DESIGN OF A VISUAL SYSTEM FOR DETECTING NATURAL EVENTS BY THE USE OF AN INDEPENDENT VISUAL ESTIMATE: A HUMAN FALL DETECTOR

P. A. BROMILEY, P. COURTNEY AND N.A. THACKER
Imaging Science and Biomedical Engineering, Stopford Building, University of Manchester, Oxford Road, Manchester, M13 9PT
E-mail: paul.bromiley@man.ac.uk

We describe the development of a vision system to detect natural events in a low-resolution image stream. The work involved the assessment of algorithmic design decisions to maximise detection reliability. This assessment was carried out by comparing measures and estimates made by the system under development with measures obtained independently. We show that even when these independent measures are themselves noisy, their independence can serve to guide rational design decisions and allow performance estimates to be made. Although presented here for one particular system design, we believe that such an approach will be applicable to other situations when an image-based system is to be used in the analysis of natural scenes, in the absence of a precise ground truth.

1 Introduction

Performance evalution is esssential for providing a solid scientific basis for machine vision, and yet its importance is often understated. Current work in this area [1,2,3] has tended to emphasise the importance of an objective ground truth (see for example work in medical image registration [4], face recognition [5], photogrammetry [6] and graphics recognition [7]). We present a case study of the evaluation of a machine vision system, which exhibits two examples of performance characterisation in the absence of a suitable ground truth. We believe that our approach will be applicable to other situations when an image-based system is to be used in the analysis of natural scenes, operating in the absence of a precise ground truth.

The system described here is a human fall detector based on a novel infrared sensor with limited spatial resoultion. It was developed by an industrial/academic collaboration involving the University of Manchester, the University of Liverpool, IRISYS Ltd. and British Telecom Plc in the UK. It observed a natural scene of a person in their home, typically an elderly person living alone. When left undetected, falls amongst the elderly often lead to aggravated injuries, admission to hospital, and sometimes death. However, this kind of event is rather poorly defined and occurs randomly under normal circumstances, making the design of a system to detect it problematic. In order

to be practically useful the system had to detect falls reliably whilst generating the minimum number of false alarms, making performance evaluation a vital part of this work.

The detector operated by recognising the patterns of vertical velocities present during a fall by a human subject, and issuing a fall detection warning when such a pattern was detected in the output from the thermal sensor. The initial identification and the detection of these patterns was performed automatically using an MLP neural network, thus removing any requirement for in-depth study of the dynamics of human movement. Image data showing simulated fall and non-fall (e.g. sitting) scenarios were captured from both the infrared sensor and a colour CCD camera. Vertical velocity measurements were extracted from both sets of images, and the colour data were used as a gold standard to demonstrate a correlation between the infrared velocity estimates and the physical velocities present in the scene. Despite the fact that the velocity measurements extracted from the colour data did not represent a genuine ground truth, we show that this form of performance evaluation can be used to guide algorithmic development in an objective manner. Finally, a range of neural networks were trained on a sub-set of the infrared data and tested on the remainder, and the best-performing network was identified using ROC curves. We demonstrate that, using appropriate definitions of true and false detections, it is possible to evaluate the performance of a system for identifying events in a temporal data stream in this manner.

2 Approach

The purpose of the fall detector was to monitor the image stream from the thermal detector for characteristic signals associated with falls in human subjects, and to issue a fall detection warning when such a motion was observed. At the most basic level, the primary characteristic associated with falls is vertical downwards motion. Therefore the analysis focussed on measuring vertical velocities, and identifying features in the pattern of velocities over time that were characteristic of falls. Basic physics guarantees that, if the velocity of a subject moving around a room is resolved into its horizontal and vertical components, the vertical acceleration of a subject falling under gravity acts independently of any horizontal motion. Therefore, the analysis was restricted to studying vertical motion, and any horizontal component was discarded.

The first requirement was to obtain realistic image data from which the characteristic motions associated with falls could be identified. Sequences of images showing an actress simulating a wide variety of fall and non-fall (e.g.

sitting) scenarios were captured simultaneously from both the infrared sensor and a colour CCD video camera.

Next, software was written to calculate the velocity of the actress both from the infrared image sequences and from the colour video. The approach taken to extracting velocity information from the colour video relied on the use of colour segmentation. During the simulations, the actress wore a shirt of a different colour to any other object in the scene. A colour segmentation algorithm was used to extract the shirt region from the colour video images, allowing the centroid of the actresses upper body to be calculated. The vector of these centroid positions over time could then be differentiated to obtain velocity measurements from the colour images.

The approach taken to extracting velocity information from the infrared images was quite different, and exploited the basic physics of the detector itself. The infrared sensor used was a differential sensor i.e. it registered changes in temperature. Stationary objects in the scene were therefore ignored. Any moving object warmer than the background created two significant regions in the image. The first was a region of positive values covering the pixels that that the object was moving into, which were becoming warmer. This was trailed by a region of negative values covering the pixels that the object was moving out of, which were becoming colder. If the object was colder than the background, the signs of the two regions were reversed. In either case, a zero-crossing existed between the two regions that followed the trailing edge of the moving object. The approach taken was to track this zero-crossing to give measurements of the position of the actress in the infrared images, which could in turn be used to calculate the velocity.

The velocity estimates derived from the colour video data were used as a gold standard against which to compare the velocities calculated from the infrared images. It was therefore possible to calculate the extent to which the infrared velocity estimates were correlated with the physical velocities present in the scene. This correlation analysis was performed for a subset of around 2% of the data. Several methods were used to perform this analysis, including linear correlation coefficients, Gaussian fitting to the noise on the infrared measurements after outlier rejection, and the use of ROC curves. The latter provided a generic method for ranking the performance of various smoothing filters applied to the data.

The use of two independent estimators of motion was a key part of this work. Although both used radiation from the scene, they were independent in that they sensed different kinds of information, and processed it in fundamentally different ways. The colour segmentation algorithm was based on the interaction of visible light and reflectance on clothing in spatial re-

gions, whereas the algorithm for the infrared sensor was based on signal zero-crossings on thermal radiation using a completely different lens system. We would therefore expect the information in the two estimators to suffer from noise, bias and distortion independently.

