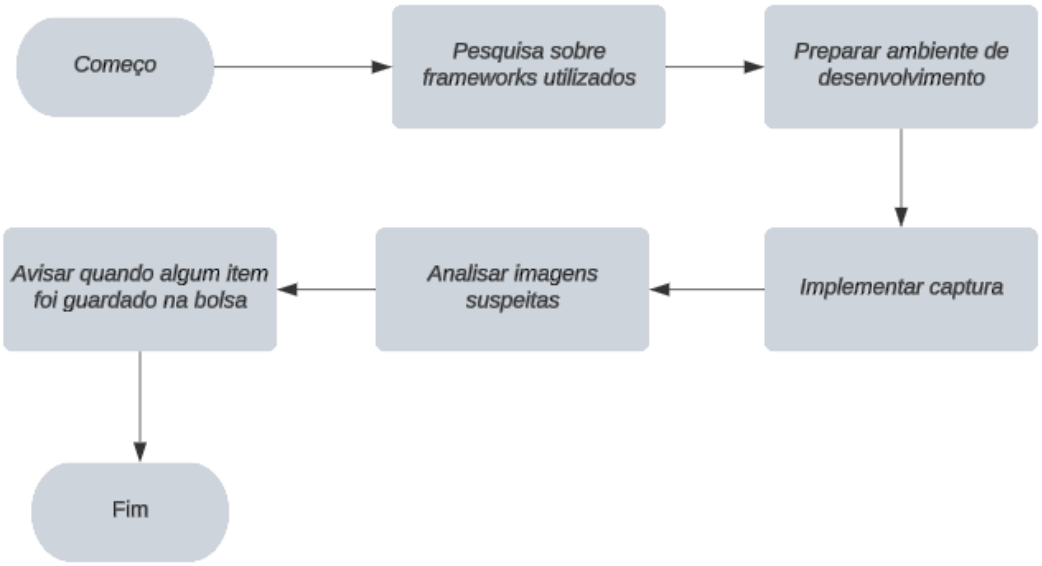
Essa etapa envolveu a instalação do Python 3.8 e a criação de um ambiente virtual, permitindo o isolamento das dependências do projeto. Com o ambiente configurado, prossegui para a instalação das bibliotecas essenciais, como *Ultralytics* e OpenCV, garantindo o suporte necessário para a implementação do método proposto.

Para garantir a execução do sistema, utilizou-se um ambiente de desenvolvimento com as seguintes configurações:

- Processador Intel Core i5 de 2,9 GHz (12CPUs);
- 2 Memórias *RAM* de 8 GB 3000 MHz;
- Sistema Operacional *Windows* 10;
- Disco de Memória 126 GB SSD;
- Placa Gráfica GTX 1660 SUPER 6GB *VRAM*;

E para fazer a captura de imagens foi utilizado uma câmera Logitech com resolução de 720p

Figura 6 – Procedimentos do desenvolvimento



Fonte: Elaborado pelo autor

Com a configuração do ambiente finalizada e as ferramentas definidas, o próximo passo foi preparar os dados de entrada, garantindo que as imagens capturadas fossem adequadas para a análise pelo modelo YOLOv8.

4.2 Pré-Processamento

Após a preparação inicial, foi necessário configurar o ambiente e realizar ajustes nas imagens capturadas. Esta etapa é fundamental para garantir que as imagens estejam nos formatos adequados para serem processadas pelo *YOLO* e para otimizar a precisão do algoritmo.

4.2.1 Captura da Imagem

A captura das imagens foi realizada por câmeras configuradas para operar em tempo real, cobrindo as áreas de interesse com ângulos estratégicos. A escolha do modelo de cores *RGB (Red, Green, Blue)* foi feita porque a maioria das câmeras digitais e algoritmos de visão computacional utilizam essa representação para capturar e processar imagens. O sensor da câmera registra a luz nas três cores primárias

e converte essas informações em dados digitais, garantindo fidelidade cromática na análise realizada pelo sistema.

(FILHO; NETO, 1999).

Além disso, optou-se por um sistema de rastreamento contínuo, onde *frames* específicos do vídeo eram extraídos em intervalos regulares para análise. Isso possibilitou a redução do volume de dados processados, sem comprometer a detecção de eventos relevantes. Cada *frame* foi identificado com um *timestamp*, garantindo que eventos pudessem ser correlacionados ao tempo exato de sua ocorrência.

