Identificação de Quedas com Visão Computacional

João Vitor Garcia Carvalho
UTFPR - Campus Apucarana
Apucarana, Brasil
jcarvalho.2020@alunos.utfpr.edu.br

Maria Eduarda Pedroso
UTFPR - Campus Apucarana)
Apucarana, Brasil
mpedroso@alunos.utfpr.edu.br

Gabriel Finger Conte
UTFPR - Campus Apucarana
Apucarana, Brasil
gabifcont@gmail.com

***Resumo*—Este relatório descreve a implementação de algoritmos de visão computacional para a identificação de quedas, utilizando câmeras como fonte de dados. A obtenção dos dados foi feita utilizando OpenCv em conjunto com o OpenPose, e para a classificação dos movimentos foi implementado o algoritmo K-Nearest Neighbor (Knn) em Python. A metodologia incluiu análises frame a frame e em conjunto. Embora os resultados tenham demonstrado boa precisão durante a validação, desafios foram identificados devido à limitação na precisão do OpenPose na obtenção de dados antropométricos. A aplicação em cenários reais destacou a inconsistência na detecção dos pontos-chave, impactando o desempenho do modelo. Sugerimos explorar técnicas avançadas de processamento de imagem, avaliar diferentes algoritmos e considerar dados temporais para futuras melhorias.**

***Index Terms*—Reconhecimento de Quedas, Processamento de Imagens, Aprendizado de Máquina, OpenPose, K-Nearest Neighbor, Segurança do Idoso.**

I. Introdução

O mundo está envelhecendo, isso é um fato. Conforme a expectativa de vida aumenta devido aos avanços tecnológicos e da medicina, cresce o número de pessoas idosas, como pode ser observado na plataforma online 'OUR WORLD IN DATA' [1].

Conforme envelhecemos nossa saúde se deteriora, em decorrência disso, processos acidentais como quedas passam a impactar seriamente na nossa sobrevivência. Tal fato constata-se tendo em vista que entre os anos de 2000 e 2019, a taxa de mortalidade por queda de idosos somente no Brasil apresentou um comportamento crescente. [2]

Emergências decorrentes de quedas podem ocorrer a qualquer momento, em qualquer lugar. Dependendo da situação o indivíduo pode não ser capaz de acionar algum serviço de emergência, perdendo muitas vezes as chances de se salvar. Com isso em mente, ressalta-se como a tecnologia de detecção de quedas é extremamente importante para agir em conjunto com sistemas de alerta médico e melhorar a chance de sobrevivência das pessoas. [3]

Mas para detectar as quedas são necessários sensores, os quais podem ser agrupados em três diferentes categorias: sensores de movimento, que utilizam dados sobre os movimentos do indivíduo; físicos, que analisam dados de sinais biológicos, ou ambientais, que utilizam uma série de sensores para detectar quedas. [4]

Dentre esses diferentes tipos, no presente trabalho focou-se especificamente em um tipo de sensor ambiental, as câmeras. A aplicação de algoritmos de identificação de quedas poderiam ser empregados nas mesmas, de modo a remover a limitação de necessitar estar com um dispositivo de medição equipado a todo momento, o qual pode ser esquecido, ter de ser removido para recarregar ou mesmo causar algum desconforto.

Além do mais, devido a sua fácil instalação, não dependência de outro conjunto de sensores e uma maior área de cobertura. Buscou-se desenvolver um algoritmo simples em linguagem de programação Python, capaz de detectar quedas através das gravações obtidas através de câmeras.

Objetivou-se com isso, aplicar os conceitos desenvolvidos na disciplina de Sistemas Inteligentes 1 do Curso de Engenharia da Computação. Outrossim, ampliar a experiência prática no campo de visão computacional, trabalhando em cima de um tema relevante e útil para a sociedade.

II. Trabalhos Relacionados

Abd et al. [5] propõem um forma de detectar quedas por meio de rastreio do corpo humano com Open Pose. Em seu trabalho, os autores apresentam um algoritmo iterativo para poder obter resultados, este começa com a obtenção de dados com Open Pose, seguido de de uma estimação rastreio da pose do corpo e, por fim, é feito uma análise dos dados para calcular os ângulos dos corpos para detectar queda. No algoritmo proprosto por Abd et al. há duas formas de decisões para definir queda, a primeira baseada no ângulo entre os pontos do esqueleto e a segunda baseada na distância entre os pontos quando, pelo ângulo, a posição vertical e horizontal são iguais.

