Count on me: learning to count on a single image

Francesco Setti, Davide Conigliaro, Michele Tobanelli and Marco Cristani, *Member, IEEE*

***Abstract*—Individuating and locating repetitive patterns in still images is a fundamental task in image processing, typically achieved by means of correlation strategies. In this paper we provide a solid solution to this task using a differential geometry approach, operating on Lie algebra, and exploiting a mixture of templates. The proposed method asks the user to locate few instances of the target patterns (seeds), that become visual templates used to explore the image. We propose an iterative algorithm to locate patches similar to the seeds working in three steps: first clustering the detected patches to generate templates of different classes, then looking for the affine transformations, living on a Lie algebra, that best link the templates and the detected patches, and finally detecting new patches with a convolutional strategy. The process ends when no new patches are found. We will show how our method is able to process heterogeneous unstructured images with multiple visual motifs and extremely crowded scenarios with high precision and recall, outperforming all the state of the art methods.**

***Index Terms*—Object counting, Repetitive patterns, Lie Algebra, Template matching, Congealing**

I. Introduction

INDIVIDUATING repetitive complex patterns – *i.e.* multiple instances of the same class of visual entities – in a single image is a crucial operation in many real-world computer vision and image processing applications, like counting cells in microscopic images, profiling crowds in surveillance videos, and performing wildlife census (see Fig. 2 for some examples). Very often, these tasks have to be carried out in very diverse conditions: think for example to biologists monitoring birds migrations or policemen counting people at public event using hand held cameras; such a task is dubbed *texton detection* (or *texton learning*) when the basic patterns to be counted are composed by few pixels, carry no semantic meaning, and are located in a strongly structured layout [1]; conversely, the expression *object counting* is adopted when the patterns are instances of a given class of objects (*e.g.* cells, pedestrians, animals, *etc.*). In this paper we will focus on this latter task.

In the literature, object counting is performed in very different ways: by adopting detection strategies [2]–[5], by regression [6]–[8], or by segmentation [9], [10] (see more details in Section II); in all these cases, a learning phase on labeled data is required to learn the specific model of the object whose instances have to be individuated. This opens up to some problems: the need of training data demands considerable time and resources; moreover, some heterogeneity in the labeled data is compulsory to ensure generalization in the counting stage. Moreover, collecting "good" training data is an issue in itself for object detection and recognition [11], [12].

The authors are with the Vision, Image Processing & Sound Lab (VIPS), Department of Computer Science, University of Verona, strada Le Grazie 15, 37135 – Verona (Italy)

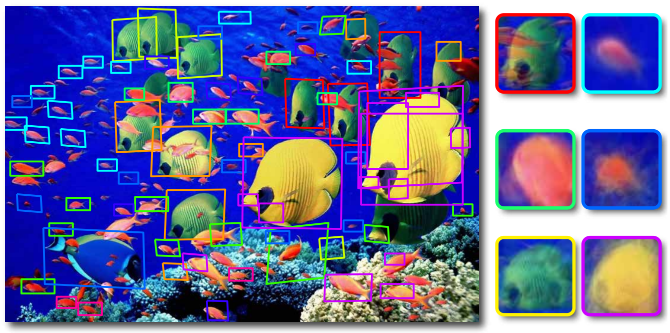

Fig. 1. A sample image used in our tests and the patches detected with our method, automatically clustered into six different classes (highlighted with different colors). On the right hand side, the final templates of each class.

For example, let consider Fig. 1: a standard fish detector would reasonably give very poor results due to the intraclass variability and perspective distortions. In addition, the objects' classes to be counted could be disparate, and co-present in the same picture, as in the above example image where there are diverse fish species. This amounts to say that understanding which particular detector(s)/regressor(s) have to be used given a particular image is a crucial issue and, in the case of novel objects' classes to be considered, we fall back again on the problem of proper training data collection. This is especially true in the case of medical image analysis, where the nature of the items to be counted is very different to the standard set of objects whose detectors are available in the literature, such as Deformable Parts Model (*DPM*) [13] . This actually brings to approaches where the user is asked to mark tens of images or hundreds of instances [14], or to ad-hoc solutions, as in the case of specific classes of entities [15], producing a classifier which actually estimates densities of objects, rather than individuating every single object instance [16].

In this paper, we provide a solid solution to the counting problem, offering a framework that: 1) does not require any preliminary training step; 2) requires minimal supervision by the user, who is asked to mark a minimal number of instances (seeds); and 3) is able to count radically different types of objects, without necessarily asking the user to explicitly individuate different typologies of items; vice versa, the algorithm provides a componential counting report – that is, it automatically individuates different kinds of objects, and then counts them.

The obtained results are convincing, also compared to all those techniques which work in similar circumstances, as in [16], [17]. In particular, we will discuss both qualitative and quantitative results, showing that our approach provides functionalities so far unreachable by any other counting technique, and is applicable to a vast heterogeneity of images, ranging from crowd photos to traffic shots, from naturalistic pictures to medical images, with just a single image.

Summarizing, the main contributions of this work are:

- a novel framework that learns to count similar objects – *i.e.* repetitive patterns – on a single image without the need of a training step and requires a minimal supervision by the user;
- a novel dataset of images taken from diverse applicative fields (*e.g.* surveillance, medical image analysis, *etc*.) and containing heterogeneous multiple objects and drastically different geometrical layouts.

The rest of the paper is organized as follows. In Section II we review the related literature; in Section III we present our proposed framework, while in Section IV we detail extensive experiments. Finally, Section V concludes the paper with some discussions and directions for future works.

II. Related Work

The counting problem has been widely studied in the last years, and several different approaches have been proposed in literature. A first way to perform object counting is by using detection strategies: in [2], Barinova *et al.* embed a Hough transform-based search into a probabilistic framework for object detection bypassing the problem of multiple peaks identification, avoiding non-maximum suppression heuristics; Leibe *et al.* [4] consider object categorization and figure-ground segmentation as two interleaved processes that closely collaborate towards a common goal, they also use Generalized Hough Transform to represent the object shape. Dong *et al.* [3] propose an exemplar-based algorithm which maps the global shape by Fourier descriptors to various configurations of humans for people detection in crowd; they also use locally weighted aeraging to interpolate for the best possible candidate configuration. Wu *et al.* [5] address the partial occlusion of the object to be detected by using whole-object and part detectors, defining a part hierarchy and computing a joint likelihood which includes an inter-object occlusion reasoning based on the object silhouettes extracted by the whole-object detector.

