Vertical Axis Detection for Sport Video Analytics

Rui Zeng, Ruan Lakemond‡, Simon Denman*, Sridha Sridharan*, Clinton Fookes*, Stuart Morgan†*

*Speech, Audio, Image and Video Technology Lab, Queensland University of Technology, Brisbane, Australia
†Australian Institute of Sport, Canberra, Australia
‡Imagination Technologies, Kings Langley, United Kingdom

{r5.zeng, s.denman, c.fookes, s.sridharan}@qut.edu.au; ruan.lakemond@imgtec.com; stuart.morgan@ausport.gov.au

***Abstract*—When processing video, it is normally assumed that cameras are vertically oriented such that people appear upright, which helps simplify subsequent processing such as person detection. In real situations, due to the need to provide maximum coverage of the viewing space, cameras are usually placed with arbitrary orientations so the apparent vertical axis of the videos captured may not correspond to the true vertical direction of the captured scene. To rectify this situation, we propose a classification-based system, which normalizes the video compensating for the camera orientation. We demonstrate the performance of the system for outdoor sports video. Our system works as follows: From an arbitrary set of sports videos, we first automatically create a training/testing image dataset, in which players have various orientations. Our classifier is a stacked autoencoder connected to a softmax output layer, which is trained using this dataset for estimating the orientation of players. The orientation of an input video is normalized according to the orientations of player patches, whose angles of orientation are estimated by the above trained classifier. The experiments conducted on hockey field video dataset show that the proposed system is able to estimate the true vertical axis of an input video accurately.

***Keywords* — sparse autoencoder, video orientation normalisation, human detection**

I. Introduction

Human detection is one of the fundamental challenges in human-centred vision systems [1-3]. While calibration is invariably helpful in this task, most uncalibrated methods assume that the image’s vertical axis is roughly aligned with the true vertical axis of the scene, or the 'up' direction [4, 5]. However, this assumption may not be valid in real cases. For example, in most camera installations, maximizing camera coverage is often more important than an obtaining a correct viewing orientation so that cameras may have arbitrary rotations, and the vertical axis of the videos captured may not correspond to the true vertical direction of the scene.

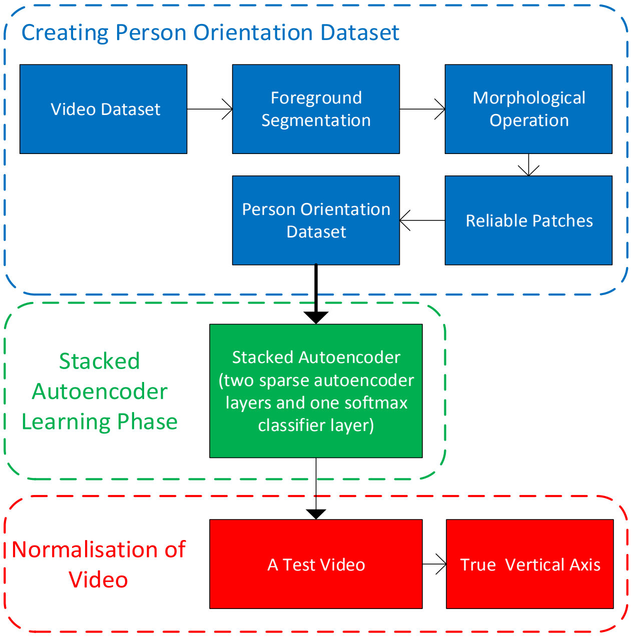

Creating Person Orientation Dataset
Video Dataset
Foreground Segmentation
Morphological Operation
Person Orientation Dataset
Reliable Patches
Stacked Autoencoder Learning Phase
Stacked Autoencoder (two sparse autoencoder layers and one softmax classifier layer)
Normalisation of Video
A Test Video
True Vertical Axis

Figure 1: The overview of our approach for vertical axis detection

Camera used for capturing outdoor sports scene usually suffer from above issues because they need to cover the entire play area. Furthermore, camera calibrations information is usually not available in these situations to normalize the acquired video. The performance of human detectors may be degraded when applied to rotated cameras because the detection algorithms are mostly based on the assumption that the scene and the camera vertical axis are aligned.