Ramirez et al. [6] utilizaram de quatro modelos de classificação sendo eles, Random Forest (RF), Support Vector Machine (SVM), Multilayer Perceptron (MLP) e K-Nearest Neighbor (KNN), para detectar quedas através de uma estimativa de pose para extrair características de imagens RGB, os resultados demonstraram que dentre os modelos treinados no conjunto de dados UP-FALL apresentaram um desempenho notável em termo de precisão. Uma das limitações desse trabalho é que devido a utilização da estimativa de pose podemos não capturar todos os nuances e sutilezas associados aos movimentos humanos, por outro lado um diferencial citado por Ramirez et al. é o reconhecimento e detecção de uma pessoa ou mais.

Htoo e Sein [7] desenvolveram em seu trabalho um sistema de reconhecimento de quedas baseado na extração de características discriminantes do equilíbrio e movimento do corpo humano, utilizando do aprendizado de classificação

multi-SVM para testar as atividades humanas desconhecidas e reconhecer um evento de queda humana. O sistema utiliza tecnologias de sensor Kinect da Microsoft para rastrear e detectar as atividades do corpo humano. Como resultado o trabalho obteve uma precisão média de 91,75% na classificação das atividades humanas, com uma precisão de 90,96% e um grau de certeza de 91,15%. No entanto, o sistema ainda apresenta algumas limitações, como a necessidade de um ambiente controlado e por utilizar da detecção e rastreamento das articulações esqueléticas do corpo humano pelo sensor Kinect, movimentos rápidos, ocultação parcial das articulações acabam atrapalhando seu desempenho.

III. Metodologia

O modelo proposto neste projeto pode ter seu processo de construção e validação dividido em 3 etapas: (1) Obtenção dos dados, (2) Desenvolvimento do modelo de Aprendizado de Máquina (AM), (3) Teste e validação. Cada uma dessas etapas é abordada de forma mais detalhada nas seções subsequentes.

No processo de desenvolvimento do sistema de reconhecimento de quedas, que abrange desde a aquisição dos dados das câmeras até o processamento e extração de parâmetros, bem como a implementação da do algoritmo de reconhecimento de quedas, a escolha da linguagem de programação Python foi deliberada e fundamentada.

Optamos por Python devido à sua vasta gama de bibliotecas disponíveis, que cobrem tanto o processamento de imagem quanto o desenvolvimento de algoritmos de inteligência artificial. Além disso, Python é notável por sua otimização para análise de dados, o que o torna uma escolha ideal para a tarefa complexa de reconhecimento de quedas.

Essa seleção estratégica de linguagem nos proporciona as ferramentas necessárias para criar um sistema de detecção de quedas eficaz, aproveitando a flexibilidade e a capacidade de personalização que Python oferece em combinação com bibliotecas de ponta.

A. Obtenção de Dados

Dentro do ambiente Python, dispomos de diversas bibliotecas que possibilitam a aquisição de dados através de câmeras. Para a execução deste projeto, optou-se pela utilização do OpenCV, uma biblioteca versátil e amplamente suportada em diversas plataformas. Através do OpenCV, conseguimos capturar vídeos e estimar o estado do corpo de por meio dos pontos do esqueleto.

Além disso, para a precisa estimativa da posição dos pontos do esqueleto com o OpenCV, recorremos a uma biblioteca auxiliar conhecida como Open Pose. Cada ponto identificado por essa biblioteca corresponde a uma parte específica do corpo, como ilustrado na figura 1, um exemplo de obtenção de dados por meio do Open Pose durante a ação de abaixar pode ser vista na figura 2. Essa abordagem permitiu uma análise detalhada e robusta dos dados provenientes das imagens de vídeo, possibilitando assim o desenvolvimento de um sistema eficaz de detecção de quedas e reconhecimento de atividades.

Figura 1. Pontos de esqueleto em Open Pose, [7]

Figura 2. Exemplo de obtenção de dados por meio de Open Pose

Para a classificação dos dados, essenciais para o treinamento da rede neural, a obtenção de dados foi categorizada em duas classes: ”Cair”e ”Abaixar”. Na primeira categoria, gravaram-se vídeos de pessoas em queda, enquanto na segunda, utilizaram-se vídeos de pessoas se abaixando.