To produce the fall detector itself, an MLP neural network was trained to take temporal windows of velocity measurements from the infrared sensor and produce a fall/non-fall decision. The advantage of this approach was that the training process allowed the neural network to automatically identify the regions in the input pattern space that contained the fall data points i.e. the patterns of velocities characteristic of a fall. Therefore no in-depth, manual study of those patterns of velocities was required. A subset of the infrared velocities were extracted at random and used to train a neural network to perform the fall/non-fall classification. A number of neural networks were trained, varying all available parameters in order to find the best architecture for this problem. ROC curves were used to select the best network.

A nearest neighbour classifier was applied to the same data. It can be shown that the nearest neighbour classifier approaches Bayes optimal classification performance for large data sets such as this, and so this gave an indication of how close the best-performing neural network approached to the optimum performance. Finally, the classification decision was made based on individual velocity measurements, without using the neural network, in order to estimate how much of a performance gain had been realised through the application of a neural network to this temporal recognition problem.

3 Data collection

Data was collected to provide realistic video data on both fall and non-fall scenarios that could then be used in the construction of a fall detector. A list of types of fall (e.g. slips, trips etc.), together with non-fall scenarios that might generate significant vertical velocities (e.g. sitting), was prepared. An actress was employed to perform the scenarios and thus simulate falls and non-falls, and sequences of images were captured from both the infrared sensor and a colour CCD video camera. Each scenario was performed in six orientations of the subject and the cameras, giving a total of 84 fall and 26 non-fall sequences for each camera. In addition to performing the fall or non-fall, the actress raised and lowered her arm at the beginning and end of each sequence in order to provide a recognisable signal that could later be used to synchronise the colour and infrared image sequences.

In order to simplify the simulations, they were performed at between four and six metres from the cameras, with the cameras held parallel to the

floor in the room. This maximised the ability to distinguish vertical from horizontal motion. Twenty degree optics were used on both the colour and infrared cameras, further simplifying the geometry of the scene through the foreshortening effect of narrow angle optics. The infrared data were recorded at 30fps in a proprietary format. Colour data were recorded at 15fps as RGB values (i.e. no codec was used) in the AVI format. Interpolation was later used to increase the effective frame rate of the colour data to match the infrared data.

4 Velocity Estimation

The extraction of velocity information from the images recorded during the simulations focussed on three areas:

- estimating velocities from the colour video images;
- estimating velocities from the differential infrared images;
- measuring the correlation between them.

The following sections give brief summaries of the methods adopted in each of these areas.

4.1 Colour Segmentation and velocity estimation

The extraction of velocities from the colour images relied on the use of a colour segmentation routine [8]. Fig. 1. shows an example of a single frame of colour video taken from one of the falls. During the simulations, the actress wore a shirt of a different colour to any other object in the scene. This allowed the colour segmentation routine to extract and label the pixels in the shirt region. Following this, the co-ordinates of the pixels in that region were averaged to calculate the centroid of the actresses upper body, giving a measurement of her position in the scene. The vector of these measurements over the course of an entire sequence of images could then be differentiated to produce a vector of velocities. Only the vertical component of the velocity was calculated.

The image segmentation algorithm is described in more detail elsewhere [8] but a brief description is included here for completeness. The approach adopted relied on the clustering of pixels in feature space. The subject of clustering, or unsupervised learning, has received considerable attention in the past [9], and the clustering technique used here was not original. However, much of the work in this area has focussed on the determination of suitable

Figure 1. A frame of colour video.

criteria for defining the "correct" clustering. In this work a statistically motivated approach to this question was adopted, defining the size required for a peak in feature space to be considered an independent cluster in terms of the noise in the underlying image. This maximised the information extracted from the images without introducing artefacts due to noise, and also defined an optimal clustering without the need for testing a range of different clusterings with other, more subjective criteria.

The segmentation process worked by mapping the pixels from the original images into an n-dimensional grey-level space, where n was the number of images used, and defining a density function in that space. A colour image can be represented as three greyscale images, showing for instance the red, green and blue components of the image, although many alternative three-dimensional schemes have been proposed [10,11]. Therefore a colour image will generate a three-dimensional grey-level space, although the algorithm can work with an arbitrary number of dimensions. An image showing a number of well-defined, distinct colours will generate a number of compact and separate peaks in the grey-level space, each centered on the coordinates given by the red, green and blue values for one of the colours. The algorithm then used

Figure 2. A frame of colour video - the saturation (left) and hue (right) fields.

the troughs between these peaks as decision boundaries, thus classifying each pixel in the image as belonging to one of the peaks. Each peak was given a label relating to the number of pixels assigned to it, and an image of these labels was generated as output.

In practice, illumination effects can spread out or even divide the peaks in colour space. For example, an object of a single colour may be partially in direct light and partially in shadow, and so the shadowed and directly lit regions of the object appear to have different colours. In the RGB colour scheme each component contains both chromatic and acromatic components. The object will therefore generate two peaks in colour space and be segmented as two separate regions. This is not a failure of the algorithm, since the separation of the object into several regions preserves the information present in the image, but it was undesirable in the current situation where the intention was to label the shirt as a single region regardless of illumination effects. Therefore the achromatic information was removed from the images prior to segmentation, by converting from the RGB colour space to the HSI colour space, which separates the chromatic information in the hue and saturation fields from the achromatic information in the intensity field. The intensity field was discarded and the segmentation was performed on the hue and saturation fields. This had the additional advantage of reducing the dimensionality of the problem from three to two, reducing the processor time required. Fig. 2 shows the hue

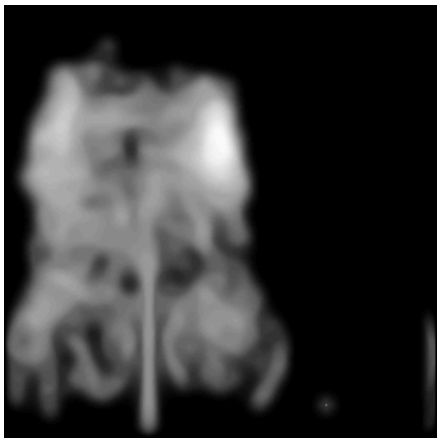


Figure 3. The scattergram for the hue and saturation fields shown in Fig. 2, corresponding to the colour video frame shown in Fig.1. The peak in the upper-right corresponds to the green shirt.

and saturation fields for the frame of colour video shown in Fig. 1, and Fig. 3 shows the scattergram of these two fields. The pixels correpsonding to the shirt form a well-defined, compact cluster in hue-saturation space.