A different way to proceed is to train a classifier to estimate the density of objects in an image; this is particularly suited for inferring the number of pedestrians in a crowd. Cho *et al.* [6] propose to extract a set of significant features from sequences of images, and then to model those features by a neural network to extract the crowd density in complex scenes. Kong *et al.* [7] take into account feature normalization mechanisms to deal with perspective projections and different camera orientations; they compute features including edge orientations and blob size histograms resulted from edge detection and background subtraction to learn the relationship between the features histograms and the number of pedestrians in the crowd. Marana *et al.* [8] extract features by means of texture analysis techniques based on grey level transition probabilities on digitised images to feed a self-organizing neural network which performs the crowd density estimation.

Other works are based on segmentation strategies: Ryan *et al.* [10] run foreground/background segmentation to extract blobs of crowd first, and then train a regressor from simple holistic features to estimate the number of pedestrians in each blob; similarly, Chan *et al.* [9] use a mixture of motion models encoding dynamic textures to segment moving crowds into homogeneous motion blobs, then the correspondence between a set of holistic features and the number of people is learned with Gaussian process regression.

Recently, Arteta *et al.* [16] propose an interactive counting system able to count object instances in an image by estimating their density with ridge regression; their results show how well bigger training sets improve the counting accuracy. Furthermore, the authors highlight that the system does not handle perspective geometry due to the fact that the extracted features consider only CIELAB color space and no geometrical aspects.

A common aspect of all these methods is the need of a training set to learn the parameters of an object detector or a density estimator.

The strategy adopted in this work is based on the detection of repetitive patterns, which is one of the classical and long-standing problems of computer vision and whose importance has been explained in [18]. In that article, Leung and Malik present an algorithm to detect, localize and group instances of repeated scene elements which exploits window matching and region growing strategies. Hays *et al.* [19] use higher-order feature matching algorithm to discover the lattices of near-regular textures in images; Park *et al.* [20] improve this work by formulating the 2D lattice detection as a spatial, multitarget tracking problem, solved within a Markov Random Field (MRF) framework using a Mean-Shift Belief Propagation (MSBP) method. Schaffalitzky and Zisserman [21] propose a RANSAC-based grouping algorithm for patterns that could be considered as lying on a plane in the captured scene, while Lin and Liu [22] exploit a region growing method to detect dynamic lattice patterns and utilize belief propagation and particle filtering to handle the dynamic tracking problem. The last methods are based on the detection of low-level features and their grouping, thanks to perspective geometric transforms or global deformable templates.

Other grouping methods are based on local alignment rather than on global strategies. Wu *et al.* [23] deal with the problem of analyzing large repetitive structures in urban scenes and propose an algorithm based on the repetition of local symmetry axes. Spinello *et al.* [24] use local descriptors and a contour codebook to detect repetitions in a lattice model. In two recent works [25], [26], Cai and Baciu propose to use Lie algebra to align repetitive patterns in near regular textures; in [17], the same authors provide an algorithm for repetitive patterns detection and grouping based on a Lie group model, inherited from visual tracking strategies published in [27] and [28].

In this paper, we present a new framework that relies on the combination of template matching and congealing strategies. The method evaluates how comparable two visually similar patches are, by mapping the affine transformations that allow to overlap them in a Lie group. We exploit the congealing method presented in [29], forcing it to work under affine transformation constraints. The proposed method is able to simultaneously manage multiple texton templates, *i.e.* different object classes in the same image or different configurations of the same object class. We also introduce a deformation constraint inside the congealing process by means of the definition of distance between two matrices given in [30]; this is needed to avoid degenerate solutions of the congealing algorithm, as highlighted by Vedaldi and Soatto in [31]. In conclusion, while most of the related works focus on the inference of a structure considering invariant repetitive patterns in images, we want to count objects in an image where the structural constraint is not essential and the repeated instances can have different configurations and intra-class variations, *e.g.* as could be found in people counting in crowded situations.

III. Method

The fundamental step of our approach is *template matching*, which consists in individuating in the image those regions that best match a given visual template. Since a simple template matching is usually not enough to achieve convincing results, due to the rigidity of the template, our proposal is to allow the template to be deformed by a set of affine transformations.

The proposed framework works in an iterative way, with an initialization phase requiring user interaction. In the initialization step, the user is asked to locate few instances of the objects to count (*seeds*). After that, the iteration runs over 4 steps:

1) **Patch Description:** patches are described by means of a Bag-of-Words (BoW) model, considering RGB values, SIFT and HOG features;
2) **Component Extraction:** an unsupervised clustering approach is carried out to identify different components, *i.e.* different objects or different configurations of the same object;
3) **Congealing:** an optimization procedure allows to describe all the patches belonging to a given component by means of a common template patch and a set of affine deformation matrices;
4) **Template Matching:** we perform cross-correlation over the entire image to identify those regions that exhibit high similarity with the template.

The process stops when a new iteration does not detect any new patch; this is usually reached in our experiments after 3 iterations on a 600×600 pixels image. The scheme of the approach is sketched in Algorithm 1. In the rest of this section each step will be explained in the details.

Algorithm 1 Main

```
Input: image $\mathbf{I}$
Output: detected patches $\mathcal{P}$, labels $\mathcal{C}$, templates $\mathcal{T}$
  $\mathcal{G} \leftarrow$ Initialization: identity matrices
  $\mathcal{C} \leftarrow$ Initialization: all patches belonging to one cluster
  repeat
    $\mathbf{P}_d \leftarrow$ Patches Description (Alg. 2)
    $\mathcal{C} \leftarrow$ Component Extraction (Alg. 3)
    $\mathcal{T}, \mathcal{G} \leftarrow$ Congealing (Alg. 4)
    $\mathcal{P}_{\text{new}} \leftarrow$ Template Matching (Alg. 5)
    $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_{\text{new}}$
  until no new patches found ($\mathcal{P}_{\text{new}} = \emptyset$)
```

A. Initialization

Given an image $\mathbf{I}$, the user is required to select a set $\mathcal{P}$ of N patches containing the objects to be detected in the rest of the image. We will call these patches *seeds*. Typically, the algorithm gives good results with a number of seeds of about 5% of the total number of instances in the image, depending also on the number of different objects to focus on.

