A Multi-Task Learning Framework for Head Pose Estimation under Target Motion

Yan Yan, *Member, IEEE*, Elisa Ricci, *Member, IEEE*, Ramanathan Subramanian, *Member, IEEE*, Gaowen Liu, Oswald Lanz, *Member, IEEE*, and Nicu Sebe, *Senior Member, IEEE*

Abstract—Recently, head pose estimation (HPE) from low-resolution surveillance data has gained in importance. However, monocular and multi-view HPE approaches still work poorly under *target motion*, as facial appearance distorts owing to camera perspective and scale changes when a person moves around. To this end, we propose *FEGA-MTL*, a novel framework based on Multi-Task Learning (MTL) for classifying the head pose of a person who moves freely in an environment monitored by multiple, large field-of-view surveillance cameras. Upon partitioning the monitored scene into a dense uniform spatial grid, FEGA-MTL simultaneously clusters grid partitions into regions with similar facial appearance, while learning region-specific head pose classifiers. In the learning phase, guided by two graphs which a-priori model the similarity among (1) grid partitions based on camera geometry and (2) head pose classes, FEGA-MTL derives the optimal scene partitioning and associated pose classifiers. Upon determining the target's position using a person tracker at test time, the corresponding region-specific classifier is invoked for HPE. The FEGA-MTL framework naturally extends to a weakly supervised setting where the target's walking direction is employed as a proxy in lieu of head orientation. Experiments confirm that FEGA-MTL significantly outperforms competing single-task and multi-task learning methods in multi-view settings.

Index Terms—Multi-task learning, graph guided, head pose classification, video surveillance, multi-camera systems

1 Introduction

Motivated by several applications such as video surveillance, human-computer interaction and human behavior analysis, extensive research has been devoted to head pose estimation (HPE) recently [1]. Several approaches precisely compute head pose when the target is close to the camera, as high resolution images enable accurate facial feature extraction and depth information can be also integrated [2], [3]. Nevertheless, despite recent advancements [4], [5], [6], [7], HPE from surveillance videos is challenging as faces are captured at very low resolution and appear blurred.

HPE accuracy on surveillance data can be improved by fusing information from multiple cameras as monocular systems are often insufficient for analyzing human behavior in large environments. Surprisingly, only a few HPE methods consider a multi-view setting [7], [8], [9], [10] and typically compute head pose as a person (target) rotates in-place [8], [9]. However, the ability to estimate head pose of moving targets is key as head orientation is primarily employed as a surrogate for gaze direction to infer social interactions [11]. HPE of moving targets is a challenging problem as illustrated in Fig. 1: facial appearance of a person exhibiting identical 3D head pose at three different scene locations varies considerably due to perspective and scale. As the target moves, the face may appear larger/smaller and some facial regions can become occluded/visible. These appearance changes severely impede HPE performance using traditional approaches [7].

In this paper, we explicitly tackle the problem of *multi-view head pose classification under target motion*. To our knowledge, only [12] (monocular) and [7] (multi-view) have explicitly studied appearance variation under target motion, while [10] is another multi-view approach that can accomplish the same. To tackle motion-induced appearance variations within a scene, [12] employs unsupervised spectral clustering to segment the scene into multiple regions and trains region-wise pose estimators. In [10], multi-view HPE under motion is performed by determining the face location on the unwrapped spherical head texture map. However, the texture synthesis is expensive and uses visual information from nine camera views. Transfer learning for multi-view HPE is proposed in [7], but this approach does not explicitly learn the relationship between head pose, scene location and facial appearance.

Differently, in this paper we present FlExible GrAph-guided Multi-Task Learning or *FEGA-MTL* for multi-view head pose classification under target motion. Given a set of related tasks, Multi-task Learning (MTL) [13] exploits their similarity to jointly learn a set of classifiers. The intuition behind FEGA-MTL is simple: upon dividing the scene ground plane into a uniform grid, one can expect some similarities as well as differences in facial appearance for a given head pose across grid partitions. For learning the

- *Y. Yan, G. Liu, and N. Sebe are with the Department of Information Engineering and Computer Science, University of Trento, Italy. E-mail: {yan, sebe}@disi.unitn.it, gaowen.liu@unitn.it.*
- *E. Ricci is with the Technologies of Vision, Fondazione Bruno Kessler, Trento and the Department of Engineering, University of Perugia, Italy. E-mail: eliricci@fbk.eu.*
- *R. Subramanian is with the Advanced Digital Sciences Center (ADSC), Singapore. E-mail: Subramanian.R@adsc.com.sg.*
- *O. Lanz is with the Technologies of Vision, Fondazione Bruno Kessler, Trento, Italy. E-mail: lanz@fbk.eu.*

Manuscript received 5 Nov. 2014; revised 17 Aug. 2015; accepted 1 Sept. 2015. Date of publication 9 Sept. 2015; date of current version 12 May 2016. Recommended for acceptance by D. Xu.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2477843

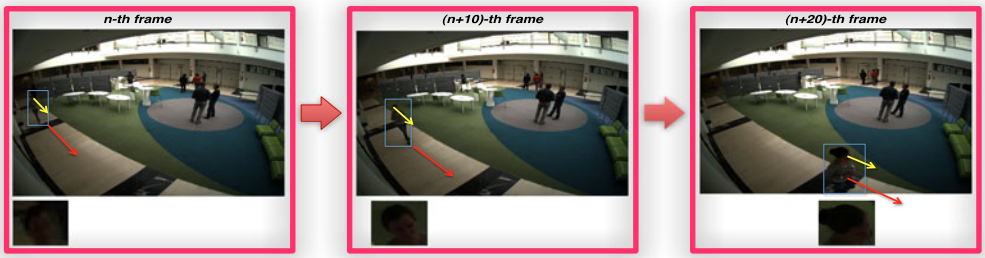


Fig. 1. *HPE under target motion.* Face crops corresponding to three different positions of a target exhibiting the same 3D head pose are shown in the bottom inset. Yellow and red arrows respectively denote head pose and motion direction. Significant changes in facial appearance can be observed as the target moves closer to the camera. These appearance differences severely impede performance of traditional head pose estimation (or classification) methods. Figure is best viewed in color and under zoom.

pose-appearance relationship within each grid segment as a task, we invoke MTL for learning a set of region-specific head pose classifiers (related tasks). Using MTL over traditional approaches for HPE under motion is advantageous as: (1) Employing a single classifier for the entire scene is inefficient as perspective and scale-based face appearance variations would impede performance, and (2) Learning an independent classifier for each grid segment is expensive and will require a large number of training samples. Instead, only few examples from each grid segment are required by FEGA-MTL, which *simultaneously* learns the pose-appearance relationship across all partitions of a dense uniform 2D spatial grid.

Also, assuming that facial appearances among all partitions are related may negatively impact head pose classification performance. Therefore, FEGA-MTL flexibly discovers appearance-wise related grid clusters learning from both *a-priori* knowledge and facial features extracted from training examples. Two graphs which respectively define appearance similarity among (i) grid partitions for a given head pose based on camera geometry, and (ii) head pose classes, model prior knowledge and guide the algorithm to output the optimal spatial partitioning and an associated set of classifiers. For head pose classification, upon determining the target position using a person tracker, the corresponding region-specific classifier is invoked. Thanks to the use of a sparse regularizer, heterogeneous descriptors with varying discriminative power can be effectively utilized for learning. We also extend the FEGA-MTL framework to employ walking direction as a weak label and eliminate the need for annotated data in line with prior works [12], [14]. Since motion direction is a noisy cue, we propose a novel strategy to discard spurious annotations and only retain those samples with consistent head and body motion for model training.

While both FEGA-MTL and the method in [12] train multiple region-specific classifiers, the two can be contrasted as follows: unsupervised spectral clustering is employed on monocular video in [12] to segment the scene into appearance-wise similar regions for HPE, and the consequent limitation is that sufficient examples are required from each of the scene regions to achieve good accuracy. For example, high HPE errors are observed when more than five region-specific classifiers are trained with 1,000-8,000 examples in [12]. Instead, our FEGA-MTL framework exploits multi-camera geometry to *a-priori* estimate appearance distortion as the target moves from one grid segment to another, and learns with few examples. A multi-camera setup also enables precise target tracking and face cropping therefrom. Finally, the use of camera geometry allows for fine-grained scene segmentation (Fig. 3) and learning of relationships among the region-specific classifiers, which is advantageous vis-à-vis learning a set of independent classifiers as discussed in Section 5.

We present extensive evaluation to demonstrate the superiority of FEGA-MTL over competing multi-view HPE and MTL approaches. Overall, this paper makes the following contributions: (1) It is one of the few works addressing multi-view head pose classification under target motion and, to our knowledge, the first work to use MTL to this end; (2) A novel graph-guided MTL is proposed for simultaneously learning a set of region-specific classifiers and the optimal scene partitioning. Our approach seamlessly connects camera geometry (traditional computer vision) with machine learning for HPE; (3) FEGA-MTL can also operate in an unsupervised setting, where head pose labels derived from motion trajectories are used for learning.

The paper is organized as follows. Section 2 reviews related work. Section 3 introduces our approach, describes pre-processing steps, the training data collection process, and the region and pose graphs that are employed to guide the learning algorithm. Section 4 describes the FEGA-MTL algorithm. Experimental evaluation is presented in Section 5, and conclusions are stated in Section 6.

2 Related Work

We now review related work on a) HPE from low-resolution surveillance data and b) multi-task learning.

2.1 HPE from Low Resolution Data

While HPE from high-resolution images and videos has been studied extensively [1], determining the coarse head orientation (i.e., head pose classification) from surveillance data has been attempted only recently. Pose classification using a Kullback-Leibler distance-based facial descriptor is proposed in [5]. The array-of-covariances (ARCO) descriptor [6] enables reliable HPE in the presence of occlusions, scale and illumination changes. Methods presented in [4], [14] use proxy information (e.g., body pose, motion direction) to estimate head pose with minimal training data. However, these algorithms work on monocular video, which is insufficient for studying human behavior in large spaces.

