**ARMY INSTITUTE OF TECHNOLOGY**

**Problem Statement :** HOME PARK ASSIANT (HPA)

**TEAM MEMBER :**

1)Vikas Saran

2) Bharat Choudhary

PROJECT DOCUMENTATION

Project Manager: Praneeth N Reddy B V and Linkesh Suresh Babu

Submission Date: 11 February 2024

| VERSION | REVISION DATE | DESCRIPTION OF CHANGE | AUTHOR |
| --- | --- | --- | --- |
| 1 | 3 February 2024 | Tried Object detection using YOLO v5 | Vikas saran , Bharat Choudhary |
| 2 | 4February 2024 | Tried Object detection using YOLO v4 tiny | Vikas saran , Bharat Choudhary |
| 3 | 6 February 2024 | Moved to Frame differencing | Vikas saran , Bharat Choudhary |
| 4 | 7 February 2024 | Tried Background substraction | Vikas saran , Bharat Choudhary |
| 5 | 9 February 2024 | Bitwise and of frame differencing with background substraction | Vikas saran , Bharat Choudhary |

|  |  |  |  |
| --- | --- | --- | --- |
| **PREPARED BY** | Team VisionX | **TITLE** | HOME PARK ASSIANT (HPA) |

Tabl of Contents

1. METHODS USED TO DESIGN STRATEGY 6 2. LIBRARIES AND FUNCTIONLITIES USED 3. HOW TO RUN THE CODE

4. FLOW CHART AND DESCRIPTIONS 6 5. CONCLUSION

**1.METHODS USED TO DESIGN STRATEGY**

>>Frame differencing

**Capture Frames:** The process begins by capturing consecutive frames from a video stream or a series of images.

**Convert Frames:** The frames are converted to a common format, such as grayscale or RGB, to enable comparison.

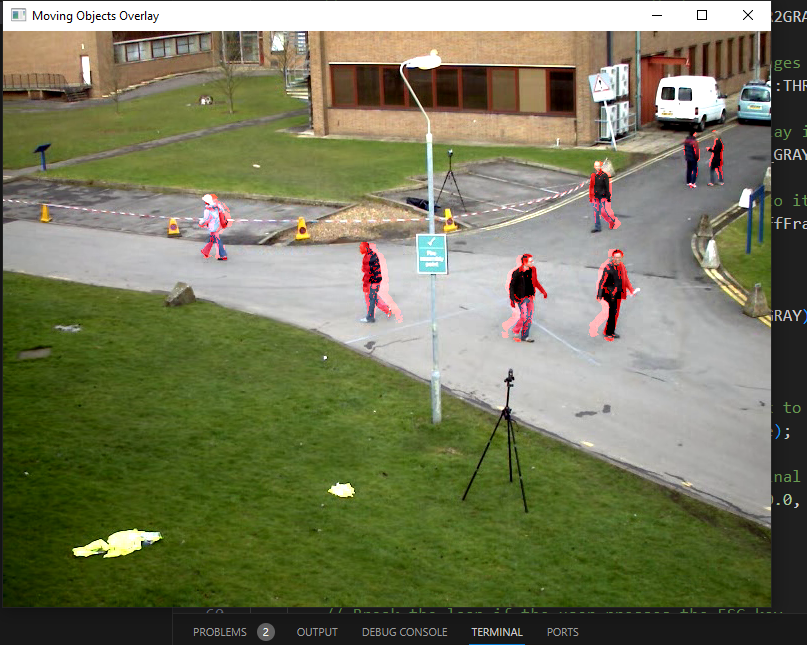
Compute Absolute Difference:

The absolute difference between corresponding pixel values in two consecutive frames is calculated.

**Thresholding:** created a binary mask that highlights areas where the differences exceed a certain threshold. This helps in isolating regions of significant motion or change.

**Post-processing:** Additional post-processing steps may be applied to the binary mask to remove noise or refine the detected motion regions. Common post-processing techniques include morphological operations like dilation and erosion.

**Visualize Results:** The final output, which typically consists of the binary mask highlighting areas of motion or change, can be visualized, or used for further analysis and processing.



>>Background subtraction

**Initialization:** In the initialization phase, the first few frames of the video are used to create an initial background model. This background model is used as a reference for subsequent frames.

**Frame differencing:** In this step, each frame of the video is compared to the background model. The absolute difference between the pixel values of the current frame and the corresponding pixel values in the background model is computed.