B. Patch Description

Each patch is described with a Bag-of-Words (BoW) representation on 3 different kinds of features: RGB pixels, dense SIFT [32] and dense HOG [33] features. To add spatial information to BoW we build spatial histograms by dividing each patch into a grid of M_p sub-patches and for each sub-patch we extract the BoW histograms of N_w words. To this sake, dense SIFT and dense HOG features are first extracted; then, the dictionary is learnt for each kind of feature separately (including RGB), by k-means clustering employing the euclidean distance; finally, the histograms are generated for each feature on each patch. All the resulting histograms are then concatenated so that the resulting description of all the patches is the matrix $\mathbf{P}_d$ of size $N \times 3N_wM_p$.

Algorithm 2 Patch description

```
Input: patches $\mathcal{P}$, transformation matrices $\mathcal{G}$
Output: Patches description $\mathbf{P}_d$
  for all patch $p^i \in \mathcal{P}$ do
    $p^i_g \leftarrow$ Apply transformation $G_{p^i} \in \mathcal{G}$ on patch $p^i$
    $p^i_d \leftarrow$ Extract SIFT, HOG and RGB descriptors
  end for
  $V \leftarrow$ Generate vocabulary $V$ using k-means over all $p_d$
  for all patch $p^i \in \mathcal{P}$ do
    Split $p^i$ into a grid of $M_p$ sub-patches
    for all sub-patch do
      $p^i_h \leftarrow$ Concatenate the BoW representations
    end for
    $\mathbf{P}_d \leftarrow$ Push $p^i_h$ to the final patches description matrix
  end for
```

C. Components Extraction

Components extraction identifies clusters of patches visually similar to each other. For this aim, clustering is performed by *affinity propagation* which is based on the idea of iteratively exchanging messages between data points (*i.e.* patches) until a proper set of exemplars are found. Such exemplars identify clusters [34]. The clustering is unsupervised on the number of components selected, minimizing thus the degree of user supervision. Starting from the set $\mathcal{P}$ of all the patches detected in the previous iteration, the output of this step is a set $\mathcal{C}$ of labels representing C clusters, each one composed by a set $\mathcal{P}_c$ of n_c patches such as $\mathcal{P} = \bigcup_c \mathcal{P}_c$, with $1 \leq c \leq C$.

The clustering algorithm acts on a dissimilarity matrix defined as the additive inverse Hadamard product of two distance matrices: one defined as the Batthacharyya distance computed over the patches descriptors described in the previous paragraph, and the second defined as pixel-wise difference of RGB colors of each pair of patches. Notably, affinity propagation works even with few samples, as in the case of the first iteration of our approach.

Algorithm 3 Components extraction

```
**Input:** patches $\mathcal{P}$, patches description $\mathbf{P}_d$
**Output:** labels $\mathcal{C}$
  $\mathbf{D}_B \leftarrow$ Compute Bhattacharya distance
  $\mathbf{D}_C \leftarrow$ Compute pixelwise difference of RGB colors
  $\mathbf{S} \leftarrow$ Compute the similarity matrix as $-(\mathbf{D}_B * \mathbf{D}_C)$
  $\mathcal{C} \leftarrow$ Affinity Propagation applied to $\mathbf{S}$
```

D. Congealing

The congealing task is defined as "the problem of aligning an ensemble of images in an unsupervised manner" [35]. The proper way to perform the alignment assumes the knowledge of the parametric nature of the misalignment (*e.g.* translation, rotation, affine transformation, *etc.*) and also the fact that patches should have similar appearance.

The congealing algorithm used in our approach is based on the conventional unsupervised least-squares congealing [35], [36] which operates on a set of N unaligned patches. The main idea is that all the patches $p_c^i \in \mathcal{P}_c$ are assumed to be related with a common template by an affine transformation matrix $G_{p_c^i}$ such as:

$$\left\| p_c^i - \mathbf{T}_c \circ G_{p_c^i} \right\|^2 < \epsilon \tag{1}$$

where ϵ is an arbitrary small positive value and $\mathbf{T} \circ G$ represents the application of an affine deformation G to the template $\mathbf{T}$, see Appendix A for more details. It is also possible to extend Equation (1) by considering the relationship between two different patches p_c^i and p_c^j belonging to the same component c:

$$\left\| p_c^i - p_c^j \circ G_{p_c^j}^{-1} \circ G_{p_c^i} \right\|^2 < \epsilon \tag{2}$$

Each patch is associated with the related template by means of the affine deformation G, and therefore, from Equation (11) (Appendix A), by means of 6 unknown transformation parameters a_k. Thus, we can define for each patch p_c^i the array of parameters $\mathbf{a}_c^i = (a_{p_c^i}^1, \ldots, a_{p_c^i}^6)$, and for each component c the collection of all the parameters' vectors $\mathbf{a}_c = (\mathbf{a}_c^1, \ldots, \mathbf{a}_c^{n_c})$. The purpose of our congealing algorithm is to estimate, for each component separately, the best parameters by minimizing the cost function:

$$\Phi(\mathbf{a}_c) = \sum_{\substack{1 \le i,j \le n_c \\ i \ne j}} \frac{1}{2} \left\| p_c^i - p_c^j \circ G_{p_c^j}^{-1} \circ G_{p_c^i} \right\|^2 \tag{3}$$

with respect to the matrix $\mathbf{a}_c \in \mathbb{R}^{6 \times n_c}$ with the constraint of Equation (1).

