FALL DETECTION IN A SMART ROOM BY USING A FUZZY ONE CLASS SUPPORT VECTOR MACHINE AND IMPERFECT TRAINING DATA

Miao Yu, Syed Mohsen Naqvi, Adel Rhuma and Jonathon Chambers

Advanced Signal Processing Group, Electronic and Electrical Engineering Department
Loughborough University, Loughborough, Leicester, UK
{elmy, s.m.r.naqvi, a.rhuma, eljac}@lboro.ac.uk

ABSTRACT

In this paper, we propose an efficient and robust fall detection system by using a fuzzy one class support vector machine based on video information. Two cameras are used to capture the video frames from which the features are extracted. A fuzzy one class support vector machine (FOCSVM) is used to distinguish falling from other activities, such as walking, sitting, standing, bending or lying. Compared with the traditional one class support vector machine, the FOCSVM can obtain a more accurate and tight decision boundary under a training dataset with outliers. From real video sequences, the success of the method is confirmed with less non-fall samples being misclassified as falls by the classifier under an imperfect training dataset.

Index Terms— voxel person, discrete Fourier transform, fuzzy one class support vector machine, fall detection, imperfect training data

1. INTRODUCTION

Fall detection has been of increasing public concern in recent years. Detecting a fall event at home is an indispensable part of elderly people's care because: 1, According to [1], falls are the leading cause of death due to injury among the elderly population and 87% of all fractures in this group are caused by falls. 2, Although many falls do not result in injuries, 47% of non-injured fallers can not get up without assistance and this period of time spent immobile also affects their health.

The most popular fall detection techniques include wearable or portable sensor-based methods, sound or vibration sensor-based approaches and computer vision-based methods. Compared with the initial two types of methods, the computer vision-based methods have advantages that the elderly persons need not wear sensors and they are not affected by the environmental noises that sound or vibration sensors will suffer. There are various ways to detect a fall event using computer vision and signal processing techniques. In [2] and [3], C. Rougier et al. use a threshold-based algorithm to compare the values of the extracted features with the corresponding thresholds to make decisions. The head's 3-D velocity and human shape information are extracted as features respectively. C. Juang and C. Chang in [4] use an elegant self-constructing neural fuzzy inference network for posture recognition to detect a fall. As an effective tool for the classification problem, the SVM technique is applied in [5], the extracted features are finally fed to a multi-class SVM for precise classification of motions and determination of a fall event. In [6], a layered hidden Markov model (LHMM)-based approach is proposed to determine the state of the person (walking or falling) from a multiview pose classification strategy.

However, for the methods mentioned above, they either construct different models for different activities [4] and [6], or build a very complex structure to distinguish falls from other activities such as the multi-class SVM method in [5]. Our work is underpinned by the observation that the fall activity shares similarities and can be ascribed to one class. This motivates us to use the one class classification technique for fall detection. In [7], M. Yu et al. propose the idea of using a one class classifier for fall detection and different one class classifiers are compared, the results show that the OCSVM achieves the best performance by obtaining the largest Geometric mean defined as $\sqrt{TPR * (1 - FPR)}$, where TPR is the true positive detection rate and FPR is the false positive detection rate. However, we found that if the training dataset is not perfect, a good classification result can not be obtained although OCSVM is robust to outliers to a certain extent. In order to solve this problem, an FOCSVM is employed in this work and small weights are given to the training points corresponding to the outliers. The features to train the FOCSVM classifier and used to make decisions are obtained from the variation of a person's 3-D angle and centroid information. Two cameras are used to reconstruct a 3-D voxel person and the 3-D angle and centroid information are then obtained. The structure of this paper is as follows: Section II describes how the video features are extracted. The concept of FOCSVM is introduced in Section III. Some experimental simulations are presented in Section IV. Conclusions and suggestions for future work are given in Section V.

2. VIDEO FEATURE EXTRACTION

We use 3-D features for the construction of the classifier, a table look-up scheme is used to reconstruct the 3-D voxel person. A codebook is built beforehand to obtain the relation between 2-D pixels and 3-D 'voxels'. Initially, we divide our 3-D room space into fixed size voxels (2.5cm*2.5cm*2.5cm), which are nonoverlapping cubes.