Configuração da câmera: As câmeras foram ajustadas para operar com resolução de 1280x720 *pixels*, equilibrando qualidade visual e desempenho computacional. A taxa de quadros foi configurada para 30 FPS (frames por segundo), o que é suficiente para captar movimentos rápidos em ambientes comerciais.

4.2.2 Conversão e Normalização

As imagens capturadas passaram por um processo de pré-processamento para garantir compatibilidade com o modelo YOLOv8. A biblioteca *Ultralytics* automatiza grande parte dessas transformações, incluindo redimensionamento e normalização, eliminando a necessidade de ajustes manuais.

O redimensionamento para 640x640 *pixels* foi realizado para padronizar o tamanho das imagens, garantindo que todas possuam a mesma escala de entrada no modelo. Embora seja possível configurar a câmera para capturar diretamente nessa resolução, essa abordagem pode comprometer a qualidade da imagem, especialmente se a câmera possuir uma resolução nativa maior. Ao capturar em uma resolução mais alta e redimensionar posteriormente, preserva-se mais detalhes visuais e minimizam-se distorções causadas por compressão ou remostrarem inadequada.

A normalização das imagens, que converte os valores dos *pixels* para um intervalo entre 0 e 1, tem um papel fundamental no desempenho e estabilidade do modelo. Em imagens digitais, os valores dos *pixels* variam geralmente entre 0 e 255 (em uma escala de 8 bits). A normalização reduz essa escala para um intervalo menor, facilitando a convergência do modelo durante o processamento. Isso melhora a eficiência dos cálculos da rede neural, reduzindo o impacto de variações de iluminação e contraste, tornando o treinamento e a dedução mais consistente.

4.2.3 Conversão de Cores

A biblioteca OpenCV, utilizada no projeto, opera no formato *BGR (Blue, Green, Red)*, que difere do padrão *RGB* adotado pela maioria dos dispositivos de captura de

imagem. Por isso, foi necessário realizar uma conversão para ajustar os canais de cores das imagens.

Essa conversão foi realizada utilizando a função *cv2.cvtColor*, que permite transformar uma imagem de RGB para BGR de maneira eficiente. Essa etapa é crucial, ao garantir que os dados estejam no formato esperado pelo modelo, prevenindo erros de detecção causados por discrepâncias nos canais de cores.

Impacto da conversão: A mudança de formato é transparente para o desempenho do modelo, mas é indispensável para manter a compatibilidade com as ferramentas e bibliotecas utilizadas. Sem ela, as cores dos objetos poderiam ser interpretadas incorretamente, comprometendo os resultados.

4.3 Classificação e Detecção

4.3.1 Envio para o *YOLO*

Após o pré-processamento, as imagens foram enviadas ao modelo *YOLOv8*, que utiliza uma arquitetura moderna e eficiente para detectar objetos em tempo real. O *YOLOv8* divide a imagem em regiões, analisando simultaneamente as características de diferentes escalas. Essa abordagem permite identificar objetos de tamanhos variados e em condições desafiadoras, como iluminação fraca ou sobreposição de itens.

A integração nativa do *YOLOv8* com a biblioteca *Ultralytics* simplifica o processo de detecção, fornecendo diretamente as caixas delimitadoras, classes e valores de confiança. Isso reduz a complexidade do desenvolvimento e aumenta a precisão das detecções.

4.4 Rastreamento e Gerenciamento de Objetos

Após a detecção de objetos na cena utilizando o modelo YOLOv8, foi implementado um módulo de rastreamento para monitorar cada objeto ao longo de múltiplos quadros. Esse módulo atribui um identificador único (ID) a cada novo objeto detectado e registra sua posição na tela. O rastreamento utiliza um sistema de correspondência baseado na proximidade das caixas delimitadoras entre quadros consecutivos, garantindo que o mesmo objeto seja identificado consistentemente, mesmo em movimento.