Several efforts have been made to counter the problems faced by sports videos with a non-normalized vertical axis. Feature-based methods [6, 7] have been proposed that can recover the vertical direction by making some assumptions about the scene, such as field marking, or geometric lines on playing area. Given that the cues on sports scene are sparse, not unique, and susceptible to noise, player occlusion, and deformation, camera calibration usually fails. Modeling based methods counter un-

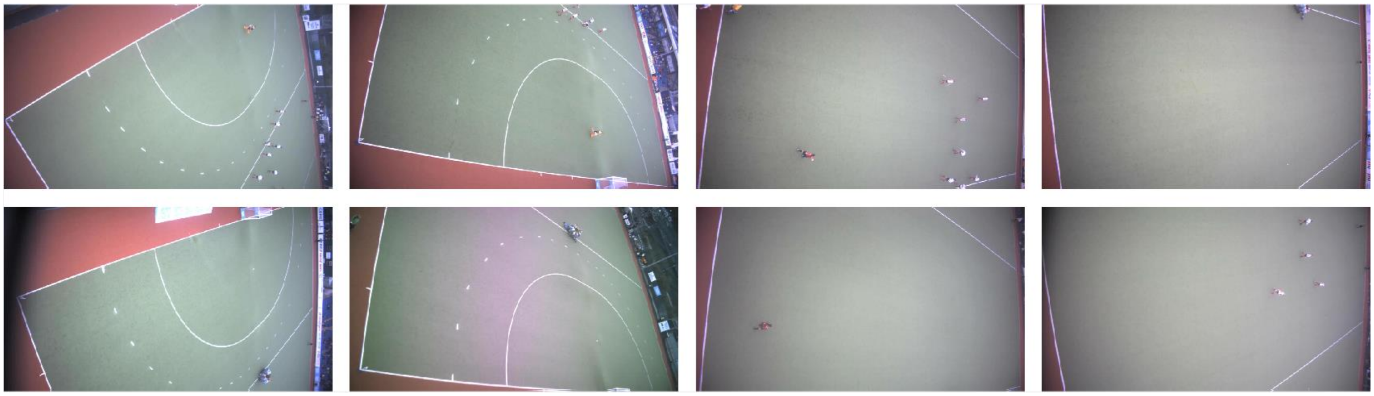


Figure 2: Example samples from field hockey video dataset.

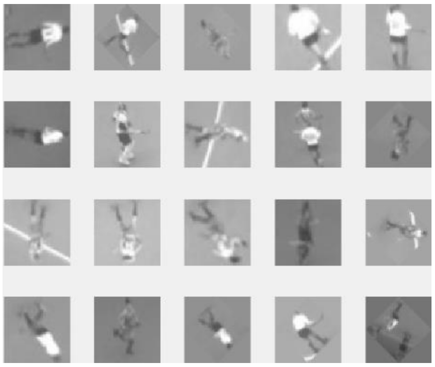


Figure 3: Some representative images in personal orientation dataset.

normalized vertical axis by fitting a human to a particular model for example built by color of the uniform, and histogram of gradients [8]. While these methods can be hand-crafted with success for some specific sport scenes and tasks, applying such approaches to a new scene or sport usually requires domain knowledge owing to the changes in the view angle, field marking, illumination or color of players' uniform. Furthermore, the features visible in a camera view are often inadequate for any feature-based matching approach, or even manual identification. Examples are shown in Figure 2, where the distinguishable surface features are absent or ambiguous.

In sports video analytics, recovering the true direction of the video will greatly reduce the difficulty of detecting and modelling players. To our best knowledge, there is no detailed study on recovering vertical axis in outdoor video so far. Our paper proposes an efficient way to achieve this task.

Figure 1 depicts the flow chart of our technique to determine the true vertical direction of video acquired using a camera with arbitrary rotation. The proposed system consists of three components. First, we automatically extract from a sports video, human images whose orientation correspond to one of eight uniformly spaced directions (i.e.: 45^o apart) in the vertical plane. The true vertical direction is manually annotated (according to the eight 45^o bins) such that this database can be used for training and testing a classifier. The annotation is based the assumption that when the players are in their upright position, the true orientation of each payer will be in the vertical direction. We call this data set "person orientation dataset." Next, we design a stacked autoencoder which contains a softmax classifier layer and train the classifier using images selected from our person orientation dataset. Subsequently, the testing images are employed to evaluate the performance of the classifier on the dataset and report the accuracy of the classification. Finally, we show how to normalize a given unseen video by estimating the true vertical direction using our classifier.