We use Newton's method to perform the minimization, considering the first order necessary condition:

$$\nabla_{\mathbf{a}_c^i} \Phi(\mathbf{a}_c) = 0 \tag{4}$$

with the operator $\nabla_{\mathbf{a}_c^i} = (\partial/\partial a_{p_c^i}^1, \partial/\partial a_{p_c^i}^2, \ldots, \partial/\partial a_{p_c^i}^6)$. Equation (4) can be solved considering the iterative scheme:

$$\mathbf{a}_c^i(k+1) = \mathbf{a}_c^i(k) - \left[H \cdot \Phi(\mathbf{a}_c^i(k))\right]^{-1} \nabla\Phi(\mathbf{a}_c^i(k)) \tag{5}$$

where k indicates the iteration index in the minimization process and H is the Hessian matrix. In such a case, instead of directly inverting the Hessian matrix, it is more efficient to compute the vector $\mathbf{d}(k) = \left[H \cdot \Phi(\mathbf{a}_c^i(k))\right]^{-1} \nabla\Phi(\mathbf{a}_c^i(k))$ as the solution of the linear system:

$$\left[H \cdot \Phi\left(\mathbf{a}_c^i(k)\right)\right] \mathbf{d}_k = \nabla\Phi\left(\mathbf{a}_c^i(k)\right) \tag{6}$$

This formulation of the minimization process relies on the image gradient computation, which is a very noise sensitive operation and could lead to degenerate solutions. In order to avoid this problem, we introduce the concept of distance between two transformation matrices in the Lie group *Aff*(2) as defined by Porikli in [30]. Thus, a hard constraint is posed on the distance between transformation matrices in two subsequent iterations as:

$$\left\| Log\left(G_{p_c^i}(k), G_{p_c^i}(k+1)\right) \right\| \le \tau_d \tag{7}$$

The last step of the congealing phase aims at generating, for each component, the template to be used for the matching phase. This template is defined as the average of all the n_c deformed patches belonging to each component c:

$$\mathbf{T}_c = \frac{1}{n_c} \sum_{1 \le i \le n_c} p_c^i \circ G_{p_c^i}^{-1} \tag{8}$$

The set of all the templates $\mathbf{T}_c$ is referred as $\mathcal{T}$, while $\mathcal{G}$ is the set of all the transformation matrices G_p.

Algorithm 4 Congealing

```
**Input:** patches $\mathcal{P}$, transformation matrices $\mathcal{G}$, labels $\mathcal{C}$
**Output:** templates $\mathcal{T}$, transformation matrices $\mathcal{G}$
  **for all** components $c \in \mathcal{C}$ **do**
    **repeat**
      $\mathbf{a}_c(k) \leftarrow$ Estimate coefficients $\mathbf{a}_c(k)$ minimizing (3)
      **for all** patches $p_c^i \in \mathcal{P}_c$ **do**
        $G_{p_c^i}(k) \leftarrow$ Solve (11) using $\mathbf{a}_c(k)$
        **if** (7) *is true* **then**
          $\mathcal{G} \leftarrow$ Update $\mathcal{G}$ with $G_{p_c^i}(k)$
        **end if**
      **end for**
      $k = k + 1$
    **until** $\mathbf{a}_c(k) = \mathbf{a}_c(k-1)$
    $\mathcal{T} \leftarrow$ Update $\mathcal{T}$ with template $\mathbf{T}_c$ computed as in (8)
  **end for**
```

E. Template Matching

Given the templates $\{\mathbf{T}_c\}$ associated with each component and the transformation matrices $\{G_{p^i}\}$ associated with each detected patch, a cross-correlation strategy is employed to detect new patches over the entire image.

It is worth noting that we do not search for structured patterns or continuous deformations; indeed, we relax the constraint of having similar deformations for objects close to each other and we search over the entire image each possible combination of templates and deformations.

For each template $\mathbf{T}_c$, $c = 1 \dots C$ and each transformation matrix G_{p^i}, $i = 1 \dots N$, we compute normalized cross-correlation (NCC) over the entire image for each composed template $\mathbf{T}_c^i = \mathbf{T}_c \circ G_{p^i}$. This generates a set of $N \cdot C$ correlation maps $\mathbf{M}_{i,c} = \text{NCC}\left(\mathbf{T}_c^i, \mathbf{I}\right)$. The final results of NCC are the global cross-correlation map $\mathbf{M}$ defined as the pixelwise maximum over all the maps $\mathbf{M}_{i,c}$

$$\mathbf{M}(x, y) = \max_{i,c} \mathbf{M}_{i,c}(x, y) \tag{9}$$

New patches are the local maxima of $\mathbf{M}$. To avoid problems related to noise in low response maps, we also impose a threshold τ_m on the minimum matching score, thus all the local maxima lower than τ_m are ignored.

Algorithm 5 Template Matching

```
Input: image $\mathbf{I}$, templates $\mathcal{T}$, transformation matrices $\mathcal{G}$, labels $\mathcal{C}$
Output: new patches $\mathcal{P}_{\text{new}}$
  for all $\mathbf{T}_c \in \mathcal{T}$ (with $c \in \mathcal{C}$) do
    for all transformation matrices $G_{p^i} \in \mathcal{G}$ do
      $\mathbf{T}_c^i \leftarrow$ Apply transformation $G_{p^i}$ to template $\mathbf{T}_c$
      $\mathbf{M}_{i,c} \leftarrow$ Compute correlation map
    end for
  end for
  $\mathbf{M} \leftarrow$ Compute the pixelwise maximum of $\mathbf{M}_{i,c}$ as in (9)
  $\mathcal{P}_{\text{new}} \leftarrow$ Detect new patches as local maxima of $\mathbf{M}$.
```

Our approach has complexity $\mathcal{O}(NC)$ where N is the number of transformation matrices (*i.e.* patches) and C is the total number of components.

IV. Experiments

We tested our framework on a brand new dataset, released with this paper, and two publicly available datasets: Cells data [16], composed by 200 synthetic images of cells; and S-Hock data [37], focusing on 93 images of people standing on a stadium watching an hockey match. The new dataset, dubbed "RepTile", exhibits several features that make it extremely challenging; it is composed by 50 pictures collected from the Internet containing heterogeneous visual objects, such as humans, animals, bacteria, *etc.*; some sample images are portrayed in Fig. 2, while the whole dataset is available at http://vips.sci.univr.it/dataset/counting/data/RepTile.zip. The images of the dataset exhibit some specific difficulties, like irregularly shaped objects and partial occlusions (Fig. 2-**B**, bacteria), strongly different appearances (Fig. 2-**D**, faces), also exhibiting object instances that are spatially very close or overlapping each other (Fig. 2-**A**, cows and birds). The objects' count ranges between 9 and 655, with an average of 67 objects per image. All the images are annotated with the ground truth positions of each object in the from of bounding boxes, for a total of 3366 annotations. Finally, the object instances employed as seeds in our experiments are also specified in the dataset.