A partir dos vídeos capturados, foram extraídos os dados dos pontos mais relevantes para o presente projeto, utilizando parâmetros comuns na literatura para a detecção de quedas. Mais especificamente, determinamos dois parâmetros a partir dos pontos de esqueleto, conforme apresentado na Tabela I.

Tabela I
COORDENADAS E PARÂMETROS ANTROPOMÉTRICOS ANALISADOS A PARTIR DOS PONTOS DE ESQUELETO

Parâmetro	Descrição
Coordenadas do ponto do pescoço	As coordenadas do ponto do pescoço no frame do vídeo analisado, representando a posição da cabeça de uma pessoa.
Coordenadas do quadril esquerdo (QE)	As coordenadas do ponto de esqueleto referentes ao quadril esquerdo.
Coordenadas do quadril direito (QD)	As coordenadas do ponto de esqueleto referentes ao quadril direito.
Coordenadas do ponto médio da cintura (PMC)	As coordenadas do ponto médio entre o quadril esquerdo e o direito.
Ângulo entre o quadril e o pescoço	Mede o ângulo formado entre as posições do quadril e do pescoço no esqueleto do corpo.
Distância vertical do pescoço	Indica a distância entre o ponto de esqueleto do pescoço em relação ao ponto médio dos quadris.

Para obter as coordenadas do ponto médio da cintura (PCM) e calcular os parâmetros de ângulo e distância, foram utilizadas as coordenadas pertinentes destacadas na tabela apresentada. A determinação dessas métricas fundamentou-se nos princípios básicos da geometria, resultando nas equações fornecidas na Tabela II. Essas formulações foram essenciais para a construção do conjunto de dados empregado no desenvolvimento do modelo de Aprendizado de Máquina (AM), como será minuciosamente abordado na seção subsequente.

Tabela II
CÁLCULOS DOS PARÂMETROS ANTROPOMÉTRICOS

Parâmetro	Fórmula de Cálculo
Coordenadas PCM	$\left(\frac{x_{\text{QD}}+x_{\text{QE}}}{2}\right), \left(\frac{y_{\text{QD}}+y_{\text{QE}}}{2}\right)$
Ângulo entre o quadril e o pescoço	$\arctan\left(\frac{y_{\text{pescoço}}-y_{\text{PCM}}}{x_{\text{pescoço}}-x_{\text{PCM}}}\right) * \frac{180}{\pi}$
Distância vertical do pescoço	$\sqrt{(x_{\text{pescoço}} - x_{\text{PCM}})^2 + (y_{\text{pescoço}} - y_{\text{PCM}})^2}$

Adicionalmente, nos casos em que o OpenPose não consegue reconhecer os pontos necessários para os cálculos de coordenadas, foi considerada a possibilidade de inserir manualmente os dados. Isso envolve a estimativa da posição do ponto perdido com base na posição e velocidade anteriores. Essa abordagem visa evitar distorções significativas nos dados fornecidos ao algoritmo de aprendizado de máquina. O objetivo é minimizar a introdução de um erro substancial em comparação com o erro associado à estimativa da posição perdida.

B. Desenvolvimento do modelo de Aprendizado de Máquina (AM)

Ao longo deste estudo, decidimos empregar o algoritmo K-Nearest Neighbor (Knn) de aprendizado de máquina, dada a sua comprovada eficácia na classificação de eventos. Escolhemos este algoritmo, pois já foi abordado em profundidade ao longo da disciplina, apresentando um nível de complexidade adequado para o tempo limitado disponível no desenvolvimento do projeto.