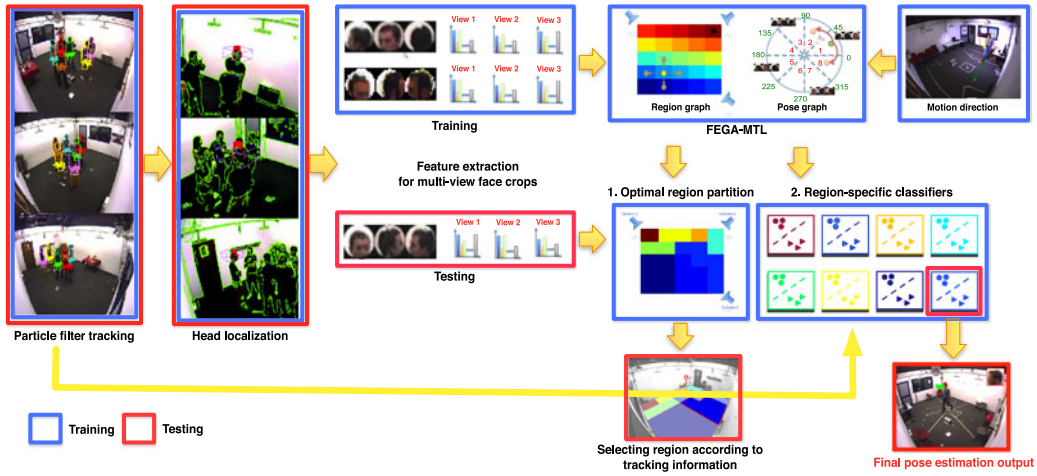


Fig. 2. FEGA-MTL HPE framework overview assuming three camera views. Blue and red blocks correspond to the training and test phases respectively. FEGA-MTL can be trained using annotated ('Training' box on top-row center) or unlabeled images, where motion direction serves as a weak label for head pose (top-row right), enabling its use in both supervised and unsupervised settings. Figure is best viewed in color and under zoom.

HPE from multi-view video has been studied in [7], [8], [9], [10], [15]. A particle filter is combined with two neural networks for independently estimating head pan and tilt in [9]. View-specific probability distributions for pose classification are computed using SVMs in [8], and are fused to obtain a more precise pose estimate. Both these works attempt HPE as a person *rotates in-place*, and motion-induced appearance variations are not considered. Multi-view HPE under motion is addressed in [10] by determining the face location in the unfolded spherical texture map synthesized using nine camera views. A transfer learning approach to compute head pose under motion in a four-view setting is presented in [7]. Weights denoting saliency of face patches for pose classification are first learned from *source* examples corresponding to stationary targets, and adapted using an online learning algorithm to the *target* scenario with moving targets. However, [7] does not seek to explicitly learn the relationship between head pose, target position and facial appearance unlike this work.

Two recent works that have expressly addressed HPE under target motion are [15] and [12]. Scene-adaptive HPE is proposed in [12]. The scene is segmented into multiple regions employing spectral clustering to tackle facial appearance variation with motion, and region-wise head pose classifiers are independently learned. Limitations of this approach are that (i) sufficient examples are required to identify scene segments, and (ii) only a coarse-grained scene segmentation is achievable. In contrast, the FEGA-MTL framework relies on camera geometry and few training examples for scene segmentation, and appearance-wise similar scene regions are modeled via MTL parameters. This paper builds on [15], where explicit learning of facial appearance variations over grid segments for multi-view pose classification under target motion is proposed using FEGA-MTL. This paper extends [15], as a more efficient solver with respect to the one presented in [15] is proposed for the underlying optimization problem of FEGA-MTL and an unsupervised setting is also considered in order to obviate the need for annotated training data (head pose labels are inferred via motion direction and a warping-based filtering technique is employed to extract sequences with consistent head and body motion).

2.2 Multi-Task Learning

Multi-task learning has recently been employed in image classification [16], visual tracking [17], multi-view action recognition [18] and egocentric daily activity recognition [19]. Given a set of related tasks, MTL [13] seeks to simultaneously learn a set of task-specific classification or regression models. The intuition behind MTL is that a joint learning procedure accounting for task relationships is more efficient than learning each task separately. Traditional MTL methods [20], [21] assume that all the tasks are related and their dependencies can be modeled by a set of latent variables. However, in many real world applications such as HPE under target motion, not all tasks are related, and enforcing erroneous (or non-existent) dependencies may lead to *negative* knowledge transfer.

Recently, sophisticated methods have been introduced to counter this problem. These methods assume *a-priori* knowledge (e.g., a graph) defining task dependencies [22], or learn task relationships in conjunction with task-specific parameters [23], [24], [25], [26], [27]. Among these, our work is most similar to [22] and our algorithm adopts two graphs (one defining appearance similarity among grid segments, and the other relating head pose classes) to specify task dependencies. FEGA-MTL further improves over [22] by automatically discovering task relationships to iteratively refine the initial graph structure. For multi-view HPE under motion, the graph structure is very useful as it defines inter-region facial appearance similarity based on camera geometry. The FEGA-MTL framework is described below.

3 Multi-View Head Pose Estimation

3.1 System Overview

The proposed approach outlined in Fig. 2 comprises two main steps: a training phase where multiple region-specific classifiers are learned with FEGA-MTL and a test phase, where head pose classification is performed on novel instances. Our approach relies on a multi-view particle filter tracker [28] for target position estimation and head localization. The output of the tracker is used both in the training and test phases.

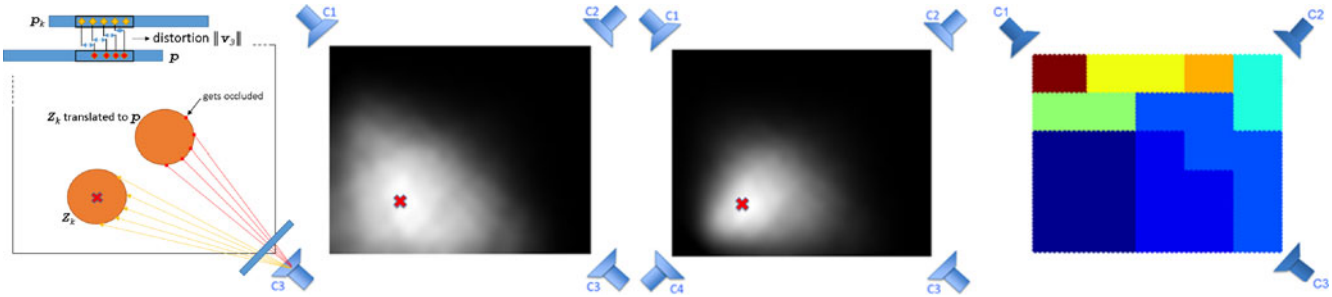


Fig. 3. (From left to right) Method to predict appearance distortion induced due to translation of the head sphere Z_k from p_k to p (exemplified with respect to camera C3), appearance similarity map computed around p_k with $U = 3$ and $U = 4$ camera views, and learned grid clusters for the three-camera setup (figure best viewed in color).

In the pre-processing step, multi-view face crops are extracted using the tracker, and Histogram of Oriented Gradient (HOG) descriptors [29] are computed from the multi-view face images. These HOG features are provided to FEGA-MTL for learning region-specific classifiers across grid partitions on the scene ground plane. The learning process is guided by two graphs: a *region graph* which quantifies the facial appearance distortion based on camera geometry as the target moves from one grid partition to another, and the *pose graph* modeling the appearance similarity among neighboring head pose classes. In this work, we are mainly interested in determining the *head pan* for detecting face-to-face interactions and seek to assign the target's head pan to one of eight classes, each denoting a quantized 45 degree range. To eliminate the need for training data, FEGA-MTL is also designed to operate in an unsupervised manner, i.e., by employing the motion direction of targets as weak labels to signify their head orientation. Post training, FEGA-MTL outputs (1) pose classifiers for each grid partition, and (2) the optimal scene partitioning, where grid regions with similar facial appearance for a given head pose constituting a cluster. During classification, the tracker provides target position based on which the appropriate region-based pose classifier is invoked to output the head pan class. We now describe each of these modules.

3.2 Tracking, Head Localization and Feature Extraction

A multi-view, color-based particle filter [28] is used to compute the 3D body centroid of moving targets. A $30 \times 30 \times 20$ cm-sized dense 3D grid (with 1 cm resolution) of hypothetic head locations is then placed around the estimated 3D head-position provided by the particle filter.[1] Assuming a spherical model of the head, a contour likelihood is computed for each grid point by projecting a 3D sphere onto each view using camera calibration information. The grid point with the highest likelihood sum is determined as the head location. The tracking and head localization procedures are illustrated in Fig. 2.

The head is then cropped and resized to 20×20 pixels in each view. Head crops from the different views are concatenated to generate the multi-view face crops as shown in Fig. 2 and similar to previous works [4], [14] we employ HOG descriptors to effectively describe the face appearance for head pose classification. The multi-view face appearance image is divided into non-overlapping 4×4 patches, and a nine-bin histogram is used as the HOG descriptor for each image patch.

3.3 Region and Pose Graph Modeling

To apply FEGA-MTL, we initially divide the scene ground plane into a uniform 5×5 grid[2] as shown in Fig. 3. We seek to learn the pose-appearance relationship in each partition. The algorithm learns from a training set $\mathcal{T}_t = \{(\mathbf{x}_i^t, y_i^t) : i = 1, 2, \ldots, N_t\}$ for each region $t = 1, 2, \ldots, R$, where $\mathbf{x}_i^t \in I\!R^D$ denote D-dimensional feature vectors and $y_i^t \in \{1, 2, \ldots, C\}$ are the head pose labels ($C = 8$ classes in our setting). One of the graphs guiding the learning process specifies the similarity in appearance for a given head pose across the grid regions based on camera geometry. If grid partitions form the graph nodes, we determine the edge set $\mathcal{E}_1$ and the associated edge weights γ_{mn} quantifying the appearance distortion between $\mathcal{T}_m$ and $\mathcal{T}_n$ due to positional change from region m to region n (these edge weights indicate whether knowledge sharing between regions m and n is beneficial or not).

As mentioned earlier, we model the target's head as a sphere. Let Z_k denote the sphere placed at the target's 3D head position p_k, and whose multi-view camera projection yields training image I_k in $\mathcal{T}_m$. Using camera calibration parameters, one can compute the correspondence between surface points in Z_k and pixels in I_k. Then, we move Z_k to position p_l corresponding to image I_l in $\mathcal{T}_n$, and determine how many surface points in Z_k are still visible in I_l. The appearance distortion over U camera views due to translation from p_k to p_l is defined as $\delta(Z_k, p_k \rightarrow p_l) = \sum_{u=1}^{U} \|v_u\| + \xi n_0$, where v_u is the flow induced by this translation in view u, and n_0 is the number of surface points in Z_k that are occluded after translation. ξ is a constant that penalizes such occlusion. Fig. 3 (left) shows an outline of the method and a comparison of the predicted distortion between three-camera and four-camera setups (discussed below).

The appearance similarity between regions m and n is then computed based on a Gaussian model by considering distortion between all image-pairs associated to $\mathcal{T}_m$, $\mathcal{T}_n$ as:

$$\gamma_{mn} = e^{-\frac{\Omega}{N_m N_n \sigma^2}}, \tag{1}$$

1. The grid size accounts for tracker's variance and horizontal/vertical offsets of the head from the body centroid due to pan, tilt and roll.

2. Upon experimenting with various grid sizes, we note that FEGA-MTL works best with a 5×5 grid as shown in Table 4.

where $\Omega = \sum_{\forall I_k \in \mathcal{T}_m, I_l \in \mathcal{T}_n} [\delta(Z_k, p_k \rightarrow p_l) + \delta(Z_l, p_l \rightarrow p_k)]$, N_m and N_n are number of images in $\mathcal{T}_m$ and $\mathcal{T}_n$. $\sigma = 1$ and $\mathcal{E}_1$ is the set of edges for which $\gamma_{mn} \geq 0.1$.