Since the actress wore a shirt of a single, uniform colour, it was labelled as a single region. A thresholding algorithm could then be used to identify the pixels covering the shirt region, and the centroid of the shirt was calculated by averaging the pixel co-ordinates. Fig. 4 shows the outputs from the colour segmentation algorithm. The vertical component of the centroid's velocity was then calculated by taking differences between the vertical position in neighbouring frames, thus producing a velocity in units of pixels per frame.

The frame rate of the colour video was 15fps, whereas the frame rate of the infrared data was 30fps. Since the colour video provided much more accurate data, and to avoid discarding data, extra data points were generated for the colour video by interpolating the centroid positions between each pair of frames before the velocity calculation. Finally, since the colour segmentation algorithm was very processor intensive, the velocity calculation procedure was only applied to a subset of 26 of the fall video sequences from the simulations, and to a window of 30 frames centered around the fall in each video. This generated 59 positional data points for each video sequence when the interpolation was applied, and 58 velocity data points.

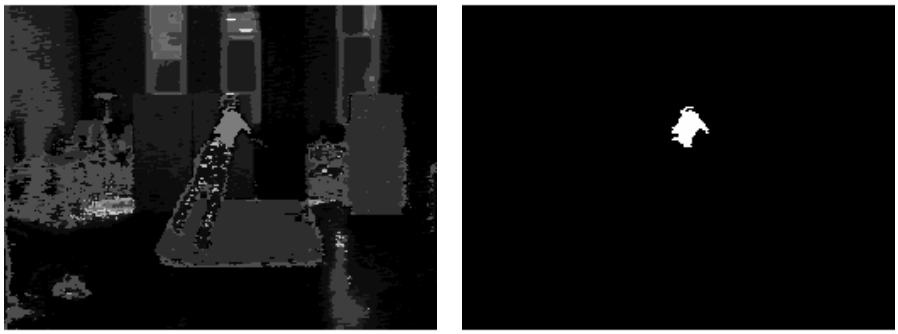
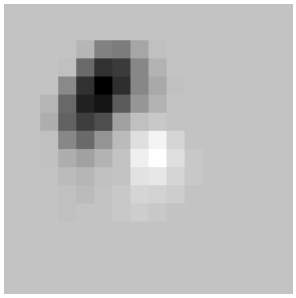


Figure 4. Segmentation of a frame of colour video (left), and the result of thresholding to extract the shirt region (right).

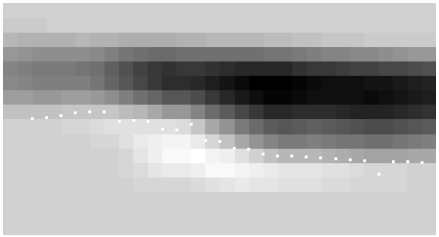
4.2 IR Velocity Estimation

The approach taken with the infrared images relied on the differential nature of the detector. Since it was sensitive to changes in temperature, only moving objects at a different temperature to the background were detected. Therefore, the analogue of the colour segmentation task was performed by the detector itself. As mentioned above, any moving object at a higher temperature than the background generated two regions in the images: a region of positive values covering the pixels into which the object was moving, trailed by a region of negative values covering the pixels out of which the object was moving. The zero-crossing between these two regions followed the trailing edge of the object. In order to remove any horizontal component of the movement, the 16x16 pixel infrared images were summed across rasters to produce a 16x1 pixel column vector. This introduced several additional advantages, providing an extremely compact method for storing the image information (the "crushed" image, produced by stacking together the column vectors from an image sequence) and reducing the effects of noise. The positions of the zero-crossings were identified initially by a simple search up the column vectors, and were refined by linear interpolation. This produced a vector of approximations to the vertical position of the actress across a sequence of images, which was differentiated to produce a vector of vertical velocities. Fig. 5a shows a single



(a) Single IR video frame.

(b) The "crushed" image.



(c) Detail from (b).

Figure 5. A single frame of IR video (a), showing the positive (white) and negative (black) regions. The crushed image for this fall sequence (b) shows the extended period of activity in the middle of the sequence corresponding to the fall itself, together with regions of activity before the fall, corresponding to the actress waving her arm as a synchronisation signal, and after the fall, corrsponding to the actress standing up. The region from the middle of the crushed image is shown in detail in (c), with the zero-crossings marked as white points.

frame from the IR video, taken simultaneously with the colour video frame shown in Fig. 1. Fig. 5b shows the result of stacking together the column vectors for all of the IR frames taken during this fall, the "crushed" image.