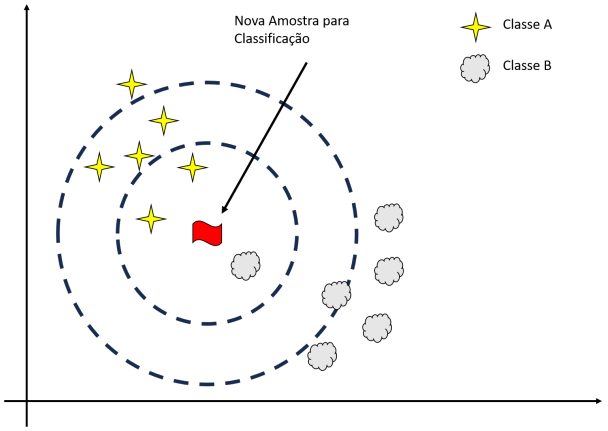

Figura 3. Ilustração do funcionamento do algoritmo K-Nearest Neighbor (Knn)

É relevante reiterar o funcionamento fundamental do algoritmo: a partir de um conjunto de dados (dataset), realiza-se a separação em conjuntos de treino e teste. Recomenda-se que o conjunto de treino contenha preferencialmente o mesmo número, ou um número relativamente próximo, de exemplos de diferentes categorias.

Durante a etapa de treinamento, o algoritmo Knn avalia a proximidade entre pontos no espaço de características, classificando uma nova instância com base na maioria dos k vizinhos mais próximos, como ilustrado na Figura 3. Esse método é particularmente eficaz em problemas nos quais a fronteira de decisão é complexa e não linear.

No processo de desenvolvimento, dedicamos atenção especial à seleção e ajuste criteriosos dos parâmetros, notadamente o valor de k, com o objetivo de otimizar o desempenho do modelo. Essa escolha foi embasada em resultados promissores observados em uma pesquisa anterior conduzida por Ramirez et al., que indicou a eficácia do Knn na detecção de quedas com base nas informações dos pontos de esqueleto.

C. Teste e validação

A fase de teste e validação é crucial para avaliar o desempenho do modelo desenvolvido. Nesta seção, detalharemos o processo de teste e as métricas utilizadas para avaliar a eficácia do modelo.

1) Divisão dos Dados: Inicialmente, dividimos o conjunto de dados em dois subconjuntos principais: conjunto de treinamento e conjunto de teste. A porcentagem típica usada para essa divisão é 70% para treinamento e 30% para teste, sendo essa a proporção considerada no presente.

2) Avaliação do Modelo: Após o treinamento, aplicamos o modelo ao conjunto de teste para avaliar sua capacidade de generalização. As métricas de avaliação utilizadas incluem:

- Matriz de Confusão: Uma tabela que mostra a frequência das classificações corretas e incorretas feitas pelo modelo para cada categoria. Isso nos permite avaliar o desempenho em termos de verdadeiros positivos (VP), verdadeiros

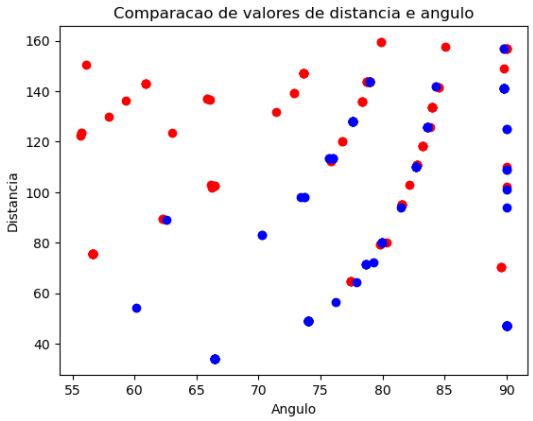

Figura 4. Relação entre dados de distância e ângulo nos dados obtidos, modelo Frame à Frame

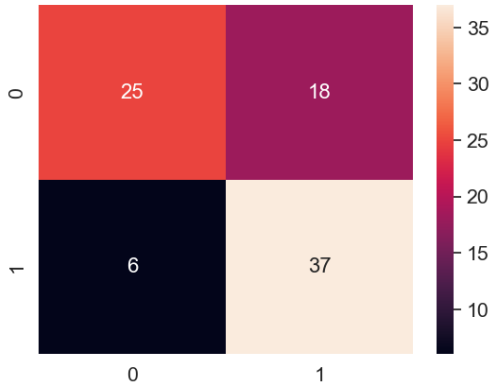

Figura 5. Matriz de Confusão do modelo Frame à Frame

negativos (VN), falsos positivos (FP) e falsos negativos (FN).

- Precisão: A proporção de VP em relação ao total de instâncias classificadas como positivas (VP + FP). Indica a capacidade do modelo de evitar classificações incorretas, sendo formalmente descrita pela Fórmula 1.

