

Vision System – Human Detection

AI31 – Coursework 1

Design

I have chosen to approach each requirement point individually to break down the task. The first task for this problem will be that of segmenting the subject, in this case the person walking through the scene, from the background. The fact that the background remains for the most part static means that the most obvious method of distinguishing movement and the presence of a person would be comparing the pixel values in each frame with the predecessors. Values that remain similar throughout the footage can be assumed as background, because the subject will always be moving.

After experimenting with a variety of segmentation and position identification techniques including sum-squared difference, frame averaging, and pixel clustering, I decided that for my data the work-flow below offered a good balance of complexity, computational cost and success rate for correctly analysing the images, proven during a small selection of individual trials during the design phase.

Logical Work-flow

Read in Video Clip → Carry out Frame differencing using background modelling method during input → Convert video data to grey-scale using a given threshold to eliminate background noise → Carry out Connected Components Analysis → Check the first element resulting from CCA to see if it's dimensions appear to be reasonable sized for a person → If so flag this frame as having a person present.

Please see *Appendix* for fully commented code detailing all commands used.

Background – Subject Segmentation

A further improvement applied during this process was the use of frame differencing, my first trials involved making calculations for each frame based on the previous¹:

$$\text{Output} = \text{abs}(\text{grey}_t - \text{grey}_{t-1})$$

This processing step is completed prior to all other processing of the file. After testing the absolute difference mechanism for background modelling I chose to instead use the Matlab function *modellingBackground()*, part of the *dch_matlab toolkit*. This function uses the background modelling approach to carry out frame differencing. The background frame is subtracted from all other frames to get the output for this function. I found that the most successful segmentation of the subject was achieved by not supplying the function with a threshold, leaving it to default. After realising the success of this frame differencing approach I subsequently replaced other elements of my design, using this robust segmentation method to remove both the static environment background and also a lot of noise.

¹ Taken from Vision Matlab Tutorial, School of Computing

Subject Identification and Tracking

Following the steps above I need to move on to tackle the problem of identifying the subject within the scene, this is now a smaller problem because the environment has been segmented and there is low noise in the image. The frames are now equivalent to binary images, the next task is to identify connected regions of pixels to identifying large continuous objects, ideally the person in the shot. For this task I used **Connected Components Analysis**, specifically a Matlab function (*connectedComponents²*) that provided a collection of regions, ordered by total size. After experimenting with the regions this function was returning I found that in most cases the first element ($L==1$), was an accurate detection of the subject within the image. In the cases where this was not correct I found that by altering my person-size threshold values explained previously I could stop the system from making incorrect identifications.

To facilitate this sensible selection of people, I included thresholds that represent the size of a human, and tested these against the size of the detected region. This allowed my script to consider whether or not the detected object is in fact a person or more similar to an erroneous detection for example a very large object such as the entire frame of a black image at the end or start of the footage. In the case of one video I found it necessary to alter this threshold to correctly classify the scene, this was a simple case of editing the threshold values, it could be argued that this is a limitation of the system and greater success and consistency could be achieved by further developing the system to always recognise people correctly, eliminating the need to adjust the thresholds.

After the task of locating the component that is most likely to be a part of someone walking in the scene, the location of this component can be used to decide where the person is within the shot. Using the extremities of the component ($L==1$) and then getting an average for X and Y coordinates a centre point can be established for each frame which represents where the person is, thus satisfying point to on the specification.

To design a method for the final task on the specification I considered what information was already extracted from the work completed to far. As each frame my system will be outputting information about the presence or absence of a person, and in the latter case the location within the scene, it is possible to record the first and last frame in which a person was present. From this information, considering the positions of the subject within these two frames, it is possible to determine which direction the subject travelled in:

IF (First Position – Last Position) > 0 → Right to Left Direction	This is an accurate test because given axis of the scene that the person travels along, if the first detected
IF <0 → Left to Right	

location in the clip has a high X position value then the subject is starting on the right. For example an entrance on the right of the scene where detection occurs at position 358 , 93 by subtracting the Last Position from the First we can test for either a negative value (indicating Right to Left) or the converse proving that the subject started at the Left with a comparatively low X coordinate, and there for travelled to the right.

