Automatic Workflow Monitoring in Industrial Environments

Galina Veres[1], Helmut Grabner[2], Lee Middleton[1] and Luc Van Gool[2,3]

[1]University of Southampton, IT Innovation Centre, UK
[2]Computer Vision Laboratory, ETH Zurich, Switzerland
[3]ESAT - PSI / IBBT, K.U. Leuven, Belgium

Abstract. Robust automatic workflow monitoring using visual sensors in industrial environments is still an unsolved problem. This is mainly due to the difficulties of recording data in work settings and the environmental conditions (large occlusions, similar background/foreground) which do not allow object detection/tracking algorithms to perform robustly. Hence approaches analysing trajectories are limited in such environments. However, workflow monitoring is especially needed due to quality and safety requirements. In this paper we propose a robust approach for workflow classification in industrial environments. The proposed approach consists of a robust scene descriptor and an efficient time series analysis method. Experimental results on a challenging car manufacturing dataset showed that the proposed scene descriptor is able to detect both human and machinery related motion robustly and the used time series analysis method can classify tasks in a given workflow automatically.

1 Introduction

Intelligent visual surveillance is an important area of computer vision research. In this work we focus on real-time workflow monitoring in industrial environments. The aim is to recognise tasks happening in the scene, to monitor the smooth running of the workflow and to recognise any abnormal behaviours. Any deviation from the workflow either by the workers or by the machinery may cause severe deterioration of the quality of the product or may be dangerous.

An example of such an industrial scenario is shown in Fig. 1. By monitoring industrial scenes, one faces several challenges such as recording data in work areas (camera positions and viewing area), industrial working conditions (sparks and vibrations), difficult structured background (upright racks and heavy occlusion of the workers), high similarity of the individual workers (nearly all of them wearing a similar utility uniform), and other moving objects (welding machines and forklifts). Furthermore, the dynamics of workflow can be quite complex and deviations typically occur. Several tasks within a workflow can have different lengths and no clear definition of beginning/ending. Moreover, the tasks can include both human actions and motions of machinery in the observed process;

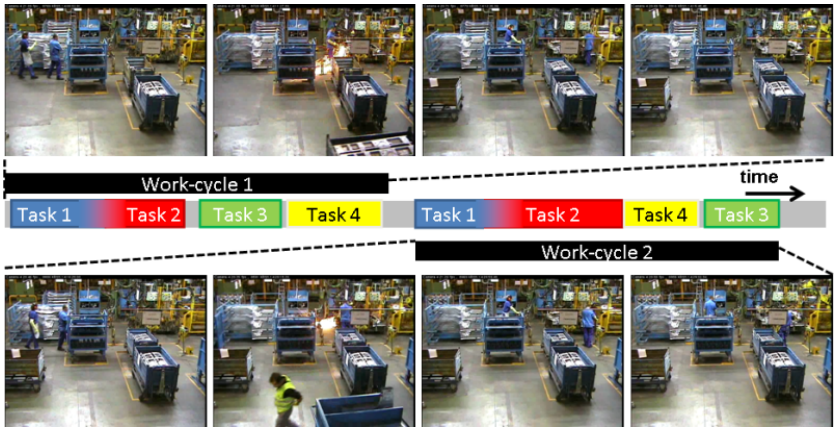


Fig. 1. Example of an industrial scenario: a workflow consists of several tasks; different length tasks; no clear definition of beginning/ending; coping with irrelevant motion and occlusions.

whereas other motions not related to the workflow have to be suppressed (e.g., other people passing or people restocking or replacing racks).

Related work. A whole field of operational research is concerned with the design and data mining from workflows. In the computer vision community, this is mainly addressed in applications such as abnormal behaviour recognition or unusual event detection. Many approaches have been suggested over the past years and a complete review is far beyond the scope of this paper. Typically they build a model of normality and the methods can differ in (i) the model used, (ii) the employed learning algorithm for learning the model parameters, and (iii) the features used. Models might be previously trained and kept fixed [11, 18, 19] or adapt over time [3] to cope with changing conditions. Various machine learning and statistical methods have been used: from clustering [2] and density estimation [8] to Hidden Markov Models for time series analysis commonly used in action recognition [12, 14]. Also a broad variety of extracted image features are used such as global scene descriptors [3], optical flow [1, 10], 3D motion [14] or object trajectories [17, 8].

Depending on the used model and prior knowledge, a lot of training data has to be collected. For example, the purely data driven approach of [3] has to observe every situation and is not able to generalise a person walking on a slightly different path. On the other hand, using more sophisticated "high-level" features (e.g., a person's trajectory) usually requires less data to generalise well. However, these features have to be extracted robustly. Recent advances in object detection and tracking showed that these methods do not perform very well in a cluttered/occluded environment such as those presented in this paper.

