Department of Electrical & Electronic Engineering (EEE)
Islamic University of Technology (IUT)
Organisation of Islamic Cooperation (OIC)

Md. Rahib-Bin-Hossain 200021241

Md. Muhidur Rahman 200021242

Shafin Ibnul Mohasin 200021244

K.M. Sirazul Monir 200021247

EEE 4602
Signals And Systems Lab

Project:

Real-Time Moving Object Detection and Background Removal in Video Streams using MATLAB.

1. Introduction

There are three types of methods mainly used in moving object detection. These methods are- the frame subtraction method, the background subtraction method and the optical flow method. In the Frame subtraction method, the difference between two consecutive images is taken to determine the presence of moving objects. The calculation in this method is very simple and easy to develop. But in this method, it is difficult to obtain a complete outline of moving object; therefore, the detection of moving object is not accurate. In the Optical flow method, calculation of the image optical flow field is done. The clustering processing is done according to the optical flow distribution characteristics of image. From this, the complete movement information of moving body is found and it detects the moving object from the quantity of calculation, poor anti-noise performance makes it unsuitable for real-time applications. The background subtraction method is the method in which the difference between the current image and background image is taken for the detection moving objects by using simple algorithm. But it is very sensitive to the changes which occur in the external environment and it also has poor anti interference ability. One advantage of this method is, it can provide the most complete object information in the case of the background is known. In the background subtraction method, in a single static camera condition, the dynamic background modeling is combined with dynamic threshold selection method which depends on the background subtraction. The background is updated on the basis of accurate detection of object.

A) Frame Separation

Frame processing is the first step in the background subtraction algorithm, the purpose of this step is to prepare the modified video frames by removing noise and unwanted objects in the frame in order to increase the amount of information gained from the frame. Preprocessing of an image is a process of collecting simple image processing tasks that change the raw input video info into a format. This can be processed by subsequent steps. Preprocessing of the video is necessary to improve the detection of moving objects, for example; by spatial and temporal smoothing, snow as moving leaves on a tree, can be removed by morphological processing of the frames after the identification of the moving object.

B) Moving Object Detection

Background subtraction is particularly a commonly used technique for motion segmentation in static scenes. It attempts to detect moving regions by subtracting the current image pixel-by-pixel from a reference background image that is created by averaging images over time in an initialization period. The pixels are classified as foreground where the difference is above a threshold. After creating a foreground pixel map, some morphological post processing operations such as erosion, dilation and closing are performed to reduce the effects of noise and enhance the detected regions. The reference background is updated with new images over time to adapt to dynamic scene changes. There are different approaches to the basic scheme of background subtraction in terms of foreground region detection, background maintenance and post processing.

The simple version of this scheme where a pixel at location (x, y) in the current image, it is marked as foreground if is satisfied.

$$|It(x, y) – Bt (x, y)| > \tau \text{ ----------(1)}$$

Where, τ is a predefined threshold. The background image Bt is up- dated by the use of an Median filter as follows:

$$Bt +1 = \alpha It + (1 - \alpha) Bt\text{----------(2)}$$

The foreground pixel map creation is followed by morphological closing and the elimination of small-sized regions. Although background subtraction techniques perform well at extracting most of the relevant pixels of moving regions even, they stop, they are usually sensitive to dynamic changes when, for instance, stationary objects uncover the back- ground (e.g. a parked car moves out of the parking lot) or sudden illumination changes occur.