In order to highlight the contribution of each step of our framework, we performed experiments with three different versions of our approach: **1)** simple template matching (**TM** – Algorithms 2 and 5), **2)** template matching enhanced with components extraction (**TM+CE** – Algorithms 2, 3 and 5), and **3)** the complete procedure. All the proposed experiments are performed with the same starting seeds and the same parameters.

As for the parameterization, two quantities have to be set: the thresholds τ_d and τ_m. About the matching threshold, our experiments show that good values are $\tau_m \in [0.2, 0.4]$; higher values enforce the method to select instances strongly similar to the template (so generating many miss-detection), while lower values tend to create many instances and thus increasing the number of false positives. For geodesic distance threshold, the best results are achieved with $\tau_d \in [1, 2]$; this parameter regulates how different the transformations have to be in order to be considered during the congealing step, and in practice it prevents the patches to be strongly deformed.

In the initialization step we manually selected a number of seeds equal to 5% of the total amount of objects in the image. As an under-the-hood rule, a good strategy for the initial selection of the patches is to focus on visual entities strongly different from each other; in this way, a large variability of the objects to consider can be obtained in the first iteration, and maintained along the entire image exploration process, thus capturing more instances. Vice versa, if the initial seeds are too similar to each other, they will collapse into a single mixture component, resulting in a too conservative exploration process. It is important to note that the approach accepts the seed patches with or without the indication of which mixture component they have to be assigned to: in the first case, the user tells the algorithm that there are two or more kinds of objects into play (*e.g.* cows and birds in Fig. 2-**A**); in the second case the algorithm is fully responsible to detect different kinds of entities. In any case, the method can refine the indication of the user further partitioning mixture components (*e.g.* different configurations of the same object class, as for birds' heads in Fig. 2-**C**). In our experiments we tested the worst scenario, thus we did not specify the component of each seed.

For the features extraction we used the MATLAB implementation of VLFeat libraries [38]. We built a codebook of 20, 20 and 40 words respectively for the SIFT, HOG and RGB color spaces, and the spatial histograms are computed by splitting each patch into a grid of 4×4 subpatches, with no overlap among them.

The evaluation metrics employed are different according to the specific task: object detection and counting. For the object detection task, the quantitative evaluation is based on the concept of *matching*: a match occurs when the intersection between the predicted and ground truth patches is higher than 50% of the union of the two areas [39]. With this definition, we can define the standard pattern recognition rates of *precision*, *recall* and F_1. For the counting task, we used two standard measures: Mean Absolute Error (MAE), defined as the mean of absolute differences between ground truth and predicted counts, and Normalized Mean Absolute Error (NMAE), which

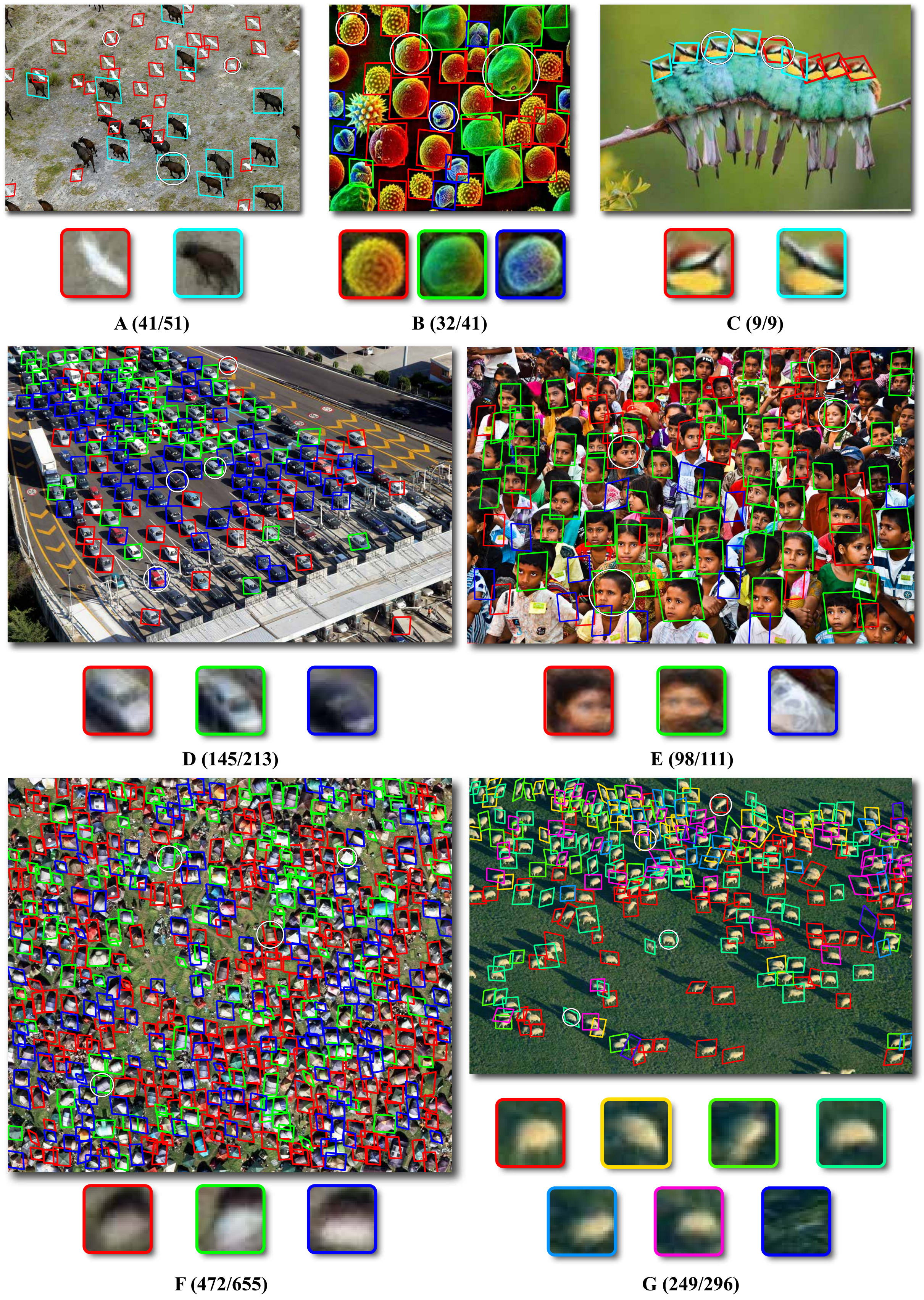

Fig. 2. Detection results obtained on 7 images of the RepTile dataset and final templates generated by the algorithm. The seeds are highlighted with a black and white circle. The label of each image represents the *ratio: predicted/GT counts*.