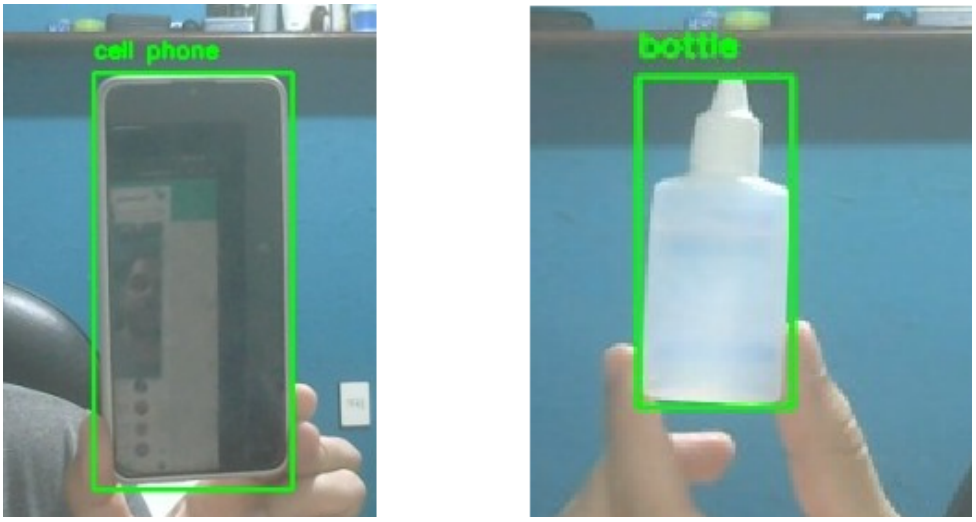
Para otimizar o gerenciamento de objetos, cada item detectado inclui informações como:

- Classe do Objeto: Nome do item detectado (ex.: mochila, garrafa).

- Posição Atual: Coordenadas da caixa delimitadora na imagem.
- Tempo de Desaparecimento: Marca temporal para determinar quando um objeto deixa de ser visível.

Demonstração do objeto e suas caixas delimitadoras na figura 6.

Figura 7 – Caixas delimitadoras e itens reconhecidos



Fonte: Elaborado pelo autor

4.5 Remoção de Objetos Não Rastreados

Um desafio identificado durante o desenvolvimento foi o acúmulo de objetos no dicionário de rastreamento, mesmo após sua saída da tela. Para resolver isso, foi implementada uma lógica de limpeza automática, que remove objetos que permanecem fora da tela por mais de dois segundos. Essa abordagem evita sobrecarga de memória e informações desnecessárias.

No entanto, para objetos que desaparecem na região de uma mochila, foi adotado um tratamento especial. Caso um objeto seja detectado numa mochila antes de desaparecer, ele não é removido imediatamente. Em vez disso, é registrado como “desaparecido” e analisado como potencialmente furtado.

4.6 Identificação de Ações Suspeitas

Com o rastreamento em funcionamento, o sistema verifica constantemente a interação entre objetos e mochilas. A lógica é baseada nas seguintes condições:

1. Quando um objeto detectado em um quadro não é mais visível nos quadros seguintes e está fora da área delimitada por uma mochila, ele é considerado irrelevante para a análise e removido do sistema.
2. Se um objeto desaparece na região delimitada por uma mochila (ou seja, suas coordenadas finais estão completamente nas coordenadas da mochila), ele é registrado como “desaparecido na mochila”. Após um período de dois segundos, o sistema emite um alerta indicando um possível furto.
3. as

Essa lógica é essencial para distinguir entre objetos que simplesmente saem do campo de visão e aqueles que podem indicar ações suspeitas. Por exemplo, um objeto que desaparece em uma prateleira pode não ser relevante, enquanto um objeto que desaparece numa mochila é considerado crítico para análise de segurança. Ao operar dessa forma, o sistema reduz significativamente falsos positivos e foca em eventos potencialmente associados a furtos.