a) Background Modeling In the background modeling process, the reference background image and some parameters associated with normalization are computed over a number of static background frames. The background is modeled statistically on a pixel-by-pixel basis. A pixel is modeled by a 4-finite sequence of pixels Ei; si; ai; bi where Ei is the expected color value, si is the standard deviation of color value which is defined in ai is the variation of the brightness distortion, and bi is the variation of the chromaticity distortion of the ith pixel. The expected color value [6] of pixel i is given by $Ei = [\mu R(i), \mu G(i), \mu B(i)]\text{----------(3)}$ Where μR(i), μG(i) and μB(i) are the arithmetic means of the ith pixel's red, green and blue values computed over N background frames. So far, we have defined Ei and si.	**b) Background Update** For accurately extracting the moving object the background needs to be updated in real time and the background model can better adapt to light changes. In the proposed method, the update algorithm is as follows: In the moving object detection, the pixels judged as belonging to the moving object maintain the original background gray values, not be updated. We update the background model according to following rule for the pixels which are judged to be the background $Bk+1(x,y) = \beta Bk(x,y) + (1-\beta) Fk(x,y)$ ----------4 Where B(x,y) is background image, Fk(x,y) is current image and F3 E (0,1) is update coefficient, in this paper f3 = 0.004. Fk(x,y) is the pixel gray value in the current frame. Bk(x,y) and Bk+1(x,y) are respectively the Background value of the current frame and the next frame. As the camera is fixed, the background model can remain relatively stable at one position for very long period of time. Using this method, we can avoid the unexpected phenomenon of the Background, such as the sudden appearance of something in the background which is not included in the original background. Moreover, the impact brought by light, weather and other changes in the external environment can be effectively adapted by the updating of pixel gray value of the background.
c) Moving Object Extraction When the background image B(x, y) is obtained, subtract the background Image B(x,y) from the current frame Fk (x, y). Set threshold as T. If the pixel difference is greater than threshold T, then determines that the pixels appear in the moving	**d) Extraction of Moving Human Body** Some accurate edge regions are got after doing the median filtering and morphological operations. But the moving human body regions could not be determined. By observation, we can find out that when moving object appears,

object, otherwise, as the background pixels. The moving object can be detected after threshold operation. Its expression is as follows: *Dk (x, y) = {1 | Fk(x,y) – Bk-1(x,y)| T -------(5)* ={0 others Where, Dk (x, y) is the binary image of differential results. T is gray-scale threshold. Its size determines the accuracy of object identification. As in the algorithm T is a fixed value, only for an ideal situation, is not suitable for complex environment with lighting changes. Therefore, this paper proposes the dynamic threshold method, we dynamically change the threshold value according to the lighting changes of the two images obtained. On this basis, add a dynamic threshold T to the above algorithm.	shadow will appear in some regions of the scene. The presence of shadow it is difficult to extract the moving object accurately. By analyzing the characteristics of motion detection, we combine the projection operator with the previous methods [6]. Based on the results of the methods above, adopting the method of combining vertical with horizontal projection to detect the height of the motion region. This can remove the impact of the shadow to a certain degree. Then we analyze the vertical projection value and set the threshold value (determined by experience) to remove the pseudo-local maximum value and the pseudo local minimum value of the vertical projection [7] to determine the number and width of the body in the motion region, we will get the moving human body with precise edge. We are assuming that people in the scene are all in upright-walking state.

C) Detection of Moving Objects in the Video

Motion detection has been done using spatio-temporal differencing. For motion detection based on the spatio-temporal filter, the motion is characterized via the entire three-dimensional (3D) spatio-temporal data volume spanned by the moving person in the image sequence. Its advantages are low computational complexity and a simple implementation process. Here, Motion detection is carried out by the background elimination algorithm as follows.

D) Background Elimination

The background subtraction system is used to provide foreground image through the threshold of difference image between the current image and reference image. As the reference image is the previous frame, this method is called temporal differencing. The temporal differencing is very adaptive to dynamic environment. Background elimination was carried out using mean squared error concept.

2. Proposed Methodology

A) Algorithm for background Elimination

1. Begin

2. Read video using *mmreader* command in MATLAB.

3. Extract number of frames, height and width of frames.

4. Pre-allocate output video structure with all its elements assigned value 0.

5. Divide each frame into blocks of size 16x16 each.

6. Compare blocks of first frame with the corresponding blocks of each frame using mean squared error (MSE) concept.

7. If MSE is less than 5 percent the blocks are considered to be matched.

8. If more than half the corresponding blocks match, the blocks are considered to be a part of background and are made white in the output video structure.