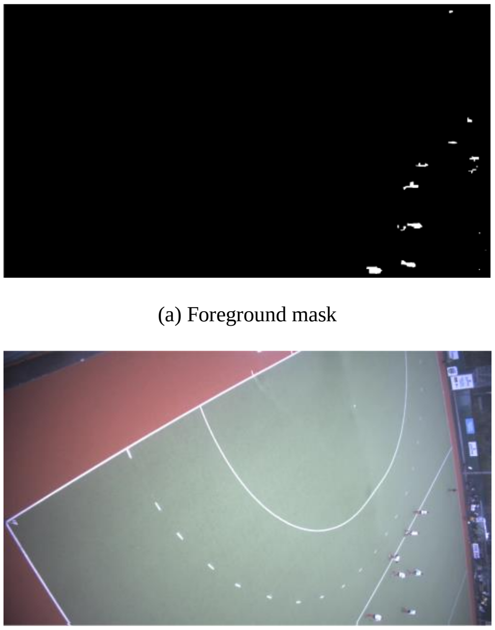
The main contribution of this paper is an efficient way to recover the true vertical axis in an unseen video. Experiments conducted demonstrate that our proposed detector works well to effectively normalize the vertical axis of a video. Although demonstrate the use of this approach for an outdoor sports scene, the technique is equally applicable to videos captured from other scenes.

The remainder of this paper is organized as follows: in Section II we describe how we create the person orientation dataset for training and testing our classifier. In Section III, we describe our classifier. Section IV describes the experimental setup for training/testing and presents/discusses the results. Section V conclude the paper.

II. The Person Orientation Dataset

In this section, we describe our process of creating the person orientation database by extracting image patches from a sports video.

The video data was collected by a sporting body and consists of eight field hockey videos captured from eight fixed cameras in the same match. Each camera captures data at 25 frames per second, at a resolution of 1888 × 1062, and with a static background, which contains only a few illumination variations and small amounts of noise. Representative frames from these eight views are shown in Figure 2. The selected frames highlight the variations among of athlete's uniform and angle of view. Five different kinds of uniform appear in these videos. They are two teams' players, a referee, and two goals keeper respectively. One may see that these eight fixed cameras have been placed in different corners around playing area, and the true vertical direction of videos are compromised by the need



(a) Foreground mask

(b) Original frame

Figure 4: Representative frame of foreground mask (a) and original frame (b).

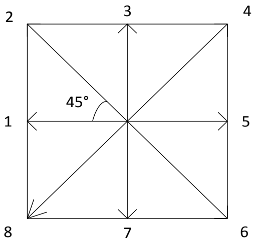


Figure 5: 8 vertical directions of player in our dataset.

to provide maximum coverage of playing filed.

The person orientation dataset created from the sporting videos consists of 14704 person images. Each person image is automatically extracted from the sporting videos, to enable the evaluation of the true vertical direction, which is based assumption players tend to move approximately upright. Figure 3 displays some representative images in this dataset. All images are transformed into gray space and the size of all images is scaled to 50×50 pixels.

A. Image patch extraction

To create the person orientation dataset, we first need to extract image patches of players with different orientation from the sport video data. A Gaussian mixture model based method [6] is used to detect foreground in each frame. In the Gaussian mixture model, each pixel in a video is modeled by a mixture of K Gaussian distribution. The probability that a certain pixel has intensity x_t at time t is estimated as:

$$\Pr(x_t) = \sum_{j=1}^{K} \frac{w_j}{(2\pi)^{\frac{d}{2}} \left|\Sigma_j\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t-\mu_j)^T \Sigma_j^{-1} (x_t-\mu_j)}, \tag{1}$$

where w_j, μ_j, and Σ_j is the weight, the mean, and covariance of the *j*th Gaussian distribution respectively. The first *B* distributions, which are used as a model of the background of the scene, is computed as:

$$B = \arg\min_{b} \left(\left(\frac{\Sigma_{j=1}^{b} w_j}{\Sigma_{j=1}^{K} w_j}\right) > T \right), \tag{2}$$

where threshold *T* is the fraction of the total weight which is set for background model. Subsequently, background subtraction is performed by marking any pixel that is more than 2.5 standard deviations away from any of the B distributions. A foreground mask with the coarse position of players is obtained for each frame in videos. Figure 4 shows binary foreground mask after background subtraction and original frame respectively.

Once a foreground mask has been generated for the estimation of player position, we analyze the blobs using morphological operations to remove noise and connect separated areas of player on foreground mask.