Contribution. To our knowledge, no state-of-the-art approach is able to cope with the challenges of workfow analysis within industrial environments

(a) Person detection [4] (b) Person tracking [5]

Fig. 2. Examples of state-of-the art methods for detection and tracking.

as reviewed above. We tried several state-of-the-art methods for person detection/tracking [4, 5][1]. However, none of them showed stable and robust results in the industrial environment. Fig. 2 (a) shows typical failures of the detector on our dataset with a recall of 24% and a precision of only 9%. Thus, tracking-by-detection approaches (e.g., [9]) cannot be used to generate trajectories. Also, the person could be hardly tracked as displayed in Fig. 2 (b). The tracker starts very well, however it soon loses the person and drifts away.

The reasons for the failures are due to the nature of the environment such as significant occlusions, clutter similar in structure/shape to a person, the workers coloured similarly to the racks, and the unstable background due to welding flare, machinery operation, and lighting changes. Any of these in isolation would cause problems for person detection and tracking, but all of them together make the problem especially difficult for both detection and tracking and violates the use of approaches which are based on the analysis of trajectories.

Hence, in this paper we propose a novel approach for *real-time workflow monitoring*. It consists of a robust, yet simple, scene descriptor and an efficient time series analysis method. The model is easy to train and update, i.e. it allows adaptation to new workflows and inclusion of prior knowledge. Several activities happening simultaneously can be incorporated in the model while the algorithm is still robust to irrelevant motions.The experimental results on a car manufacturing database show that our approach is suitable to monitor workflows robustly and thus allows for (i) obtaining statistics of the workflow in order to optimise it, as well as (ii) detecting abnormalities (for safety reasons).

The remainder of the paper is organised as follows. In Sec. 2 we present image descriptors and model learning techniques, which are then used by our novel approach for real-time workflow analysis in Sec. 3. Detailed experiments and comparisons are shown in Sec. 4. Sec. 5 concludes the paper.

[1] The code was downloaded from the authors webpages.

2 Problem formulation and modelling

Our goal is to monitor a pre-defined repetitive workflow. We describe a workflow as a sequence of defined tasks that have to be executed in some order (permutations are allowed) due to safety or quality reasons. A task is a sequence of observations that corresponds to a physical action like "take object and place it somewhere".

Problem modeling. Let $I_t \in \mathbb{R}^{n \times m}$ be gray scale image at time t. Given an image sequence $\mathcal{I} = \{I_0, \ldots, I_t\}$ and a set of $L+1$ possible tasks, $\mathcal{L} = \{1, \ldots, L, \#\}$, where $\#$ corresponds to a task not related to the workflow. We want to associate a task $l_t^\star \in \mathcal{L}$ with each image I_t at time t, using measurements obtained up to time t. This can be seen as temporal $L+1$ class classification problem.

In summary, the goal is to identify for each frame whether the frame belongs to the workflow or not. If the frame belongs to the workflow, then we need to identify which task takes place. Our approach is based on using local motion monitors which serve as scene features for a powerful time series analysis network. In the following we define and review these two components before introducing our approach.

2.1 Motion-Grid

Features extracted from the raw pixel values should be discriminative enough to capture relevant changes with respect to the tasks but at the same time be invariant to irrelevant variations. Motivated by the fact that workflow consists of actions of an object (walking of a person) and interactions of objects within the environment (using a welding machine), we use motion as our primary feature cue. Inspired by [1] we introduce local motion monitors.

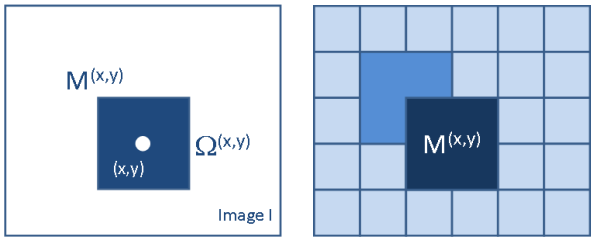
Local Motion Monitor. A local motion monitor $M^{(x,y)}$ observes a position (x, y) and the surrounding $(n \times m)$ pixel neighbourhood $\Omega^{(x,y)}$ of the image (see Fig. 3(a)). The binary output of a motion monitor applied on the image I_t is defined as

$$M^{(x,y)}(I_t) = \begin{cases} 1 & \text{if } \sum_{(i,j)\in\Omega^{(x,y)}} |I_t(i,j) - I_{t-1}(i,j)| > \theta_M \\ -1 & \text{otherwise} \end{cases} \quad (1)$$

In other words, simple frame differencing is used to get changes of the image. If the changes are significant (specified by the parameter θ_M) within the region $\Omega^{(x,y)}$ the local motion monitor $M^{(x,y)}$ "fires", i.e. returns a positive response.