Method	Detection			Counting	
	prec.	rec.	F_1	MAE	NMAE
Cai and Baciu [17]	0.478	0.473	0.396	59	1.034
Arteta *et al.* [16]	-	-	-	50	1.629
TM	0.860	0.776	0.805	18	0.186
TM + CE	0.909	0.790	0.835	18	0.164
Proposed	**0.915**	**0.839**	**0.870**	**14**	**0.109**

TABLE I
DETECTION AND COUNTING RESULTS ON REPTILE DATASET. COMPLETE FRAMEWORK (PROPOSED), TEMPLATE MATCHING ONLY (**TM**) AND TEMPLATE MATCHING WITH COMPONENT EXTRACTION (**TM+CE**), AND STATE-OF-THE-ART METHODS.

Method	Cells data		S-Hock data	
	MAE	NMAE	MAE	NMAE
Cai & Baciu [17]	149	0.809	117	0.779
Arteta *et al.* [16]	45	0.332	31	0.208
Proposed	**40**	**0.233**	**7**	**0.048**

TABLE II
COUNTING RESULTS ON CELLS [16] AND S-HOCK [37] DATASETS.

Method	RepTile	Cells data	S-Hock data
Cai & Baciu [17]	2814	753	3655
Arteta *et al.* [16]	685	184	1067
Proposed	867	272	1265

TABLE III
AVERAGE EXECUTION TIME (IN SECONDS) FOR ALL THE METHODS AND DATASETS. IMPLEMENTATIONS OF CAI & BACIU [17] AND THE PROPOSED METHODS ARE PRODUCED BY THE AUTHORS, WHILE ARTETA *et al.* [16] IS OUR IMPLEMENTATION STRICTLY FOLLOWING THE PAPER'S ALGORITHM.

is obtained by normalizing the MAE with the objects count for each image. In both these cases, less is better.

Quantitative results on RepTile data are presented in Table I. The third row shows the results obtained with template matching only (Algorithm 5), where the input transformation matrices $\mathcal{G}$ are identity matrices and each patch belongs to a different component. Thus, each patch is the template used in the matching phase, without applying any transformation. Augmenting this with the components extraction step (Algorithm 3) allows to take into account different object types, generating fewer templates to search in the image. This reduces the number of false positives, resulting in a higher precision of about 5%. Adding the congealing phase (Algorithm 4) allows to obtain a better definition of templates by taking into account the patches' deformations; this increases the number of correct detections, resulting in a higher recall of about 5%.

We also tested the proposed method on two public datasets particularly suited for the object counting task. The first one is Cells dataset [16], which contains 200 synthetic images of cells, while the second is S-Hock dataset [37], that includes the counting of people in the crowd.

For the Cells data, we tested all the 200 images separately. The experimental protocol adopted in [16] requires that a few images are used for the training (labeling them in their entirety), focusing on the remaining ones as testing set. This is not required by our approach, that instead wants to minimize the user support. We started the analysis of each image with the same random selection of 5% of cells in the image as initial seeds. The starting bounding box of each seed is a square box with size equals to 3 times the radius of the cell and centered in the center of the cell itself. Please note that the testing protocol is very different from the one used in [16], where they used N training images and 100 testing images with 3 repeated iterations to ameliorate the results (injecting false positive as negative training sample); with this testing protocol the results are about 10 times better then the one reported in this paper, but this is a definitely different task and it is not the problem addressed in this paper.

For the S-Hock dataset, we selected 100 random frames from the videos and for each image we extracted a random selection of 5% of people as initial seeds. In this case the starting bounding boxes are the ground truth annotation of each frame. All the methods were compared using the same patches as input for each image.

Table II reports counting results on Cells data [16] and S-Hock data [37]. The method presented by Cai and Baciu has worst performance because images in both datasets are unstructured, while the Arteta's method performs pretty well on its own data, but experienced problems with S-Hock data, probably due to the high appearance variability of faces in the crowd. In both cases our method outperforms the state-of-the-art of more than 10% in MAE and NMAE. In the case we decide to select few training patches on a single image, and apply our approach starting from these patches on other images, we avoid the overfitting, and the results drop down of only 10% in terms of MAE on both Cell and S-Hock data.

We also present a comparison of the execution times for the different methods and datasets (Fig. III). In our tests we used a non optimized Matlab implementation on a standard laptop with Intel i7 CPU and 8GB of RAM. The proposed method is very close to the best performing one ([16]), in particular when the number of objects is limited (RepTile and S-Hock data).

Fig. 3 shows qualitative results on a very challenging image in RepTile dataset. Comparing images **D** and **E**, one can appreciate how the component extraction phase hepls in reducing the number of false positives (*e.g.* the detection on top-left corner of **D**). In **E**, the templates generated after clustering the different components are fewer than in **D** and they average the appearance of many patches, making the matching phase less sensitive to noise. Still, template matching cannot handle the different geometry appearance of the entities, *e.g.* fishes of different sizes with respect to the initial seeds will not be properly detected. We overcome this problem by aligning the patches belonging to the same component with the congealing step (image **F**), resulting in an increment of correct detections and, thus, of the recall rate. Images **B** and **C** report results with state-of-the-art methods. In **B**, Cai and Baciu [17] also consider a congealing phase but they do not impose deformation constraints, leading some patches to degenerate deformations. Furthermore, they are not able to manage multiple texton templates, bringing to detection of one single species at a time. Image **C** shows the density estimation with the interactive framework of Arteta *et al.* [16]. Such a method does not provide proper detections, but it estimates the number of objects based on the density in specific regions. In our experiments the final estimator cannot be adequately