Fig. 3 depicts the appearance similarity maps for two different camera configurations when the head-sphere at p_k is moved around in space (the projection of p_k on the ground is denoted by the red 'x'). When p_k is close to the cameraless room corner in the three-camera setup, a number of regions around p_k share a high appearance similarity, implying that pose-appearance relationship can be learnt jointly in these regions. However, the similarity measure decreases sharply as the target moves from p_k towards any of the three cameras, and tends to zero for the upper diagonal half of the room. Also, when a camera is introduced in the fourth room corner, appearance similarity holds only for a smaller portion of space around p_k as compared to the three-camera case.

A second graph guiding the learning process models the fact that facial appearances should be more similar for neighboring head pose classes. For example, as shown in Fig. 2 (top-row), the facial appearance of exemplars from class 1 should be most similar to exemplars from classes 2 and 8. Exploiting this information, a pose graph $\mathcal{E}_2$ is defined with associated edge weights $\beta_{ij} = 1$ if i and j correspond to neighboring pose classes c_i, c_j, and $\beta_{ij} = 0$ otherwise.

3.4 Motion Direction as Weak Label

In this section, we describe the process of automatically compiling weakly labeled (instead of annotated) head crops, so that FEGA-MTL can be applied to unsupervised HPE.

Specifically, as obtaining a large repository of annotated data for HPE under target motion is costly, we exploit the fact that people usually tend to look in the direction of their motion (see Fig. 1) to collect a large set of weakly-labeled exemplars without any human intervention. We use walking direction, which can be conveniently extracted from the ground locations output by the tracker, as a proxy for head pose. Most importantly, a novel filtering technique is applied to detect short segments where head appearance is consistent with the observed motion employing this procedure. The filtering process aims to reject samples corresponding to static positions, tracking failures and sudden changes in direction, where the face may appear blurred and the walking direction may not correspond to the head orientation. The result of the filtering process is a set of short image sequences that can be used to learn head pose classifiers customized to a specific multi-view environment and lighting condition.

3.4.1 Extracting Pose Labels Using Trajectory Analysis

To compile weakly annotated training data, we exploit the tracker output both in terms of estimated target position and particle-spread. Given the tracker estimates for each target, we first employ a smoothing spline approximation to interpolate the trajectory. From the position estimates $\{p_x^k, p_y^k\}_{k=1}^M$, we interpolate the two dimensions x and y independently. To compute the interpolating function $f_I(\cdot)$, we adopt Reinsch's algorithm [30]. To filter out noisy samples, we compute the Euclidean distance between tracker estimates and their smoothed counterparts $f_I(\cdot)$, and retain those samples with distance below threshold θ_D. Furthermore, as tracking failures can also contribute to noisy labeling, we monitor the entropy of the target position distribution propagated by the particle filter which, up to a certain extent, indicates the accuracy of the target position estimate. We reject position estimates that result from large localization uncertainty, i.e., where the volume of the typical set approximated from the particle set via kernel density estimation [31] is above a threshold θ_P.

Evidently, using motion direction as a proxy for head pose has some caveats. Even when people walk along a certain path, their attention is often captured by the environment in the form of objects, artifacts, events, or other people in the scene. In such cases, it is unlikely that head orientation can reliably be predicted using walking direction as a proxy. However, attention targets are either static, or likely to move independently of the observer, and so, visual attention direction exhibits different dynamics as compared to the target's (or observer's) walking direction.[3] An effective filtering technique to detect inconsistencies between observed and expected head pose, as given by the walking direction, involves measuring the deviation between the two.

Our filtering technique involves application of the warping detailed in Section 3.3 to recover instances (frames) where head and body motion are consistent. If walking direction is assumed to be an accurate proxy for head pose at the beginning of an analysis window, the warping will produce similar face images over the window only if (i) head and body orientation are consistent and, (ii) head cropping (Section 3.2) is achieved successfully. Thus, we compute a score over each time window denoting the similarity among warped head crops. The similarity at sample i is computed as $S_i = e^{-\sum_{w=i-W}^{i+W} ||\mathbf{x}_w - \mathbf{x}_i||}$, where $W = 10$ is the number of frames in the window around i, and $\mathbf{x}_w$ is the HOG descriptor extracted at frame w. In practice, the scores will be penalized by head occlusion, inconsistent motion-induced pose variations and inaccurate head crops, and thus the filtered frames can be used to produce reliable weakly-labeled data for learning with FEGA-MTL. Samples with similarity score above θ_S are assigned a head pose label $y_i \in \{1, 2, \ldots, C\}$ based on walking direction.

Fig. 4 shows an exemplar tracked trajectory of a target, and demonstrates the appearance consistency-based filtering procedure. This strategy has some advantages over the outlier rejection scheme proposed in [12]. Since our tracker operates on the ground plane instead of the image plane, we do not need to introduce perspective-based scale factors while computing target velocity. Moreover, the warping procedure accounts for perspective and scale-based facial appearance changes under target motion. Finally, as the tracking is based on a multi-target particle filter, tracking failures can be monitored to a certain extent by analyzing the variance in the particle distribution.

4 Flexible Graph-Guided MTL

In this section, we describe the proposed Flexible Graph-guided Multi-task Learning framework in details.

3. An exception is an interacting group of people. However, this situation can be easily detected with multi-target tracking, and by analyzing closeness and similarity of motion trajectories.

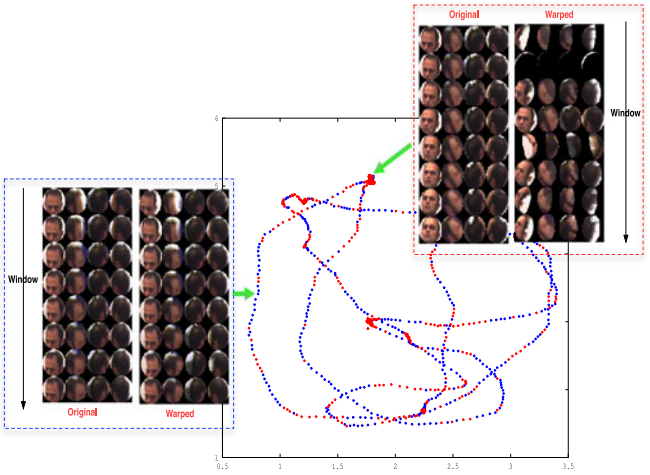


Fig. 4. Exemplar target trajectory from DPOSE [7]: blue dots correspond to samples retained after the filtering process, while red ones are discarded. The two sets of head crops correspond to filtering windows associated with two samples. In the dotted blue rectangle, high similarity among the warped crops imply consistent head and body movements, and thus this sample is used for training. In the red rectangle, warps based on trajectory-based analysis differ considerably, and this sample is rejected (best viewed under zoom).

4.1 Notation and Definitions

In this paper we denote with $\|\cdot\|_F$ and $\|\cdot\|_1$ the Frobenius and the ℓ_1 norms, respectively. The notation $(\cdot)'$ indicates the transpose operator, while $|.|$ denotes a set cardinality. The notation $\mathbf{I}_D$ and $\mathbf{0}_D$ indicate the identity and the null matrix of size $D \times D$, respectively.

Modeling spatial regions as separate tasks, for each task t we define a training set $\mathcal{T}_t$ and a matrix $\mathbf{X}_t \in I\!R^{N_t \times D}$, $\mathbf{X}_t = [\mathbf{x}_1^t, \ldots, \mathbf{x}_{N_t}^t]'$. We also define the matrix $\mathbf{X} \in I\!R^{N \times D}$, $\mathbf{X} = [\mathbf{X}_1', \ldots, \mathbf{X}_R']'$, where $N = \sum_{t=1}^{R} N_t$ denotes the total number of training samples. For each training sample, we construct a binary label indicator vector $\mathbf{y}_i^t \in I\!R^{RC}$ as $\mathbf{y}_i^t = [\underbrace{0,0,\ldots,0}_{Task\ 1}, \underbrace{0,1,\ldots,0}_{Task\ 2}, \ldots, \underbrace{0,0,\ldots,0}_{Task\ R}]$, i.e., the position of the non-zero element indicates the task and class membership of the corresponding training sample. A label matrix $\mathbf{Y} \in I\!R^{N \times RC}$ is then obtained concatenating the $\mathbf{y}_i^{t\prime}$s for all training samples.

4.2 FEGA-MTL

For each region t and pose class c, we consider the weight vectors $\mathbf{s}_{t,c}, \boldsymbol{\theta}_{t,c}, \mathbf{w}_{t,c} \in I\!R^D$ and define the associated matrices $\mathbf{S}, \boldsymbol{\Theta}, \mathbf{W} \in I\!R^{D \times RC}$, $\mathbf{S} = [\underbrace{\mathbf{s}_{1,1}, \ldots, \mathbf{s}_{1,C}}_{Task\ 1}, \ldots, \underbrace{\mathbf{s}_{R,1}, \ldots, \mathbf{s}_{R,C}}_{Task\ R}]$, $\boldsymbol{\Theta} = [\underbrace{\boldsymbol{\theta}_{1,1}, \ldots, \boldsymbol{\theta}_{1,C}}_{Task\ 1}, \ldots, \underbrace{\boldsymbol{\theta}_{R,1}, \ldots, \boldsymbol{\theta}_{R,C}}_{Task\ R}]$ and $\mathbf{W} = \mathbf{S} + \boldsymbol{\Theta}$.

In this paper we present a MTL framework to learn a set of region-specific weight vectors for pose classification $\mathbf{w}_{t,c} \in I\!R^D$, $\mathbf{w}_{t,c} = \mathbf{s}_{t,c} + \boldsymbol{\theta}_{t,c}$. Each weight vector is obtained by summing up two components, $\mathbf{s}_{t,c}$ which models the appearance relationships among regions and $\boldsymbol{\theta}_{t,c}$ accounting for region-specific appearance variations. Using a matrix notation for the sake of clarity, we propose to solve the following optimization problem:

$$\min_{\mathbf{S},\boldsymbol{\Theta}} f(\mathbf{S},\boldsymbol{\Theta}) + r(\mathbf{S},\boldsymbol{\Theta}), \tag{2}$$

where:

$$f(\mathbf{S},\boldsymbol{\Theta}) = \left\|\mathbf{U}^{\frac{1}{2}}(\mathbf{Y} - \mathbf{X}(\mathbf{S}+\boldsymbol{\Theta}))\right\|_F^2$$
$$r(\mathbf{S},\boldsymbol{\Theta}) = \lambda_\theta \|\boldsymbol{\Theta}\|_F^2 + \lambda_s \|\mathbf{S}\|_F^2 + \lambda_s\lambda_1 \sum_{(i,j)\in\mathcal{E}_1} \gamma_{ij} \|\mathbf{s}_{t_i,c} - \mathbf{s}_{t_j,c}\|_1 + \lambda_s\lambda_2 \sum_{(i,j)\in\mathcal{E}_2} \beta_{ij} \|\mathbf{s}_{t,c_i} - \mathbf{s}_{t,c_j}\|_1.$$

In the loss function $f(\cdot)$, the matrix $\mathbf{U} \in I\!R^{N\times N}$, $\mathbf{U} = \mathbf{N}(\mathbf{Y}\mathbf{Y}')^{-1}$ is obtained multiplying two terms. The normalization factor $(\mathbf{Y}\mathbf{Y}')^{-1}$ compensates for different number of samples per task, while the matrix $\mathbf{N} = \mathrm{diag}(\nu_i^t)$ aims to weight differently samples labeled by a human annotator and those automatically obtained by exploiting the information about the walking direction. Specifically we assign a weight $\nu_i^t = 1$ for samples with a true label (i.e., human annotation), while ν_i^t is set to a value $\rho \leq 1$ for weakly labeled data.