Motion-Grid. A motion grid is defined as a set of local motion monitors. The idea of motion grids is to sample an input image sequence by using a fixed overlapping grid. Each grid element corresponds to a local motion monitor as illustrated in Fig. 3(b).

We use the motion grid as a scene descriptor. In other words, the local motion monitors can be seen as features which extract high level information from each image. For one time instance t we concatenate the output of the local motion monitors within the grid into a vector.



(a) local motion monitor (b) motion-grid

Fig. 3. (a) a local motion monitor $M^{(x,y)}$ at position (x, y) analyses the intensity changes in its local neighborhood $\Omega^{(x,y)}$. (b) an overlapping grid of local monitors

2.2 Echo State Network

Often Hidden Markov Models [14, 12] are used for analysing temporal data. However, HMM require well-defined states formed by robustly extracted features. This is not always possible in industrial settings. In this paper we propose to use an alternative approach: Echo State Networks (ESN) [6] which offer several benefits such as (i) fast and simple learning of many outputs simultaneously, (ii) possibilities of both off-line and on-line learning, (iii) ability to learn complex dynamic behaviours, and (iv) directly dealing with high dimensional input data. They have been shown to perform very well in the context of supervised learning, for example emotion recognition [15] and speech recognition [16].

Principle. An ESN is a discrete time, continuous state, recurrent neural network proposed by Jaeger [6]. Learning complexity is kept low while good generalisation can be achieved on various dynamic problems. Fig. 4 shows a typical ESN[2]. The hidden layer consists of N randomly connected neurons. If the connectivity is low, this layer provides independent output trajectories. For this reason, the hidden layer is also called a "reservoir". Furthermore, there are neurons which are connected to cycles in the reservoir, so that past states "echo" in the reservoir. The neurons within the hidden layer are also randomly connected to the k-dimensional input signal, which drives the network.

The essential condition for the successful application of the ESN is the "echo state" property of their state space. If this condition is met, *read-out* neurons can be trained, which take a linear combination of the input and the hidden layer [13]. In other words, only network output weight adaptation is sufficient to train the network. Consequently an ESN can be seen as a universal dynamical system approximator, which combines the elementary dynamics contained in the reservoir. For a large and rich reservoir of dynamics hundreds of hidden units are required.

Let $\mathbf{u}_t = (u_{1,t}, \ldots, u_{K,t})$ be the input to the network at time t. Activations of hidden units are $\mathbf{x}_t = (x_{1,t}, \ldots, x_{N,t})$, and of output units are $\mathbf{y}_t =$

[2] Figure adapted from `http://www.scholarpedia.org/article/Echo_state_network`, 2010/06/05 with the author's permission.

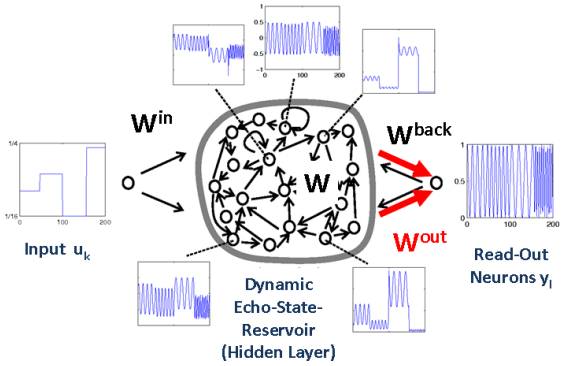


Fig. 4. Schematic view of an Echo State Network. Black arrows indicate weights chosen randomly and fixed;red arrows represent the weights to be optimised.

$(y_{1,t}, \ldots, y_{L,t})$. Further, let $\mathbf{W}^{in}_{N\times K}$ be the weights for the input-hidden connection, $\mathbf{W}_{N\times N}$ be the weights for the hidden-hidden connections, $\mathbf{W}^{back}_{N\times L}$ be the weights for the output-hidden connection, and $\mathbf{W}^{out}_{L\times(K+N+L)}$ be the weights for the read-out neurons, i.e. the connection from all units to the respective read-out neurons. The activation of internal and output units are updated at every time step by:

$$\mathbf{x}_t = f(\mathbf{W}^{in}\mathbf{u}_t + \mathbf{W}\mathbf{x}_{t-1} + \mathbf{W}^{back}\mathbf{y}_{t-1}), \tag{2}$$

where $f = (f_1, \ldots, f_N)$ are the hidden unit's activation functions. The outputs are calculated as:

$$\mathbf{y}_t = f^{out}(\mathbf{W}^{out}[\mathbf{u}_t, \mathbf{x}_t, \mathbf{y}_{t-1}]), \tag{3}$$

where $f^{out} = (f^{out}_1, \ldots, f^{out}_L)$ are the output unit's activation functions. The term $[\mathbf{u}_t, \mathbf{x}_t, \mathbf{y}_{t-1}]$ is the concatenation of the input, hidden and previous output activation vectors.