From the 2-D coordinate of a pixel in the recorded frame $[\iota_x, \iota_y]^T$, the undistorted coordinate $[u_x, u_y]^T$ on the focal plane can be obtained from the camera calibration [8], where $[\cdot]^T$ denotes vector transpose. Moreover, we can also obtain the rotation matrix $\boldsymbol{R}$ and translation vector $\mathbf{t}$, which reflect the relationship between the real world coordinate system and a camera-centered coordinate system from the procedure proposed in [8]. After obtaining these values, we can translate the origin of the camera-centered coordinate system $[0, 0, 0]^T$ and the point on the focal plane $[u_x, u_y, f]^T$ into the corresponding 3-D real-world coordinate by: $\mathbf{z} = \mathbf{R}^{-1}(\mathbf{c} - \mathbf{t})$, where $\mathbf{z}$ is the real world coordinate and $\mathbf{c}$ is the coordinate in the camera-centered coordinate system.

So, for a pixel, a ray can be constructed and we can identify a set of voxels that this ray intersects. The procedure is repeated for every pixel in the 2-D frame for each camera and we can thereby obtain a pixel-voxel table for each camera.

Two video cameras located at the corners of a room environment are then used to record the video frames and we use the codebook background subtraction technique [9] to obtain the foreground human body region in these frames. When the silhouettes of a person in the frames recorded by two cameras are extracted, for

978-1-4577-0539-7/11/$26.00

each frame, we set the 3-D space voxels corresponding to the pixels in the silhouette with the value '1' by looking up the pixel-voxel table, the remaining voxels are set to be '0'. In this way, a 3-D binary matrix $\mathbf{V}_t^i$ is constructed for the ith camera at time t. An intersection operation is then applied to the two 3-D matrices obtained by both cameras to obtain a final matrix,$\mathbf{V}_t$, with the representation as $\mathbf{V}_t = Intersection(\mathbf{V}_t^1, \mathbf{V}_t^2)$, an element of $\mathbf{V}_t$ is one if both its counterparts in $\mathbf{V}_t^1$ and $\mathbf{V}_t^2$ are ones. The '1s' region in the 3-D matrix corresponds to a 3-D voxel person.

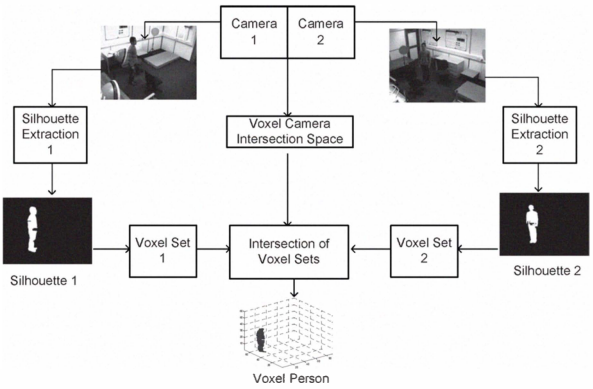

Fig. 1: The procedure of constructing the voxel person by using two cameras

The procedure of the 3-D voxel person reconstruction is shown in Figure 1. We next consider the video feature extraction process.

2.1. Video feature extraction

The corresponding video features are then extracted after we obtain the voxel person $\mathbf{V}_t$. We know that a fall is usually a short activity lasting less than 1s and the variations of orientation angle and centroid in fall activities are different from those of other activities such as lying or walking. So, we need to extract certain features which can reflect the variations of orientation angle and centroid during a short interval (here we use 1s) in order to recognize falls. In order to obtain such a feature, firstly, we calculate the centroid position and a value called $gpsim_t$ reflecting the similarity of the voxel person's primary orientation for every recorded frame.

The centroid of the voxel person at time t, $\mathbf{u}_t = [x_t, y_t, z_t]$ can be obtained by: $\mathbf{u}_t = (\frac{1}{M})\sum_{j=1}^{M}\mathbf{V}_{t,j}$, where M is the number of voxels belonging to the human body region.