The regularization function $r(\cdot)$ is made by several components. The first term penalizes large region-specific appearance variations, the second regulates model complexity, and the ℓ_1 norm terms impose the weights $\mathbf{s}_{t,c}$ of appearance-wise related regions and neighboring classes to be close together. Specifically, γ_{ij}'s and β_{ij}'s are the appearance similarity-based weights of *region* graph edges $\mathcal{E}_1$ and *pose* graph edges $\mathcal{E}_2$ respectively as described in Section 3.3. Similar parameters $\mathbf{s}_{t,c}$ for neighboring head orientations are obtained as λ_2 increases. Region clusters are formed as $\lambda_1 \to \infty$. Importantly, this effect is feature-specific: cluster structure varies from feature to feature. Less important features are used similarly by all tasks, while discriminative features are used differently by different tasks. This is one of the main reasons why our method is termed flexible.

The optimization problem in Eq. (2) is convex. To solve it we propose an algorithm based on smoothing proximal gradient method [22]. The optimization algorithm is outlined in Algorithm 1 and is described in details in the following section.

Algorithm 1. FEGA-MTL

Input: $\mathcal{T}_t, \forall t = 1,\ldots,R$, λ_s, λ_θ, λ_1, λ_2, $\mathbf{E}$, the desired accuracy ϵ.

Initialize $\mathbf{V}_0 = \mathbf{B}_0 = [\mathbf{S}_0;\ \boldsymbol{\Theta}_0]$, $\alpha_0 = 0$.
Set $\mu = \frac{\epsilon}{|\mathcal{E}_1|+|\mathcal{E}_2|}$.
for $n = 1, 2, \ldots,$ until convergence **do**
  Compute the gradient $\nabla H(\mathbf{V}_n)$ using Eq. (12).
  $\mathbf{B}_{n+1} = \mathbf{V}_n - \frac{1}{L_n}\nabla H(\mathbf{V}_n)$
  $\alpha_n = \frac{1}{2}(1 + \sqrt{1 + 4\alpha_{n-1}^2})$
  $\gamma_n = \frac{1-\alpha_n}{\alpha_{n+1}}$
  $\mathbf{V}_{n+1} = \gamma_n \mathbf{B}_n + (1-\gamma_n)\mathbf{B}_{n+1}$
Output: The optimal $\mathbf{V} = [\mathbf{S};\ \boldsymbol{\Theta}]$.

After the training phase, the computed weights $\mathbf{w}_{t,c}$ are used for classification. While testing, upon determining the region $\bar{t}$ associated to a test sample $\mathbf{x}_{test}$ using the person tracker, the corresponding weights vectors are used to compute the head pose label, i.e.,:

$$y_{test} = \arg\max_{c=1,\ldots,C} \mathbf{w}_{\bar{t},c}' \mathbf{x}_{test}. \tag{3}$$

4.3 Optimization

In our previous work [15], as the optimization problem in Eq. (2) belongs to the category of convex smooth/non-smooth problems, we proposed to solve it adopting an approach based on the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [32]. In particular, to handle the non-smooth part, we developed a method based on the Alternating Direction Method of Multipliers (ADMM) [33]. However, the ADMM involves solving a linear system at each iteration and may not scale well for high dimensional problems. In this paper, to solve Eq. (2) a more efficient approach based on smoothing proximal approximation [22] can be employed. The proposed approach is described below and the overall procedure is outlined in Algorithm 1.

Defining $\mathbf{V} = [\mathbf{S};\ \Theta]$ and $\tilde{\mathbf{X}} = [\mathbf{X}\ \mathbf{X}]$, the proposed optimization problem in Eq. (2) can be rewritten as follows:

$$\min_{\mathbf{V}} \left\|\mathbf{U}^{\frac{1}{2}}(\mathbf{Y} - \tilde{\mathbf{X}}\mathbf{V})\right\|_F^2 + \lambda_s\|\Lambda\mathbf{V}\|_F^2 + \lambda_1\|\mathbf{E}\mathbf{V}'\Gamma\|_1, \tag{4}$$

where the matrices $\Gamma, \Lambda \in \mathbb{R}^{2D\times 2D}$ are defined as $\Gamma = \text{blkdiag}(\mathbf{I}_D, \mathbf{0}_D)$ and $\Lambda = \text{blkdiag}(\mathbf{I}_D, \sqrt{\frac{\lambda_0}{\lambda_s}}\mathbf{I}_D)$. The matrix:

$$\mathbf{E} = \begin{bmatrix} \mathbf{E_1} \\ \frac{\lambda_2}{\lambda_1}\mathbf{E_2} \end{bmatrix}$$

is defined in terms of the edge-vertex incident matrices $\mathbf{E}_1 \in \mathbb{R}^{|\mathcal{E}_1|\times RC}$, $\mathbf{E}_2 \in \mathbb{R}^{|\mathcal{E}_2|\times RC}$,

$$\mathbf{E}_{1\ e=(i,j),h} = \begin{cases} \gamma_{ij} & i = h \\ -\gamma_{ij} & j = h \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and:

$$\mathbf{E}_{2\ e=(i,j),h} = \begin{cases} \beta_{ij}, & i = h \\ -\beta_{ij}, & j = h \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

To solve Eq. (4), following [22], [34] we first consider a smooth approximation of the non-smooth term in Eq. (4):

$$\Omega_{\mathcal{E}}(\mathbf{V}) = \|\mathbf{E}\mathbf{V}'\Gamma\|_1 \tag{7}$$

in two steps. First, $\Omega_{\mathcal{E}}(\mathbf{V})$ is reformulated into a linear transformation of $\mathbf{V}$ via the dual norm. Specifically, for each vector $\tilde{\mathbf{v}}^d$, where $\tilde{\mathbf{v}}^d$ is the dth column of $\tilde{\mathbf{V}} = \mathbf{V}'\Gamma$, $d = 1, \ldots, 2D$, we can reformulate $\left\|\mathbf{E}\tilde{\mathbf{v}}^d\right\|_1 = \max_{\|\mathbf{q}_d\|_\infty \le 1}(\mathbf{q}_d)'\mathbf{E}\tilde{\mathbf{v}}^d$, where $\mathbf{q}_d$ is a vector of auxiliary variables. By defining the matrix $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_{2D}]$, $\mathbf{Q} \in \mathcal{Q} = \{\mathbf{Q} : \|\mathbf{q}_d\|_\infty \le 1, \mathbf{q}_d \in \mathbb{R}^{|\mathcal{E}_1|+|\mathcal{E}_2|}, \forall d = 1, \ldots, 2D\}$, the non-smooth term $\Omega_{\mathcal{E}}(\mathbf{V})$ can be equivalently reformulated as:

$$\Omega_{\mathcal{E}}(\mathbf{V}) = \max_{\mathbf{Q}\in\mathcal{Q}} \langle \mathbf{E}\mathbf{V}'\Gamma, \mathbf{Q}\rangle, \tag{8}$$

where $\langle \mathbf{A}, \mathbf{B}\rangle = \text{Tr}(\mathbf{A}'\mathbf{B})$ is the inner product of two matrices. Even after the reformulation, Eq. (8) is still a non-smooth function of $\mathbf{V}$ and this makes the optimization challenging. To tackle this problem, following [22] a smoothing approximation function $h_\mu(\cdot)$ is introduced, i.e.,:

$$h_\mu(\mathbf{V}) = \max_{\mathbf{Q}\in\mathcal{Q}} \langle \mathbf{E}\mathbf{V}'\Gamma, \mathbf{Q}\rangle - \frac{1}{2}\mu\|\mathbf{Q}\|_F^2. \tag{9}$$

The optimization problem which must be solved is then:

$$\min_{\mathbf{V}} H(\mathbf{V}) = \left\|\mathbf{U}^{\frac{1}{2}}(\mathbf{Y} - \tilde{\mathbf{X}}\mathbf{V})\right\|_F^2 + \lambda_s\|\Lambda\mathbf{V}\|_F^2 + \lambda_1 h_\mu(\mathbf{V}). \tag{10}$$

Since Eq. (10) is convex and smooth, it can be efficiently solved with standard gradient methods. In [22], it has been shown that for any μ, the gradient of $h_\mu(\mathbf{V})$ can be computed as $h_\mu(\mathbf{V}) = \Gamma(\mathbf{Q}^*)'\mathbf{E}$, where $\mathbf{Q}^*$ is the optimal solution to Eq. (9). Specifically, the optimal solution $\mathbf{Q}^*$ is composed of $\mathbf{q}_d{}^* = S(\frac{\mathbf{E}\tilde{\mathbf{v}}^d}{\mu})$, $\forall d$, where S is a projection operator such that for any vector $\mathbf{x}$, $S(\mathbf{x})$ is defined by applying on each entry of $\mathbf{x}$:

$$S(x) = \begin{cases} x & \text{if } -1 < x < 1, \\ 1 & \text{if } x > 1, \\ -1 & \text{if } x < -1. \end{cases} \tag{11}$$

Then, the gradient of $H(\mathbf{V})$ can be easily computed as:

$$\nabla H(\mathbf{V}) = \tilde{\mathbf{X}}'\mathbf{U}(\tilde{\mathbf{X}}\mathbf{V} - \mathbf{Y}) + \lambda_s\Lambda'\Lambda\mathbf{V} + \lambda_1\Gamma(\mathbf{Q}^*)'\mathbf{E}. \tag{12}$$

As standard gradient schemes have a slow convergence rate, in this paper we follow the method in [22], [32]. The detail of the optimization are described in Algorithm 1.

Computational complexity. In the update of $\mathbf{V}$, the computational cost at each iteration is dominated by the gradient computation. As the product of some matrices can be precomputed and typically $D \ll N$, the time complexity at each iteration is $\mathcal{O}(D^2RC + |\mathcal{E}_1| + |\mathcal{E}_2|)$. With respect to the approach proposed in [15], this method is faster since at each iteration ADMM requires to solve D linear systems with cost $\mathcal{O}((RC)^2)$. This step can be avoided with the novel solver. As demonstrated in [22], the rate of convergence of the proposed algorithm is $\mathcal{O}(\frac{\sqrt{(|\mathcal{E}_1|+|\mathcal{E}_2|)}}{\epsilon})$.