Training. Assume a given training sequence $\mathcal{T} = \{(\mathbf{u}^{teach}_0, \mathbf{y}^{teach}_0), \ldots, (\mathbf{u}^{teach}_T, \mathbf{y}^{teach}_T)\}$ with known input-output relations. When the network is updated according to Eq. (2), then under certain conditions the network state becomes asymptotically independent of initial conditions. So, the network will asymptotically depend only on the input history, and the network is said to be an echo state network. A condition for the echo state property is to ensure that the maximum eigenvalue (the spectral radius) $|\lambda_{max}|$ of $\mathbf{W}$ is less than 1 [7].

The hidden-hidden weights $\mathbf{W}$, the input-hidden weights $\mathbf{W}^{in}$ and the output-hidden weights $\mathbf{W}^{back}$ are set randomly (typically in a sparse random connectivity pattern) and kept fixed during training. Only the output weights $\mathbf{W}^{out}$ are learned so that the teaching output is approximated by the network. In the ESN approach this task is formulated as minimising the training error:

$$\epsilon_{train}(t) = (f^{out})^{-1}\mathbf{y}^{teach}_t - \mathbf{W}^{out}(\mathbf{u}^{teach}_t, \mathbf{x}_t) \tag{4}$$

in the mean square sense. For example, when considering the read-out as linear combination (i.e., $f^{out}(x) = (f^{out}(x))^{-1} = x$) this can be done quite easily by standard methods of linear regression.

Testing. For usage on test sequences, the trained network can be driven by new input sequences and the output is computed as:

$$\hat{\mathbf{y}}_t = \mathbf{W}^{out}\mathbf{x}_t. \tag{5}$$

3 Workflow monitoring

In the ideal world the surveillance system would monitor workflows and detect abnormalities automatically by observing the scene. However, this is a very challenging goal. Therefore we make use of the specifics of workflows in manufacturing environments. In an industrial environment, a lot of time is spent on setting up machinery, designing work cycles and testing the smooth operation of designed workflows. We use this time and the accumulated knowledge to build the mode of normality which is then used during run-time. Hence, our proposed approach consists of two phases: (i) an off-line setup/maintenance phase and (ii) an on-line run-time phase,which are described in the following.

3.1 Setup/Maintaince

Given a labelled training set $\{(\mathbf{u}_t, \mathbf{y}_t)\}_{t=1,\ldots,T}$ of input/output pairs we train an ESN. One read-out neuron y_l is trained for each individual task $l = 1, \ldots, L$, according to the annotation for each frame. Additionally, one read-out neuron y_{L+1} is trained where no specific task definition is given to capture the remaining "class".

Prior knowledge. Unfortunately in real-life applications it is not always possible to set up cameras which are focusing only on the relevant areas. As a result, our scene descriptor will detect motion both associated with the workflow and usual behaviour not associated with it. An extra motion in the feature space can result in an increase of intraclass variations for some tasks and potentially in reduction of the correct classification rates. One might use feature selection methods to remove the redundancy and focus on the relevant parts of the images. However, when the training set is relatively small (and the intra-class variance compared to it is quite high) it is hard to establish the robust statistics required.

As an alternative, we suggest to use prior information to identify regions in the frame where the actions related to the workflow can potentially take place. Note, the user is only needed once during the setup of the monitoring process. As soon as no significant changes are happening to the workflow for a given camera view no further interaction is necessary. That is, we mask out local motion monitors, which carry no significant information for the workflow description, i.e. are not within a relevant region R_{rel}. The remaining motion monitors form

Algorithm 1: Setup/Maintaince (Off-line Training)

Data: Given labelled training image sequence $(\mathbf{x}_t^{teach}, \mathbf{y}_t^{teach})$
Result: Trained ESN, $(\mathbf{W}^{in}, \mathbf{W}, \mathbf{W}^{back}, \mathbf{W}^{out})$

1. Generate randomly the matrices $(\mathbf{W}^{in}, \mathbf{W}, \mathbf{W}^{back})$
2. Scale $\mathbf{W}$ such that its maximum eigenvalue $|\lambda_{max}| \leq 1$
3. Apply the motion grid to obtain the scene descriptor $\mathbf{u}_t^{teach}$ for image $\mathbf{x}_t^{teach}$.
4. Run this ESN by driving it with the teaching input signal $(\mathbf{u}_t^{teach}, \mathbf{x}_t^{teach})$
5. Dismiss data from the initial transient. Collect remaining input and network states row-wise into a matrix $\mathbf{M}$. Simultanously, collect the remaining training pre-signals $(f^{out})^{-1}\mathbf{y}_t^{teach}$ into a teacher collection matrix $\mathbf{R}$.
6. Calculate the output weights $\mathbf{W}^{out} = (\mathbf{M}^{+}\mathbf{R})^T$, where $\mathbf{M}^{+}$ is the pseudo inverse of M.