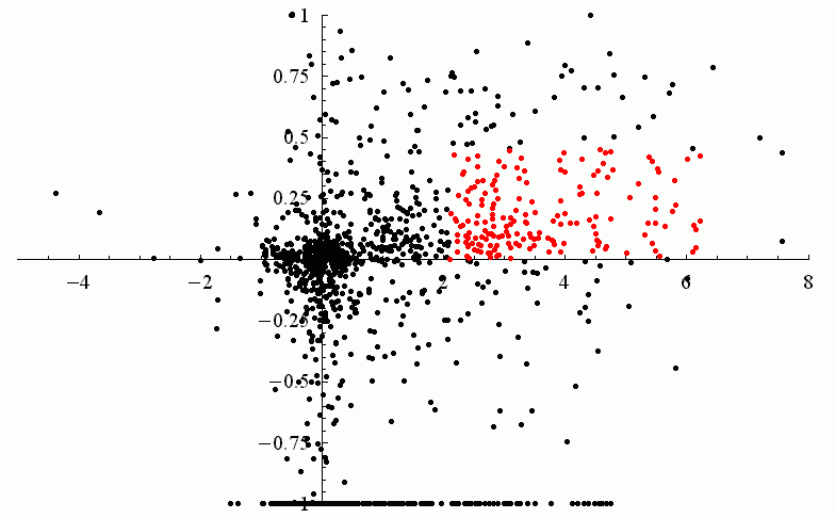


Figure 6. The raw velocity data, showing infrared velocity against colour video velocity. The units of both axes are pixels per frame, where one frame corresponds to 1/30 th of a second, but the definition of a pixel is different for the two axes.

Fig. 5c shows the region of the crushed image covering the fall itself, with the zero-crossing marked as white points.

4.3 Velocity Correlation

The extent of the correlation between the velocities extracted from the colour images and those extracted from the infrared images was measured using a number of different techniques.

The simulation data consisted of 108 data sets, covering a variety of both fall and non-fall scenarios, recorded on both a colour CCD camera and the infrared sensor. In order to test the velocity calculation routines a sub-set of 26 of these data sets, covering the whole range of fall scenarios, were selected and plotted against one another, as shown in Fig. 6. The units of velocity in this plot are pixels per frame, where a frame corresponds to 1/30 th of a second, but it must be remembered that there are 240 pixels along the y-axis of the colour video images, and only 16 along the y-axis of the infrared images.

As can be seen, there is some correlation, but it is somewhat poor. This was expected given the exacerbation of noise in the position data by the

differentation used to calculate the velocity. Therefore methods of combining data within the temporal stream were studied.

4.4 Data combination

Smoothing functions were applied in order to improve the correlation. A variety of smoothing techniques were tested in order to find the best technique for this data set. These included:

- a five-point moving window average, which replaced each data point with the result of averaging the five points centered around that data point;
- a five-point median filter, which took the same window of data and replaced the central data point with the median of the five values;
- a combination of the two which was termed a median rolling average (MRA) filter which took a window of five data points centered on the point of interest, dropped the highest and lowest values, and then averaged the remaining three points.

Each of these techniques had advantages. The median filter tended to be more robust to outliers, whereas the moving window average was very susceptible to outliers, but had a stronger smoothing effect in the absence of outliers. The MRA filter combined the advantages of each, providing stronger smoothing whilst retaining resistance to outliers.

Figs. 7-9 show the result of plotting the processed estimates from the thermal sensor against the velocity estimates derived from the colour processing. Several facts were immediately obvious from these plots. It is clear that the smoothed data produced tighter distributions than the unsmoothed data, but none of the three smoothing methods had an obvious advantage from simple visual inspection of the plots. The plots for the three smoothing methods had a typical shape: the velocity estimates from the infrared images were directly proportional to those from the colour video up to velocities of approximately 2 pixels per frame on the x-axis of the plots (corresponding to the colour video velocity estimates), and then they flattened out. The infrared velocity at which this occured was around 0.19 pixels, approximately the value that would be expected from the ratio of pixel sizes in the two image types. Above this point, the infrared velocity estimates no longer increasd with colour video velocity estimate. This was due to the basic physics of the detector itself, rather than the methods used to extract velocity estimates from the data. The thermal sensor took several seconds to saturate,

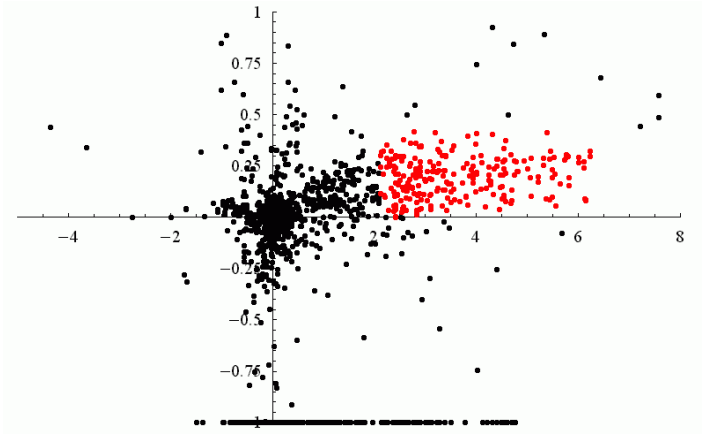


Figure 7. The velocity data after 5 point moving window average, showing infrared velocity against colour video velocity.

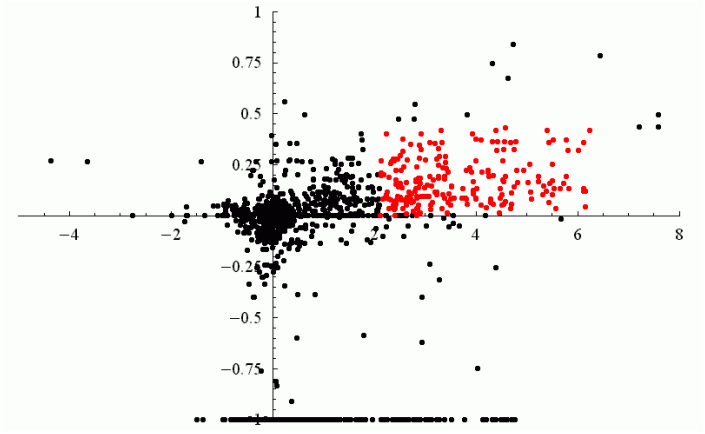


Figure 8. The velocity data after 5 point median average, showing infrared velocity against colour video velocity.

and this placed an upper limit on the rate of temperature change it could detect. The colour video velocity estimates had a maximum velocity of around 6 pixels per frame, corresponding to around 2.25 metres per second. The infrared velocity estimates flattened at around one third of this value, and so