The sample covariance matrix used to define the eigen information is: $(\frac{1}{M})\sum_{j=1}^{M}(\mathbf{V}'_{t,j} - \mathbf{u}_t)(\mathbf{V}'_{t,j} - \mathbf{u}_t)^T$.

The eigenvalues and orthonormal eigenvectors of the covariance matrix are calculated and the eigenvector corresponding to the largest eigenvalue at time t is denoted as $eigenvec_t$ and a value denoted by $gpsim_t$ is calculated by: $gpsim_t = max(eigenvec_t \cdot \langle 0,0,1\rangle^T, -eigenvec_t \cdot \langle 0,0,1\rangle^T)$, where $(\cdot)$ represents the dot product.

If the person is upright, the value is near unity; if he or she is on the ground, the value is near zero. The value of $gpsim_t$ is in the range of $[0, 1]$.

We obtain three sequences $sequence_1$, $sequence_2$ and $sequence_3$ which represent the variations of the centroid's horizontal position and vertical position, and the orientation angle from a video clip of one second. We firstly apply first order differences on each sequence, as: $dsequence_n(i) = |sequence_n(i) - sequence_n(i+1)|$, where $dsequence_n$ is the first order difference result for $sequence_n$.

Next for $dsequence_n$, we apply the discrete Fourier transform (DFT) operation as:

$$DFTdsequence_n(k) = abs(\sum_i dsequence_n(i) exp(-j*2*\pi*i*k/N)) \quad (1)$$

where k=0,..,N-1, and N is the length of $dsequence_n$.

Because the DFT amplitudes are invariant to time shifting, the $dsequence_n s$ with similar shapes but different time shifting will have $DFTdsequence_n s$ with similar magnitudes. That means for the video clips which contain falls occurring at different times, the extracted $DFTdsequence_n$ (n=1,2,3) magnitudes will be similar.

In our experimental setting, the frame rate is 15fps, the lengths of $sequence_n$, $dsequence_n$ and $DFTdsequence_n$ are all 15. For each video clip lasting one second, we calculate three $DFTdsequence_n s$ and sample the initial 5 coefficients for each $DFTdsequence_n$, the obtained 15 coefficients are then concatenated to form a 15-dimensional vector which is the video feature we use for training or testing. From the experimental results, we show that the video features for falls have similar patterns and differ from those of other activities.

3. FUZZY ONE CLASS SUPPORT VECTOR MACHINE CLASSIFICATION

The traditional one class support vector machine (OCSVM) is proposed in [10]. The basic idea behind OCSVM is that given a data set drawn from an underlying probability distribution P for the minority class, the OCSVM estimates a function f that is positive in a region S and negative in the complement, where S is the 'most-likely region'– a subset of the input space such that a test point drawn from P lies outside of S equals some a priori specified value between 0 and 1. In the application of fall detection, the minority class corresponds to fall activities. We obtain the training video features from the video clips containing the fall activities and use them to train an OCSVM classifier. The classifier can then capture the 'most-likely' region where the falling video features fall. And if the test video feature point is within this region, it is recognized as fall; otherwise, it is regarded as non-fall.

For a non-separable dataset, a kernel OCSVM [10] can be used to separate the mapped dataset in a high dimension feature space. The strategy of a kernel OCSVM is to map the training data into the feature space $\mathbf{x}_i \rightarrow \Phi(\mathbf{x}_i)$ to separate them from the origin with maximum margin. The popularly used kernels include Polynomial kernel, Gaussian kernel and Tangent kernel. In this paper, we use Gaussian kernel with the form $K(\mathbf{x}, \mathbf{y}) = e^{-\gamma\|\mathbf{x}-\mathbf{y}\|^2}$.

To design the classifier, we try to solve the following quadratic problem:

$$\min_{\mathbf{w},\mathbf{h},\rho} \quad \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu\ell}\sum_i h_i - \rho$$
$$\text{subject to } (\mathbf{w}\cdot\Phi(\mathbf{x}_i)) \geq \rho - h_i, \quad h_i \geq 0 \quad (2)$$

Here, $\nu \in (0, 1]$ and ℓ is the number of training data samples. The nonzero slack variables h_i are introduced to allow for the possibility of outliers (the data points which are not drawn from the distribution P).