5 Experimental Results

In this section, we first conduct experiments with synthetic data to demonstrate the effectiveness and the flexibility of our MTL algorithm. Then we perform real-world data experiments to show that FEGA-MTL outperforms the state-of-the-art for multi-view head pose classification.

5.1 Synthetic Data Experiments

To demonstrate the generality of FEGA-MTL, we simulate two toy experiments, one for a classification task and the other for regression.

In case of classification, we consider a multi-class problem with three classes and $R = 8$ tasks. The input data $\mathbf{x}_{t_i} \in \mathbb{R}^D$, $D = 10$ are generated from multivariate normal distributions as follows: for each task, the first $D/2$ feature vector components are obtained from $x_{t_i}^d \sim \mathcal{N}(0, 1)$, while for the $d = D/2 + 1, \ldots, D$ components, we group the tasks into three different clusters, namely $t = \{1, 2\}$, $t = \{3, 4\}$ and $t = \{5, 6, 7, 8\}$ and generate $x_t^d \sim \mathcal{N}(\mu, \sigma)$ according to μ, σ values listed in Table 1. This toy data problem is meant to simulate a realistic scenario, where one expects some

TABLE 1
Synthetic Data Generation for Classification

	μ for the three different classes	σ
Task 1	10, 12, 14	7
Task 2	10.1, 12.1, 14.1	7
Task 3	20, 22, 24	20
Task 4	20.1, 22.1, 24.1	20
Task 5	2, 4, 6	10
Task 6	2.1, 4.1, 6.1	10
Task 7	2.2, 4.2, 6,2	10
Task 8	2.3, 4.3, 6.3	10

discriminative features and some non-discriminative ones. The associated graph describing task dependencies is defined, appropriately setting $\gamma_{ij} = \pm 1$ if two tasks are related, and $\gamma_{ij} = 0$ otherwise. The pose graph is not used in these experiments, i.e., $\beta_{ij} = 0\ \forall i, j$. We generate 100 samples for training, 50 for validation and 100 for test.

In case of regression, we consider $R = 9$ tasks. The input data $\mathbf{x}_{t_i} \in R^d$, $D = 20$ are generated from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, while the outputs are obtained with $y_{t_i} \sim \mathbf{w}_t \mathbf{x}_{t_i} + \mathcal{N}(0, 100)$. The weight vectors $\mathbf{w}_t$ for each task are obtained by generating the first $D/2$ components from a common cluster $w_t^d \sim \mathcal{N}(0, 1)$, while grouping the tasks into three different clusters for the last $D/2$ features, i.e., $w_t^d \sim \mathcal{N}(0,1), t = 1, 2, 3$, $w_t^d \sim 10 + \mathcal{N}(0,4), t = 4, 5, 6$, $w_t^d \sim 20 + \mathcal{N}(0,9), t = 7, 8, 9$. We generate 100 samples for training, 50 for validation and 100 for test.

We compare FEGA-MTL with state-of-the-art MTL approaches. For competing methods, the publicly available implementations in the MALSAR (Multi-tAsk Learning via StructurAl Regularization) [35] library are adopted. In particular, we consider the regularized MTL with a single global model (ℓ_{21} MTL) [20], the Flexible Task Clusters (FTCMTL) method [24], the dirty model MTL method (DMTL) [25], the Cluster MTL (CMTL) [26], the Robust MTL method (RMTL) [27]. The validation sets are used to tune the regularization parameters of all the methods. All the regularization parameters vary in the range [0.01, 0.1, 1, 10, 100]. The results are shown in Fig. 5. Fig. 5 (left) shows the classification accuracy while Fig. 5 (middle) depicts the mean square error (MSE) (lower numbers indicate better performance). It is evident from the plots that in this situation, an MTL method assuming all tasks are related does not suffice, since tasks are clustered in groups. Therefore, ℓ_{21} MTL approach has the highest regression MSE and lowest classification accuracy. Moreover, considering methods which assume grouping among tasks, our method performs best. This is probably due to the fact that features are considered independently, thus discarding the contribution of non-discriminative features. Fig. 5 (right) shows the learned **S** matrix in the regression task. Here, we can clearly see a common cluster for the first $D/2$ dimensions and three different clusters for the last $D/2$ dimensions.

5.2 Multi-View Head Pose Classification

We now present head pose classification results and demonstrate the superiority of our method with respect to other multi-view head pose estimation and multi-task learning algorithms.

5.2.1 Datasets

To assess quantitatively the performance of our method, we conduct our experiments on the publicly available DPOSE dataset [7]. DPOSE comprises over 50,000 4-view synchronized images recorded by distant, large field-of-view cameras for 16 moving targets, with associated positional and head pose measurements (target positions are computed using the person tracker [28]). To our knowledge, there are no other databases for benchmarking multi-view head pose classification performance under target motion. We also manually annotated a video sequence of 30 minutes duration capturing six persons involved in an informal social gathering. Denoted as the PARTY sequence (Fig. 8), this dataset is very challenging, as it involves several targets freely moving around in a room and affected by persistent and substantial occlusions.

5.2.2 Experimental Setup

As we consider faces at very low resolution (i.e., 20×20 pixels) and estimating the head pose orientation is very challenging in these conditions, we only focus on classifying the *head-pan* into one of eight classes, each denoting a 45 degree pan range. For each dataset, we consider an initial, uniformly spaced grid with $R = 25$ regions (Fig. 3) and define mutually exclusive training/validation/test sets. For all considered classification methods, the regularization parameters are tuned using the validation set. In particular, we set $\lambda_s = 2$, $\lambda_\theta = 2^2$, $\lambda_1 = 2^2$, $\lambda_2 = 1$, $\rho = 0.25$ for FEGA-MTL in our experiments. To extract short sequences with consistent head and body motion when annotated training

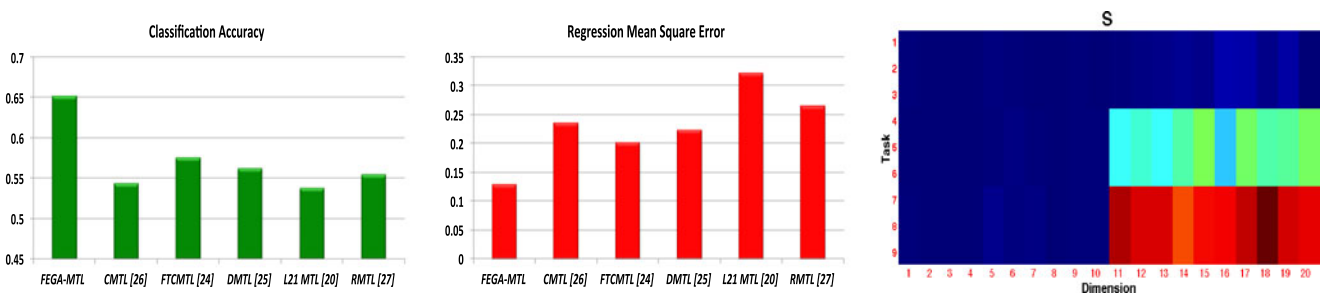


Fig. 5. Synthetic data experiments: (left) Comparison with several MTL methods: classification accuracy. Higher numbers indicate better performance. (middle) Comparison with several MTL methods for the regression problem. Lower numbers indicate better performance. (right) **S** matrix for regression task comprising task clusters.

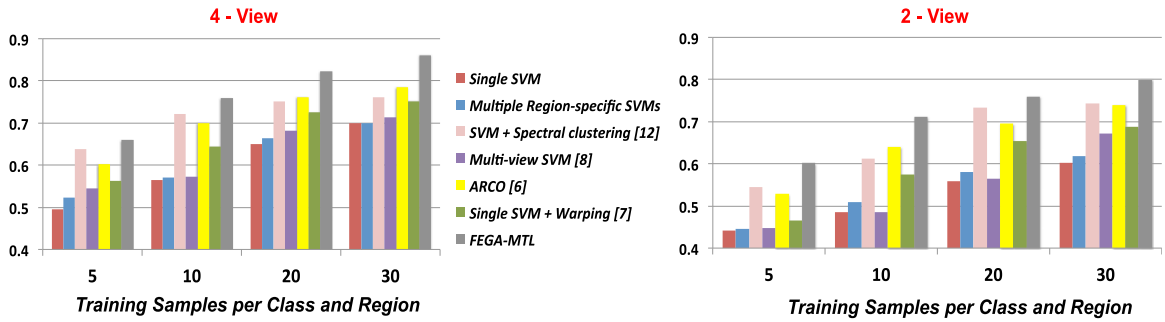


Fig. 6. DPOSE dataset: Comparison with state-of-the-art head pose classification methods.

data are unavailable, thresholds are set to $\theta_P = 0.5$ m^2 using Gaussian kernel with variance 0.15 m for entropy estimation of tracking particle set, $\theta_D = 0.1$ and $\theta_S = 0.001$.

5.2.3 Evaluating HPE Performance

Fig. 6 presents the results obtained comparing FEGA-MTL with competing head pose classification methods. The mean classification accuracies obtained from five independent trials are reported, where a randomly chosen training set is employed in each trial. In this series of experiments, we consider *annotated* training data. We gradually increase the training set size from 5 to 30 samples/class/region, while the test set comprises images from all regions. To underline the usablity of FEGA-MTL with arbitrary camera configuration, we show the results obtained with both four (Fig. 6 left) and two camera views (Fig. 6 right). As expected, all the considered methods perform better when information from four cameras is used.

As baselines, we consider recent multi-view approaches, namely, the warping algorithm in [7] combined with RBF-SVMs for classification (no transfer learning is required in this case), the approach in [8] which probabilistically fuses the output of multiple SVMs, and the monocular ARCO [6] (image features from multiple views are used to extend ARCO to multi-view) and SVM+Spectral Clustering [12] methods. As shown in Fig. 6, both ARCO and the method in [8] perform poorly with respect to FEGA-MTL as they are not designed to account for facial distortions due to scale/perspective changes.

Considering baselines that have explicitly accounted for motion-based facial appearance variations while predicting head pose, the warping method in [7] achieves lower accuracy with respect to FEGA-MTL, despite considerably outperforming Single SVM. Here, it is also important to point out two differences between our approach and [7]. The scene is *a priori* divided into four quadrants in [7], which is not necessarily optimal for describing the pose-appearance relationship under arbitrary camera geometry. Second, task dependencies are ignored in [7], and an independent classifier is used for each quadrant. In contrast, FEGA-MTL discovers the optimal configuration of grid clusters that best describes the pose-appearance relationship given camera geometry. Considering task relationships enables FEGA-MTL to achieve higher classification accuracy than a single global classifier (Single SVM), Single SVM+Warping and separate region-specific classifiers that do not consider inter-region appearance relationships (Multiple Region-specific SVMs).