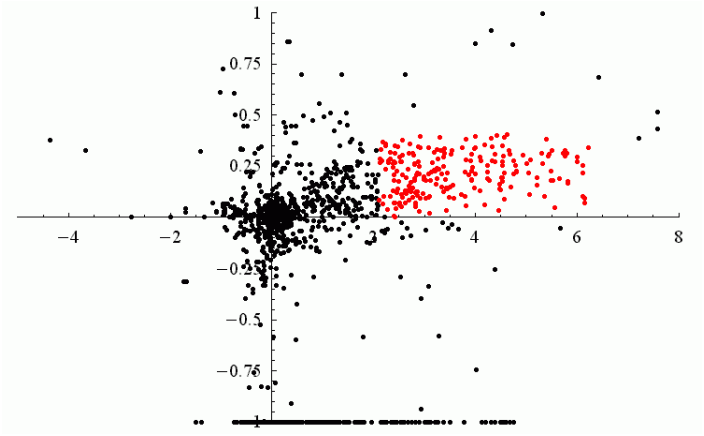


Figure 9. The velocity data after 5 point median/ 3 point moving window average, showing infrared velocity against colour video velocity.

the maximum velocity measureable for the given geometry was 0.75 metres per second. Therefore, the sensor design discarded approximately two thirds of the dynamic range of velocities present during typical fall scenarios. Following consultation with the sensor manufacturer, it emerged that this may have been due to the preprocessing performed by the sensor itself, which has since been modified.

In order to determine which smoothing method was the most effective for this data, the correlations between the infrared and colour video velocity estimates were measured in several ways. Firstly, a simple linear correlation coefficient was calculated for the whole data set for each smoothing option, and the results are given in Table 1. This was not in itself particularly informative, since it assumed a simple linear relationship between the two variables studied, and furthermore was highly sensitive to outliers. The correlation coefficients showed that smoothing the data gave a better correlation than no smoothing, but did not show a significant difference between the three smoothing methods.

Therefore, a more complex technique was applied. As mentioned above, the infrared velocity data reached a plateau above a value of around 2 pixels per frame on the x-axis (colour video velocity estimate) and 0.19 pixels per frame on the y-axis (infrared velocity estimate), and did not increase further with colour velocity. Any difference in the infrared velocity for data points above this threshold therefore corresponded only to noise. The data from this

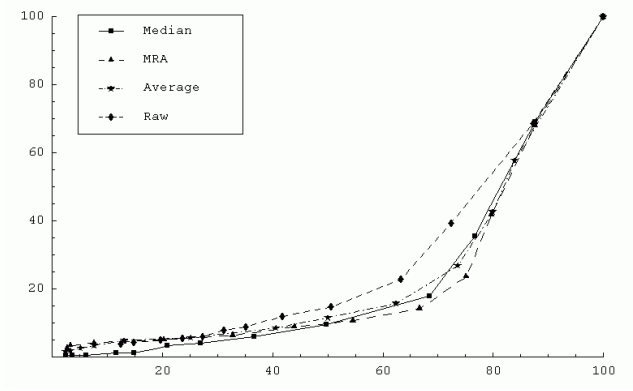
Table 1: Statistics for the four smoothing methods used.

Smoothing method	Correlation	Inliers	Outliers	μ	σ
None	0.176	178	133	0.178	0.129
5 pt Average	0.201	239	72	0.193	0.098
5 pt Median	0.204	228	83	0.168	0.110
M.R.A.	0.199	239	72	0.212	0.098

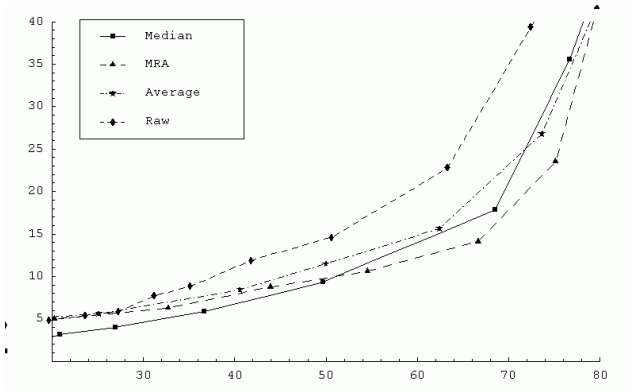
region were projected onto the y-axis (i.e. the x-coordinates were discarded) and the mean and standard deviation were calculated, assuming a normal distribution. Since these calculations were also sensitive to outliers, the outlying data points were identified with a simple visual inspection and discarded. The grey points on the graphs show the data points selected for these calculations: the black points are the discarded data points. The number of inliers (grey points) and outliers (black points above the 2 pixels per frame threshold) were also counted. The results are again given in Table 1.

Inspection of the data given in Table 1 shows that the median rolling average filter gave the best performance of the four smoothing options (three smoothing methods plus no smoothing). It had the highest inlier/outlier ratio and lowest standard deviation (although the difference between the median rolling average and moving average was largely insignificant), showing that it produced a tighter distribution and had the largest noise-reducing effect of the three methods. It also produced the highest mean, and so gave more separation of fall data points from the noise.

In order to conclusively demonstrate the superiority of the median rolling average filter over the other filters tested for this data set, ROC curves were plotted for the four smoothing options. A threshold was specified on the color velocities to arbitrarily split the data into "fast" and "slow" velocities. This threshold was set at 2 pixels per frame, the point at which the infrared velocity estimates stopped increasing with colour velocity estimate, and therefore the threshold which gave the maximum ability to differentiate high velocities from noise. Then a second, varying threshold was applied to the infrared velocity estimates, and was used to make a fast/slow decision based on the infrared data. The points above this second threshold therefore fell into two groups. Firstly, those defined as fast using the colour data, i.e. those with x values higher than 2 pixels per frame, represented correct decisions based on the infrared data. Secondly, those defined as slow using the colour data represented incorrect decisions. The number of data points in each of these categories was counted, and a percentage was calculated by dividing by the



(a)



(b)

Figure 10. ROC curves (showing percentage false acceptance rate plotted against percentage true acceptance rate) for the four velocity calculation methods (a) and a detail from this plot (b).

total number of points either above or below the threshold applied to the colour data. The two probabilities calculated in this way produced a point on the ROC curve, and by varying the threshold applied to the infrared velocity

the whole ROC curve was plotted for each smoothing option.