For a new test point $\mathbf{x}$, the decision function is: $f(\mathbf{x}) = sgn((\mathbf{w}\cdot\Phi(\mathbf{x})) - \rho)$, where $sgn(\cdot)$ is a sign function which yields the sign of the term in the bracket.

Using multipliers $a_i, b_i \geq 0$, we introduce a Lagrangian [11]:

$$L(\mathbf{w}, \mathbf{h}, \rho, \mathbf{a}, \mathbf{b}) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 + \frac{1}{\nu\ell} \sum_i h_i - \rho - \sum_i a_i((\mathbf{w} \cdot \Phi(\mathbf{x})) - \rho + h_i) - \sum_i b_i h_i$$

We set the derivatives with respect to the primal variables $\mathbf{w}, \mathbf{h}, \rho$ equal to zero respectively and obtain:

$$\mathbf{w} = \sum_i a_i \Phi(\mathbf{x}_i) \tag{3}$$

$$a_i = \frac{1}{\nu\ell} - b_i \leq \frac{1}{\nu\ell} \tag{4}$$

$$\sum_i a_i = 1 \tag{5}$$

According to the KarushKuhnTucker conditions (KKT) [11], the following constraints are satisfied:

$$a_i((\mathbf{w} \cdot \Phi(\mathbf{x})) - \rho + h_i) = 0 \tag{6}$$

$$b_i h_i = 0 \tag{7}$$

We substitute equations (3), (4) and (5) into the Lagrangian function and obtain a dual problem as:

$$\min_{\mathbf{a}} \quad \frac{1}{2} \sum_{ij} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j)$$
$$\text{subject to } 0 \leq a_i \leq \frac{1}{\nu\ell}, \quad \sum_i a_i = 1 \tag{8}$$

Here a_i is the component of vector $\mathbf{a}$, $k(\mathbf{x}_i, \mathbf{x}_j) = (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j))$ is the 'kernel function' (we use Gaussian).

According to the KKT condition [11], the decision function follows as: $f(\mathbf{x}) = sgn(\sum_i a_i k(\mathbf{x}_i, \mathbf{x}) - \rho)$.

For the value of ρ, we can obtain it from the KKT condition. According to equations (6) and (7), we can see if a_i and b_i are non-zero, the corresponding pattern $\mathbf{x}_i$ satisfies: $\rho = (\mathbf{w} \cdot \Phi(\mathbf{x}_i)) = \sum_j a_j k(\mathbf{x}_j, \mathbf{x}_i)$.

However, in some real world applications (such as fall detection), the effects of the training points are different. It is obvious that we can not always obtain a perfect training dataset, some training samples would be outliers and they should be less important than other 'good' training samples in the design of the classiifier. In order to reflect the importance of different training samples, we assign each training data point with an associated fuzzy membership and the training dataset becomes: $(\mathbf{x}_1, u_1), ..., (\mathbf{x}_{N'}, u_{N'})$, where N' is the number of samples in the training dataset. The fuzzy membership u_i which represents the likelihood of the corresponding point $\mathbf{x}_i$ being the target class is calculated as proposed in [12]: $u_i = 1 - \frac{||\mathbf{x}_i - \mathbf{x}_{mean}||}{r_{target}}$, where $r_{target} = max_i ||\mathbf{x}_i - \mathbf{x}_{mean}||$.

The constrained optimization problem of the fuzzy one-class SVM is then formulated as:

$$\min_{\mathbf{w}, \mathbf{h}, \rho} \quad \frac{1}{2} \parallel \mathbf{w} \parallel^2 + \frac{1}{\nu\ell} \sum_i u_i h_i - \rho$$
$$\text{subject to } (\mathbf{w} \cdot \Phi(\mathbf{x}_i)) \geq \rho - h_i, \quad h_i \geq 0 \tag{9}$$

From the similar procedure of solving the traditional OCSVM problem, we can obtain the dual problem as:

$$\min_{\mathbf{a}} \quad \frac{1}{2} \sum_{ij} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j)$$
$$\text{subject to } 0 \leq a_i \leq u_i \frac{1}{\nu\ell}, \quad \sum_i a_i = 1 \tag{10}$$