Algorithm 2: Run-Time (On-line Testing)

Data: Trained ESN, $(\mathbf{W}^{in}, \mathbf{W}, \mathbf{W}^{back}, \mathbf{W}^{out})$, new input image $\mathbf{x}_t$
Result: Task $\hat{y}(t)$ for $\mathbf{x}(t)$

1. Apply the motion grid to obtain the scene descriptor $\mathbf{u}_t$ for image $\mathbf{x}_t$.
2. Update the ESN states with $\mathbf{u}_t$
3. Calculate the read outs, i.e., $\hat{\mathbf{y}}_t = \mathbf{W}^{out}\mathbf{x}_t$.
4. Assign concrete task $\hat{y}$ using Eq. (7) and post-processing.

then the motion grid matrix and are used as inputs for the ESN

$$\mathbf{u}_t = [u_t^{(1,1)}, \ldots, u_t^{(x,y)}, \ldots, u_t^{(n,m)}], \text{where } u_t^{(x,y)} = \begin{cases} M^{(x,y)} & \text{if } (x,y) \in R_{rel} \\ \text{not used} & \text{otherwise} \end{cases}. \tag{6}$$

3.2 Run-time

During run-time, the scene descriptors are obtained as in Eq. (6) and processed by the ESN. The predicted output produced by the ESN will not be binary in the general case and more than one output can have non-zero values. Furthermore, since the individual read-out neurons are trained independently and usually from highly unbalanced data, we propose to normalize the response with respect to their mean responses $\bar{\mathbf{y}}$, calculated on the training data. To identify a task for each time instance the significant maximum is taken by

$$\hat{y}_t = \begin{cases} l^\star & \text{if } \dfrac{\max\limits_{l=1,\ldots,L+1} \mathbf{y}_t(l) - \bar{\mathbf{y}}(l)}{\max\limits_{\substack{l'=1,\ldots,L+1 \\ l^\star \neq l'}} \mathbf{y}_t(l') - \bar{\mathbf{y}}(l')} > \theta_L \\ L+1 \text{ otherwise} \end{cases}, \text{where } l^\star = \arg \max_{l=1,\ldots L+1} \mathbf{y}_t(l) - \bar{\mathbf{y}}(l) \tag{7}$$

In other words, the maximum of the $L+1$ outputs is considered to be significant, if the ratio to the second highest value is above some defined threshold θ_L. This threshold influences the precision of our method.

Post-processing. Since the tasks do not switch within short time intervals we apply median filtering to remove outliers.

Training and testing are summarised in Alg. 1 and Alg. 2, respectively. Please note, no temporal pre-segmentation has to be done for testing. Moreover, testing is performed on-line and is very efficient, which allows for real-time processing.

4 Experimental results

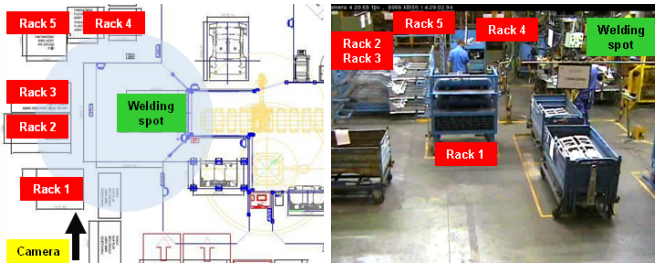
While our approach is quite general, we focus on an concrete example within a car assembly environment. During each day the same workflow is performed many times. The purpose of the surveillance in this application is to monitor the smooth running of assigned operations.

4.1 Car assembly example setup

We recorded approximately 8 hours of video from a single working cell (including gaps between workflows and breaks) due to environment sensitivity. Additonally, restrictions on the possible areas of camera installation were also imposed. A dataset was captured by a PTZ camera at a workcell inside a car manufacturing plant. Omnidirectional cameras cannot be used in this setting due to height restrictions on the camera position. We recorded data at 25 frames per second with relative jitter bounded by 1.6% on frame rate with resolution of 704 by 576 pixels. The camera view for workflow and task recognition and a schematic top-down view of the setting are given in Figure 5.According to the manufacturing requirements each workflow consists of the following 7 tasks, which are usually, however not always, executed sequentially:

Task1: A part from Rack 1 (upper) is placed on the welding spot by a worker(s).
Task2: A part from Rack 2 is placed on the welding spot by worker(s).
Task3: A part from Rack 3 is placed on the welding spot by worker(s).
Task4: Two parts from Rack 4 are placed on the welding spot by worker(s).
Task5: A part from Rack 1 (lower) is placed on the welding spot by worker(s).
Task6: A part from Rack 5 is placed on the welding spot by worker(s).
Task7: Worker(s) grab(s) the welding tools and weld the parts together.
Task8: Any frame, where no actions from Tasks 1-7 take place.