**Thresholding:** The absolute difference image obtained from frame differencing is typically threshold to create a binary mask. Pixels with values above a certain threshold are considered part of the foreground (moving objects), while pixels with values below the threshold are considered part of the background.

**Post-processing:** Post-processing techniques such as morphological operations (e.g., dilation, erosion) are often applied to the binary mask to remove noise and smooth the boundaries of the detected objects.

**Object detection and tracking:** Once the foreground objects have been detected using background subtraction, additional techniques such as object detection and tracking can be applied to further analyze and identify the objects in the scene.

A screenshot of a computer screen

Description automatically generated

On taking bitwise and of frame differencing with background subtraction and drawing contours we get, more precise



# **2.LIBRARIES AND FUNCTIONLITIES USED**

|  |
| --- |
| * OPENCV   - Frame differencing  - Background substruction   * Object detection using YOLO v4 tiny |

>>**LIBRARIES**

* **<opencv2/opencv.hpp>**:

This is the main header file for OpenCV (Open Source Computer Vision Library). OpenCV is a popular open-source computer vision and machine learning software library. It provides various tools and functions for image processing, computer vision tasks, and machine learning algorithms. Including this header file gives access to many of OpenCV's core functionalities

* **<opencv2/highgui/highgui.hpp>**:

This header file provides functions for graphical user interface (GUI) operations in OpenCV. It includes functions for displaying images, creating windows, capturing video streams from cameras, and handling user input events such as mouse clicks and keyboard inputs

* **<opencv2/imgproc/imgproc.hpp>**:

This header file contains functions for image processing operations in OpenCV. It includes functions for various tasks such as image filtering, edge detection, morphological operations, color space conversion, geometric transformations, and many more. These functions are essential for manipulating and analyzing images in computer vision applications

**>>object detection using yolo v4 tiny**

* **Input Image or Video:**

Object detection begins with an input image or video frame in which you want to detect objects.

* **Preprocessing:**

The input image is preprocessed to meet the requirements of the YOLOv4 Tiny model. This typically involves resizing the image to a fixed size and normalizing pixel values.

* **Object Detection Process:**

YOLOv4 Tiny adopts a single neural network architecture that directly predicts bounding boxes and class probabilities for the objects present in the image. It divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell.

* **Bounding Box Prediction:**

It predicts bounding boxes for objects along with confidence scores ,class probabilities for each bounding box , indicating the likelihood that the detected object belongs to a particular class.

* **Non-Maximum Suppression (NMS):**

After obtaining bounding boxes and their associated confidence scores, a technique called non-maximum suppression .

* **Output:**

The final output typically includes the bounding boxes drawn around the detected objects along with their class labels and confidence scores.

* **Integration with OpenCV:**

YOLOv4tiny can be integrated with the OpenCV library to perform object detection tasks in c++ applications.

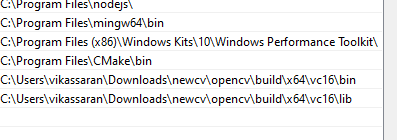
OpenCV provides functionalities for loading images and videos, as well as processing the outputs of the YOLOv5 model, such as drawing bounding boxes and labels on the detected objects.