4. EXPERIMENTS AND EVALUATIONS

The experiments were carried out in Loughborough University's Smart Room. Two cameras are located at the corners for which the intersection of their covering spaces is the whole room space (4.5m*3.5m*3m). The two cameras are connected to two PCs. The StreamPix 3 software [13] is installed on the PCs to perform video recordings and the format of the obtained video is AVI. Video recordings are converted to consecutive 320*240 frames for further processing by MatLab. A synchronizer which is connected to the server is used to ensure the synchronization of the cameras' recordings.

A stuntman simulates the fall and non-fall activities, and three datasets (available from the first author) are recorded by each of the four cameras respectively. The first dataset containing video clips containing falls is used for training. The second dataset (23 falls and 23 non-falls) is for validation purpose—-to aid in the learning of the parameters of the FOCSVM. Finally, a test video dataset (46 falls and 46 non-falls) is recorded and used for evaluation of the performances of the constructed classifiers.

A pixel-to-voxel table is pre-built for the two cameras, the computation amount for obtaining the human body voxels after background subtraction only involves looking up two tables so that it can be real-time. From the reconstructed voxel person, we can get the person's 3-D centroid position and angle information. For a 1s video clip, we can obtain the variations of the person's 3-D centroid position and angle values which are represented as $sequence_1$, $sequence_2$ and $sequence_3$ from 15 frames. From the procedure proposed in Section 2.1, we can obtain the video features of the DFT amplitudes. Figure 2 and 3 show the comparisons of video features of fall activities and non-fall activities respectively.

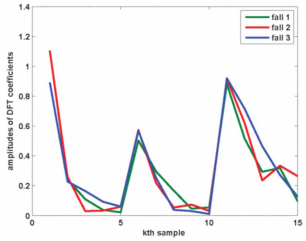

Fig. 2: Video features of three fall activities confirming their similar behavior

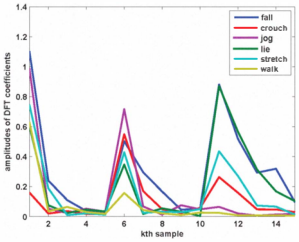

Fig. 3: Video features of five non-fall activities showing their variability and difference from one fall example (in blue)

By visual inspection, we can see that the feature sequences for fall activity are similar and different from those of non-fall activities.

The obtained features are fed into our FOCSVM for training and testing. The validation dataset is used and a grid search method is applied to obtain the optimal parameters ν and γ which maximize

the Geometric mean defined as $\sqrt{TPR * (1 - FPR)}$ for this validation dataset.

Table 1 shows the fall detection results by using OCSVM and FOCSVM under a perfect and three imperfect training datasets, the perfect dataset is composed of 22 well obtained training samples for fall activity, while the imperfect datasets are composed of the same 22 well obtained samples and some poorly obtained fall training samples (due to some reasons, such as poor background subtraction result or incorrect 3-D person reconstruction), which can be regarded as outliers.

Table 1: The performance comparison between OCSVM and FOCSVM under perfect and imperfect training datasets

	OCSVM			FOCSVM		
No. of outliers	TPR	FPR	Geometric mean	TPR	FPR	Geometric mean
0	100%	2%	0.99	100%	2%	0.99
1	100%	12%	0.94	100%	2%	0.99
2	100%	17%	0.91	100%	2%	0.99
3	100%	20%	0.89	100%	2%	0.99

From Table 1, we can see the TPR, FPR and Geometric mean are the same for OCSVM and FOCSVM with the perfect training dataset; however, for imperfect training datasets, the FPR of OCSVM increases (so the Geometric mean decreases) with the number of the outliers while those of FOCSVM remain the same as the obtained results from the perfect training dataset. This advantage has been found for various other datasets with more outliers but these three examples just provide appropriate illustration.