We have also compared our approach against SVM +Spectral Clustering adopted as a proxy for [12] (a rigorous comparison is not possible as the approach in [12] is monocular). In our implementation of SVM+Spectral Clustering, we use the training images and the spectral clustering algorithm described in [12] to learn a set of spatial regions according to facial appearance similarity. The number of clusters is set to five. Then, five independent SVM classifiers are trained (one for each learned region). As shown in Fig. 6, by learning the optimal region partitioning and the classifiers simultaneously, we achieve higher accuracy than SVM+Spectral Clustering.

5.2.4 Comparison with MTL Approaches

Table 2 compares HPE performance of various MTL methods. Here again, we consider annotated training data. The advantage of employing MTL for head pose classification under target motion is obvious since all MTL approaches greatly outperform a single SVM. However, assuming that all tasks share a common component, i.e., using the ℓ_{21} MTL approach [20] is sub-optimal, and having a flexible learning algorithm which is able to infer appearance relationships among regions improves classification accuracy. This is confirmed by the fact that in all situations (varying training set sizes and number of camera views), FTC MTL [24], Clustered MTL [26] and FEGA-MTL achieve superior performance. FEGA-MTL, which independently considers features and employs graphs to explicitly model region and head pose-based appearance relationships, achieves the best performance. The usefulness of modeling both region and pose-based task dependencies through FEGA-MTL is evident when observing the results in Table 2. Using the region graph alone is beneficial as such, while employing the region and pose graphs in conjunction produces the best classification performance.

When multiple targets move freely in the environment such as a *party* scenario shown in Fig. 8 (bottom), many occlusions usually exist making head pose estimation even harder. Table 3 compares FEGA-MTL with other MTL methods on the PARTY sequence. Even if inferior classification is achieved with respect to the DPOSE dataset given the more challenging nature of the scene, and more training examples per class typically needed to achieve satisfactory performance, the advantages of FEGA-MTL over competing methods can be clearly observed.

To further demonstrate the advantages of FEGA-MTL, we compare it with the other graph-guided MTL methods [22], [35]. Fig. 7 shows that higher accuracy is obtained

TABLE 2
DPOSE Dataset: Comparing Head Pose Classification Accuracy with Competing MTL Methods

	5 training samples/class/region			**10** training samples/class/region		
	2-view	3-view	4-view	2-view	3-view	4-view
Single SVM	0.441 ± 0.011	0.494 ± 0.024	0.523 ± 0.016	0.486 ± 0.012	0.549 ± 0.008	0.564 ± 0.013
ℓ_{21} MTL [20]	0.525 ± 0.010	0.567 ± 0.009	0.589 ± 0.012	0.642 ± 0.012	0.675 ± 0.015	0.696 ± 0.014
Flexible Task Clusters MTL [24]	0.555 ± 0.008	0.598 ± 0.009	0.621 ± 0.007	0.65 ± 0.005	0.681 ± 0.008	0.715 ± 0.006
Dirty model MTL [25]	0.546 ± 0.006	0.585 ± 0.008	0.603 ± 0.011	0.655 ± 0.011	0.686 ± 0.009	0.696 ± 0.008
Clustered MTL [26]	0.540 ± 0.007	0.590 ± 0.007	0.619 ± 0.009	0.639 ± 0.014	0.682 ± 0.011	0.711 ± 0.010
Robust MTL [27]	0.550 ± 0.012	0.580 ± 0.011	0.581 ± 0.009	0.655 ± 0.005	0.689 ± 0.004	0.705 ± 0.008
FEGA-MTL (region graph only, $\lambda_2 = 0$)	0.581 ± 0.002	0.623 ± 0.004	0.643 ± 0.006	0.677 ± 0.006	0.718 ± 0.003	0.733 ± 0.007
FEGA-MTL (pose graph only, $\lambda_1 = 0$)	0.564 ± 0.006	0.605 ± 0.006	0.637 ± 0.007	0.661 ± 0.009	0.699 ± 0.011	0.728 ± 0.005
FEGA-MTL (region graph + pose graph)	$\mathbf{0.602 \pm 0.002}$	$\mathbf{0.643 \pm 0.003}$	$\mathbf{0.660 \pm 0.004}$	$\mathbf{0.711 \pm 0.003}$	$\mathbf{0.748 \pm 0.004}$	$\mathbf{0.759 \pm 0.005}$

TABLE 3
PARTY Dataset: Head Pose Classification Accuracy

	20 training samples/class/region			**30** training samples/class/region		
	2-view	3-view	4-view	2-view	3-view	4-view
Single SVM	0.422 ± 0.021	0.463 ± 0.016	0.498 ± 0.014	0.508 ± 0.015	0.529 ± 0.018	0.541 ± 0.013
ARCO [6]	0.501 ± 0.009	0.513 ± 0.013	0.561 ± 0.012	0.529 ± 0.017	0.554 ± 0.014	0.606 ± 0.015
ℓ_{21} MTL [20]	0.491 ± 0.013	0.525 ± 0.011	0.552 ± 0.009	0.557 ± 0.008	0.573 ± 0.004	0.596 ± 0.011
Flexible Task Clusters MTL [24]	0.526 ± 0.021	0.538 ± 0.004	0.541 ± 0.014	0.578 ± 0.014	0.611 ± 0.009	0.625 ± 0.006
Robust MTL [27]	0.521 ± 0.005	0.532 ± 0.007	0.551 ± 0.008	0.575 ± 0.014	0.599 ± 0.012	0.61 ± 0.011
FEGA-MTL (region graph only, $\lambda_2 = 0$)	0.543 ± 0.006	0.569 ± 0.004	0.571 ± 0.002	0.601 ± 0.004	0.637 ± 0.003	0.652 ± 0.008
FEGA-MTL (pose graph only, $\lambda_1 = 0$)	0.534 ± 0.007	0.553 ± 0.003	0.572 ± 0.004	0.611 ± 0.005	0.629 ± 0.006	0.643 ± 0.006
FEGA-MTL (region graph + pose graph)	$\mathbf{0.575 \pm 0.006}$	$\mathbf{0.592 \pm 0.001}$	$\mathbf{0.606 \pm 0.004}$	$\mathbf{0.631 \pm 0.005}$	$\mathbf{0.663 \pm 0.002}$	$\mathbf{0.681 \pm 0.004}$

Comparison with state-of-the-art approaches.

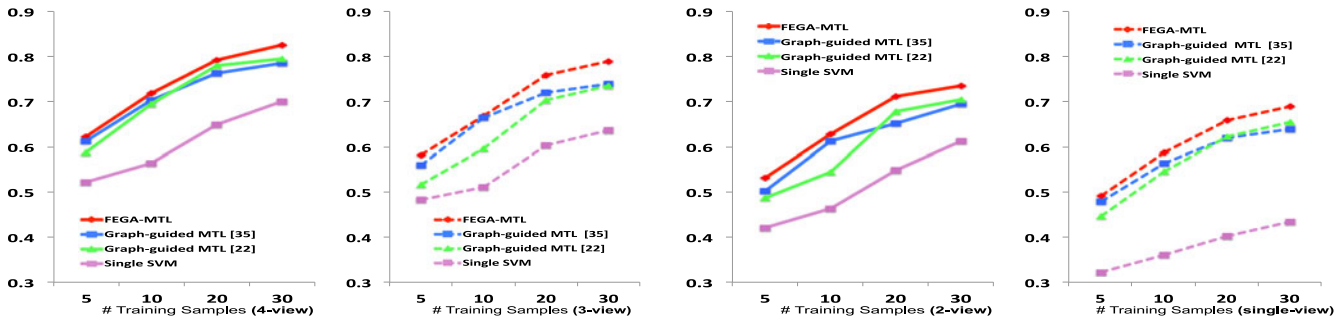


Fig. 7. (Left to right) Classification accuracies with Graph-guided MTL methods using 4, 3, 2 and single-view information.

with our approach for different training set sizes. A main difference between FEGA-MTL and the methods described in [22], [35] is that they do not decompose $\mathbf{W}$ as $\mathbf{S} + \mathbf{\Theta}$, and due to the non-consideration of task-specific components $\mathbf{\Theta}$, they have less flexibility. Moreover, in [35] (due to the use of the ℓ_2 norm) and [22] (due to smoothing) task-clustering is encouraged but not enforced, i.e., the weights corresponding to a cluster are similar but not identical. As discussed above, FEGA-MTL can also be used with an arbitrary number of cameras and even in a monocular setting. However, the use of several views is typically advantageous and improves performance. The performance gain achieved using FEGA-MTL over a single SVM classifier trained for the entire scene is evident, irrespective of the number of camera-views used.

5.2.5 Qualitative Results

Fig. 8 shows some qualitative results obtained with FEGA-MTL on the DPOSE and PARTY sequences. For illustration, identical colors are used to denote the pose direction frustum and face crop rectangle for each target. As discussed above, the party scene is quite challenging as six targets are interacting naturally (resulting in prolonged and substantial occlusions) and freely moving around in the scene. However, as demonstrated by the results in Table 3, FEGA-MTL generally estimates head orientation correctly despite the presence of occlusions and low scene resolution.

Fig. 8 also shows the optimal spatial partitioning learned for a three-camera system with five training images/class/region. The learned grid clusters are also shown in Fig. 3 together with the initial spatial grid. Clustered regions correspond to identical columns of the task similarity matrix $\mathbf{S}$, i.e., two regions t_i and t_j merge if $\mathbf{s}_{t_i,c} = \mathbf{s}_{t_j,c}\ \forall c$. Constrained by the appearance similarity graph weights, spatially adjacent regions tend to cluster together. While regions closer to the camera-less room corner tend to form large clusters, smaller clusters are observed as one moves closer to the

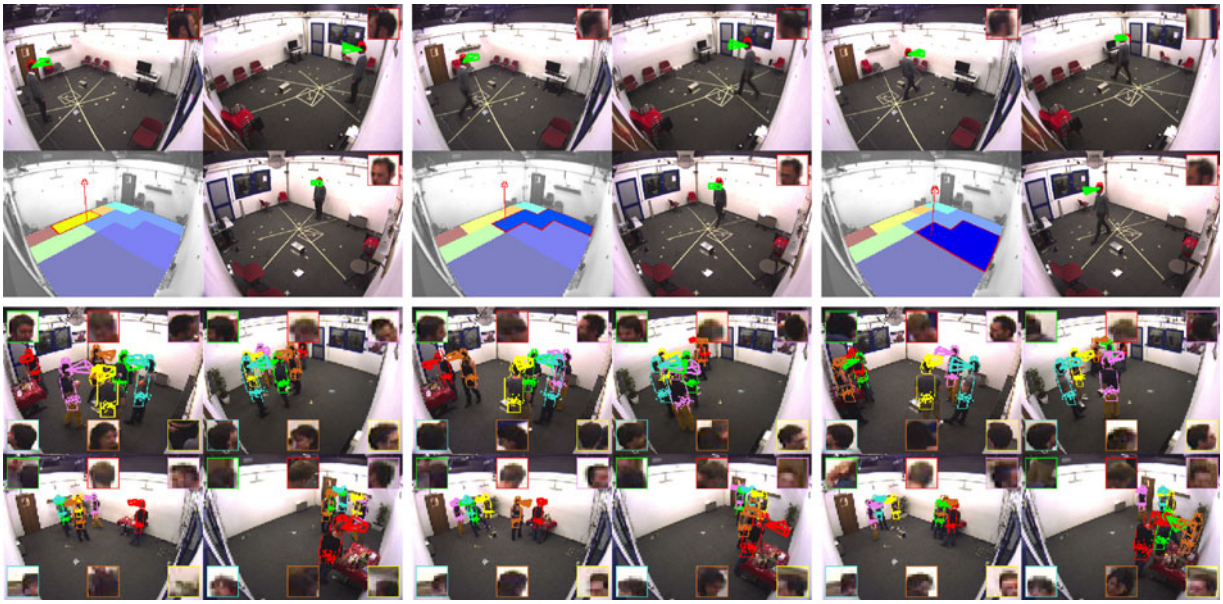


Fig. 8. (Top) Head pose classification results for a target moving freely within a three-camera setup are shown two-by-two. The learned clusters, as seen from a fourth view, are shown on the bottom-left inset. Cluster corresponding to the target position (denoted using a stick model) is highlighted. (Bottom) Head pose classification results for the PARTY sequence involving mobile targets (best viewed under zoom).

cameras owing to larger facial appearance distortions caused by perspective and scale changes. Apart from the region and pose-based appearance similarity graph weights, facial appearance features also influence the clustering of related regions, and therefore, the computed optimal partitioning.