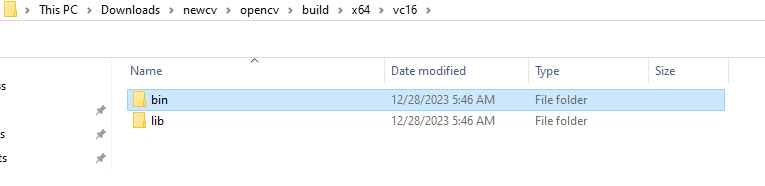


**3.HOW TO RUN THE CODE**

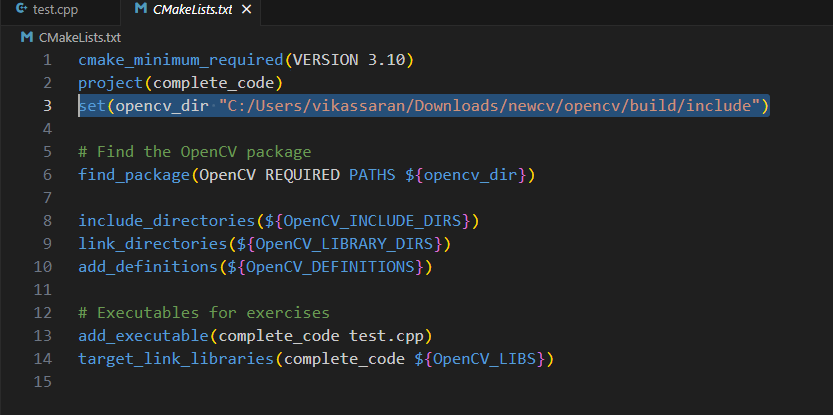
* **Prerequisite for the program**

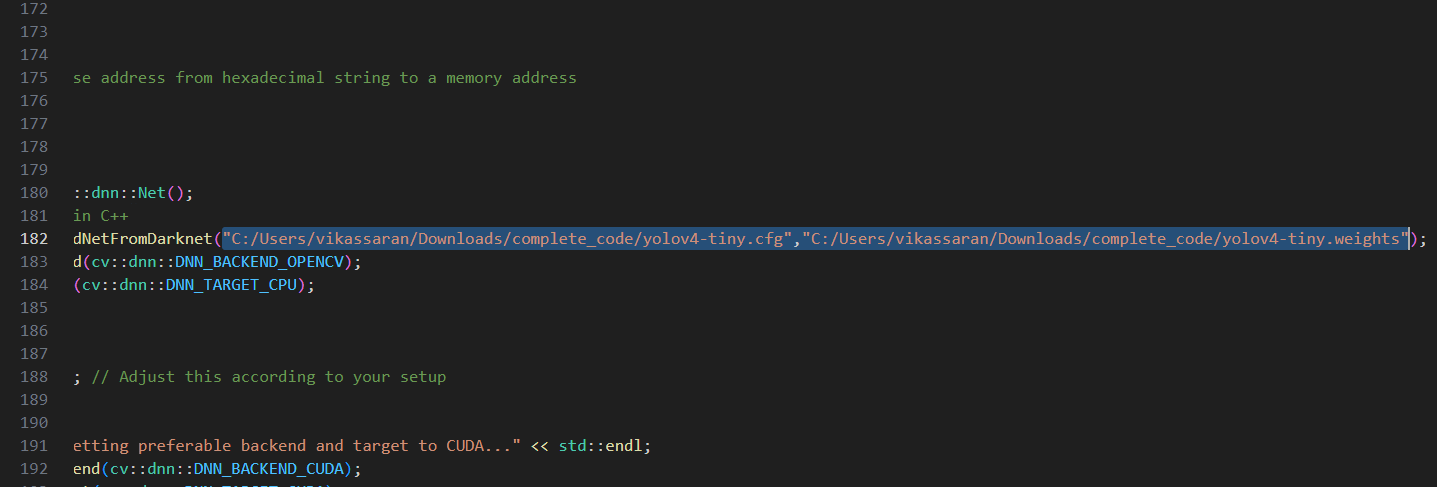
**Opencv and cmake installed latest version and file address set in environment variable a64bit compiler needed to compile opencv**

****



* **Extract the zip file and open Cmake file and change the opencv include file address according to what is in yours pc and want to use gpu**A group of icons of a computer

  Description automatically generatedA screenshot of a computer program

  Description automatically generated
* **Change the yolov4-tiny and yolov4-tiny.weights path in the main file .**A group of icons of a computer

  Description automatically generated
* **Finally to run the code either build and run from vs code "C:/Users/vikassaran/Downloads/complete\_code/build/Debug/complete\_code.exe" "C:/Users/vikassaran/Downloads/hacathon/Problem\_Statement\_1/video.mp4" "0x1e1780" ///X64bitaddress**