The ROC curves therefore provided a generic method for calculating the correlations between the colour and infrared velocity estimates, by measuring how often they agree or disagree on a data point being fast for some arbitrary definitions of "fast" and "slow". The resulting graph is shown in Fig. 10.

The best smoothing option was the one whose ROC curve most closely approached the point (100,0) on the graphs, and it is clear that the MRA filter was the best for this data set. Applying no smoothing was shown, as expected, to be the worst choice. The median rolling average was always better than the simple average, due to its ability to reject outliers. At very low correct decision rates the median average proved better than the median rolling average due to its superior resistance to outliers. However, the median rolling average was clearly better than the other smoothing choices in the regime in which the final classifier would operate.

4.5 Conclusions

The statistical analysis of the results from the four smoothing options clearly showed that the median rolling average was the best smoothing filter for this data set. Several other conclusions can also be drawn from the data presented. Firstly, the time taken for the detector to saturate placed an upper limit of around 0.2 pixels per frame, or 6 pixels per second, on the dynamic range of velocities that could be detected. Given this information, the effects of changes to the detector design can be deduced. Fitting a Gaussian to the infrared velocity measurements in the flat part of the velocity curves produced a mean of approximately 0.2 pixels per frame and a standard deviation of around 0.1. It is therefore clear that, in order to separate the velocities measured during a fall from the noise on measurements of zero velocity, a resolution of around 0.1 pixels per frame is required, and furthermore that this was being achieved by the current method.

It is probable that the correlation between the velocities calculated from the infrared images and the physical velocities present in the scene was better than the correlation between the velocities calculated from the infrared and colour video. The two velocity extraction techniques measured slightly different quantities. In the case of the colour video, the actresses upper body was segmented from the scene and its centroid calculated. The limbs and head were ignored, and so the calculated velocities corresponded closely to the movement of the centre of gravity. In contrast, the infrared images measured the movements of all body parts. As an example, if the actress waved her arms whilst otherwise standing still this movement was detected in the

infrared images but not in the colour video. The aim of this work was to demonstrate the feasibility of extracting velocity estimates from the data provided by the thermal sensor, and the analysis presented here placed a lower limit on the correlation between the velocity estimates from the infrared data and the physical velocities present in the scene. This approach was intended to give an overall ranking to the various velocity estimation algorithms tested, rather than calculate an absolute fall detection efficiency measure.

5 Neural Network Fall Detector

Once the method for extracting estimates of velocity from the infrared images had been produced, and a correlation with the physical velocities present in the scene had been demonstrated, the next stage was the construction of a fall detector that took the infrared velocity estimates as input and produced a fall/non-fall decision as output.

Neural networks represent a well-established computational technique for making classification decisions on data sets that can be divided into two or more classes. In essence a neural network is a method for encoding a set of decision boundaries in an arbitrarily high-dimensional space. The network typically takes some high dimensional vector as input, and produces a classification of the data as output, based on the position of the vector in the space compared to the positions of the decision boundaries. The main advantage of the technique is that the positions of the decision boundaries can be determined automatically during the training phase. Data is provided in which the classification is known, and a variety of algorithms exist that optimise the decision boundaries. The trained network can then be used to classify data points for which the classification is not previously known.

In this case, the input data for the network was high-dimensional vectors of velocity measurements representing temporal windows. Each temporal window of velocity measurements defined a point in the high-dimensional space in which the neural network was operating. The network then attempted to define decision boundaries which encompassed the region of the space containing the points corresponding to falls, thus identifying the characteristic patterns of velocities present during falls. The dimensionality of the input vectors, and thus the number of input nodes in the network, was selected on the basis of the timespan of the falls recorded during the simulations: temporal windows of nine velocity measurments were used.

5.1 Data Preparation, Network Design, and Training

In order to provide known classifications for the data points used in the neural network training, some method for labelling the positions of the falls in the data was required. Therefore, the infrared images were synchronised with the colour images, and the approximate positions of the falls were identified by visual comparison. Then the three points of highest velocity during each fall were labelled as falls, and the remaining data points were labelled as non-falls. This provided approximately 50,000 classified data points, of which 20% were extracted at random for neural network training, leaving 80% for testing the networks.

A variety of neural networks were trained, in an attempt to find the optimum architecture for this problem, varying all available parameters including:

- the number of hidden nodes, varied between 2 and 21;
- the number of hidden layers (1-2);
- the initialisation factor for the network weights;
- the training algorithm - either RPROP (resilient back-propagation) or CGM (conjugate gradient minimisation);
- number of iterations, from 200 to 2000.

A total of 120 combinations of MLP architecture and training regime were used.

5.2 Testing

The trained neural networks were tested in two ways: by plotting ROC curves describing their performance on the full data file (including the data not used in training), and by comparison with the ROC curve for a nearest neighbour classifier applied to the same data. For large data sets such as this the nearest neighbour classifier approaches a Bayes optimal classification, and so provided an upper bound on the performance of the neural networks, which was used as a benchmark during network training. However, it requires a prohibitively large amount of processor time to run and so did not form a viable solution to the problem of fall detection in real time.

The trained neural networks gave output in the form of a probability i.e. ranging from 0 to 1, with higher values indicating that the input data were more likely to represent a fall. Rather than apply a simple threshold to this output, counting all values above the threshold as falls and all values below

it as non-falls, a more sophisticated method was applied in order to increase detection reliability. The output from the network was monitored for local peaks in the probability, and then the height of the peak was compared to a threshold. This ensured that, during extended high-velocity events such as falls, the network issued only one detection, rather than issuing a series of fall detections for every point during an event that generated a network output higher than the threshold.