In order to visualize the classification result, we use principle component analysis (PCA) to project the 15-dimensional features into 2-dimensional space, and corresponding classifiers are built up according to these 2-dimensional features. In Figure 5 the 2-dimensional spaces of the two principle components extracted from the length 15 feature vectors are shown on these plots, the two axe pc1 and pc2 stand for the two principle components obtained from PCA, the blue crosses correspond to falls and the red circles correspond to non-falls for the test dataset, we can see that the 2-dimensional projections of fall feature sequences are in a cluster which is nearly non-overlapped with the projected non-fall feature sequences. So, they are distinguishable and a proper boundary could be constructed to enclose the cluster of fall samples.

One example is presented to show the comparison of OCSVM and FOCSVM under a perfect training dataset and an imperfect training dataset with one outlier. The decision boundaries for OCSVM and FOCSVM are shown in red and blue respectively. We can see that for the perfect training dataset, the obtained decision boundaries are approximately similar (a); however, FOCSVM can obtain a much accurate decision boundary compared to OCSVM for the imperfect training dataset (b) and this accounts for the lower FPR shown in Table 1 for the one outlier case. So, from this example, FOCSVM appears more suitable to be used in the real application of fall detection to cope with the uncertanties in the training dataset.

5. CONCLUSION

In this paper, we have proposed a new fall detection method based on FOCSVM with novel 3-D features. A voxel person is extracted from the 2-D human silhouettes obtained from the images of two cameras. The video features which reflect the variations of centroid position and orientation angle are extracted from a series of reconstructed voxel persons during a time interval (1s). FOCSVM is used with the obtained video features as the classifier. The experimental results show as compared with the conventional OCSVM, the FOCSVM can achieve a more accurate fall detection result under an imperfect training dataset with outliers.

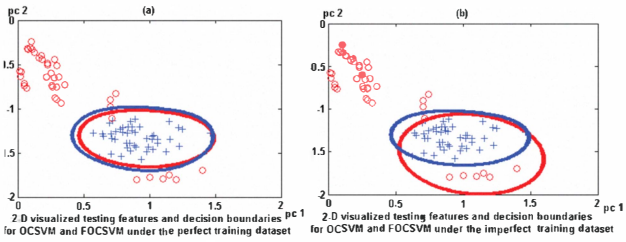

Fig. 4: Two-dimensional feature spaces and decision boundaries for OCSVM and FOCSVM

6. REFERENCES

[1] "Old people- NMC Facts And Figures, UK," *http://www.medicalnewstoday.com/articles/142487.php*.

[2] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Monocular 3d head tracking to detect falls of elderly people," *Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE*, pp. 6384–6387, 2006.

[3] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Fall detection from human shape and motion history using video surveillance," *In Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pp. 875–880, 2007.

[4] C. Juang and C. Chang, "Human body posture classification by a neural fuzzy network and home care system application," *IEEE Transactions on Systems, Man, and Cybernetics, Vol. 37*, pp. 984–994, 2007.

[5] H. Foroughi, A. Rezvanian, and A. Paziraee, "Robust fall detection using human shape and multi-class support vector machine," *Computer Vision, Graphics and Image Processing, 2008. ICVGIP '08. Sixth Indian Conference on*, pp. 413–420, 2008.

[6] N. Thome, S. Miguet, and S. Ambellouis, "A real-time, multiview fall detection system: A LHMM-based approach," *IEEE Transactions on Circuits and Systems for Video Technology, Vol. 18*, pp. 1522–1532, 2008.

[7] M. Yu, S. Naqvi, and J. Chambers, "Video-based fall detection system by using the boundary method," *Submitted to IET Proc. Image Processing*.

[8] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation, Vol. 3*, pp. 323–344, 1987.

[9] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using code-book model," *Real-Time Imaging, Vol. 11*, pp. 172–185, June 2005.

[10] B. Scholkopf, J. Platt, J. Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation, Vol. 13*, pp. 1443–1471, 2001.

[11] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, California*, pp. 232–237, 1998.

[12] P. Hao, "Fuzzy one-class support vector machines," *Fuzzy Sets and Systems*, vol. 159, no. 18, pp. 2317 – 2336, 2008.

[13] "Norpix, digital video recording software," *http://norpix.com/products/streampix.php*.