Since the detected blobs contains several holes due to the impact of the noise, an eight-point square neighborhood is used to identify an individual player included in foreground mask as one region. Then, the binarized foreground mask is processed by closing and opening operations to obtain a refined mask showing only the players in the scene. The closing and opening operator are respectively represented as:

$$A \bullet B = (A \oplus B) \ominus B, \tag{3}$$

$$A \circ B = (A \ominus B) \oplus B, \tag{4}$$

where *A* is a binary mask, *B* is structural element [7] and $\oplus$ $\ominus$ represent dilation and erosion operation respectively.

In practice, not all blobs detected above are suitable for inclusion in the dataset, largely because there is some misdetection caused by noise. Rapid change in intensity among a few frames may lead to unstable background and blobs of noise. Thus, we need to filter out these erroneous blobs to prevent false candidates from being included in the dataset. To achieve this, each blob is constrained into a bounding box according to its centroid. Subsequently, bounding boxes whose sizes are too small or are too large are eliminated. Furthermore, bounding boxes with improper aspect ratio are also deleted from binarized foreground mask. In experimental setting, the proper size of bounding box is set to 900 to 4000 pixels squared to filter out small blobs of noise and the aspect ratio is constrained to the range [0.6 1.4]. In final processing, some blobs only appear in a few discontinuous frames and these are removed as well. The remaining blobs are therefore reliable to form the person orientation dataset.

Next, the image patches that contains reliable blobs are manually inspected and classified into one of eight directions. The diagram of directions in our classification is shown in Figure 5. The angle between each adjacent direction pair is 45 degrees.

B. Dataset augmentation and labelling

Since the number of samples in each category is not the same, the distribution is not appropriate to train a robust and reliable

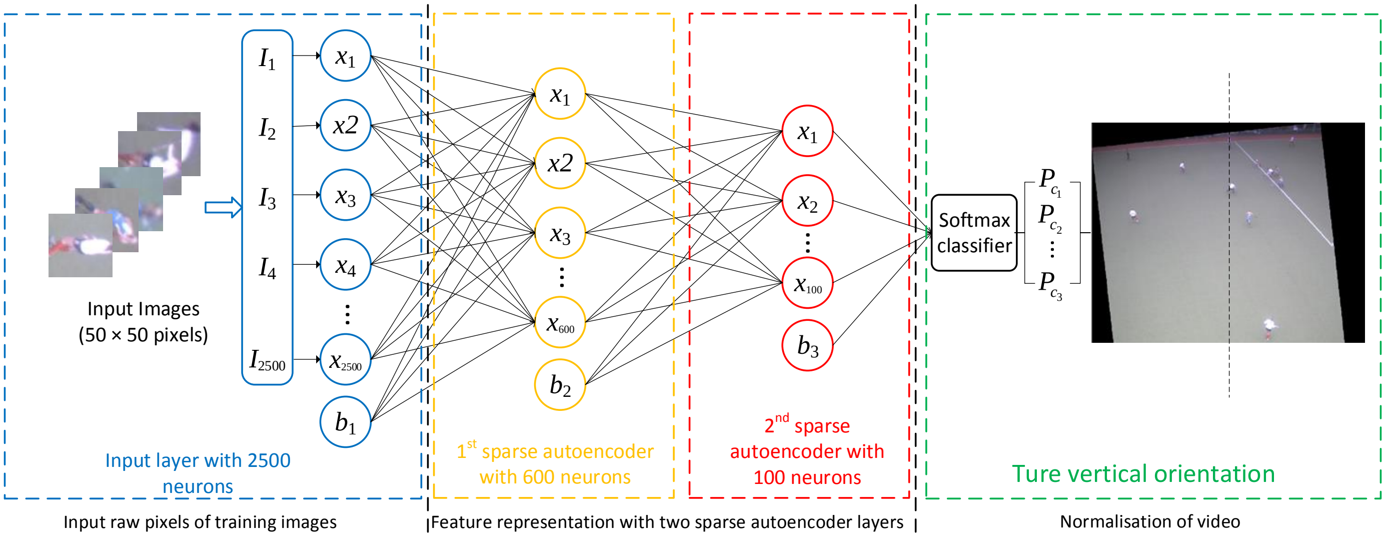

Input Images
(50 × 50 pixels)
I_1
I_2
I_3
I_4
I_{2500}
x_1
$x2$
x_3
x_4
x_{2500}
b_1
Input layer with 2500 neurons
x_1
$x2$
x_3
x_{600}
b_2
1st sparse autoencoder with 600 neurons
x_1
x_2
x_{100}
b_3
2nd sparse autoencoder with 100 neurons
Softmax classifier
P_{c_1}
P_{c_2}
P_{c_3}
Ture vertical orientation
Input raw pixels of training images
Feature representation with two sparse autoencoder layers
Normalisation of video

Figure 6: Two layer stacked autoencoder used for testing performance of up direction dataset.

classifier for estimating the true vertical direction of videos.

