Relatório Técnico: Extração de Frames de Vídeo e Análise de Desempenho

# 1. Introdução

Este documento descreve o código utilizado para extrair frames de um vídeo, gerando uma imagem para cada segundo de vídeo, e calcular uma série de KPIs (Key Performance Indicators) relacionados ao desempenho do processo. O código é projetado para ser executado em um ambiente Google Colab, aproveitando suas capacidades de processamento para análise eficiente de vídeos.  
O objetivo deste processo é extrair um frame por segundo, realizar uma análise detalhada de desempenho e fornecer uma maneira fácil de realizar o download dos frames extraídos.

# 2. Requisitos

## 2.1. Ambiente de Execução

Google Colab: A plataforma baseada em nuvem é utilizada para permitir a execução remota e aproveitar a infraestrutura de processamento de vídeos.

## 2.2. Bibliotecas e Ferramentas

- OpenCV: Usado para manipulação de vídeos, leitura de frames e gravação das imagens extraídas.  
- psutil: Biblioteca utilizada para monitorar o uso de recursos do sistema, como memória RAM.  
- shutil: Usado para compactar os arquivos de saída, facilitando o download.  
- time: Para medir o tempo de execução do processo de extração.

# 3. Estrutura do Código

## 3.1. Etapa 1: Upload do Vídeo

A primeira etapa consiste no upload do vídeo pelo usuário. Utilizando a biblioteca `files` do Google Colab, o arquivo de vídeo é enviado e seu nome é capturado para as etapas subsequentes de processamento.  
```python  
uploaded = files.upload()  
video\_filename = list(uploaded.keys())[0]  
```

## 3.2. Etapa 2: Instalação de Bibliotecas

Em seguida, a biblioteca OpenCV é instalada caso não tenha sido previamente instalada, utilizando o comando:  
```python  
!pip install opencv-python-headless  
```  
Isso garante que o ambiente de execução tenha as dependências necessárias para manipulação de vídeos.

## 3.3. Etapa 3: Extração de Frames por Segundo

O código utiliza a biblioteca OpenCV para ler o vídeo e extrair um frame a cada segundo. O número de frames por segundo (FPS) é obtido e utilizado para determinar o intervalo entre os frames extraídos. O código salva cada frame extraído em uma pasta específica (`frames\_1fps`).  
```python  
cap = cv2.VideoCapture(video\_path)  
fps = cap.get(cv2.CAP\_PROP\_FPS)  
```

## 3.4. Etapa 4: Cálculo dos KPIs

Após a extração dos frames, o código calcula uma série de KPIs para avaliar o desempenho do processo. As KPIs extraídas são:  
1. Tempo de processamento total: O tempo total necessário para a extração de todos os frames.  
2. Uso de memória RAM: A quantidade de memória RAM utilizada durante o processamento.  
3. Espaço total dos frames extraídos: O espaço ocupado pelos arquivos de imagem extraídos.  
4. Tamanho médio por frame: O tamanho médio de cada frame extraído.  
5. Eficiência do processamento: A eficiência de extração, com base no número de frames extraídos em comparação com o total de frames do vídeo.  
6. Velocidade de extração: O número de frames extraídos por segundo durante o processamento.  
```python  
def gerar\_kpis(video\_path, pasta\_frames, tempo\_inicio, tempo\_fim, frames\_extraidos):  
 # Cálculos de KPIs  
 tempo\_total = tempo\_fim - tempo\_inicio  
 memoria\_usada = processo.memory\_info().rss / (1024 \* 1024)  
 total\_tamanho = sum(os.path.getsize(os.path.join(pasta\_frames, f)) for f in arquivos) / (1024 \* 1024)  
 tamanho\_medio\_frame = (total\_tamanho \* 1024) / len(arquivos)  
 eficiencia = (frames\_extraidos / total\_frames) \* 100  
 velocidade\_extracao = frames\_extraidos / tempo\_total if tempo\_total > 0 else 0  
```

## 3.5. Etapa 5: Compactação e Download

Após a extração dos frames e cálculo das KPIs, os frames são compactados em um arquivo `.zip` para facilitar o download. O código utiliza a biblioteca shutil para criar o arquivo compactado:  
```python  
shutil.make\_archive("frames\_extraidos", 'zip', output\_folder)  
files.download("frames\_extraidos.zip")  
```

# 4. KPIs Detalhados

As métricas calculadas são fundamentais para avaliar o desempenho do processo de extração de frames. Abaixo, explicamos cada KPI:

1. ⏱️ Tempo de Processamento Total: Mede o tempo total necessário para o processo de extração de frames, desde o início até o término da execução.  
2. 🧠 Uso de Memória RAM: Indica a quantidade de memória RAM usada durante o processamento, ajudando a entender a eficiência do código e seu impacto nos recursos do sistema.  
3. 💾 Espaço Total dos Frames Extraídos: Calcula o espaço ocupado por todos os frames extraídos, em MB. Isso pode ser útil para entender o armazenamento necessário para processar vídeos grandes.  
4. 📁 Tamanho Médio por Frame: Este KPI calcula o tamanho médio de cada imagem extraída, o que pode ser útil para otimizar o armazenamento e o tempo de processamento.  
5. 🧩 Eficiência do Processamento: Reflete a eficácia do processo, calculando a porcentagem de frames extraídos em relação ao número total de frames disponíveis no vídeo.  
6. ⚙️ Velocidade de Extração: Este KPI calcula a taxa de extração de frames, ou seja, quantos frames são extraídos por segundo durante o processamento.

# 5. Conclusão

Este código oferece uma solução eficiente para a extração de frames de vídeos e a avaliação do desempenho do processo. As métricas de desempenho (KPIs) fornecem informações valiosas sobre o tempo de processamento, o uso de memória, a eficiência e a velocidade do processo, permitindo melhorias futuras no código.  
Com a extração de frames a cada segundo, o código é ideal para analisar vídeos em baixa resolução ou quando não é necessário capturar todos os frames. A compactação dos frames extraídos também facilita o armazenamento e o compartilhamento dos dados processados.  
O código pode ser facilmente adaptado para outras necessidades, como a extração de frames a intervalos diferentes ou a adição de novos KPIs para avaliar diferentes aspectos do desempenho.

# 6. Referências

- OpenCV Documentation: https://opencv.org/  
- psutil Documentation: https://psutil.readthedocs.io/  
- Google Colab Documentation: https://colab.research.google.com/