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{1}$$

- Revocação (Sensibilidade): A proporção de VP em relação ao total de instâncias realmente positivas (VP + FN). Avalia a capacidade do modelo de identificar todas as instâncias positivas, sendo formalmente descrita pela Fórmula 2.

$$\text{Revocação} = \frac{VP}{VP + FN} \tag{2}$$

- Acurácia: A proporção de instâncias classificadas corretamente em relação ao total de instâncias. Embora seja uma métrica comum, pode ser enganosa em conjuntos de dados desbalanceados. Sendo formalmente descrita pela Fórmula 3.

$$\text{Acurácia} = \frac{VP + VN}{VP + VN + FP + FN} \tag{3}$$

IV. Resultados

Tomando em consideração a abordagem por meio de uma análise frame à frame dos dados, foi montado um modelo de KNN com 70% dos dados obtidos em cada categoria. Após a seleção dos dados do modelo obteve-se gráfico presente na Figura 4, nela é possível observar a semelhança entre as categorias em determinados momentos, no entanto, também é possível observar a diferença entre elas em valores mais baixos da distância. A região em comum às duas categorias se dá pelo início do movimento onde o corpo ainda está quase em pé.

Realizando a validação deste modelo com os outros 30% dos dados obteve-se a matriz de confusão presente na Figura 5, onde a categoria 0 descreve a categoria ”Abaixar”e a categoria 1, a categoria ”Cair”. Calculando as métricas de avaliação, observa-se que o resultado frame a Frame apresentou uma precisão de 58,14%, contando com uma revocação de 80,64%. O que indica que o modelo obteve um bom desempenho, com uma acurácia de 72,09%. Observa-se um desempenho melhor na classificação da categoria ”Cair”, no entanto, com relação à categoria ”Abaixar”houve uma queda no desempenho. A partir desta matriz de confusão é possível teorizar que em um ambiente real, o modelo estaria sujeito a errar na análise de movimentos onde o sujeito estaria apenas abaixando.

Em um segundo momento, foi feita uma análise dos dados em conjunto, diferente da análise frame à frame, essa considera não apenas um frame em singular, mas uma sequência de frame descrevendo uma trajetória dos elementos em função do tempo. Realizando a obtenção dos dados para a elaboração de um modelo, obteve-se o gráfico descrito pela Figura 6, desta vez, os dados presentes no modelo representam 66% dos dados totais obtidos. Cada linha preta do gráfico descreve uma trajetória de um movimento da categoria ”Cair”as linhas vermelhas caracterizam os movimentos da categoria ”Abaixar”. Analisando o gráfico é possível observar que o começo das trajetória de ambos os movimentos são semelhantes, mas depois divergem uma da outra.

Da mesma forma com o primeiro modelo, este foi validado realizando com os dados restantes do modelo, então, obteve-se a matriz de confusão presente na Figura 7. Nela é possível observar um desempenho semelhante ao modelo frame à frame, onde a classificação da categoria ”Cair”atingiu um desempenho melhor que a classificação da categoria ”Abaixar”. Todavia, a acurácia geral apresentou uma redução em dez pontos percentuais, passando para 60%.

Por fim, ambos os modelos foram implementados de forma iterativa em um programa contínuo para validar a sua aplicação em um cenário real. Para o modelo frame à frame, obteve-se os resultados presentes na Figura 8, demonstrando uma acurácia aceitável de aproximadamente 75% para os 4 casos analisados. Já para o modelo conjunto, os resultados presentes

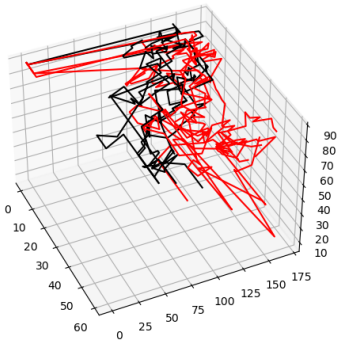

Figura 6. Relação entre dados de distância e ângulo nos dados obtidos, modelo Conjunto