The tasks are strongly overlapped: all tasks (except Task1) will start/finish at the welding machine. It is difficult to identify, even by eye, which task the frame belongs to at the start/end of the task. Moreover, some tasks can have overlapping paths for a number of frames. In the workflow, the duration of the tasks is different. Moreover, the duration of the same task changes from the workflow to the workflow. All of these difficulties and the high level of occlusions in the car assembly environment present challenges to workflow monitoring and task recognition (cf. Fig. 1).



(a) Schematic top-down view (b) Camera view

Fig. 5. Schematic and camera view in the car assembly environment.

4.2 Implementation details

Here we give specific implementation details of our method, which allow reproduction of our experimental results easily. Where the parameters have been specified manually, most of them are not particularly sensitive.

Motion Grid. The initial motion grid matrix was calculated for 140 patches overlaid onto the whole image. The size of patches (local motion region) were selected as $\Omega = 100 \times 100$ with an overlap of 0.5. Activation motion threshold is set to $\theta_M = 250$.

Prior knowledge. The human operative manually specifies the region where the workflows can potentially take place including welding machine. In fact, 65 local motion monitors in the top half of the image were selected. These monitors form the scene descriptor for each frame. Fig. 6 depicts the descriptors over time for three consecutive tasks and three workflows. Red vertical lines indicate the beginning of the new task according to the annotation.

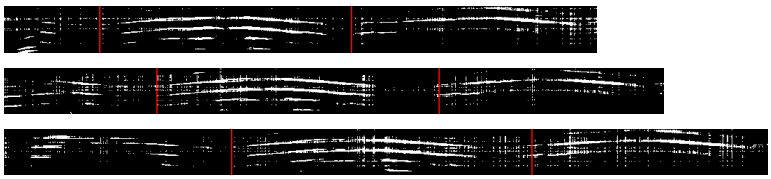


Fig. 6. Examples of scene descriptors over time (each column represents a motion grid feature vector) for the first three tasks of three workflows.

Echo-State-Network. We apply a plain ESN with 12,000 hidden units, 65 inputs and 8 outputs. We used the ESN toolbox written in MATLAB by H. Jaeger and group members[3] and the recommendations given in the on-line

[3] `http://www.reservoir-computing.org/node/129`, 2009/08/05.

tutorial on how to select some ESN's parameters. Furthermore, we investigated dependencies between the ESN parameters and normalised mean-square error. There are many suboptimal sets of parameters which will yield very similar results. Here the spectral radius is $|\lambda| = 0.98$, input scaling and teacher scaling are chosen as 0.1 and 0.3 respectively. Furthermore, additional noise is added to the ESN during the training process to improve the stability.

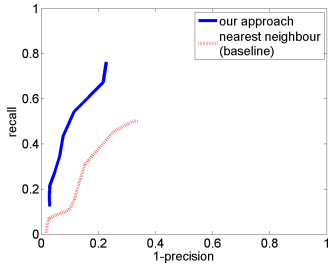
Post-processing. Median filtering with a filter length of 51 is performed.

4.3 Evaluation and results

On the dataset, 19 workflows and their tasks are manually labelled. Each workflow consists of $3,550$ to $7,300$ frames, which correspond to 2 to 5 minutes. Ten of them ($51,520$ frames) were used for training, the remaining nine workflows ($48,950$ frames) were used for testing. Training takes approximately 50 hours using the MATLAB implementation on a 2.83 GHz computer running Windows Vista. However, testing can be done very efficiently online at 20 fps. For a quantitative evaluation, we use recall-precision measurements. The recall corresponds to the correct classification rate, whereas the precision relates to the trust in a classification. The F-measure is the harmonic mean of these two measurements.