² Part of *dch_matlab_toolkit* , Mark Everingham, Leeds 04-Oct-06

Justification of Design

It should be considered that the samples used to build this system do not vary a great deal, therefore future video clips provided may require a rethink in design. If the quality of the footage was reduced more efforts would be needed to remove noise before processing. A static camera also made background modelling for frame differentiation possible to provide a binary image, this method would not work effectively if future samples captured on a non-static device where requiring the same analysis. An alternative here would be to create a system that used reasonably large averaging areas to determine what was background and foreground.

Please refer to **Design** for additional justification and alternatives considered / trialled.

Implementation & Results

Presenting Results

I chose to update the user each frame with textual information answering the questions posed for this task:

- *Is there anyone in this frame?*
- *If so where are they?*
- *Which direction have they walked in this video?*

In addition to text feedback describing the information taken from each frame, I chose to visualise this data in the form of a red bounding box, with a green cross signifying the centre of the located object.

Illustrating Presentation of Task 1 Information

```
>>
>>
*****
Info for frame: 60.000000
Person present at frame number: 60.000000
The chap/lass was at position: 148.000000, 130.000000

FirstPosition =

    148    130

*****
*****
Info for frame: 60.000000
Person present at frame number: 60.000000
The chap/lass was at position: 148.000000, 130.000000

FirstPosition =

    148    130

*****
:K>>
```

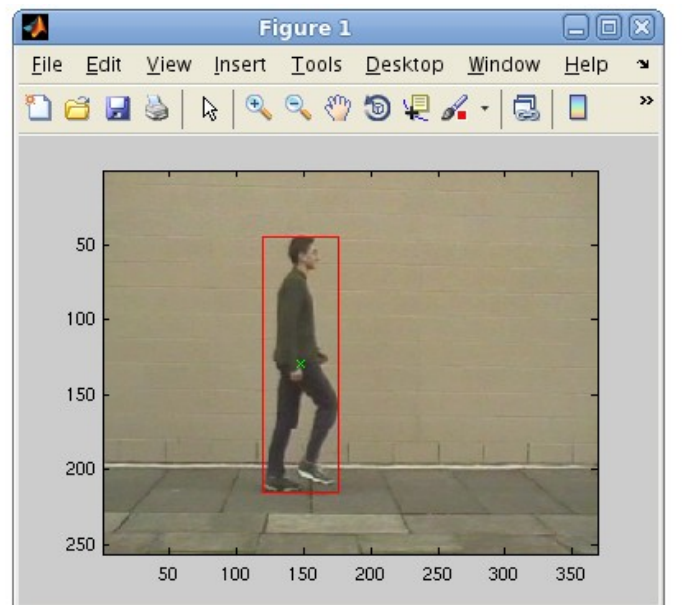


Illustration 1: Shows the second and first task being tackled, the detection of a person and the presentation, both textually and visually, of persons position. Green cross used to signify centre of the person. This data was provided for each frame.

Results Table

Details Provided		Results Obtained from Solution						
Vid No.	Walk Direction	Walk Direction	Entrance Frame	Position	Exit Frame	Position	>Person Threshold	<Person Threshold
1	LR	LR	30	18 , 192	104	363, 191	20	200
2	LR	LR	37	19 , 168	125	185, 129	20	200
3	RL	RL	28	358 , 93	125	6, 115	20	200
4	RL	RL	33	355 , 200	141	185, 129	20	200
5	LR	LR	40	15 , 159	141	185, 129	20	200
6	LR	LR	39	12 , 133	201	363, 163	20	200
7	LR	LR	47	15 , 105	145	177, 28	20	200
8	RL	RL	19	364 , 146	117	4, 206	5	200
9	RL	RL	29	356 , 197	105	185, 129	20	200
10	LR	LR	30	14 , 130	141	358, 142	20	200