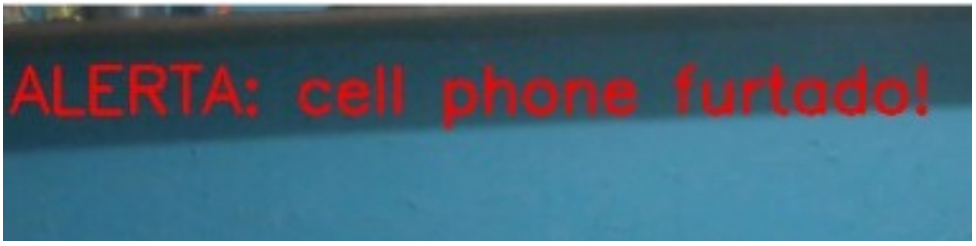
Com essa abordagem, o sistema consegue identificar ações suspeitas de maneira eficiente. No entanto, existem cenários que podem levar a falsos positivos, especialmente quando um cliente coloca um objeto pessoal na própria mochila. Por exemplo, alguém pode guardar um celular, carteira ou um item recém-adquirido em outra loja, e o sistema pode interpretar essa ação como um furto. Isso ocorre porque a lógica atual não distingue entre objetos pertencentes ao indivíduo e itens disponíveis na loja.

Para minimizar esse tipo de erro, seria interessante aprimorar o sistema com mecanismos adicionais de verificação. Uma abordagem possível seria a implementação de um código que, em vez de emitir um alerta de furto imediato, acione discretamente um segurança para analisar a situação antes de qualquer intervenção direta. Esse profissional poderia utilizar câmeras adicionais, revisar gravações anteriores ou até mesmo observar o comportamento do indivíduo antes de tomar qualquer medida.

Outra alternativa seria a integração com sistemas de inteligência artificial que analisam o contexto da ação, como a correlação entre o tempo de permanência da pessoa no local, o histórico de movimentação dos objetos e até o cruzamento de dados com sensores *RFID* ou reconhecimento facial para identificar se o objeto já

pertencia ao cliente. Dessa forma, o sistema se tornaria mais robusto e reduziria abordagens desnecessárias, evitando constrangimento para o cliente enquanto mantém a segurança eficiente.

Figura 8 – Mensagem de furto



Fonte: Elaborado pelo autor

4.7 Saída e Visualização

As detecções realizadas foram apresentadas visualmente por meio de caixas delimitadoras desenhadas sobre os objetos identificados. As ações suspeitas foram destacadas com cores específicas, permitindo aos operadores de segurança identificar rapidamente situações de risco. Além disso, os dados das detecções foram armazenados para análises futuras, possibilitando ajustes no modelo e o aprimoramento do sistema.

4.8 Implementação

Com o objetivo de validar o funcionamento e a eficiência do método de detecção de ações suspeitas e rastreamento de objetos, foi desenvolvido um software capaz de analisar imagens únicas ou sequências de *frames* em tempo real, utilizando a biblioteca *YOLO* para detecção de objetos. O sistema foi projetado para identificar ações como o movimento de esconder itens em mochilas, seguindo todas as etapas especificadas no método. A configuração do ambiente foi realizada para garantir a implementação correta e o desempenho. Para isso, foram utilizados os seguintes softwares e bibliotecas:

- *OpenCV*: para captura de imagens, manipulação de *frames* e exibição visual das detecções.

- *YOLOv8m*: para a detecção de objetos em tempo real, utilizando um modelo pré-treinado.

4.8.1 Observação em ambiente controlado

O teste em ambiente controlado foi realizado em um ambiente bem iluminado, onde foram posicionados diversos objetos, como frascos, copos, mochilas e pessoa. A iluminação foi cuidadosamente ajustada para ser uniforme e constante, reduzindo possíveis interferências externas que poderiam afetar a detecção. O objetivo principal foi validar a capacidade do sistema de reconhecer corretamente os objetos na cena e avaliar sua confiabilidade na identificação de ações suspeitas, como o movimento de inserção de objetos em uma mochila.

As imagens foram capturadas em tempo real por uma câmera. Essa configuração garantiu uma perspectiva clara e uniforme, permitindo ao sistema analisar o vídeo de maneira precisa, com foco na identificação dos objetos.

4.8.2 Observação no supermercado

O teste foi realizado em um supermercado local com diversos produtos nas prateleiras. A iluminação do ambiente variava, e o elevado número de objetos adicionava complexidade ao cenário. Durante um período de 1 hora, avaliou-se a robustez do sistema na identificação de objetos e sua capacidade de detectar ações suspeitas em uma simulação de furto. Para isso, cerca de 10 objetos diversificados, entre garrafas, caixas e sacos de alimento, foram colocados em uma mochila.

As imagens foram capturadas por uma câmera posicionada a uma altura de 2 metros, garantindo uma visão ampla do ambiente e permitindo a análise de objetos e movimentos em uma área considerável.