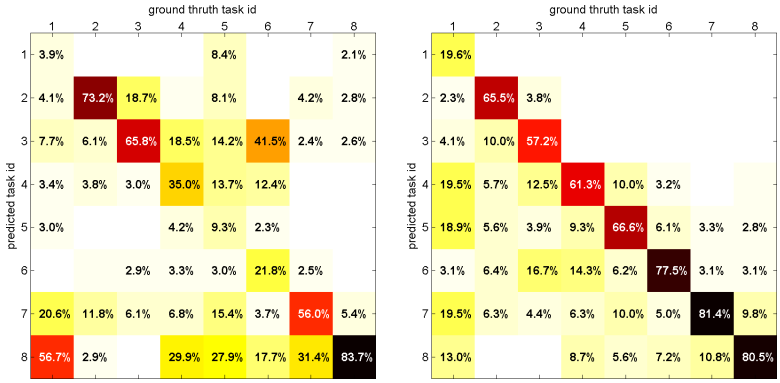
Comparison. At first, we extract features from images using the proposed motion-grid approach. As previously stated, no current state-of-the-art method is able to produce robust results on this kind of data. Hence, for a comparison, we implemented a simple data driven approach inspired by the recent work of [3]. Its principle is to store all training data in respective clusters and detect outliers using the principle of meaningful nearest neighbours from these clusters. Since we have labelled training data, we use the following approach. All training images are stored in a large database and during run-time we look for the best matching one. The label of the frame is returned if it is meaningful. Similar to Eq. (7) of our proposed approach this stabilizes results and allows us to specify the precision.

Results are presented in Fig. 7 and in more detail by a confusion matrix in Fig. 8. Whereas the performance of the frame based nearest neighbour voting already achieved good results, our proposed approach, using the ESN as a complex dynamics time series predictor, gains a further 15% during monitoring of tasks. Additionally, it is significantly faster. Confusion matrices (Fig. 8) indicate that the most difficult task for monitoring is Task 1. Though this task is well separated from manufacturing requirements point of view, it is not that easy to distinguish from other tasks using the video recordings, since it shares the same paths as other tasks for some periods of time. Task 7 is the best recognised task from the workflow when the proposed approach is used. It consists of the smaller proportion of human motion (the majority of the time the humans are stationary) and larger proportion of machinery motion and sparks from the welding process. In many cases the tasks are misclassified either as belonging to Task 7 or Task 8. Misclassifications to Task 7 happen since start and end of all tasks are concentrated around the welding machine. Overall, our approach is able to



	Recall	Prec.	F-m.
Motion-grid + NN (baseline)	54.1 ±14.8%	68.2 ±10.8%	60.1 ±13.2%
Motion-grid + ESN (our approach)	78.6 ±11.5%	77.4 ±10.2%	78.0 ±10.7%

Fig. 7. Performance for all 9 test workflows.



(a) Motion-grid + NN (base-line) (b) Motion-grid + ESN

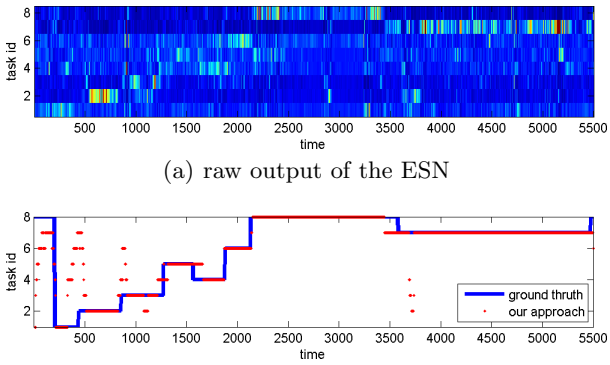
Fig. 8. Confusion matrices for the whole testing set (at maximum f-Measure).

practically achieve above 50% correct classification rate for each of the tasks. Additionally, the raw output of the ESN and the results after post-processing of the our approach with respect to the ground truth is shown in Fig. 9.

Some examples of successful estimated task and typical cases of failures are depicted in Fig. 10. The similarity of the tasks can be demonstrated by the last two images in the fourth row. Task 2 is misclassified as Task 3 (the second image), since the Task 2 is ending in this frame, and our system believes that the Task 3 is starting. Task 1 is misclassified as Task 4 in the third image, since for this frame Task 1 has very similar path to Task 4.

5 Conclusion

In industrial environments automatic workflow monitoring faces several challenges: restrictions on where data is recorded and hostile conditions. Object detection/tracking in such settings is a nontrivial matter and not able to produce robust results. Hence, no robust features based on detection/tracking can be extracted. In this paper we proposed an approach based on local motion monitors as a scene descriptor and Echo State Networks as a time series predictor



(a) raw output of the ESN

(b) final result

Fig. 9. Result of predicted task of our approach for one test workflow sequence.

for workflow monitoring. Experimental results show that monitoring the smooth operation of workflow is achievable. With a simple scene detector we are able to robustly detect both human and machine motion relevant to pre-defined workflows. Using the ESN as a classifier of complex dynamics we achieve 78.6% recall rate and 77.4% precision. Further research will investigate how the proposed approach can cope with more complex scenarios when 2 or more tasks occur in parallel.