The easiest and most common method to address the issue is to artificially enlarge the dataset using a transformation [8-10]. The form of data augmentation used here is to generate image rotations. Each image is augmented such that seven extra images are generated, by applying seven rotations of 45°, i.e. the transformation increases the size of our dataset by a factor of eight. Due to rotating, we cannot avoid losing pixels at the boundary of the image, and the pixels that are lost are replaced with the mean value of the image.

III. Learning Stacked Sparse Autoencoder

For classification, we use a sparse auto-encoder. In this section, we review principles of a typical three-layer autoencoder and then describe the stacked sparse autoencoder used as the classifier. At the end of this section, the reasons for choosing stacked sparse autoencoder are enumerated.

A. *Typical sparse autoencoder*

A sparse autoencoder (AE) [11] is an unsupervised feature learning algorithm which aims to generate a sparse feature representation of high-dimensional input data. A simple, sparse autoencoder is usually represented as a neural network consisting of three layers, in which the number of neurons in the input layer is the same as that of the output layer, trained by a back-propagation algorithm. The cost function to be minimized can be written as:

$$\frac{1}{2N}\sum_{i=1}^{N}\left\|h_{W,b}(X_i)-X_i\right\|^2+\alpha\left\|W\right\|_2^2+\beta\sum_{j=1}^{l_h}\mathrm{KL}(\rho\,||\,\hat{\rho}_j). \quad (5)$$

The first term is an average sum-of-squares error term where N is the number of training images, X_i is the ith input image, W, and b represent weights and biases parameters in the whole sparse autoencoder respectively. Here, $h_{W,b}(X_i)$ is defined as the output of the sparse autoencoder, which may be obtained through a forward propagation of the neural network. The second term is a regularization term that tends to decrease the magnitude of the weights, and helps prevent overfitting. In the third term, l_h, is the number of neurons in the hidden layer, and $\mathrm{KL}(\rho\,||\,\hat{\rho}_j)$ is the Kullback-Leibler (KL) divergence between $\hat{\rho}_j$, i.e. the output value of the neuron j in the hidden layer and presetting sparsity parameter, ρ. α, β are the weight factors of the second term and the third term respectively.

B. *The stacked autoencoder with softmax layer*

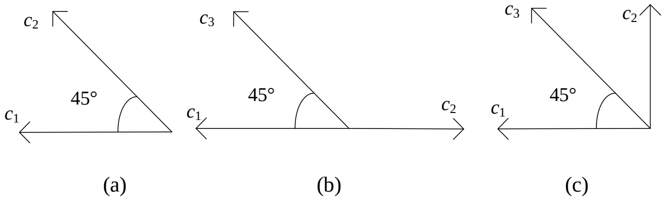
A stacked autoencoder (SAE) [11] is a deep neural network consisting of multiple basic sparse autoencoders. In this paper, a stacked autoencoder with two basic sparse autoencoder (AE) layers and a softmax layer is used for classification of a person's vertical orientation. Figure 6 illustrates the architecture of the SAE utilized.

The set of training images are denoted as $\{I_i\}_{i=1}^{N}$. For the first AE in the SAE system, the cost function shown in Eq.(5) is minimized over parameters W^1 and b^1 via image set $\{I_i\}_{i=1}^{N}$, where superscript 1 of W^1 and b^1 indicates that they are the parameter model in the first SAE. We assume that W_1^1 and b_1^1 are the weight and the bias between the input layer and the hidden layer respectively. Thus, the feature of each image, F_i^1, extracted by the first SAE can be written as

$$F_i^1 = f(W_1^1 I_i + b_1^1), \quad i = 1, 2, \ldots N, \quad (6)$$

where f is defined by a sigmoid logistic function as $f(x) = 1/(1+\exp(-x))$.

Almost repeating the same process as the first SAE, we can get a new feature set $\{F_i^2\}_{i=1}^{N}$ from the old feature set $\{F_i^1\}_{i=1}^{N}$ through the second SAE in our system. The new feature set is a further abstract representation of the original image set at which data has lower dimensionality and better classification performance.