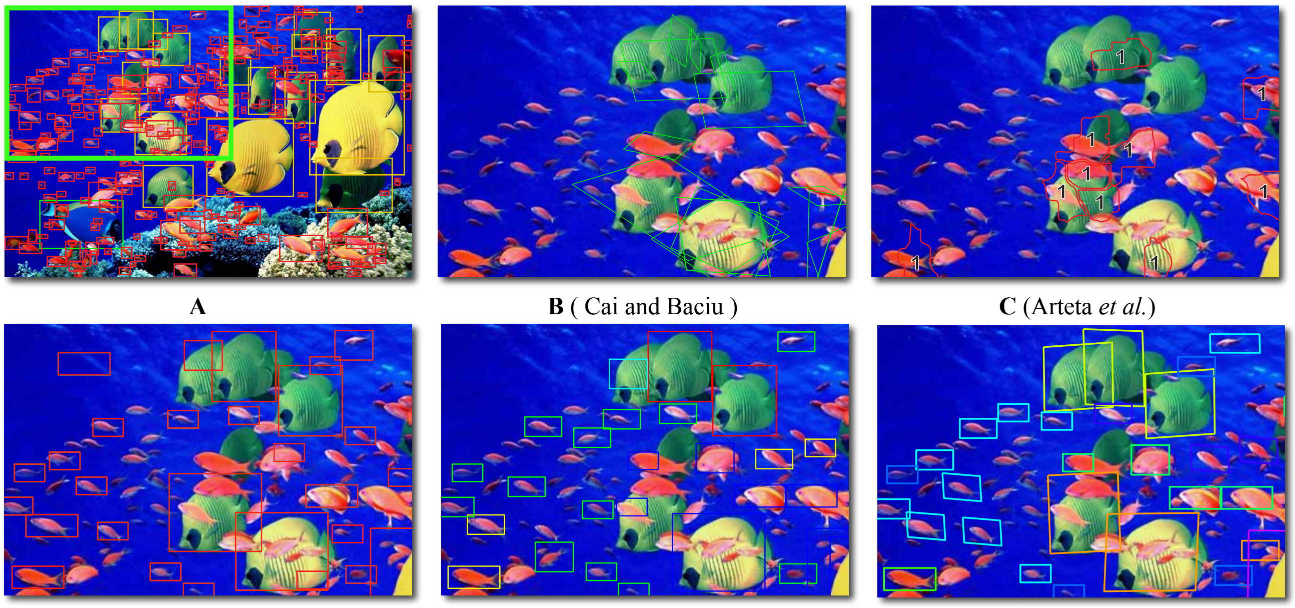

Fig. 3. Qualitative results of our experiments. The green rectangle in image **A** highlights the area where the images **B**-**F** are focused.

learned due to the low number of training samples.

More qualitative results are shown in Fig. 2, where detections and generated templates are reported. As for the number of components automatically found, in some cases they are intuitive – as the kind of animals in **A** and bacteria in **B**, or the head orientation in **C** –, while in other cases the components capture small variations on the visual appearance of the objects, supposedly partitioning the visual intraclass variance in separate smaller groups as visible in **F** and **G**. Moreover, templates give to the user an immediate and useful information about which object the component is related to, eventually allowing the user to select and remove the misleading components. *e.g*. the blue template in image **E** is definitely not a face and the elements belonging to this component could be removed from the final counting.

V. Conclusions

In this paper we presented an online learning by example approach, capable to individuate visual motifs in an image with a minimal supervision by the user, who only needs to select few visual objects of interest. The approach outputs all the instances of the target objects in the image, individuating and extracting different object typologies (*e.g*. cows and birds). The approach has been tested on a brand new dataset, dubbed “RepTile”, characterized by very challenging characteristics for object counting: different object classes in the same image, a variable number of instances per type of object, and non-rigid geometric displacement. This problem cannot be faced with any of the techniques present in the literature, since they need training data to work, or images where the repeated patterns form a rigid structures, as the windows of a skyscraper or a texture. Future perspectives are to embed the proposed method into an interactive tool that allows the user to select “good” and “bad” templates, automatically removing all the detections provided by undesired components. Another direction will focus on images where the objects to be analyzed are composed by few pixels: actually, in our experiments, reasonable results are obtained if the patches are at least of 30×30 pixels. This in practice prevents us to apply our framework to some instances of crowd counting. In order to reduce the computational complexity, we can reduce the number of total transformation matrices (N) by clustering them into groups of similar matrices, thus reducing the number of cross-correlation maps to be computed. Finally, a segmentation module based on superpixels can be included in the pipeline to refine the final output, providing a proper foreground/background segmentation. In such a case, our method could also be embedded into image post processing tool as Photoshop or Gimp.

Appendix A
Affine transformation and Lie algebra

In geometry, an *affine transformation* is a function between affine spaces which preserves points, straight lines and planes. A generic 2-D affine transformation is represented in the homogeneous coordinates by means of the matrix G, such as:

$$[x', y', 1]^T = G[x, y, 1]^T \tag{10}$$

where x, y are the coordinates of a generic point in the original space and x', y' are the coordinates after the mapping. $G = \left[\begin{smallmatrix} M & t \\ 0 & 1 \end{smallmatrix}\right]$, where $M \in GL(2)$ is an invertible 2×2 matrix and $t \in \Re^2$ is a 2×1 translation vector.

The matrix G can be identified as the group of all invertible affine transformations from the affine space into itself and it is a Lie group $\mathbf{G}$ called 2-D affine group *Aff*(2). A Lie group $\mathbf{G}$ is a differentiable manifold such as the group operations, multiplication and inverse, are differentiable maps. The tangent space to the identity element $\mathbf{I}$ of the group forms a Lie algebra $\mathbf{g}$. A Lie group $\mathbf{G}$ and its algebra $\mathbf{g}$ are related by the exponential map $\exp\colon \mathbf{g} \to \mathbf{G}$ such that

$$G = Exp\left(\sum_{k=1}^{6} a_k E_k\right) \tag{11}$$

where $\{a_k\}_{k=1,\ldots,6}$ are the coefficients related to the basis E_k chosen as

$$E_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad E_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$E_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad E_5 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad E_6 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Each geometric transformation corresponds to each E_k.

The inverse mapping is given by $\log\colon \mathbf{G} \to \mathbf{g}$

$$\mathrm{Log}(G) = \sum_{k=1}^{6} a_k E_k \tag{12}$$

The Lie algebra $\mathit{aff}(2)$ is represented as $\left[\begin{smallmatrix} U & v \\ 0 & 0 \end{smallmatrix}\right]$, where $U \in gl(2)$ and $v \in \Re^2$. A further description of the Lie groups and Lie algebra can be found in [40].

