**Task 1 (40 points)**

  
Figure 1: Bounding box of the person in frame 62 of the "walkstraight" sequence.

Write a Matlab function called find\_bounding\_box that takes in as argument the name of an image file from the "walkstraight" sequence, and computes the bounding box of the person. The function should RETURN the bounding box, as a matrix of four numbers: [top row, bottom row, left column, right column]. Furthermore, as a side effect, the function should display a figure that shows the original image, with a yellow (color code: [255 255 0]) rectangle superimposed, representing the detected bounding box. Your function can use data from any frame of the sequence in order to determine the bounding box for the frame in question.

Your function should be named find\_bounding\_box, and should take a single argument, i.e., the filename specifying a frame of the sequence. For example:

> find\_bounding\_box('walkstraight/frame0052.tif');

Don't worry about how the function works when the person is not visible, or is only partially visible.

**Task 2 (40 points)**

Write a Matlab function called person\_present that can tell when no person is present. Don't worry about how your algorithm performs on borderline cases, like frames 5-32 when the person is not fully visible. However, your algorithm should be able to tell, for example, that there is no person at frame 3, and that there is a person at frame 62. The function should return 1 if the person is present, and 0 otherwise.

Your function should be named person\_present, and should take a single argument, i.e., the filename specifying a frame of the sequence. For example:

> person\_present('walkstraight/frame0052.tif');

**Task 3 (20 points)**

Write a Matlab function called person\_speed that returns the average velocity of the person, between two frames. The function should return a 1x2 matrix [rows\_per\_frame, cols\_per\_frame], that specifies, in pixels, the velocity of the person along the vertical direction (rows, increasing from top to bottom) and the horizontal direction (columns, increasing from left to right).

Your solution can be built on top of your find\_bounding\_box function: call find\_bounding\_box twice, to find the person in both frames, and calculate the velocity based (somehow) on the results of the find\_bounding\_box function. Notice that your function must return the velocity, which specifies the direction of motion (and that is why it needs to be a 2D vector), NOT the speed, which is a single number.

Your function should be named person\_speed, and should take two arguments, i.e., the filenames specifying two frames of the sequence. For example:

> person\_speed('walkstraight/frame0052.tif', 'walkstraight/frame0062.tif');

Again, don't worry about how the function works when the person is not visible, or is only partially visible.

**Hints and Suggestions**

* The walkstraight sequence can be downloaded from here.
* Most of the solution for Task 1 is included in the code we covered in the introductory slides. You just need to package it up nicely as a single function.
* Use the addpath function if you need to let Matlab know where to find directories containing user-defined functions. Type help addpath to see how that works, or see examples in the code posted on the course website.
* In general, familiarize yourselves with the code we used in the introductory slides. You will find lots of Matlab tricks there that can be handy for this assignment.
* File draw\_rectangle.m implements a function that draws a rectangle.
* Files parse\_frame\_name.m and make\_frame\_name.m contain code that you should feel free to use, and that you may find useful if you want your code to automatically figure out the filename of the next frame, or previous frame, and so on. For example, try:
* [sequence\_name, frame] = parse\_frame\_name('walkstraight/walkstraight0062.tif');
* filename = make\_frame\_name(sequence\_name, frame+1);