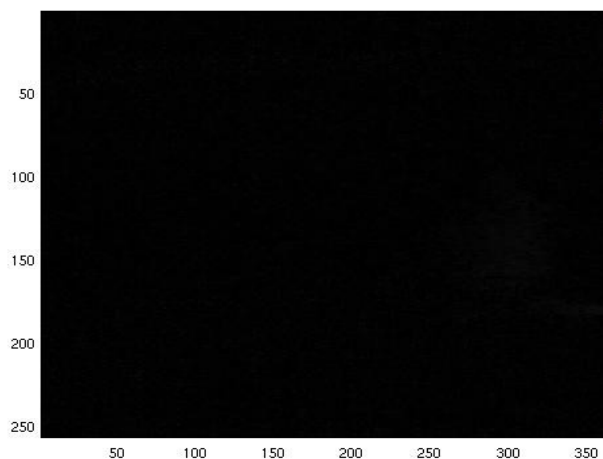
Validating Results

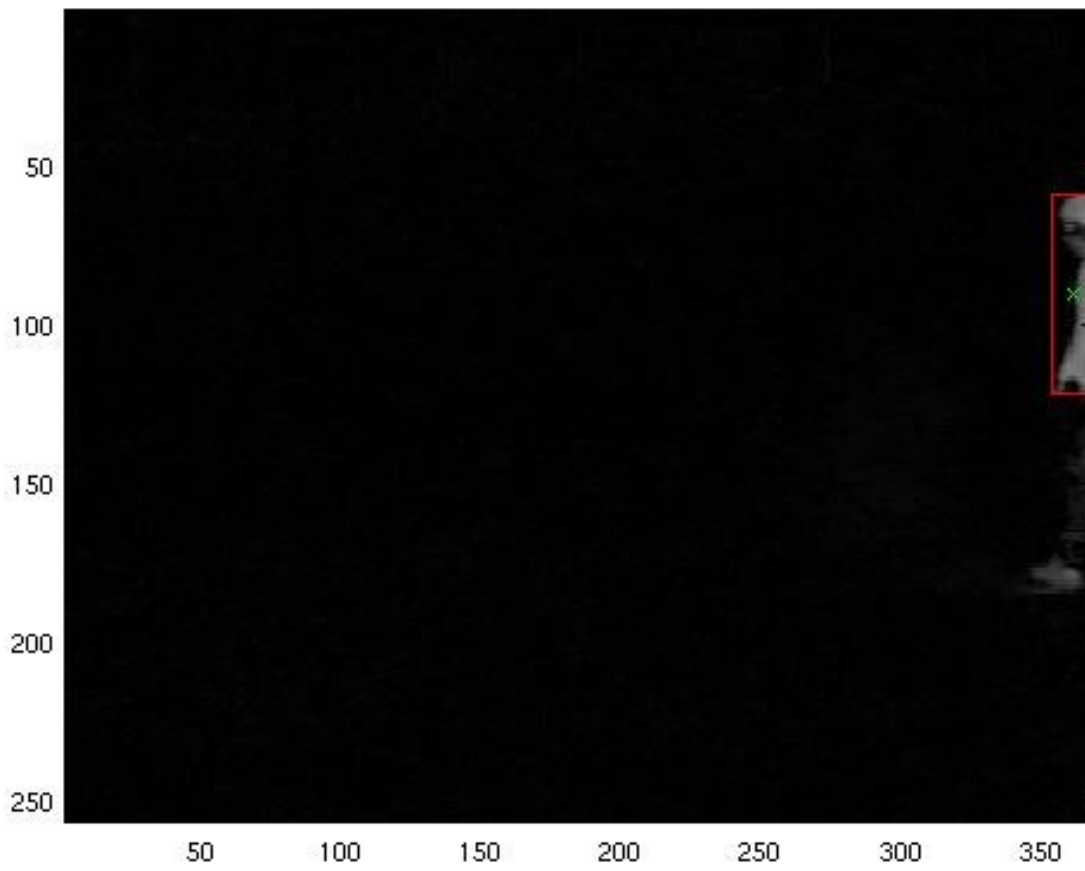
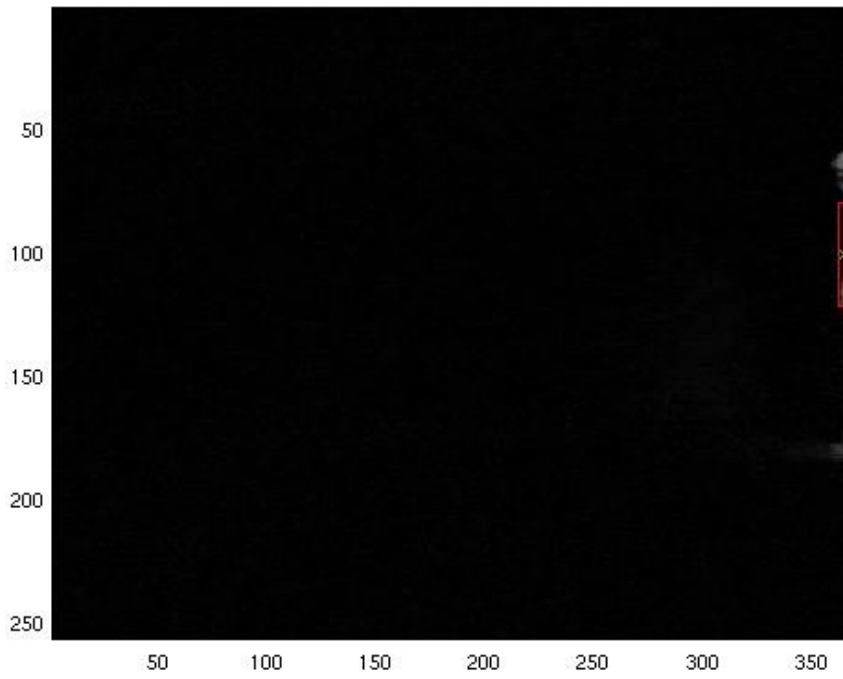
Result Confirmed through Manual Observation?