**A screenshot of a computer

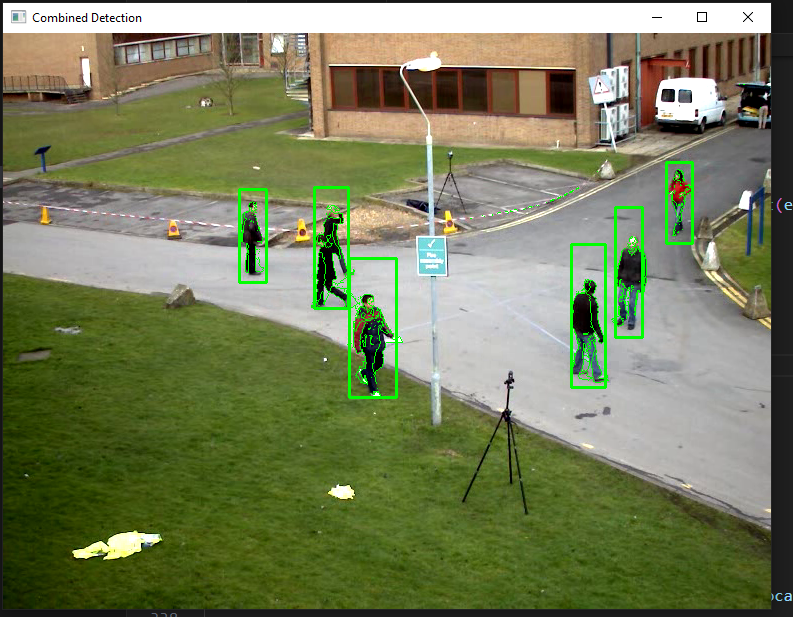
Description automatically generated**

**Or go to build/file/debug/complete\_code.exe file to execute**

A screenshot of a computer

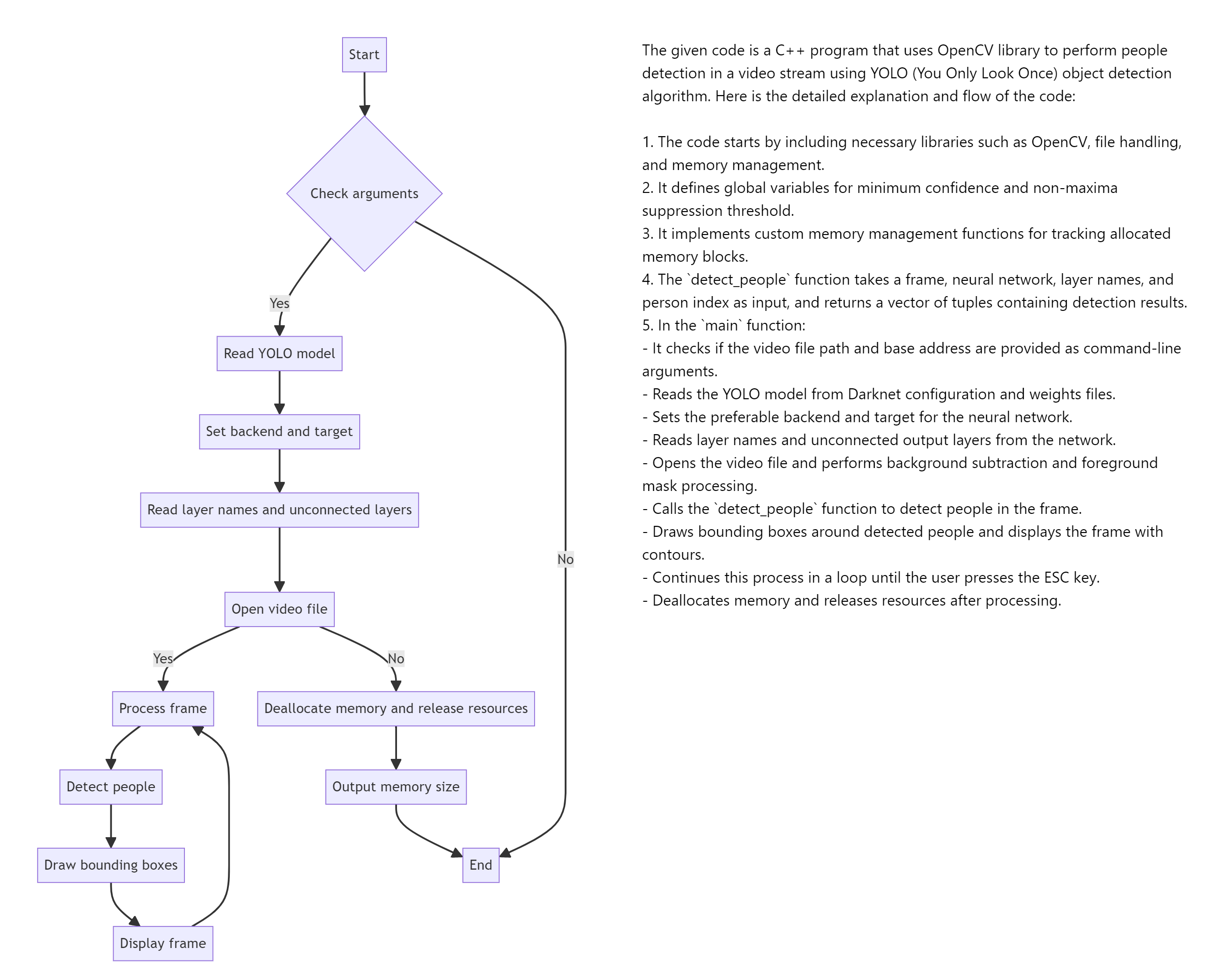
Description automatically generatedA screenshot of a computer program