(a) (b) (c)

Fig. 7: The diagram of computing angles of player's orientation. (a) The two categories c_1 and c_2 are adjacent. (b) c_1 and c_3 are in opposite direction. (c) c_1 is orthogonal to c_2, and c_3 is in the middle of c_1 and c_2

In the last layer, we link these two basic AE with a softmax layer, which is an output layer used in classification task, to estimate the probability of each category.

There are several reasons for selecting such a stacked autoencoder. As shown in figure 3, lots of patches are synthesized by a rotating operation and hence the difference between them is only the orientation of the person within them, and slight illumination variation. Traditional image classification methods which using a combination of hand-crafted features and a classifier largely depend on proper feature descriptor and knowledge of the specific tasks. In our case, it is difficult to choose a proper descriptor to obtain discriminative features, and more advanced classification algorithms based on unsupervised feature learning are needed to classify images in this dataset.

Since we have a large the number of samples in the dataset is 14, 704, it is possible for us to train a classifier with a complex structure. The autoencoder, which is a mature deep learning architecture, is easier to train and has more concise structure than other deep learning methods (e.g. deep belief network, convolutional neural network, etc.). Thus, autoencoder classifier is utilized in our dataset under consideration of the balance of performance and efficiency.

Another reason for choosing autoencoder as a classifier is due to the fact that the last layer, which is a softmax layer, may output probability of each category of one image. The next subsection will describe how to compute the angle of true vertical direction according to the probabilities of the reliable image patches.

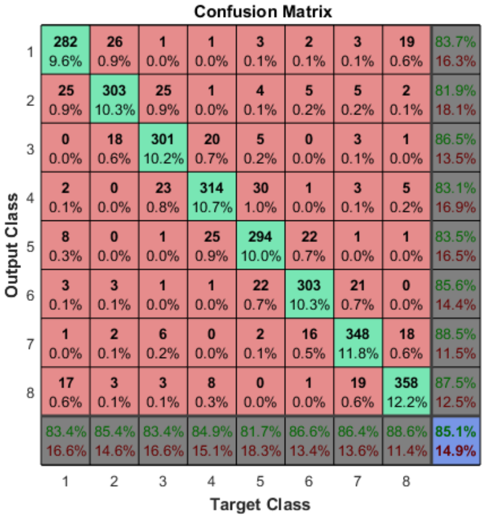
C. *Normalization of Video*

Suppose that there is a reliable image patch *A* extracted from a test video to estimate the vertical direction. After being processed by the stacked autoencoder, eight probabilities $\{P_i\}_{i=1}^{8}$ in descending order which corresponding to categories $\{c_i\}_{i=1}^{8}$ may be obtained. In practice, three cases may occur when computing angle are shown in Figure 7 and illustrated as follows.

1. Classes c_1 and c_2 are adjacent. This case is the simplest case when computing angle. One may compute the angle θ from c_1 to c_2 as follows:

$$\theta = 45^\circ\left(\frac{P_{c_2}}{P_{c_1}}\right). \tag{7}$$

2. The first two classes $c1$ and $c2$ are in opposite direction or nearly opposite direction. The third largest probability c_3 is introduced to assist in angle computation. By comparing c_1 with c_3 and c_2 with c_3, the class whose direction is far away from that of the others is deleted and then compute angle using the probabilities of reaming two classes like Eq. (7).



Confusion Matrix

Output Class									
1	282 9.6%	26 0.9%	1 0.0%	1 0.0%	3 0.1%	2 0.1%	3 0.1%	19 0.6%	83.7% 16.3%
2	25 0.9%	303 10.3%	25 0.9%	1 0.0%	4 0.1%	5 0.2%	5 0.2%	2 0.1%	81.9% 18.1%
3	0 0.0%	18 0.6%	301 10.2%	20 0.7%	5 0.2%	0 0.0%	3 0.1%	1 0.0%	86.5% 13.5%
4	2 0.1%	0 0.0%	23 0.8%	314 10.7%	30 1.0%	1 0.0%	3 0.1%	5 0.2%	83.1% 16.9%
5	8 0.3%	0 0.0%	1 0.0%	25 0.9%	294 10.0%	22 0.7%	1 0.0%	1 0.0%	83.5% 16.5%
6	3 0.1%	3 0.1%	1 0.0%	1 0.0%	22 0.7%	303 10.3%	21 0.7%	0 0.0%	85.6% 14.4%
7	1 0.0%	2 0.1%	6 0.2%	0 0.0%	2 0.1%	16 0.5%	348 11.8%	18 0.6%	88.5% 11.5%
8	17 0.6%	3 0.1%	3 0.1%	8 0.3%	0 0.0%	1 0.0%	19 0.6%	358 12.2%	87.5% 12.5%
	83.4% 16.6%	85.4% 14.6%	83.4% 16.6%	84.9% 15.1%	81.7% 18.3%	86.6% 13.4%	86.4% 13.6%	88.6% 11.4%	85.1% 14.9%
	1	2	3	4	5	6	7	8	

Target Class

Fig. 8: The confusion matrix of classification results from the vertical direction dataset trained by the stacked autoencoder.