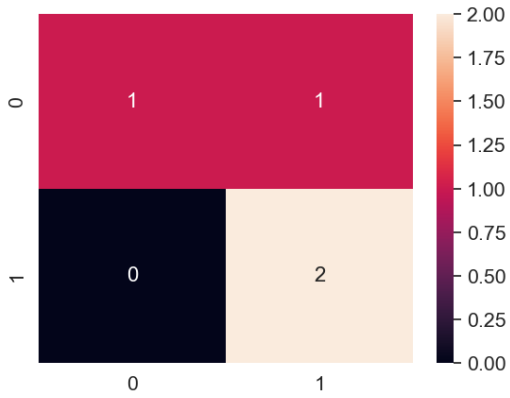

Figura 7. Matriz de Confusão do modelo Conjunto

na Figura 9, demonstrando assim uma acurácia aceitável de aproximadamente 70%. Nos gráficos é possível analisar um desempenho diferente do obtido na validação dos modelos, onde a taxa de acerto da categoria ”Abaixar”era menor que a categoria ”Cair”.

A implementação do Openpose para a obtenção de dados pode ser considerado um dos grandes fatores no erro presente no modelo, não é a todo momento em que o modelo obtém as valores das juntas conforme era esperado. Por exemplo, na Figura 10 é possível observar que a ferramenta consegue obter os valores das juntas o suficiente para estimar o movimento realizado. No entanto, também há momentos em que estes valores não são obtidos, por exemplo na Figura 11, onde a ferramenta descreve a posição do pescoço, mas não dos quadris, influenciando na decisão do modelo implementado.

Quando uma parte do corpo não é detectada pelo Openpose o valor é apenas mantido em seu valor anterior, este fato influencia na distância entre os valores de distância, influenciando na decisão do modelo. Contudo, a implementação do modelo provou-se capaz de reconhecer a queda de pessoas em alguns casos, provando seu potencial como algoritmo aplicável, seria