Description automatically generated

* **Enter the full address of the video a 64bit address within the “” as in the photo and press enter. You should get output ,the same could be achieved by using visual studio with 64bit compiler and c++developr kit also and memory allocation and delocation and total allocated memory shown in command prompt at end is n0 byte as it’s all delocated**

\*\*You will also find the video of the output in the zip file\*\*

4. FLOW CHART AND DESCRIPTIONS



**Header includes:** The code includes necessary header files from OpenCV and other standard C++ libraries.

**Global Variables:** The code declares global variables for minimum confidence (MIN\_CONF) and non-maximum suppression threshold (NMS\_THRESH). These variables determine the confidence threshold for person detection and the threshold for removing overlapping bounding boxes, respectively.

**Custom Memory Management:** The code defines custom memory management functions (customAllocate and customDeallocate) to track memory allocation and deallocation using malloc and free. It also maintains a map of allocated memory blocks and their sizes.

**Memory allocation:**

Function Signature:

The function is declared with a return type of void\*, indicating that it returns a pointer to the allocated memory block.

It takes a single parameter size of type std::size\_t, which specifies the size (in bytes) of the memory block to be allocated.

Memory Allocation:

Inside the function body, memory is allocated using the malloc function, which is a standard C function for dynamic memory allocation.

**Memory Tracking:**

After allocating memory, the function tracks the allocated memory block and its size.

It stores the pointer to the allocated memory block (ptr) along with its size (size) in a data structure (presumably a map or similar container).

In the provided code, a std::map<void\*, std::size\_t> named allocatedMemory is used to store the mapping of memory addresses to their respective sizes.

Additionally, the total allocated memory size (totalAllocatedMemory) is updated by adding the size of the newly allocated memory block.

**detect\_people Function**: This function takes a frame, YOLOv4 Tiny network, layer names, and person index as input parameters. It performs object detection using the YOLO model on the input frame to detect people. It returns a vector of tuples containing the confidence score, bounding box coordinates, and centroid of detected people.

**main Function:**

Parses command-line arguments for the video file path and base address.

Loads the YOLOv4 Tiny model from Darknet configuration and weights files.

Performs background subtraction and people detection on each frame of the video.

Uses background subtraction to detect moving objects in the video.

Applies YOLOv4 Tiny model to detect people in the foreground region.

Draws bounding boxes around detected people on the frame.

Displays the processed frame with real-time monitoring.

Tracks memory allocation and deallocation for various data structures and objects.

Releases resources, closes video file, and destroys OpenCV windows upon program completion.

**Frame Differencing:**

Frame differencing is a technique used to detect motion in video sequences.

It involves computing the absolute difference between consecutive frames.

In the code, frame differencing is performed using the OpenCV function cv::absdiff(prevGrayFrame, grayFrame, fgMaskDiff), where:

prevGrayFrame is the grayscale version of the previous frame.

grayFrame is the grayscale version of the current frame.

fgMaskDiff is the resulting difference mask.

**Background Subtraction:**

Background subtraction is a method used to separate foreground objects from the background in a video sequence.

It involves modeling the background of a scene and identifying pixels that significantly differ from the background.

In the code, background subtraction is applied using the OpenCV cv::BackgroundSubtractorMOG2 class:

A background subtractor object bgSubtractor is created using the MOG2 algorithm.

The apply() method of the background subtractor is called to obtain a foreground mask (fgMaskSub) indicating moving objects.

Combining Results:

After performing frame differencing and background subtraction, the code combines the resulting foreground masks (fgMaskDiff and fgMaskSub) using bitwise AND operation:

cv::bitwise\_and(fgMaskDiff, fgMaskSub, fgMaskCombined) combines the two masks to obtain a final foreground mask fgMaskCombined.

This operation helps to refine the detection of moving objects by considering both frame differences and background subtraction.

Contour Detection and Visualization:.

6