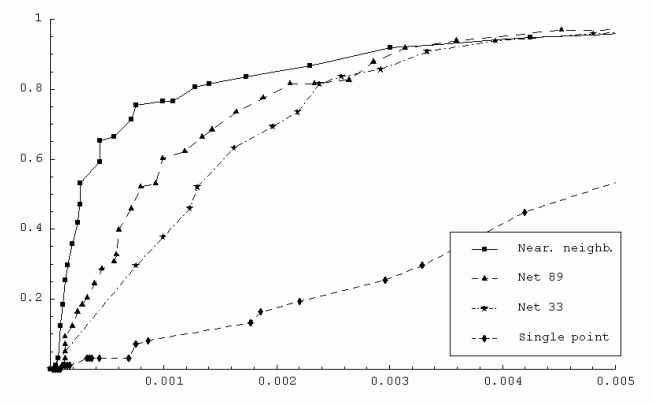
The issue of which quantities to plot on the axes of the ROC curves was problematic in this case. Firstly, falls were marked in the input data only at the three points of highest velocity during the fall. The falls were, however, extended events covering more than three frames, and so it was reasonable to expect the network to issue detections shortly before or after the marked three-frame window. It was therefore unreasonable to count as correct only those detections which coincided exactly with the marked positions of falls. Secondly, the network might issue more than one detection during the course of a fall. This provided the choice of whether to count all of the detections occurring close to a marked fall as correct detections, or whether to reverse the problem and look at how many marked falls had one or more detections in their temporal vicinity. Finally, a similar issue applied to false detections. If false detections could be caused by an event such as the subject sitting down, which was not marked as a fall but nevertheless caused a local period of high velocities, then several false fall detections might be issued within this period. This raised the problem of whether to count these as multiple false detections or as a single false detection, which in turn raises the logical problem of how to specify non-events. It should however be noted that any choice of how to count correct and false detections would allow the network performances to be compared as long as the same procedure was applied to the outputs of all networks.

The approach chosen for plotting the ROC curves for the neural network outputs was as follows. The outputs from the neural networks were monitored, and the heights of local peaks were compared with some threshold. Peaks higher than the threshold represented fall detections, and the positions of these fall detections in the input data file were recorded. A second loop scanned through the data looking for the specified positions of falls. Any true fall that had one or more fall detections by the neural network within 20 frames in either temporal direction (0.66 seconds at 30fps recording rate) was counted as a correct detection, and any specified fall that did not have such a detection by the network within that time period was counted as a real fall not detected. Therefore the reconstructed signal axis of the ROC curves showed the number of genuine falls detected by the network as a proportion of

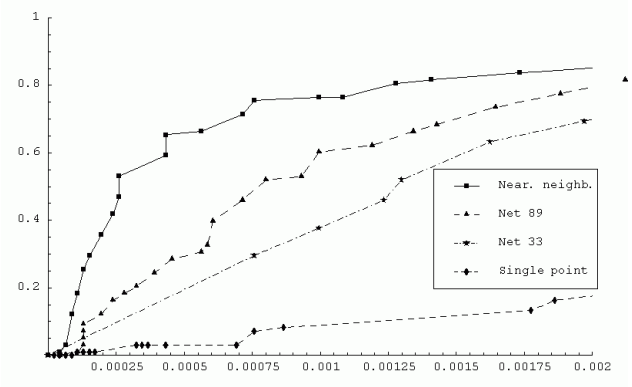
the total number of genuine falls. This treatment ensured ease of comparison between the different networks, as the maximum possible number of events represented in the recovered signal measurement was limited to the number of labelled falls in the data. It avoided the potential problems inherent in examining the raw number of detections by the network e.g. if a particular network produced multiple detections during a small percentage of the labelled falls, but did not detect the remaining labelled falls, looking at the raw number of detections would give that behaviour an unfair advantage, whereas looking at the number of genuine falls which had one or more detections did not. Finally, the whole procedure was repeated, varying the threshold to which the heights of local probability peaks were compared, to plot out the whole span of the ROC curve.

The treatment applied to generating data for the error rate axis of the ROC curves was slightly different. The same arguments applied, in that there were underlying events in the data (e.g. sitting down) which might generate one or more false fall detections by the neural network. However, in this case there was a logical problem of how to specify a non-event. Therefore, in the absence of a viable solution to this problem, the error rate was calculated as the number of false detections divided by the total number of data points which did not fall within a forty-point window around one of the falls.

Plotting the ROC curves for the neural networks provided a method for picking the best-performing network, but it was also desirable to compare them with the optimal classification performance. Therefore, the results were compared to the results obtained from a nearest neighbour classifier. The nearest neighbour classifier can be shown to approach the Bayes optimal classification i.e. the classification that would be obtained if the underlying probability distributions which generated the data were known and were used at each point in the input pattern space to make the classification decision. A nearest neighbour classifier operates by searching for the n nearest points to each data point using e.g. a Euclidean distance metric, and then taking an average of the labels assigned to these points i.e. whether the point has been specified as belonging to a fall or not. A threshold can then be applied to this score, and points with higher values represent fall detections by the nearest neighbour classifier. As with the neural network, this output threshold was the parameter that was varied to plot the whole range of the ROC curve. The treatment applied to convert these detections into percentages for ROC curve plotting was kept exactly the same as the procedure used with the neural network ROC curves, including scanning for local peaks in the output, to ensure that the curves could be compared. The number of nearest neighbour points used to make the classification decision was varied between 10 and 50, and



(a)



(b)

Figure 11. ROC curves (a) for the nearest neighbour classifier using 30 neighbours, net 89, net 33 and for classifications based on single velocity measurements. The x-axis shows the error rate as a proportion of the total number of data points and the y-axis shows the proportion of true falls detected. A detail from the plot is shown in (b).

the best-performing nearest neighbour classifier was selected using the ROC

curves.