3. c_1 is orthogonal to c_2. If the third class is in the middle of the first two, the angle from $c2$ to $c1$ may be computed as

$$\theta = 45^\circ\left(1+\frac{P_{c_3}}{P_{c_1}}\right)\left(\frac{P_{c_2}}{P_{c_1}+P_{c_3}}\right). \tag{8}$$

For those the third class is not in the middle of c_2 and c_1, we ignore them.

Finally, the true vertical axis of video may be computed as the average of the angles of the reliable image patches.

IV. Experimental Design and Results

The person orientation dataset created above contains 14, 704 images which have 8 direction classes. For each class, 80 percent of each class are randomly selected for training and the remaining images are used for testing.

Since the size of images is 50×50, the number of neurons in the input layer for vectorized image is set to 2500. Two SAE layers for intermediate feature extraction contains 600 and 100 neurons respectively. In the last softmax layer, 8 neurons are employed to output the probability of each category. The architecture of the used autoencoder is shown in Figure 6.

We employ a greedy layer-wise approach for pre-training SAE by training two AE and one softmax layer in order. After pre-training is done, all three layers are combined together to form a compact SAE system. Finally, fine tuning is applied to the SAE system to get a classification model which has good accuracy.

The classification results evaluated by the stacked autoencoder are shown in Figure 8 via confusion matrix. The rows of this confusion matrix plot the predicted class of vertical direction, and the columns show the true class. The diagonal

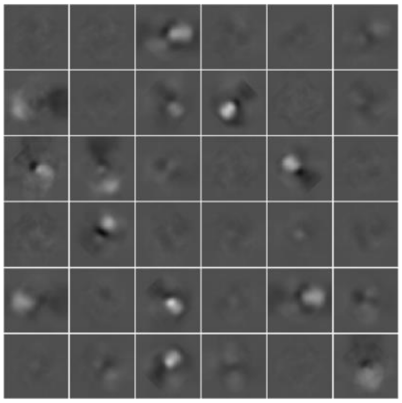


Fig. 9: Visualization of 600 neurons from the first layer of the SAE. As expected, the plot shows detailed boundary features of player and orientation of player.

cells display where the true class and predicted class match. Other cells in off-diagonal show instances where the classifier has made errors. One may see that the misclassified samples are concentrated around the true class they should be. It should be noted that there is a small amount of error in the ground truth labels due to the difficulty of the manual annotation. The large variation in player pose, e.g. running, squatting, makes the data be difficult to label. Thus the method should achieve a higher accuracy than its classification accuracy based on the manual annotation. The right-most column shows the accuracy for each predicted class, while the row at the bottom of the plot shows the accuracy for each true class. The cell in the right-bottom of the plot shows the overall accuracy 85.1%. This accuracy indicates that the performance with the dataset is acceptable. It should be noted that there are still a few samples that are misclassified into opposite direction. In the real case, their correct angle may be computed by the strategy proposed above.

The learned stacked autoencoder filters are shown in Figure 9. An intriguing pattern is observed in the filters. One may see that all the filters attempt to capture the directional pattern of the players and the details of uniform as well as background are ignored automatically by the filters. When there is some background in images, several filters become low-pass, to secure the responses from background images.

The normalizing vertical axis for any given test video is based on the classifier trained above. The normalization is based angles estimated by the stacked autoencoder classifier. Fig. 10 shows the results of the estimated angle for a particular video. One may see that only reliable patches are used for the estimation of angles. From left to right, the estimated angles are 40.853°, 27.398°, 0.67059°, -34.702°, 0.0005° respectively. We compute the angle of the (true) vertical axis of the video as the average of above angles, i.e. 6.3568°. The normalization is done by rotation operation on original video as illustrated in Fig. 11. Rotating the video rectifies the camera rotation to recover the true vertical axis. This simplifies further automatic processing of the video, for example for person detection tasks, as well as provide a good view of the scene for human perception.