5.2.6 FEGA-MTL Analysis

We now examine the impact of (i) grid size, (ii) considered visual features, (iii) prediction strategy and (iv) the head localization accuracy on FEGA-MTL performance. Finally, we also show the results of the parameter sensitivity study and computational cost analysis.

Table 4 shows the classification accuracy of FEGA-MTL when different grid sizes are considered for partitioning the scene ground plane. For this experiment, we consider four-views and use a fixed set of 250 training samples/class that are uniformly distributed over the scene. From the table, it is evident that the best performance corresponds to a 5×5 grid. Too coarse (higher within-region appearance distortion) or too fine scene partitioning (fewer training samples/class/region) typically hampers HPE performance.

We also evaluate FEGA-MTL performance with different visual features and their combinations. In addition to HOG features [29], we consider three other descriptors: Kullback-Leibler (KL) divergence features [5], Local Binary Pattern (LBP) [36] and skin color features [12]. KL features are computed as described in [5] by indexing each pixel with respect to the mean appearance template of different head pose classes. For LBP, we use 256-dimensional histogram features (16 cells $\times$ 16 bins). For computing skin color features we first detect skin pixels using a Gaussian Mixture Model. Then, we divide the face image into 4×4 cells and count the number of skin pixels in each cell, obtaining a 16-dimensional feature vector.

Fig. 9 shows the head pose classification accuracy obtained with different methods employing various features. Among the different features, HOG and skin color histograms are respectively the most and least effective features. This justifies our choice of HOG in this work. Furthermore, Fig. 9 demonstrates that the performance of FEGA-MTL (and other methods) can be improved by combining different descriptors. For all methods, maximum classification accuracy is obtained with HOG and KL feature combination.

We also examine if a weighted voting strategy for combining classifiers from neighboring regions is beneficial in the test phase. Specifically, we compare three different approaches for prediction, namely, using a single classifier (as discussed in Section 4.2), and employing a combination of classifiers from adjacent scene regions according to a four-neighbor or eight-neighbor connection scheme. When multiple classifiers are employed, the class label is assigned

TABLE 4
DPOSE Dataset: HPE Accuracy with Varying Grid Sizes

Size	3×3	5×5	8×8	15×15
Acc	0.745 ± 0.007	0.759 ± 0.005	0.736 ± 0.006	0.717 ± 0.004

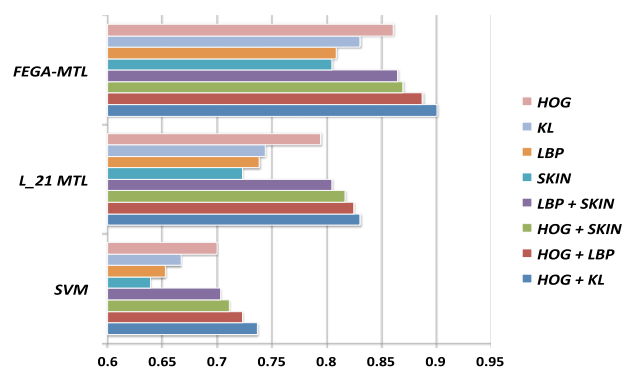


Fig. 9. DPOSE dataset: Head pose classification accuracy obtained with different methods employing different features and features combination.

TABLE 5
DPOSE Dataset: Accuracy with Different Prediction Strategies

	4-view	3-view	2-view
Single Classifier (Eq.3)	0.759 ± 0.005	0.748 ± 0.004	0.711 ± 0.003
Classifier-comb (4-neighbor)	0.772 ± 0.005	0.762 ± 0.008	0.733 ± 0.012
Classifier-comb (8-neighbor)	0.753 ± 0.011	0.752 ± 0.005	0.702 ± 0.006

by computing the mode of the classifier-score distribution. Table 5 shows the empirical results when 10 training samples/class/region are used. By considering classifiers from four nearby regions, the classification accuracy generally improves with respect to the use of a single region-specific classifier. However, little performance improvement is observed when eight neighboring classifiers are considered, probably due to large appearance changes in the area covered by the eight regions.

To study the influence of head localization accuracy on HPE performance (we use a multi-view particle filter tracker as described in Section 3.2), we compare manually annotated head crops with the crops obtained with the proposed automatic procedure adding noise to the estimated head coordinates. To this end, we manually marked the head coordinates of all targets in the PARTY sequence and regenerated head crops upon adding different levels of Poisson noise to the estimated target head locations, which were then input to the FEGA-MTL. Corresponding results (Table 6) clearly indicate the importance of accurate head localization. λ_p indicates the Poisson noise level. Indeed, even perturbing head location estimates by few pixels in x and y (cropsize is 20×20) reduces HPE accuracy. These results also confirm the effectiveness of the proposed head localization method.

We also examine the effect of varying FEGA-MTL regularization parameters (results correspond to one of our experiments on the DPOSE dataset). In Fig. 10 (left), the role of λ_1 and λ_2, i.e., the parameters which regulate the importance of the *region* and *pose* graphs respectively are analyzed. It is interesting to observe that when λ_s and λ_θ are fixed, very small or large values of λ_1 and λ_2 correspond to decreased classification accuracy, thereby evidencing the importance of both graph terms. Fig. 10 (right) presents classification accuracies on varying λ_s and λ_θ when λ_1 and λ_2 are fixed. These parameters balance the importance of the two regularization terms $\|\mathbf{S}\|_F^2$ and $\|\mathbf{\Theta}\|_F^2$ or in other words, regulate the influence of the common and region-specific components of the classifier parameters. As expected, the highest classification performance is achieved when $\lambda_s \sim \lambda_\theta$, i.e., equal importance is given to the shared and task-specific components.

TABLE 6
PARTY Dataset: Head Localization versus Classification Accuracy

Poisson noise	4-view	3-view	2-view
Annotated head location	0.712 ± 0.003	0.691 ± 0.003	0.663 ± 0.005
$\lambda_p = 0.05 \times$ cropsize	0.694 ± 0.004	0.677 ± 0.006	0.641 ± 0.007
$\lambda_p = 0.10 \times$ cropsize	0.652 ± 0.006	0.647 ± 0.007	0.599 ± 0.008
$\lambda_p = 0.15 \times$ cropsize	0.612 ± 0.007	0.599 ± 0.011	0.557 ± 0.013
$\lambda_p = 0.25 \times$ cropsize	0.576 ± 0.013	0.554 ± 0.015	0.516 ± 0.018
Localization via tracking	0.681 ± 0.004	0.663 ± 0.002	0.631 ± 0.005

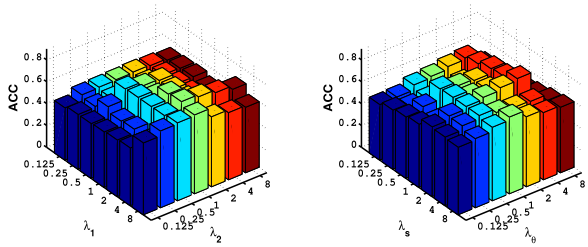


Fig. 10. Sensitivity analysis. Classification accuracy on varying regularization parameters λ_1 and λ_2 when λ_s and λ_θ are fixed (left); λ_s and λ_θ with λ_1 and λ_2 fixed (right).

Finally, we examine the computational efficiency of the training phase of the proposed FEGA-MTL. As discussed in Section 4.3, in this paper to solve Eq. (2) we propose a different approach with respect to the one introduced in [15] that was based on ADMM. Compared with the ADMM solver in [15], the novel approach is more efficient, as no linear systems must be solved. To confirm this fact, in Table 7 we report the computation time required by the two solvers in some of our experiments on the DPOSE dataset (the associate accuracy is reported in Table 2). Specifically, we compute the times associated to the experiments done using five training samples/class/region and fixed regularization parameters chosen with cross-validation. Both the proposed approach and the method in [15] have been implemented in MATLAB and our experiments run on a desktop computer with Intel (R) Xeon (R) CPU E5-2620 0 @ 2.00 GHz processor.

5.2.7 Extending MTL to a Weakly-Supervised Setting

We now evaluate FEGA-MTL performance when head pose labels extracted using motion trajectories (Section 3) are used for learning. We again consider the DPOSE data in this series of experiments, and evaluate FEGA-MTL performance in three different settings: supervised (as in [15]), semi-supervised and unsupervised. For unsupervised learning, we train FEGA-MTL exclusively using 1,000 images (five images/class/region) with head pose labels computed using motion direction. For supervised learning, we train the classifier only using annotated examples (i.e., 5/10 training samples/class/region). We also evaluate FEGA-MTL performance in the semi-supervised case where the training set comprises the above annotated-plus-weakly labeled examples.

