# All media file is available for download as a zip file (See description)

import tkinter

import cv2 # pip install opencv-python

import PIL.Image, PIL.ImageTk # pip install pillow

from functools import partial

import threading

import time

import imutils # pip install imutils

stream = cv2.VideoCapture("clip.mp4")

flag = True

def play(speed):

global flag

print(f"You clicked on play. Speed is {speed}")

# Play the video in reverse mode

frame1 = stream.get(cv2.CAP\_PROP\_POS\_FRAMES)

stream.set(cv2.CAP\_PROP\_POS\_FRAMES, frame1 + speed)

grabbed, frame = stream.read()

if not grabbed:

exit()

frame = imutils.resize(frame, width=SET\_WIDTH, height=SET\_HEIGHT)

frame = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(frame))

canvas.image = frame

canvas.create\_image(0,0, image=frame, anchor=tkinter.NW)

if flag:

canvas.create\_text(134, 26, fill="black", font="Times 26 bold", text="Decision Pending")

flag = not flag

def pending(decision):

# 1. Display decision pending image

frame = cv2.cvtColor(cv2.imread("pending.png"), cv2.COLOR\_BGR2RGB)

frame = imutils.resize(frame, width=SET\_WIDTH, height=SET\_HEIGHT)

frame = PIL.ImageTk.PhotoImage(image=PIL.Image.fromarray(frame))

canvas.image = frame

canvas.create\_image(0,0, image=frame, anchor=tkinter.NW)

# 2. Wait for 1 second

time.sleep(1.5)

# 3. Display sponsor image

frame = cv2.cvtColor(cv2.imread("sponsor.png"), cv2.COLOR\_BGR2RGB)

frame = imutils.resize(frame, width=SET\_WIDTH, height=SET\_HEIGHT)

frame = PIL.ImageTk.PhotoImage(image=PIL.Image.fromarray(frame))

canvas.image = frame

canvas.create\_image(0,0, image=frame, anchor=tkinter.NW)

# 4. Wait for 1.5 second

time.sleep(2.5)

# 5. Display out/notout image

if decision == 'out':

decisionImg = "out.png"

else:

decisionImg = "not\_out.png"

frame = cv2.cvtColor(cv2.imread(decisionImg), cv2.COLOR\_BGR2RGB)

frame = imutils.resize(frame, width=SET\_WIDTH, height=SET\_HEIGHT)

frame = PIL.ImageTk.PhotoImage(image=PIL.Image.fromarray(frame))

canvas.image = frame

canvas.create\_image(0,0, image=frame, anchor=tkinter.NW)

def out():

thread = threading.Thread(target=pending, args=("out",))

thread.daemon = 1

thread.start()

print("Player is out")

def not\_out():

thread = threading.Thread(target=pending, args=("not out",))

thread.daemon = 1

thread.start()

print("Player is not out")

# Width and height of our main screen

SET\_WIDTH = 650

SET\_HEIGHT = 368

# Tkinter gui starts here

window = tkinter.Tk()

window.title("CodeWithHarry Third Umpire Decision Review Kit")

cv\_img = cv2.cvtColor(cv2.imread("welcome.png"), cv2.COLOR\_BGR2RGB)

canvas = tkinter.Canvas(window, width=SET\_WIDTH, height=SET\_HEIGHT)

photo = PIL.ImageTk.PhotoImage(image=PIL.Image.fromarray(cv\_img))

image\_on\_canvas = canvas.create\_image(0, 0, ancho=tkinter.NW, image=photo)

canvas.pack()

# Buttons to control playback

btn = tkinter.Button(window, text="<< Previous (fast)", width=50, command=partial(play, -25))

btn.pack()

btn = tkinter.Button(window, text="<< Previous (slow)", width=50, command=partial(play, -2))

btn.pack()

btn = tkinter.Button(window, text="Next (slow) >>", width=50, command=partial(play, 2))

btn.pack()

btn = tkinter.Button(window, text="Next (fast) >>", width=50, command=partial(play, 25))

btn.pack()

btn = tkinter.Button(window, text="Give Out", width=50, command=out)

btn.pack()

btn = tkinter.Button(window, text="Give Not Out", width=50, command=not\_out)

btn.pack()

window.mainloop()