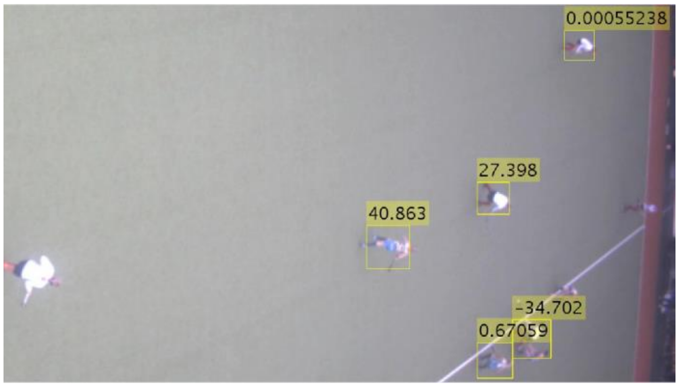


Fig. 10: Estimation of angles in video via patches. From left to right, the estimated angles of players are 40.853°, 27.398°, 0.67059°, -34.702°, 0.0005° respectively. The angles are computed in clockwise from the class 1 shown in figure 4 to the class 8.

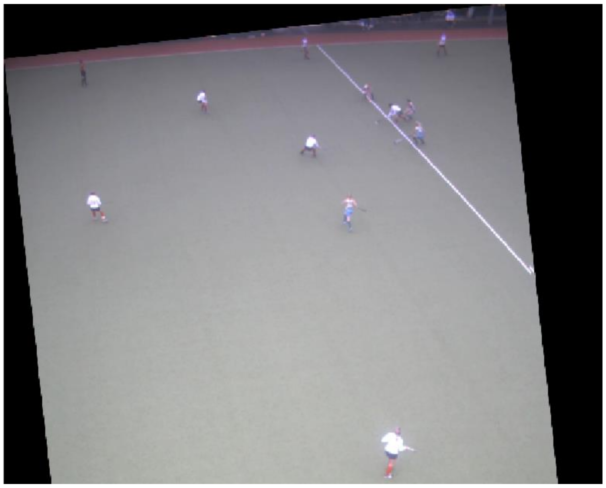


Fig. 11: Rotated video which has normalized the world vertical axis.

V. Conclusion and Future Works

This paper has proposed the semi-automatic creation of a person orientation image dataset from sports video data and proposes a method for estimating the true vertical axis of a given video to normalize the orientation of the video to further analytics and to provide improved video for human perception. Evaluation of our classifier on test data shows an accuracy of over 85%. The experiments conducted on hockey field video dataset show that the proposed system is able to estimate the true vertical axis of an input video accurately. In future work, the normalised video, which is more in line with human vision expectations and the assumptions applied in training various object classifiers, will be used for camera calibration.

References

[1] J. Xu, S. Denman, S. Sridharan, and C. Fookes, "Activity modelling in crowded environments: A soft-decision approach," in *Proceedings of the International Conference on Digital Image Computing techniques and Applications (DICTA)*, 2011, pp. 107-112.

[2] J. Xu, S. Denman, V. Reddy, C. Fookes, and S. Sridharan, “Real-time video event detection in crowded scenes using MPEG derived features: A multiple instance learning approach,” *Pattern Recognition Letters*, vol. 44, pp. 113-125, 2014.

[3] S. Denman, T. Lamb, C. Fookes, V. Chandran, and S. Sridharan, “Multi-spectral fusion for surveillance systems,” *Computers & Electrical Engineering*, vol. 36, no. 4, pp. 643-663, 2010.

[4] P. Kovesi, "Video Surveillance: Legally Blind?," in *Proceedings of the International Conference on Digital Image Computing Techniques and Applications (DICTA)*, 2009, pp. 204-211.

[5] G. Yu, and J. Yuan, "Fast action proposals for human action detection and search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1302-1311.

[6] C. Stauffer, and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[7] A. Antoniou, Digital signal processing, McGraw-Hill Toronto, Canada, 2006.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 1097-1105.

[9] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3642-3649.

[10] D. C. Ciresan, U. Meier, J. Masci, L. Maria Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2011, pp. 1237.

[11] J. Xu, L. Xiang, R. Hang, and J. Wu, "Stacked Sparse Autoencoder (SSAE) based framework for nuclei patch classification on breast cancer histopathology," in *Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI)*, 2014 pp. 999-1002.