Table 8 shows the results of our evaluation (note that the case corresponding to zero annotated samples in the semi-supervised setting exemplifies the unsupervised setting). While classification accuracy achieved with unsupervised learning is expectedly lower than with supervised learning,

TABLE 7
DPOSE Dataset: Computation Time Comparison

Solver	2-views	3-views	4-views
[15] (ADMM based)	107 sec	267 sec	490 sec
This paper	78 sec	185 sec	312 sec

TABLE 8
FEGA-MTL Classification Accuracy Obtained with Different Training Sets

	supervised		semi-supervised						no learning
			with filtering			without appearance filtering			motion direction
n^{o} annotated samples/class/region	5	10	0	5	10	0	5	10	-
4-view	0.66	0.75	0.64	0.74	0.82	0.61	0.73	0.78	0.45
3-view	0.64	0.74	0.61	0.69	0.78	0.61	0.69	0.74	
2-view	0.60	0.71	0.58	0.66	0.76	0.55	0.64	0.70	
1-view	0.49	0.58	0.45	0.59	0.67	0.44	0.59	0.63	

weakly labeled examples nevertheless boost performance when used in conjunction with annotated ones. An improvement of 8.8 and 9.6 percent respectively is obtained by adding motion-based examples with five annotated examples/class/region with four and single-view information. This result implies that FEGA-MTL can be used to effectively estimate head pose in practice with few annotated and sufficient number of automatically labeled examples. Finally, the filtering approach employed for weak labeling is also found to enhance classification performance. Higher accuracies are observed by using only those examples where head and body motion are consistent using the filtering process (using appearance filtering, i.e., the samples where the appearance similarity score exceeds the threshold θ_S), with higher relative improvements observed when a larger proportion of (clean) annotated data is used for training. Overall, the obtained empirical results confirm the efficacy of the FEGA-MTL framework when unlabeled examples are used for training, and the usefulness of the proposed filtering procedure to extract image sequences with consistent head and body motion. As a reference, we also compute the accuracy obtained in estimating the head pose when motion direction is used as a label and no learning and no filtering (no spline smoothing, entropy and appearance filtering) are performed. This corresponds to estimating the level of noise of the weakly annotated samples. As expected performance significantly degrade (note that the last column report just one number since there is no learning involved).

6 Conclusions

The proposed FEGA-MTL framework for estimating the head pose of moving targets is found to outperform a host of monocular/multi-view HPE approaches as well as multi-task learning methods via extensive experiments. FEGA-MTL efficiently leverages on camera geometry information and sparsely annotated training data from different grid partitions to discover scene regions where the head pose-appearance relationship is consistent, and can also be utilized when no labeled training data are available through the use of motion direction as a proxy for head orientation. Since camera geometry is incorporated in the learning process, model training may be scene-specific as discussed in [15]. Nevertheless, this does not limit the applicability of our method as multi-camera installations are easy to calibrate nowadays, and efficient HPE is possible with few labeled examples even on the challenging DPOSE and PARTY datasets. Finally, it worth noting that the FEGA-MTL algorithm is a general framework, potentially applicable to many other computer vision and pattern recognition problems such as action recognition and event detection.

Future works will be devoted to extend the proposed FEGA-MTL to deal with sparse training data and arbitrary camera configurations. Currently, FEGA-MTL cannot be used when the spatial distribution of training data is highly unbalanced across the scene. In this case, typically no grid partitioning with sufficient samples/class/region can be determined to learn robust region classifiers. Moreover, in this work an early fusion approach is adopted to combine features corresponding to multiple cameras, hindering the use of FEGA-MTL in case of cameras with non-overlapping field of view. Addressing these limitations will involve new research towards a distribution-sensitive MTL approach with late fusion scheme for combining multiple views.

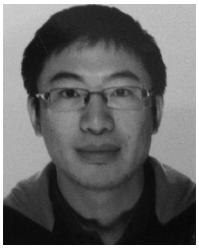
Acknowledgments

This work was partially supported by the MIUR Cluster project Active Ageing at Home, the EC project ACANTO, EIT ICT Labs SSP 12205 Activity TIK—The Interaction Toolkit tasks T1320A-T1321A and A*STAR Singapore under the Human-Centered Cyber-physical Systems (HCCS) grant.

References

[1] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 4, pp. 607–626, Apr. 2009.
[2] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 617–624.
[3] M. Breitenstein, D. Kuettel, T. Weise, L. Van Gool, and H. Pfister, "Real-time face pose estimation from single range images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2008, pp. 1–8.
[4] C. Chen and J.-M. Odobez, "We are not contortionists: Coupled adaptive learning for head and body orientation estimation in surveillance video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 1544–1551.
[5] J. Orozco, S. Gong, and T. Xiang, "Head pose classification in crowded scenes," in *Proc. Brit. Mach. Vis. Conf.*, 2009, pp. 1–11.
[6] D. Tosato, M. Farenzena, M. Cristani, M. Spera, and V. Murino, "Multi-class classification on Riemannian manifolds for video surveillance," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 378–391.
[7] A. K. Rajagopal, R. Subramanian, E. Ricci, R. L. Vieriu, O. Lanz, and N. Sebe, "Exploring transfer learning approaches for head pose classification from multi-view surveillance images," *Int. J. Comput. Vis.*, vol. 109, no. 1-2, pp. 146–167, 2014.
[8] R. Muñoz-Salinas, E. Yeguas-Bolivar, A. Saffiotti, and R. M. Carnicer, "Multi-camera head pose estimation," *Mach. Vis. Appl.*, vol. 23, no. 3, pp. 479–490, 2012.
[9] M. Voit and R. Stiefelhagen, "A system for probabilistic joint 3d head tracking and pose estimation in low-resolution, multi-view environments," in *Proc. 7th Int. Conf. Comput. Vis. Syst.*, 2009, pp. 415–424.
[10] X. Zabulis, T. Sarmis, and A. A. Argyros, "3d head pose estimation from multiple distant views," in *Proc. Brit. Mach. Vis. Assoc. Conf.*, 2009, pp. 1–12.
[11] R. Subramanian, Y. Yan, J. Staiano, O. Lanz, and N. Sebe, "On the relationship between head pose, social attention and personality prediction for unstructured and dynamic group interactions," in *Proc. 15th ACM Int. Conf. Multimodal Interaction*, 2013, pp. 3–10.
[12] I. Chamveha, Y. Sugano, D. Sugimura, T. Siriteerakul, T. Okabe, Y. Sato, and A. Sugimoto, "Head direction estimation from low resolution images with scene adaptation," *Comput. Vis. Image Understanding*, vol. 117, no. 10, pp. 1502–1511, 2013.
[13] R. Caruana, *Multitask Learning*. New York, NY, USA: Springer, 1998.
[14] B. Benfold and I. Reid, "Unsupervised learning of a scene-specific coarse gaze estimator," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 2344–2351.
[15] Y. Yan, E. Ricci, R. Subramanian, O. Lanz, and N. Sebe, "No matter where you are: Flexible graph-guided multi-task learning for multi-view head pose classification under target motion," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1177–1184.

[16] X. Yuan and S. Yan, "Visual classification with multi-task joint sparse representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 3493–3500.
[17] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via structured multi-task sparse learning," *Int. J. Comput. Vis.*, vol. 101, no. 2, pp. 367–383, 2013.
[18] Y. Yan, E. Ricci, R. Subramanian, G. Liu, and N. Sebe, "Multitask linear discriminant analysis for view invariant action recognition," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5599–5611, Dec. 2014.
[19] Y. Yan, E. Ricci, G. Liu, and N. Sebe, "Egocentric daily activity recognition via multitask clustering," *IEEE Trans. Image Process.*, vol. 24, no. 10, pp. 2984–2995, Oct. 2015.
[20] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 41–48.
[21] T. Evgeniou and M. Pontil, "Regularized multi-task learning," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 109–117.
[22] X. Chen, Q. Lin, S. Kim, J. G. Carbonell, and E. Xing, "Smoothing proximal gradient method for general structured sparse regression," *The Ann. Appl. Statist.*, vol. 6, no. 2, pp. 719–752, 2012.
[23] Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 521–528.
[24] L. W. Zhong and J. T. Kwok, "Convex multitask learning with flexible task clusters," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 49–56.
[25] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan, "A dirty model for multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 964–972.
[26] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 702–710.
[27] P. Gong, J. Ye, and C. Zhang, "Robust multi-task feature learning," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 895–903.
[28] O. Lanz, "Approximate Bayesian multibody tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1436–1449, Sep. 2006.
[29] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2005.
[30] C. H. Reinsch, "Smoothing by spline functions," *Numerische Mathematik*, vol. 10, no. 3, pp. 177–183, 1967.
[31] O. Lanz, "An information theoretic rule for sample size adaptation in particle filtering," in *Proc. 14th Int. Conf. Image Anal. Process.*, 2007, pp. 317–322.
[32] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.
[33] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[34] B. Zhao, F. Li, and E. P. Xing, "Large-scale category structure aware image categorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1251–1259.
[35] J. Zhou, J. Chen, and J. Ye. (2011). *MALSAR: Multi-tAsk Learning via StructurAl Regularization*, Arizona State University. [Online]. Available: http://www.public.asu.edu/ jye02/Software/MALSAR
[36] X. Wang, T. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 32–39.



Yan Yan received the PhD degree in computer science from the University of Trento, Italy, in 2014. He is currently a research fellow with the MHUG group at the University of Trento, Italy. He was a visiting scholar with Carnegie Mellon University in 2013 and a visiting research fellow with the Advanced Digital Sciences Center (ADSC), UIUC, Singapore in 2015. His research interests include computer vision, machine learning, and multimedia. He received the Best Student Paper Award in ICPR 2014 and Best paper candidate in ACM Multimedia 2015. He has been PC members for several major conferences and reviewers for referred journals in computer vision and multimedia area. He will serve as a guest editor in *IEEE Transactions on Pattern Analysis and Machine Intelligence*. He is a member of the IEEE.

Elisa Ricci received the PhD degree from the University of Perugia in 2008. She is an assistant professor at the University of Perugia and a researcher at Fondazione Bruno Kessler. She has since been a post-doctoral researcher at Idiap, Martigny, and Fondazione Bruno Kessler, Trento. She was also a visiting researcher at the University of Bristol. Her research interests are mainly in the areas of computer vision and machine learning. She is a member of the IEEE.

Ramanathan Subramanian received the PhD degree in electrical and computer engineering from the National University of Singapore in 2008. He is currently a research scientist at the Advanced Digital Sciences Center, Singapore. His research interests span human-centered computing, brain-machine interfaces, human behavior understanding, computer vision, and multimedia processing. He is a member of the IEEE and ACM.

Gaowen Liu received the BS degree in automation from Qingdao Univerisity, China and the MS degree in system engineering from the Nanjing University of Science and Technology, China, in 2006 and 2008, respectively. She is currently working toward the PhD degree in the MHUG group at the University of Trento, Italy. Her research interests include computer vision and machine learning.

Oswald Lanz received the masters in mathematics and the PhD degree in computer science from the University of Trento, in 2000 and 2005, respectively. He then joined Fondazione Bruno Kessler, and post his tenure in 2008, now heads the Technologies of Vision research unit of FBK. His research interests are mainly in the areas of computer vision and more recently in machine learning for vision. He holds two patents on video tracking. He is a member of the IEEE and ACM.

Nicu Sebe received the PhD degree from Leiden University, The Netherlands, in 2001. Currently, he is with the Department of Information Engineering and Computer Science, University of Trento, Italy, where he is leading the research in the areas of multimedia information retrieval and human behavior understanding. He was a general co-chair of FG 2008 and ACM Multimedia 2013, and a program chair of CIVR 2007 and 2010, and ACM Multimedia 2007 and 2011. He is a program chair of ECCV 2016 and ICCV 2017. He is a senior member of the IEEE and ACM and a fellow of IAPR.