5 Resultados

Com base no referencial teórico apresentado neste trabalho e nas pesquisas utilizadas durante a implementação e desenvolvimento, foi proposto um método para a detecção de furtos em ambientes comerciais, com foco em ações como o movimento de inserção de objetos em mochilas. O objetivo do método é aliar precisão e eficiência em tempo real, utilizando a combinação de técnicas de visão computacional e aprendizado profundo.

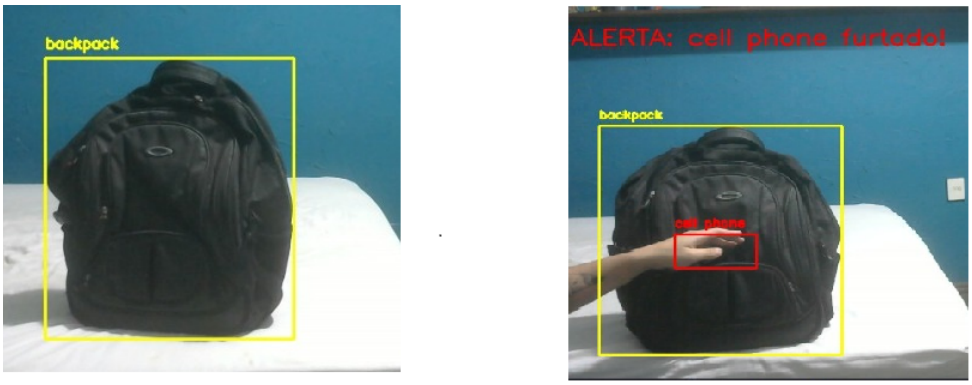
O sistema desenvolvido consegue identificar objetos em até 80 categorias diferentes, distinguir ações suspeitas em ambientes com alta densidade de informações visuais e rastrear o movimento de objetos ao longo de múltiplos quadros. Além disso, as ações suspeitas, como inserir um item em uma mochila, são registradas em tempo real para posterior análise.

5.1 Teste em ambiente controlado

Os resultados comprovam que o algoritmo se mostrou altamente confiável na identificação de objetos e na detecção de movimentos associados a furtos em ambientes controlados. Durante os testes, um total de 20 objetos foram posicionados em frente à câmera para avaliação do sistema. Destes, 18 foram reconhecidos e nomeados corretamente, enquanto 2 foram confundidos durante o registro, sendo inicialmente confundidos, mas ainda assim identificados nas categorias corretas.

Além disso, a detecção de furtos relacionados ao uso de mochilas demonstrou uma boa precisão. Mesmo ao tentar forçar falsos positivos, colocando objetos próximos à mochila sem intenção de roubo, o sistema não gerou alertas indevidos. Da mesma forma, falsos negativos não foram registrados, já que, conforme as especificações do sistema, todos os testes em que um objeto foi inserido na mochila foram corretamente identificados como furto. Esse desempenho reforça a eficácia do algoritmo e sua robustez na prevenção de perdas no varejo.