9. If none or less than half the number of corresponding blocks does not match, they are considered to be moving object and original video values are assigned to those blocks in the output video structure.

10.The output video now contains only the moving object and background is eliminated.

11.End

B) Human Detection from background eliminated video

A shape-based approach for classification of objects is used following background subtraction based on frame differencing. The goal is to detect the humans for threat assessment. The target intruder is classified as human or animal or vehicle based on the height to width ratio (H/W) of the moving object detected during background subtraction.

a) Algorithm for human detection

1. Begin

2. Read the original and background eliminated videos using *mmreader*

function in MATLAB.

3. Extract the number of frames and frame size.

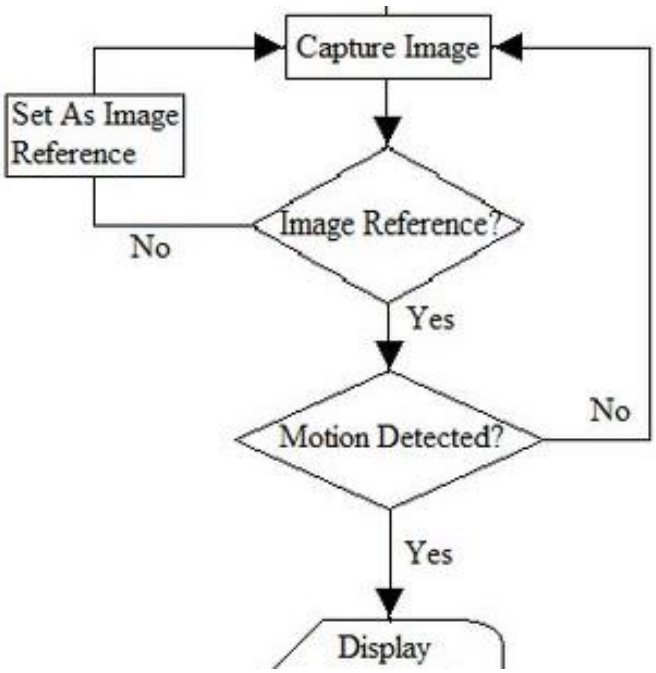
4. Pre-allocate output video structure assigning zeros to all its elements.

5. Make the pixels corresponding to moving object white and the rest black.

6. For each frame number of blocks containing moving object are checked to satisfy H/W ratio depending on area covered by camera.