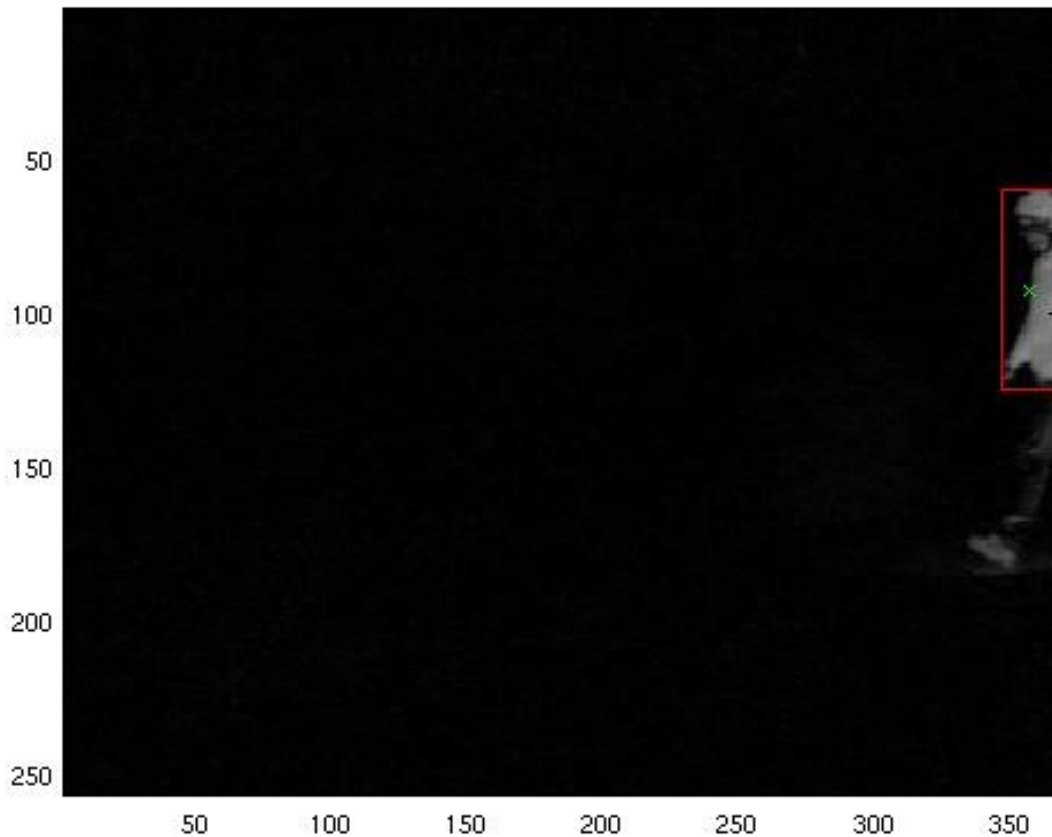
Video Number	Notes
1	Yes
2	Yes
3	Yes
4	Yes
5	Yes
6	Yes
7	Threshold adjustment required!
8	Yes
9	Yes
10	Yes