Figura 9 – Detecção de mochila e detecção e mensagem de furto



Fonte: Elaborado pelo autor

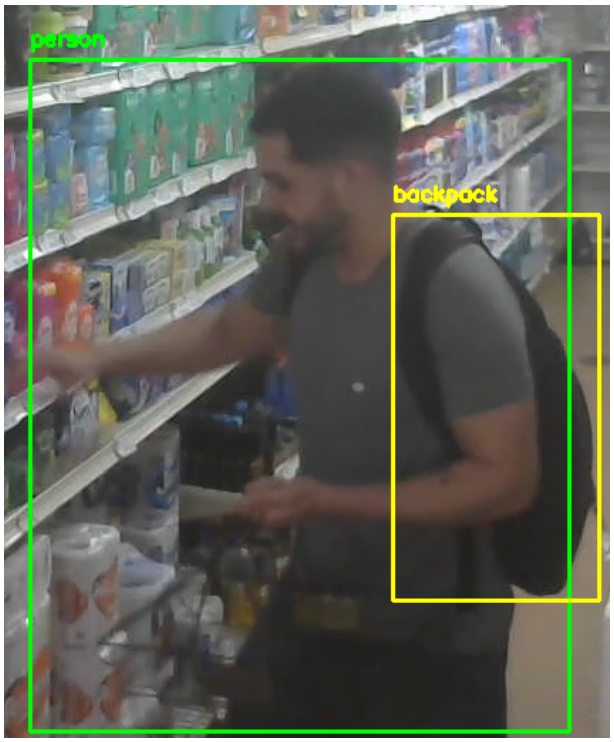
5.2 Teste no supermercado

Essa configuração foi eficaz para avaliar tanto os movimentos suspeitos quanto os objetos nas prateleiras, demonstrando a capacidade do sistema de identificar ações, como a inserção de itens em mochilas.

No entanto, foi observado que em situações onde os objetos estavam muito distantes da câmera, a detecção deixou de ser consistente. Essa limitação ocorreu devido à resolução reduzida dos objetos no quadro, o que dificultou o reconhecimento pelo modelo. Por exemplo, itens pequenos posicionados em prateleiras mais afastadas ou em regiões fora do campo central da câmera apresentaram falhas na identificação.

Essa limitação, entretanto, não compromete a aplicabilidade do sistema em cenários reais, como supermercados, onde normalmente há uma rede de câmeras estrategicamente posicionadas para cobrir múltiplos ângulos do ambiente. Em um sistema com múltiplas câmeras, essas falhas seriam suavizadas, já que cada câmera seria responsável por monitorar uma área específica, garantindo maior cobertura e precisão na detecção de objetos e ações suspeitas.

Figura 10 – Gravação em um supermercado, mochila detectada



Fonte: Elaborado pelo autor

6 Conclusão

6.1 Desafios enfrentados e resultados obtidos

Um dos maiores desafios enfrentados foi ajustar o algoritmo para operar em condições variáveis de iluminação e em ambientes com grande número de objetos. No ambiente do supermercado, foram observadas dificuldades específicas, como a baixa iluminação em determinadas áreas, que comprometeu a visibilidade de alguns itens, e a distância entre os objetos e a câmera, que reduziu a precisão do reconhecimento, especialmente para itens menores ou em prateleiras mais afastadas. Apesar dessas limitações, o sistema conseguiu detectar o movimento de inserir objetos em mochilas. Em cenários controlados, no entanto, o desempenho foi significativamente melhor, confirmando a eficácia do método em condições ideais.

6.2 Trabalhos futuros

Com o método implementado, diversas possibilidades de evolução do projeto foram identificadas:

- Expansão do treinamento do modelo com imagens capturadas em tempo real no ambiente-alvo, para personalização e maior acurácia.
- Integração com sistemas de segurança, permitindo alertas automáticos por dispositivos conectados.
- Detecção de ações mais complexas, como transferência de objetos entre pessoas.
- Detecção de outros objetos que possam apresentar capacidade de furto, como bolsas, sacolas e mochilas.

6.3 Considerações finais

Os resultados obtidos demonstraram que o método proposto é viável e eficiente para a identificação de objetos e ações suspeitas em tempo real, mesmo em cenários desafiadores. Durante os testes, o sistema apresentou alta assertividade, com uma taxa de falsos positivos de apenas 10% no teste controlado, sendo estes de itens que foram erroneamente identificados pelo sistema. A robustez na detecção

de ações específicas, como furtos, aliada à sua adaptabilidade, reforça o potencial do método para aplicações comerciais.

Embora os resultados sejam promissores, melhorias adicionais, como o aumento da base de treinamento e a integração com redes de câmeras, podem ampliar ainda mais a precisão e a aplicabilidade do sistema. Esse trabalho representa um avanço significativo no uso de visão computacional para prevenção de perdas em ambientes comerciais, estabelecendo uma base sólida para futuras evoluções no campo da segurança automatizada.

Referências

ALVES, G. *Detecção de Objetos com YOLO - Uma abordagem moderna*. 2020. Citado na página 15.

ASSERJ. Furtos em supermercados: Onde impacta e como os varejistas podem reduzir seus danos nos resultados? 2023. Citado nas páginas 17 e 18.

AWARI. Conceito básico de python: Aprenda os fundamentos da linguagem de programação. 2023. Citado na página 17.

BARELLI, F. *Introdução à visão computacional: Uma abordagem prática com Python e OpenCV*. [S.l.]: Editora Casa do Código, 2018. Citado nas páginas 9 e 14.