7. Draw top, bottom, left and right lines in red color to highlight the detected human.

8. End



3. MATLAB CODE

```
% Task 1: Read the video
videoFile = 'project.mp4'; % Replace with your video file path
videoObj = VideoReader(videoFile);

% Initialize the foreground detector
foregroundDetector = vision.ForegroundDetector('NumGaussians', 5, ...
    'NumTrainingFrames', 100, 'MinimumBackgroundRatio', 0.5, 'LearningRate',
0.005);

% Initialize the blob analysis object
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', false, 'CentroidOutputPort', false, ...
    'MinimumBlobArea', 500);

% Create a video player to display the video frames
videoPlayer = vision.VideoPlayer;

% Read the first frame and use it as the initial background model
backgroundFrame = readFrame(videoObj);
[height, width, ~] = size(backgroundFrame);

% Ensure the number of blocks is an integer
numBlocksHeight = floor(height / 16);
numBlocksWidth = floor(width / 16);

% Adjust the frame size to be divisible by the block size
adjustedHeight = numBlocksHeight * 16;
adjustedWidth = numBlocksWidth * 16;
backgroundFrame = backgroundFrame(1:adjustedHeight, 1:adjustedWidth, :);

% Divide the first frame into blocks
backgroundBlocks = mat2cell(backgroundFrame, repmat(16, 1, numBlocksHeight),
repmat(16, 1, numBlocksWidth), 3);

% Initialize the background model
backgroundModel = zeros(size(backgroundFrame));
mseThreshold = 15; % Threshold for MSE

% Process each frame in the video
while hasFrame(videoObj)
    frame = readFrame(videoObj);

    % Compute the foreground mask
    foregroundMask = step(foregroundDetector, frame);

    % Perform morphological filtering to remove noise and fill in holes
    cleanForeground = imopen(foregroundMask, strel('Disk', 3));
    cleanForeground = imclose(cleanForeground, strel('Disk', 30));
    cleanForeground = imfill(cleanForeground, 'holes');

    % Find the boundaries of the foreground object
    boundingBoxes = step(blobAnalysis, cleanForeground);

    % Adjust the frame size to match the background model
    frame = frame(1:adjustedHeight, 1:adjustedWidth, :);
```

```
    % Divide the current frame into blocks
    currentFrameBlocks = mat2cell(frame, repmat(16, 1, numBlocksHeight),
repmat(16, 1, numBlocksWidth), 3);

    % Initialize the mask for the current frame
    frameMask = zeros(adjustedHeight, adjustedWidth);

    % Update the background model and compare each block with the
corresponding block in the background model
    for i = 1:numBlocksHeight
        for j = 1:numBlocksWidth
            % Calculate the Mean Squared Error (MSE) between blocks
            mse = immse(cell2mat(backgroundBlocks(i,j)),
cell2mat(currentFrameBlocks(i,j)));

            % If MSE is below the threshold, mark as background
            if mse < mseThreshold
                frameMask((i-1)*16+1:i*16, (j-1)*16+1:j*16) = 1;
            else
                % Update the background model for blocks that are not part
of the background
                backgroundBlocks(i,j) = currentFrameBlocks(i,j);
            end
        end
    end

    % Apply the mask to the current frame to eliminate the background
    frame(repmat(frameMask, [1, 1, 3]) == 1) = 255;

    % Draw red outlines around the detected objects
    for i = 1:size(boundingBoxes, 1)
        frame = insertShape(frame, 'Rectangle', boundingBoxes(i, :), ...
            'Color', 'red', 'LineWidth', 2);
    end

    % Display the result with background removed and object highlighted
    step(videoPlayer, frame);
end

% Release the video player
release(videoPlayer);
```

CODE EXPLANATION

1. **Loop Initialization**:
matlab while hasFrame(videoObj)
- This line starts a while loop that continues as long as there are frames available in the videoObj. - hasFrame(videoObj) checks if there is still a frame available in the video object videoObj. If there is, the loop continues; otherwise, it terminates.

2. **Reading the Frame**:
matlab frame = readFrame(videoObj);
- Inside the loop, readFrame(videoObj) reads the next frame from the video file and stores it in the variable frame.

3. **Foreground Mask Computation**:
matlab foregroundMask = step(foregroundDetector, frame);
- The foregroundDetector object applies a foreground detection algorithm to the current frame (frame), resulting in a binary mask (foregroundMask). - Foreground pixels are typically represented by ones, while background pixels are represented by zeros.

4. **Morphological Filtering**:
matlab cleanForeground = imopen(foregroundMask, strel('Disk', 3)); cleanForeground = imclose(cleanForeground, strel('Disk', 30)); cleanForeground = imfill(cleanForeground, 'holes');
- Morphological operations are performed on the foreground mask to refine the detected foreground regions. - imopen and imclose operations remove noise and fill small holes within the foreground regions. - imfill operation fills any remaining holes within the foreground regions.

5. **Blob Analysis**:
matlab boundingBoxes = step(blobAnalysis, cleanForeground);
- The blobAnalysis object analyzes the cleaned foreground mask (cleanForeground) to detect connected components or blobs. - The step function applies the blob analysis algorithm, resulting in bounding boxes (boundingBoxes) around the detected blobs.

6. **Background Model Update and Block Comparison**:
matlab for i = 1:numBlocksHeight for j = 1:numBlocksWidth mse = immse(cell2mat(backgroundBlocks(i,j)), cell2mat(currentFrameBlocks(i,j))); if mse < mseThreshold frameMask((i-1)*16+1:i*16, (j-1)*16+1:j*16) = 1; else backgroundBlocks(i,j) = currentFrameBlocks(i,j); end end end
- This nested loop updates the background model and compares each block of the current frame with the corresponding block in the background model. - Mean Squared Error (MSE) is calculated between

each pair of blocks. - Blocks with MSE below a certain threshold (MSE Threshold) are considered part of the background, while others are updated in the background model.