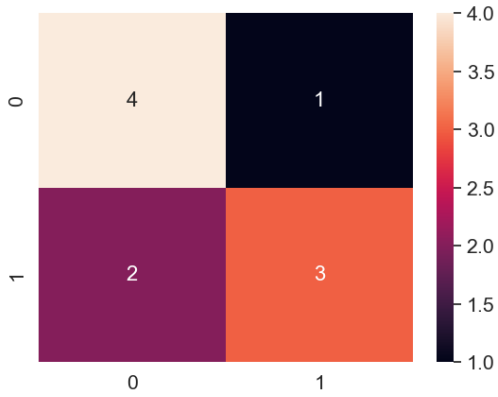

Figura 8. Análise resultados do modelo frame à frame em cenário real

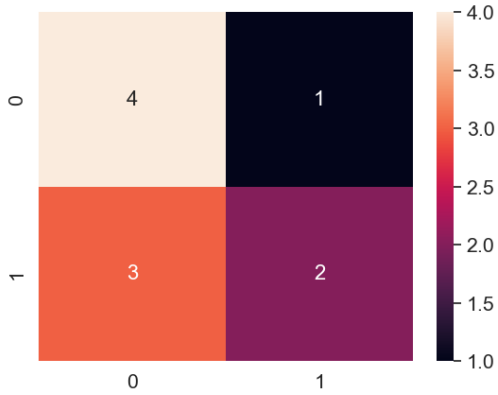

Figura 9. Análise resultados do modelo conjunto em cenário real

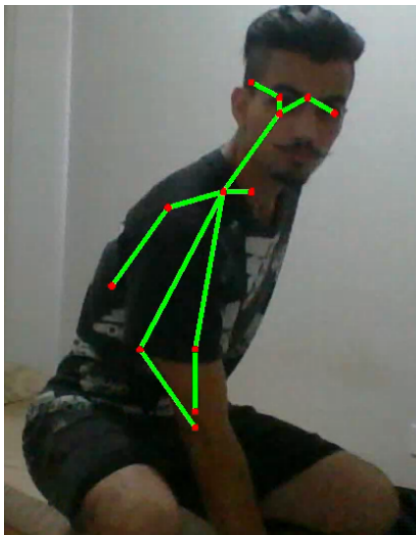

Figura 10. Exemplo de movimento Abaixar com obtenção satisfatória de dados

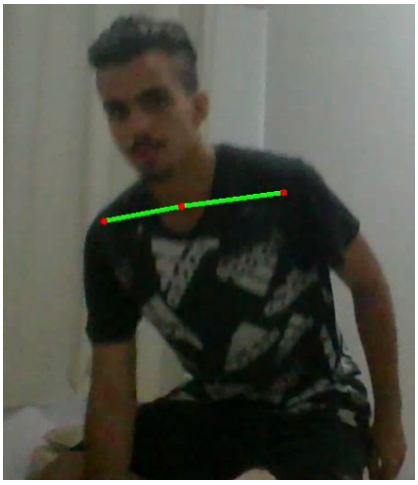

Figura 11. Exemplo de movimento Abaixar com obtenção não satisfatória de dados

interessante o aprimoramento do modelo como trabalho futuro, uma ideia que pode ser adotada é a análise do tempo entre o estado inicial e o estado final, ou até pela implementação de outra ferramenta capaz de definir com melhor precisão a pose do corpo.

V. Conclusão

Em síntese, este trabalho abordou a implementação de algoritmos de visão computacional para a identificação de quedas, utilizando câmeras como principal fonte de dados. A obtenção dos dados foi realizada por meio do OpenCv em conjunto com o OpenPose, e a classificação dos movimentos foi efetuada utilizando o algoritmo K-Nearest Neighbor (Knn) em Python. A metodologia empregada incluiu análises frame a frame e em conjunto, proporcionando uma compreensão abrangente dos movimentos analisados.

Durante a fase de validação, os resultados revelaram uma precisão satisfatória na detecção de quedas, alcançando uma acurácia de 72,09%. É relevante destacar que esse valor é um indicativo positivo do desempenho do algoritmo. A capacidade de identificar quedas com tal grau de precisão sugere que o sistema possui potencial para contribuir significativamente para a prevenção de acidentes, especialmente em ambientes frequentados por pessoas vulneráveis, como idosos.

Embora os resultados tenham revelado uma boa precisão durante a fase de validação, é crucial reconhecer os desafios inerentes à limitação na precisão do OpenPose na obtenção de dados antropométricos. A aplicação prática em cenários reais revelou a inconsistência na detecção dos pontos-chave, impactando diretamente o desempenho do modelo.

Diante desses desafios, recomendamos a exploração contínua de técnicas avançadas de processamento de imagem, a avaliação de diferentes algoritmos de aprendizado de máquina e a consideração de dados temporais para aprimorar a eficácia do sistema. Essas melhorias têm o potencial de tornar o modelo mais robusto e preciso em situações do mundo real.

Referências

[1] H. Ritchie and M. Roser. (2019) Age structure. Our World in Data. [Online]. Available: https://ourworldindata.org/age-structure

[2] I. C. M. Gonçalves, R. F. Freitas, E. C. Aquino, J. A. Carneiro, and A. do Carmo Lessa, "Tendência de mortalidade por quedas em idosos, no brasil, no período de 2000-2019," *Revista Brasileira de Epidemiologia*, vol. 25, 2022. [Online]. Available: https://scielosp.org/article/rbepid/2022.v25/e220031/

[3] M. E. Karar, H. I. Shehata, and O. Reyad, "A survey of iot-based fall detection for aiding elderly care: Sensors, methods, challenges and future trends," *Appl. Sci.*, vol. 12, no. 7, p. 3276, 2022. [Online]. Available: https://doi.org/10.3390/app12073276

[4] N. Mozaffari, J. Rezazadeh, R. Farahbakhsh, S. Yazdani, and K. Sandrasegaran, "Practical fall detection based on iot technologies: A survey," *Journal of Network and Computer Applications*, vol. 149, p. 102490, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S2542660519302355

[5] W. H. Abd, A. T. Sadiq, and K. A. Hussein, "Human fall down recognition using coordinates key points skeleton," in *2022 3rd Information Technology To Enhance e-learning and Other Application (IT-ELA)*, 2022, pp. 232–237.

[6] H. Ramirez, S. A. Velastin, I. Meza, E. Fabregas, D. Makris, and G. Farias, "Fall detection and activity recognition using human skeleton features," *IEEE Access*, vol. 9, pp. 33 532–33 542, 2021.

[7] C. K. Htoo and M. M. Sein, "Privacy preserving human fall recognition using human skeleton data," in *2023 IEEE Conference on Computer Applications (ICCA)*, 2023, pp. 276–281.