Images

Images from Video Sequence 3, tracking the path taken by person in Sequence 3. Showing the first 4 frames tracked as containing a person, initially only the edge of the persons head is detected, then the centre point moves down as more of the body comes into view.







Appendix

```
%% Liam Sullivan %%  
%% Computer Vision Coursework %%  
  
%% A vision system that will process moving images of people walking and output:  
%% â€¢ The start and end frames at which a person (or part of a person) is present.  
%% â€¢ The â€œlocationâ€ of the individual at each frame.  
%% â€¢ The direction of the persons walk in the scene, left to right or right to left  
%% Setup  
  
% Get the tools available  
addpath(genpath('/home/csunix/vislib/dch_matlab'));  
  
% Read in the video file  
CleanVid1 = mmread('/usr/not-backed-up/sc07ls/sequence1.avi');  
  
% Read in the video file using background modelling on the first frame
```

```
vid1 = modellingBackground_mm('/usr/not-backed-up/sc07ls/sequence1.avi');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
GrayScaleThreshold = 0.1;

PersonPresent = 0;

ComponentMinSize = 20;
ComponentMaxSize = 200;

FirstPosition = [0,0];
LastPosition = [0,0];
Direction = 'Unknown';

FirstFrameWithPerson = 0;
LastFrameWithPerson = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure,imagesc(CleanVid1.frames(50).cdata);

hold on;

for frame=1:vid1.nFramesTotal
%for frame=60
%Used to run on individual frames for the purpose of displaying evidence
%of the process working.

    % For each frame, carry out the following steps to result in a bounding
    % box telling us where the person is.

    ThisFrame = vid1.frames(frame).cdata;

    % Extract a single frame and save as a matrix

    GrayThisFrame = rgb2gray(ThisFrame)>GrayScaleThreshold;

    % Convert to greyscale with the given global threshold

    %Run connected components analysis on this frame

    [L, nr, rp] = connectedcomponents ( GrayThisFrame );

    % L is a collection of components, ordered by relative size

    % To track the movement of the object we need to plot the edges of this main
    component, as it seems to provide an accurate identifier for movement of the subject,
    i.e. L==1 in all test so far will track all of the walker's body minus the head in
    some cases.

    [rows columns] = find(L==1);
```

```
%Now we have the size of the major object from connected components
%analysis we can get width and height, to check if our first component
%is infact likely to be a person, or just noise.

ComponentWidth = max(rows)-min(rows);

ComponentHeight = max(columns)-min(columns);

%Stop the complaints due to comparisons for empty matrixes

if isempty(ComponentWidth)
    ComponentWidth = 0;
end

if isempty(ComponentMinSize)
    ComponentMinSize = 0;
end
if isempty(ComponentMaxSize)
    ComponentMaxSize = 0;
end
if isempty(ComponentHeight)
    ComponentHeight = 0;
end

%Using this info make a decision on person being present or not, toggle
%PersonPresent if the object appears to be person sized

if ComponentWidth > ComponentMinSize && ComponentWidth < ComponentMaxSize &&
ComponentHeight > ComponentMinSize && ComponentHeight < ComponentMaxSize
    PersonPresent = 1;
end

% Then:

% Fill up X and Y to draw bounding box, based on the major component above (L==1)
Y = [min(rows), min(rows), max(rows), max(rows), min(rows)];
X = [min(columns),max(columns),max(columns), min(columns), min(columns)];

YCollection = [min(rows), min(rows), max(rows), max(rows), min(rows)];
XCollection = [min(columns),max(columns),max(columns), min(columns), min(columns)];
PosX = (min(columns)+max(columns))/2;
PosY = (min(rows)+max(rows))/2;

%YCollection = YCollection + [min(rows), min(rows), max(rows), max(rows),
min(rows)];

%XCollection = XCollection + [min(columns),max(columns),max(columns), min(columns),
min(columns)];
```

```
msg = sprintf('*****');
disp(msg)

msg = sprintf('Info for frame: %f', frame);
disp(msg)

if PersonPresent>0
    msg = sprintf('Person present at frame number: %f', frame);
    disp(msg)

    msg = sprintf('The chap/lass was at position: %f, %f', PosX, PosY);
    disp(msg)

    %% Is this the first time a person has been present if so record to
    %% we can determine direction...i.e. which side did he enter the
    %% scene from?
    if FirstPosition == [0,0]

        %%If it is the first position then we can also save the frame
        %%number for easy reference to FirstFrameWithPerson, allowing
        %%this info to be added to the Video Summary at the end of the
        %%program run.

        FirstPosition = [PosX, PosY]
        FirstFrameWithPerson = frame;
    end

    %% Also update LastLocation variable so that until we lose the
    %% person each frame we'll update LastLocation to the most recent
    %% position, thus after exiting the loop LastLocation will in fact
    %% contain the latest position we detected, giving us the place
    %% where person x disappeared, and more importantly where he left
    %% the scene

    %%But then to keep stable for detection problems within isolated
    %%frames

    if [PosX, PosY] > [0.0]
        LastPosition = [PosX, PosY];
        LastFrameWithPerson = frame;
    end

end

else
    msg = sprintf('No one present at frame number: %f', frame);
    disp(msg)
end

msg = sprintf('*****');
disp(msg)
```

```
%Then draw a bounding box, like in the plotSquare function
```

```
plot(PosX,PosY,'gx');  
plot(XCollection,YCollection,'r-');
```

```
end
```

```
msg = sprintf('@@@@@@@@@@@@');  
disp(msg)  
msg = sprintf('Walk Analysis');  
disp(msg)  
msg = sprintf('@@@@@@@@@@@@');  
disp(msg)
```

```
msg = sprintf('First Position of subject: %d , %d', int16(FirstPosition));  
disp(msg)
```

```
msg = sprintf('At frame: %d', int16(FirstFrameWithPerson));  
disp(msg)
```

```
msg = sprintf('Last Position of subject: %d, %d', int16>LastPosition));  
disp(msg)
```

```
msg = sprintf('At frame: %d', int16>LastFrameWithPerson));  
disp(msg)
```

```
%%% Detirmine direction
```

```
%%Takes first position from last position, to detirmine direction of  
%%walk
```

```
DirectionTest = FirstPosition(1) - LastPosition(1);
```

```
%%A negative value here will suggest the person started on the left and  
%%walked left to right, a value >0 will be the opposite, a right to  
%%left walk
```

```
if DirectionTest>0  
    Direction = 'Right to Left';
```

```
else  
    Direction = 'Left to Right';
```

```
end
```

```
msg = sprintf('Walking Direction: %s', Direction);  
disp(msg)
```