7. **Foreground Removal and Object Highlighting**:
matlab frame(repmat(frameMask, [1, 1, 3]) == 1) = 255; for i = 1:size(boundingBoxes, 1) frame = insertShape(frame, 'Rectangle', boundingBoxes(i, :), 'Color', 'red', 'LineWidth', 2); end
- The frameMask is applied to the current frame to remove the background, resulting in a frame with the background eliminated. - Red rectangles are drawn around the detected objects using the bounding box coordinates obtained from blob analysis, highlighting the detected objects.

8. **Displaying the Result**:
matlab step(videoPlayer, frame);
- The processed frame is displayed in real-time using the videoPlayer object, showing the result with the background removed and objects highlighted.

9. **Loop Continuation or Termination**: After processing the current frame, the loop checks again if there are more frames available in the video object. - If there are still frames remaining (hasFrame(videoObj) returns true), the loop goes back to the beginning and processes the next frame. - If there are no more frames available, the loop terminates, and the processing ends. This loop iterates through each frame of the video, processing them one by one until the end of the video file is reached. Each frame undergoes foreground detection, morphological filtering, blob analysis, background model update, foreground removal, object highlighting, and display.

In summary, foreground detection is a computer vision tool built in function in MATLAB. At first, set the background model by no. Of gaussian distribution of components fixed by the user & no. of training initial frames.

Then it reads frame from the video player 1 by 1 and detect the change in the pixel values so that it can be detected as a foreground. It's done using the MSE thresholding algorithm. If the Mean Squared Error (MSE) between the blocks is below a threshold (MSE Threshold), the block is considered part of the background. Otherwise, the background model is updated with the current frame block.

numBlocksHeight * 16 calculates the total height required to accommodate the integer number of 16-pixel high blocks. numBlocksWidth * 16 calculates the total width required to accommodate the integer number of 16-pixel wide blocks. This adjustment ensures that when the frame is divided into blocks, there are no leftover pixels at the edges, which could cause issues during subsequent. Here, height and width represent the dimensions of the video frame. The division by 16 is because the code intends to divide the frame into smaller blocks, each with a size of 16x16 pixels.

floor (height / 16) calculates how many full 16-pixel high blocks can fit vertically within the frame. floor (width / 16) calculates how many full 16-pixel wide blocks can fit horizontally within the frame. The floor function is used to ensure that if the frame's height or width is not perfectly divisible by 16, any partial block at the edges is excluded.

Blob analysis: Once the foreground pixels are detected...we needed to perform an operation that could tell us the area of a blob (similar pixels that is changing simultaneously).

Morphological filtering: background removing + denoising the foreground blobed pixel Explanation of codes foregroundMask = step (foregroundDetector, frame); step function applies the algorithm, resulting in a binary mask where foreground pixels are indicated by ones and background pixels by zeros.

frame(repmat(frameMask, [1, 1, 3]) == 1) = 255; for i = 1:size(boundingBoxes, 1) frame = insertShape(frame, 'Rectangle', boundingBoxes(i, :,….'Color', 'red', 'LineWidth', 2);

