**FALL DETECTION**

**MODELS USED:**

1. YOLOv8x (available on “https://docs.ultralytics.com/models/yolov8/#performance-metrics”)
2. YOLOv8x-pose-p6 (available on “https://docs.ultralytics.com/tasks/pose/”)

**REQUIREMENTS:**

1. Pytorch (version depends on whether GPU is available or not)
2. Opencv
3. Ultralytics
4. Python file of code
5. Input: video file(mp4)
6. Output: video file(mp4)

**FORMULA:**

Aspect Ratio = Height of the Bounding Box / Width of the Bounding Box

Normalized X coordinate= X coordinate / Width of frame

Normalized Y coordinate= Y coordinate / Height of frame

Height change= Height of bounding box in current frame - Height of bounding box in previous frame

Width change= Width of bounding box in current frame - Width of bounding box in previous frame

**THEORY:**

The current version of code classifies a person into two classes: STANDING and FALLEN, but uses two other classes CROPPED and FALLING for making accurate predictions.

STANDING person has generally has a Aspect Ratio higher than a FALLEN person. But using a single aspect ratio to differentiate between STANDING and FALLEN classes gives different results depending on the whether the video is zoomed in or not. So we employed other ways to detect fall, which were changes in dimension of bounding box and movement of keypoints using yolo-pose model.

Solution employed in code:

1. Identify if a person is FALLING or not. Condition for FALLING used is:
   1. Height change <-5(adjustable)
   2. Width change >3(adjustable)

Differentiate between FALLING and CROPPED:

* Height decrease + Width does not change -> CROP
* Height decrease + Width increases -> FALLING

1. If a person has ever been detected as FALLING in the entire video use Aspect ratio=1.5 to differentiate between STANDING and FALLEN.

* Aspect ratio>1.5 : STANDING
* Aspect ratio<1.5 : FALLEN (more lenient)

If a person has never been detected as FALLING in the entire video use Aspect ratio=1.25 to differentiate between STANDING and FALLEN.

* Aspect ratio>1.25 : STANDING (more lenient)
* Aspect ratio<1.25 : FALLEN

1. If a person is detected as FALLEN without FALLING (Aspect ratio<1.25 : FALLEN) further use relevant keypoints(shoulder+knee+waist) from yolo-pose to confirm fall.
2. If the prediction from yolo-pose is different from the original prediction display the prediction from yolo-pose.

**HOW TO DETECT FALL USING KEYPOINTS:**

1. Extract the coordinates of the of the relevant keypoints (shoulder, knee and waist)
2. Normalize all the coordinates so that the model is generalized across all levels of zoom.
3. Take x-coordinates of two shoulders -> add them ->divide the sum by 2 to get the average x coordinate of shoulder.
4. Similarly find average x coordinates of knee and waist.
5. Check for alignment:

* If (average x-coordinate of shoulder- average x-coordinate of waist < threshold value) -> Aligned
* If (average x-coordinate of waist- average x-coordinate of knee < threshold value) -> Aligned

Typical threshold value=0.03

1. If (average y-coordinate of shoulder> average y-coordinate of waist) ->Shoulder above waist
2. Check condition for STANDING and return the prediction:

* (if knee and waist or waist and shoulder are aligned) and (shoulder are above waist) -> STANDING
* Else -> FALLEN

1. If relevant keypoints are not detected return FALLEN

**HOW TO MERGE YOLO-POSE AND YOLO:**

Since both yolov8x and yolov8x-pose-p6 are compatible with each other they can be imported in the same virtual environment with one file.

The steps followed to confirming fall were:

1. Use YOLOv8x model to get the bounding box coordinates of a person if it is detected as FALLEN without FALLING.
2. Use the coordinates of the particular bounding box and give them as an input to a the function which implements yolo-pose model.
3. Run yolo-pose on the area inside those coordinates and check for the condition that detects fall using keypoints.

**IMPOROVEMENTS POSSIBLE**

1. Can change the thresholds values to increase accuracy for construction site videos.
2. Colour scheme can be improved, we can set-up some sort of priority system for colours (FALLEN>FALLING>STANDING>CROPPED).
3. The current version of code tracks a person using the centroid of the bounding box. In every frame it calculates the centroid of bounding box and the centroid of bounding box nearest to it in the previous frame is assumed to be the same person. Try using some tracking algorithm like “sort” to track every person more accurately.

**FOLDER STRUCTURE:**

WAS FALLING

START

NO

NOT FALLING

MARK AS “WAS FALLING”

YES

NO

STANDING

YES

ASPECT RATIO >1.5

FALLING

INCONCLUSIVE

[(KNEE WAIST ALIGNED) OR (WAIST SHOULDER ALIGNED)]

AND

(SHOULDER ABOVE WAIST)

FALLEN

FALLEN

NO

STANDING

YES

NO

ASPECT RATIO >1.25

YES

STANDING

HEIGHT CHANGE < -5

WIDTH CHANGE > 3

FALLEN

NO

YES