In order to produce a measurement of the performance improvement gained through applying a neural network to this problem, software was written to make the classification decision based on single velocity data points (the lower bound). The central velocity point from each nine-point window was used in place of the neural network output, scaled so that all downwards velocities lay between 0 and 1, but the remainder of the decision code was kept exactly the same as for the neural network. The ROC curve for this classification system was produced and compared to those for the neural networks and nearest neighbour classifiers.

Fig. 11 shows the ROC curves produced by the methods outlined above for the best-performing nearest neighbour classifier (using $n = 30$); the best-performing neural network trained with CGM (net 89); the best performing neural network trained with RPROP (net 33); and the single data point decision system. The best performing neural network was therefore net 89, which had 18 hidden nodes in one layer and was trained with 2000 iterations of CGM. No further improvements in performance were gained either through increasing the number of training iterations, or by using more data in the training phase. The ROC curve for this network lay reasonably close to that for the nearest neighbour classifier, with error rates around 2 times higher. However, the error rate for the single data point decision system was a further factor of five higher, showing that considerable performance gains were realised through the application of a neural network to this problem.

6 Conclusions

Overall, it is clear that the approach of calculating vertical velocities from the infrared images and classifying them as fall or non-fall using a neural network is sufficient to produce a practical fall detector.

Close examination of the scenarios that led to false fall detections showed that most were due to movements specific to the simulations, to movements occurring immediately after a fall, or to changes in the viewing angle of the cameras during the simulations. All of these can justifiably be ignored. The remaining false detections occurred during high-speed sitting events that could more accurately be described as falls into chairs. It might be expected from a comparison of the basic physics of falls into chairs and falls to the ground that these two classes of events would be indiscriminable in terms of the distributions of vertical velocities they generate, and the performance of the nearest neighbour classifier, an honest classifier, when applied to such events strongly supports this. In order to calculate the performance that would be

seen on realistic data additional information, such as the number of times per day that the average subject sits down, would be required. The detection efficiency for true falls can be varied by changing the threshold applied to the output of the neural network. The percentage of true falls detected will also be the percentage of events in the classes indistinguishable from falls that are falsely detected as falls. For example, if the system is tuned to detect 50% of all true falls, then 50% of all high-speed sitting events, i.e. those sitting events involving a period of free-fall, will generate false fall alarms. This proportion, multipled by the number of such events that occur each day with a typical subject, will give the average daily false alarm rate. It is probable that, given the target subject group for this system, such events would be rare and thus the false alarm rate would be low, but only a further study involving evaluation of the prototype system in a realistic environment could determine this.

The study of performance evaluation is vital in placing machine vision on a solid scientific basis. We have described a case study: the development of a vision system to detect natural events in a low-resolution image stream. The work has involved two distinct examples of the assessment of algorithmic design decisions to maximise detection reliability. In the first example this assessment was carried out by comparing measures and estimates made by the system under development with measures obtained independently, in the absence of genuine ground truth data. We have shown that even when these independent measures are themselves noisy, their independence can serve to guide rational design decisions and allow performance estimates to be made. In the second example we have shown that the temporal identification of events can be subjected to a similar performance analysis, and that upper and lower bounds placed on the data by independent classifiers can guide algorithmic design in a different way, providing an estimate of the proximity of the system to optimal performance. In both cases the analyses were performed using ROC curves, showing that, with suitable consideration of the definitions of true and false detection rates, such curves can provide a unified, generic approach to performance evaluation in a wide range of machine vision problems. We therefore believe that, although presented here for one specific system design, such an approach will be applicable to other situations when an image-based system is to be used in the analysis of natural scenes in the absence of a precise ground truth.

Acknowledgments

The authors would like to acknowledge the support of the MEDLINK programme, grant no. P169, in funding part of this work. The support of the Information Society Technologies programme of the European Commission is also gratefully acknowledged under the PCCV project (Performance Characterisation of Computer Vision Techniques) IST-1999-14159. All software is freely available from the TINA website www.niac.man.ac.uk/Tina.

References

1. K.W. Bowyer and P.J. Phillips, *Empirical Evaluation Techniques in Computer Vision*, IEEE Computer Press, 1998.
2. H. I. Christensen and W. Foerstner, *Machine Vision Applications: Special issue on Performance Characteristics of Vision Algorithms*, vol. 9 (5/6), 1997, pp.215-218.
3. R. Klette, H.H. Stiehl, M.A. Viergever and K.L. Vincken, *Performance Characterization in Computer Vision*, Kluwer series on Computational Imaging and Vision, 2000.
4. J. West, J.M. Fitzpatrick, *et al.*, *Comparison and Evaluation of Retrospective Intermodality Brain Image Registration Techniques*, J. Comput. Assist. Tomography, 21, 1997, pp.554-566.
5. P.J. Phillips, H. Moon, S.A. Rizvi and P.J. Rauss, *The FERET Evaluation Methodology for Face-Recognition Algorithms*, IEEE Trans PAMI, 2000.
6. E. Guelch, *Results of Tests on Image Matching of ISPRS III/4*, Intl. Archives of Photogrammetry and Remote Sensing, 27(III), 1988, pp.254-271.
7. I.T. Phillips and A.K. Chhabra, *Empirical Performance Evaluation of Graphics Recognition Systems*, IEEE Trans PAMI, 21(9), 1999, pp.849-870.
8. P.A. Bromiley, N.A.Thacker and P. Courtney, *Segmentation of colour images by non-parametric density estimation in feature space*, Proc. BMVC 2001, BMVA, 2001.
9. E.J. Pauwels and G. Frederix, *Finding Salient Regions in Images: Non-Parametric Clustering for Image Segmentation and Grouping*, Computer Vision and Image Understanding, 1999 75 nos. 1/2 p73-85.
10. J.D. Foley, A. van Dam, S.K. Feiner and J.F. Hughes, *Computer Graphics, Principles and Practice*, Addison-Wesley, Reading, 1990.
11. R.W.G. Hunt, *Measuring Colour*, Second Edition, Ellis Horwood, 1991.