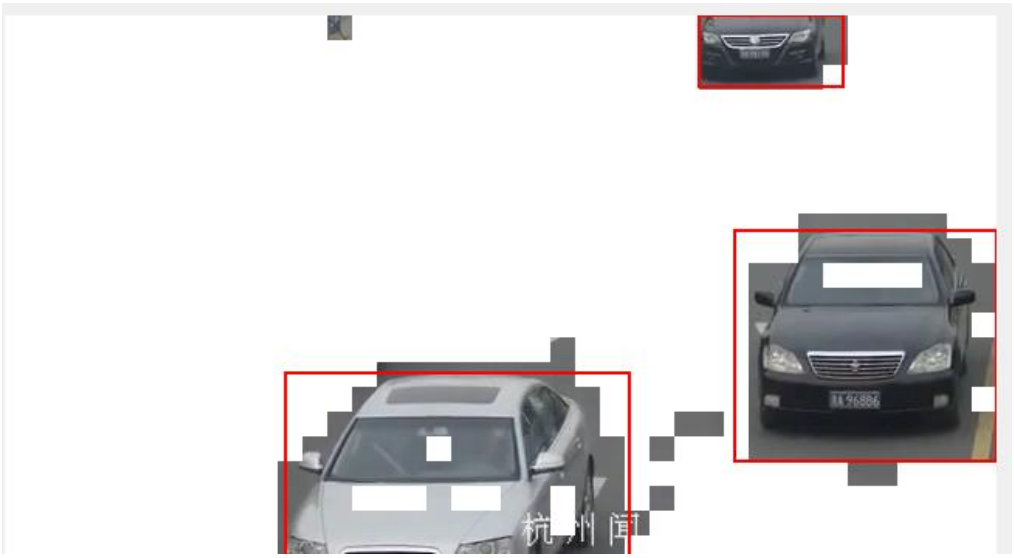
end

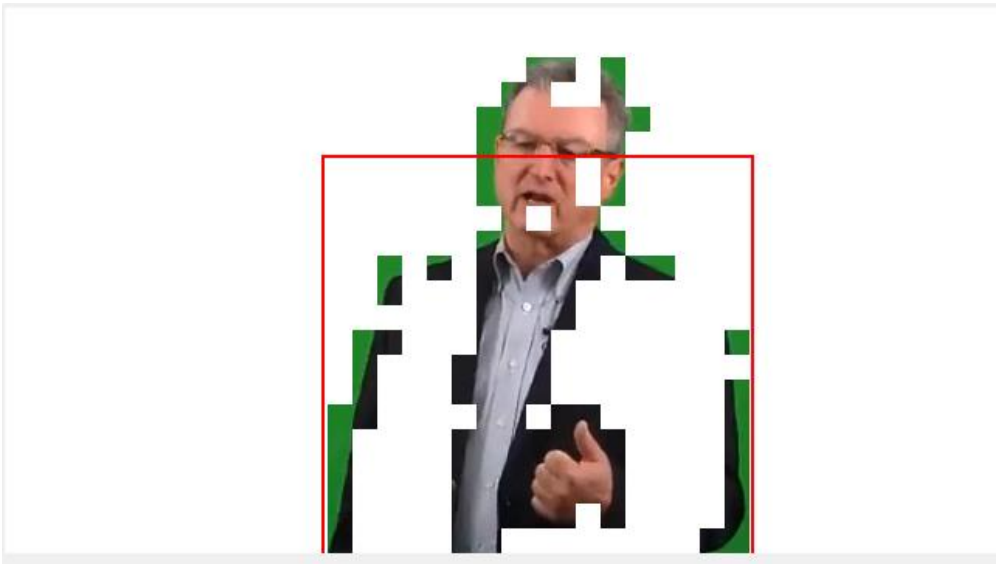
The background is removed from the current frame using the frameMask, resulting in a frame with the background eliminated. Red rectangles are drawn around the detected objects using the bounding box coordinates obtained from blob analysis, highlighting the detected object.