CHIARA, M. D. Taxa de furtos em mercadinhos de condomínio é o dobro da média do varejo comum; entenda. 2022. Citado na página 18.

CORDEIRO, G. A. et al. Detecção de roubo de computadores em laboratório usando visão computacional. *Anais do Computer on the Beach*, v. 11, p. 054–055, 2020. Citado nas páginas 20 e 21.

DATACAMP. Introduction to yolov8 for object detection. *DataCamp*, 2024. Disponível em: <https://www.datacamp.com/community/tutorials/introduction-to-yolov8-for-object-detection>. Citado na página 15.

FERREIRA, L. R. Visão computacional para segurança pública. UFMA, 2020. Citado nas páginas 9, 19 e 20.

FILHO, O. M.; NETO, H. V. *Processamento digital de imagens*. [S.l.]: Brasport, 1999. Citado nas páginas 13, 14 e 26.

GALLON, L. *Sistema de visão computacional para classificação de pedras naturais através de vídeo em tempo real*. 2014. Dissertação (B.S. thesis), 2014. Citado na página 13.

GARDIN, E. Como agir com produtos de alto risco? a prevenção é sempre a melhor forma de evitar desperdícios e perdas. 2022. Disponível em: <https://checkoutrh.com.br/como-agir-com-produtos-de-alto-risco-a-prevencao-e-sempre-a->. Citado na página 18.

GONZALEZ, R. C.; WOODS, R. E. *Processamento de imagens digitais*. [S.l.]: Editora Blucher, 2000. Citado nas páginas 13 e 14.

LIN, T.-Y. et al. Microsoft coco: Common objects in context. In: SPRINGER. *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. [S.l.], 2014. p. 740–755. Citado na página 16.

MATTOS, A. Furtos no varejo voltam a crescer e atingem maior nível desde 2019. 2024. Citado nas páginas 9 e 17.

MQL5. Understanding yolov8: Innovations in object detection. *MQL5 Community*, 2024. Disponível em: <https://www.mql5.com/en/articles/understanding-yolov8-innovations-in-object-detection>. Citado na página 16.

RAHANGDALE, K.; KOKATE, M. Event detection using background subtraction for surveillance systems. *International Research Journal of Engineering and Technology*, v. 3, n. 01, p. 1300–1304, 2016. Citado na página 9.

RESEARCHGATE. Yolov8: A comprehensive overview. *ResearchGate*, 2024. Disponível em: <https://www.researchgate.net/publication/YOLOv8_A_Comprehensive_Overview>. Citado na página 15.

ROMANI, B. *yolo-conheca-a-inteligencia-artificial-no-cerebro-de-carros-autonomos*. 2020. Citado na página 15.

SHALINI, K. et al. Comparative analysis on deep convolution neural network models using pytorch and opencv dnn frameworks for identifying optimum fruit detection solution on risc-v architecture. In: IEEE. *2021 IEEE Mysore Sub Section International Conference (MysuruCon)*. [S.l.], 2021. p. 738–743. Citado na página 14.

SHIROLE, S. Theft detection using deep learning. 2023. Citado nas páginas 21 e 22.

SOUZA, E. C.; SUZANO, J. L. Q. Desenvolvimento de um jogo sério usando opencv para reabilitação da função manual. Universidade Presbiteriana Mackenzie, 2021. Citado na página 12.

SZELISKI, R. *Computer vision: algorithms and applications*. [S.l.]: Springer Nature, 2022. Citado na página 12.

TRINDADE, A.; PEREIRA, G.; HOUNSELL, M. Chão interativo e jogos sérios ativos para autistas: A plataforma t-tea e o jogo repetea. In: SBC. *Anais Estendidos do XXI Simpósio Brasileiro de Jogos e Entretenimento Digital*. [S.l.], 2022. p. 512–521. Citado na página 12.

ULTRALYTICS. Yolov8: The next generation of object detection. *Ultralytics Blog*, 2024. Disponível em: <https://ultralytics.com/yolov8-the-next-generation-of-object-detection>. Citado nas páginas 16 e 24.

ZHAO, Z.-Q. et al. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, IEEE, v. 30, n. 11, 2019. Citado na página 24.