References

[1] S.-C. Zhu, C.-E. Guo, Y. Wang, and Z. Xu, "What are textons?" *International Journal of Computer Vision*, vol. 62, no. 1-2, pp. 121–143, 2005. 1

[2] O. Barinova, V. S. Lempitsky, and P. Kohli, "On detection of multiple object instances using hough transforms," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 34, no. 9, pp. 1773–1784, 2012. 1, 2

[3] L. Dong, V. Parameswaran, V. Ramesh, and I. Zoghlami, "Fast crowd segmentation using shape indexing," in *IEEE International Conference on Computer Vision (ICCV)*, 2007. 1, 2

[4] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 259–289, 2008. 1, 2

[5] B. Wu, R. Nevatia, and Y. Li, "Segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 1, 2

[6] S.-Y. Cho, T. W. S. Chow, and C.-T. Leung, "A neural-based crowd estimation by hybrid global learning algorithm," *IEEE Transactions on Systems, Man, and Cybernetics (Part B: Cybernetics)*, vol. 29, no. 4, pp. 535–541, 1999. 1, 2

[7] D. Kong, D. Gray, and H. Tao, "A viewpoint invariant approach for crowd counting," in *International Conference on Pattern Recognition (ICPR)*, 2006. 1, 2

[8] A. N. Marana, S. A. Velastin, L. F. Costa, and R. A. Lotufo, "Estimation of crowd density using image processing," in *IEEE Colloquium on Image Processing for Security Applications*, 1997. 1, 2

[9] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 1, 2

[10] D. Ryan, S. Denman, C. Fookes, and S. Sridharan, "Crowd counting using multiple local features," in *Digital Image Computing: Techniques and Applications (DICTA)*, 2009. 1, 2

[11] D. S. Cheng, F. Setti, N. Zeni, R. Ferrario, and M. Cristani, "Semantically-driven automatic creation of training sets for object recognition," *Computer Vision and Image Understanding*, vol. 131, pp. 56–71, 2015. 1

[12] X. Zhu, C. Vondrick, D. Ramanan, and C. Fowlkes, "Do we need more training data or better models for object detection?" in *British Machine Vision Conference (BMVC)*. BMVA Press, 2012, pp. 80.1–80.11. 1

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 9, pp. 1627–1645, 2010. 1

[14] V. S. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in Neural Information Processing Systems (NIPS)*, 2010. 1

[15] H. Idrees, I. Saleemi, C. Seibert, and M. Shah, "Multi-source multi-scale counting in extremely dense crowd images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1

[16] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman, "Interactive object counting," in *Proc. of the European Conference on Computer Vision (ECCV)*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, vol. 8691, pp. 504–518. 1, 2, 5, 7

[17] Y. Cai and G. Baciu, "Detecting, grouping, and structure inference for invariant repetitive patterns in images," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2343–2355, 2013. 1, 2, 7

[18] T. K. Leung and J. Malik, "Detecting, localizing and grouping repeated scene elements from an image," in *IEEE European Conference on Computer Vision (ECCV)*, 1996. 2

[19] J. Hays, M. Leordeanu, A. A. Efros, and Y. Liu, "Discovering texture regularity as a higher-order correspondence problem," in *IEEE European Conference on Computer Vision (ECCV)*, 2006. 2

[20] M. Park, K. Brocklehurst, R. T. Collins, and Y. Liu, "Deformed lattice detection in real-world images using mean-shift belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 31, no. 10, pp. 1804–1816, 2009. 2

[21] F. Schaffalitzky and A. Zisserman, "Geometric grouping of repeated elements within images," in *Shape, Contour and Grouping in Computer Vision*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1999, vol. 1681, pp. 165–181. 2

[22] W.-C. Lin and Y. Liu, "A lattice-based MRF model for dynamic near-regular texture tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 29, no. 5, pp. 777–792, 2007. 2

[23] C. Wu, J.-M. Frahm, and M. Pollefeys, "Detecting large repetitive structures with salient boundaries," in *Computer Vision ECCV*, ser. Lecture Notes in Computer Science, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Springer Berlin Heidelberg, 2010, vol. 6312, pp. 142–155. 2

[24] L. Spinello, R. Triebel, D. Vasquez, K. O. Arras, and R. Siegwart, "Exploiting repetitive object patterns for model compression and completion," in *Computer Vision – ECCV*, ser. Lecture Notes in Computer Science, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Springer Berlin Heidelberg, 2010, vol. 6315, pp. 296–309. 2

[25] Y. Cai and G. Baciu, "Detection of repetitive patterns in near regular texture images," in *IEEE IVMSP Workshop*, 2011. 2

[26] ——, "Higher level segmentation: Detecting and grouping of invariant repetitive patterns," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[27] J. Kwon, K. M. Lee, and F. C. Park, "Visual tracking via geometric particle filtering on the affine group with optimal importance functions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 2

[28] J. Kwon and F. C. Park, "Visual tracking via particle filtering on the affine group," *International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 198–217, 2010. 2

[29] E. Learned-Miller, "Data driven image models through continuous joint alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 28, no. 2, pp. 236–250, 2006. 2

[30] F. Porikli, "Learning on manifolds," in *Structural, Syntactic, and Statistical Pattern Recognition*, ser. Lecture Notes in Computer Science, 2010, vol. 6218, pp. 20–39. 3, 4

[31] A. Vedaldi and S. Soatto, "A complexity-distortion approach to joint pattern alignment," in *Advances in Neural Information Processing Systems (NIPS)*, 2006. 3

[32] D. G. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision (ICCV)*, 1999. 3

[33] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 3

[34] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007. 3

[35] M. Cox, S. Sridharan, S. Lucey, and J. F. Cohn, "Least squares congealing for unsupervised alignment of images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 4

[36] Y. Tong, X. Liu, F. W. Wheeler, and P. H. Tu, "Automatic facial landmark labeling with minimal supervision," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 4

[37] D. Conigliaro, P. Rota, F. Setti, C. Bassetti, N. Conci, N. Sebe, and M. Cristani, "The S-Hock dataset: Analyzing crowds at the stadium," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5, 7

[38] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008. 5

[39] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010. 5

[40] B. Hall, *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*, ser. Graduate Texts in Mathematics. Springer, 2003. 9