Acknowledgement. This work is supported by the EC Framework 7 Program (FP7/2007-2013) under grant agreement number 216465 (ICT project SCOVIS).

References

1. Adam, A., Rivlin, E., Shimshoni, I., Reinitz, D.: Robust real-time unusual event detection using multiple fixed-location monitors. PAMI **30** (2008) 555–560
2. Boiman, O., Irani, M.:Detecting irregularities in images and in video. In Proc. ICCV (2005)
3. Breitenstein, M., Grabner, H., Gool, L.V.: Hunting nessie: Real time abnormality detection from webcams. In Proc. ICCV workshop on Visual Survaillance (2009)
4. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In Proc. CVPR (2008)
5. Grabner, H., Bischof, H.: On-line boosting and vision. In Proc. CVPR **1** (2006) 260–267
6. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Institute for Computer Science (2001)
7. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In Proc. NIPS **15** (2003) 593–600
8. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. In Proc. BMVC (1996)

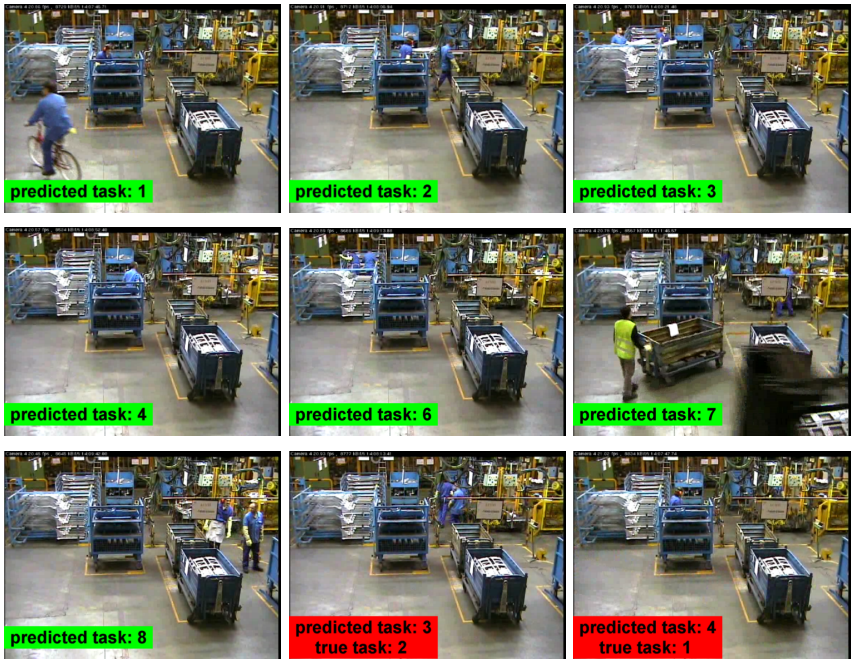


Fig. 10. Typical results of successful and failed task recognition over a single working cycle (figure is best viewed in color; full video avalibale on the authors' web-page).

9. Huang,C., Wu,D., Nevatia,R.: Robust Object Tracking by Hierarchical Association of Detection Responses. In Proc ECCV (2008) 788–801
10. Li, J., Gong, S., Xiang, T.: Scene segmentation for behaviour correlation. In Proc. ECCV (2008)
11. Makris, D., Ellis, T.:Learning semantic scene models from observing activity in visual surveillance. Trans. on Systems, Man, and Cybernetics **35** (2005) 397–408
12. Lv, F., Nevatia, R.: Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. In Proc. ECCV (2006)
13. M.C.Ozturk, D.Xu, J.C.Principe: Analysis and design of echo state networks. Neural Computation **19** (2007) 111–138
14. Padoy, N., Mateus, D., Weinland, D., Berger, M., Navab, N.: Workflow monitoring based on 3d motion features. In: Proc. ICCV/WS on Video-oriented Object and Event Classification (2009)
15. Scherer, S., Oubbati, M., Schwenker, F., Palm, G.: Real-time emotion recognition from speech using echo state networks. Artificial Intelligence (2008) 205–216
16. Skowronski, M., Harris, J.: Automatic speech recognition using a predictive echo state network classifier. Neural Networks **20** (2007) 414–423
17. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In Proc. CVPR **2** (1999) 246–252
18. Wang, X., Ma, K., Ng, G., Grimson, W.: Trajectory analysis and semantic region modeling using a nonparametric bayesian model. In Proc. CVPR (2008)
19. Zhong, H., Shi, J., Visontai, M.: Detecting unusual activity in video. In Proc. CVPR (2004)