4. Conclusions and Results

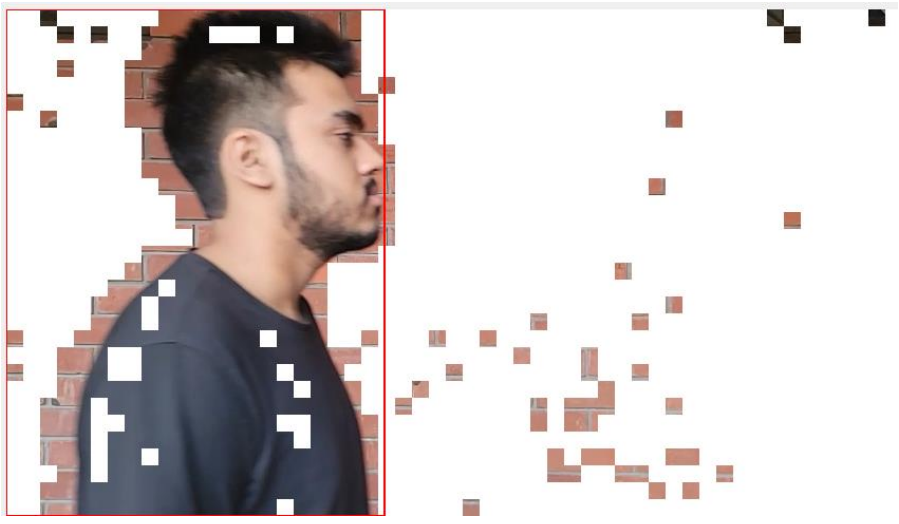
A video monitoring detecting system was thus developed successfully in this project. The evaluation cases show the accurate detection of moving object and the detection result do not affect by the body pose.

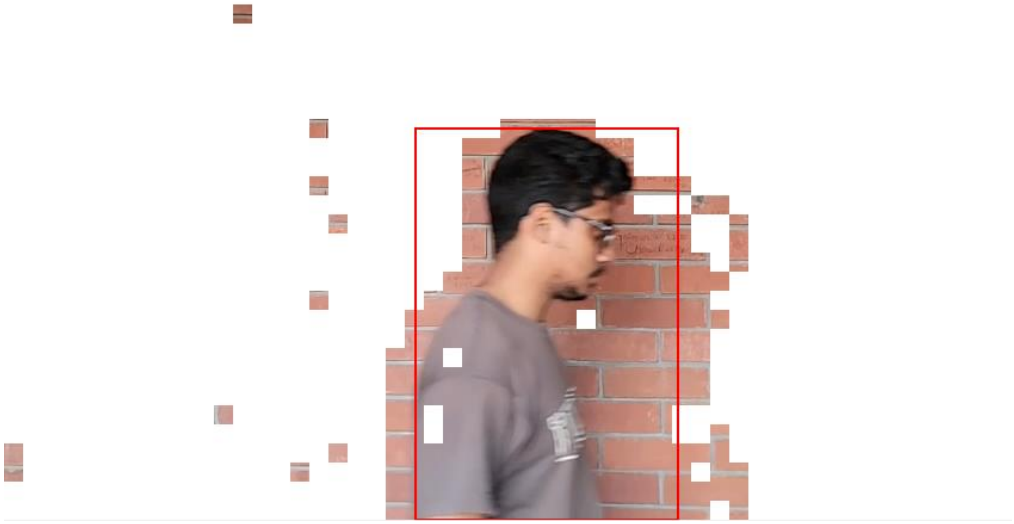
Sample Videos:





We also tested the code on one of our own filmed videos. Here are the results:





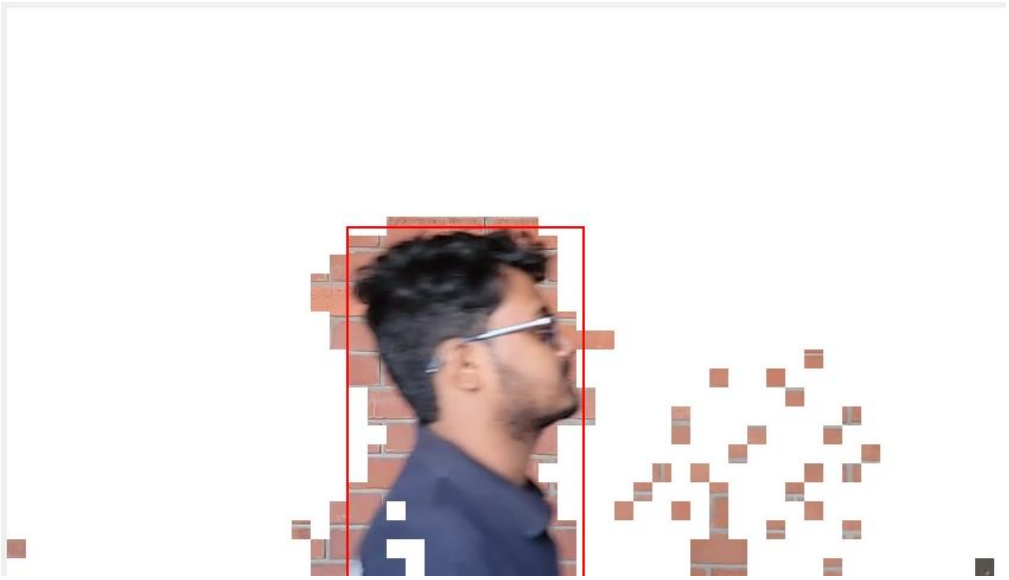


Table 1: Analysis of different cases

Case No.	*Case*	*Fail/ Success*
1	Single person case	Successful
2	Body pose change(standing to sitting) Successful	Successful
3	Two person crossing case Successful	Successful
4	Front crossing case Successful	Successful
5	Half body detection case Successful	Successful
6	Multiple persons case Successful	Successful
7	Shadow Eliminations a] Horizontal Shadow Successful b] Vertical Shadow Failed	Successful Failed
8	Moving camera Failed	Failed

Overall efficiency= (Successful cases/Total no. of cases) ×100

Overall efficiency (%)=7/8×100

Overall efficiency (%)=87.5%

Overall Efficiency of the project is about 87.5%, i.e., most of cases gave the successful result by the proposed detection algorithm. The algorithm is robust to noise and can detect the human bodies under complex circumstance.

5. Real Life Applications

a) Abnormal event detection:

The most obvious application of detecting humans in surveillance video is to early detect an event that is not normal. Abnormal events can be classified as single person loitering, multiple-person interactions (e.g. fighting and personal attacks), person-vehicle interactions (e.g. vehicle vandalism), and person-facility/location interactions (e.g. object left behind and trespassing).

b) Person detection in dense crowds and people counting:

Detecting and counting persons in a dense crowd is challenging due to occlusions. Multiple height homo graphics for head top detection can be used to overcome this problem. Advantage of the stationary cameras to perform background subtraction and jointly learn the appearance and the foreground shape of people in videos can be taken.

c) Person tracking and identification:

A person in a visual surveillance system can be identified using face recognition and gait recognition techniques. The detection and tracking of multiple people in cluttered scenes at public places are difficult due to a partial or full occlusion problem for either a short or long period of time.

d) Fall detection for elderly people:

Automatic detection of a fall for elderly people is one of the major applications of human detection in surveillance videos. Projection histograms of segmented human body silhouette can be used as the main feature for vector posture classification and the speed of fall can be used to differentiate real fall incident and an event where a person is simply lying without falling.

e) Consumer Surveillance System:

Automatic surveillance requires a sufficiently high accuracy and the computation complexity should enable a real-time performance. For such a system, we need to analyze not only the motion of people, but also the posture of the person, as the postures of the persons can provide important clues for the understanding of their activities. Hence, accurate detection and recognition of various human postures contribute to the scene understanding.

Contribution of Each member:

IDs	Names	Contribution
200021241	Md. Rahib-Bin-Hossain	Code modification & improvement to ensure high accuracy & precision through trial and error, running test cases **(Solely).** Idea finalization **(Partially).**
200021242	Md. Muhidur Rahman	Project Idea & Final project report submission along with making the PowerPoint slides **(Solely).** Idea finalization **(Partially).**
200021244	Shafin Ibnul Mohasin	Code modification & improvement to ensure high accuracy & precision through trial and error, running test cases **(Solely).** Idea finalization **(Partially).**
200021247	K.M. Sirazul Monir	Viable Idea proposition, resource & scratch code collection **(Solely).** Code modification **(Partially).**