Spring Lattice Counting Grids: Scene Recognition Using Deformable Positional Constraints

Alessandro Perina and Nebojsa Jojic

Microsoft Research, Redmond

Abstract. Adopting the Counting Grid (CG) representation [1], the Spring Lattice Counting Grid (SLCG) model uses a grid of feature counts to capture the spatial layout that a variety of images tend to follow. The images are mapped to the counting grid with their features rearranged so as to strike a balance between the mapping quality and the extent of the necessary rearrangement. In particular, the feature sets originating from different image sectors are mapped to different sub-windows in the counting grid in a configuration that is close, but not exactly the same as the configuration of the source sectors. The distribution over deformations of the sector configuration is learnable using a new spring lattice model, while the rearrangement of features within a sector is unconstrained. As a result, the CG model gains a more appropriate level of invariance to realistic image transformations like view point changes, rotations or scales. We tested SLCG on standard scene recognition datasets and on a dataset collected with a wearable camera which recorded the wearer's visual input over three weeks. Our algorithm is capable of correctly classifying the visited locations more than 80% of the time, outperforming previous approaches to visual location recognition. At this level of performance, a variety of real-world applications of wearable cameras become feasible.

1 Introduction

The counting grid model [1,2] is a result of an interesting thought experiment: Imagine a large 2-D grid of features, or rather, a grid of tight distributions over features. Take an arbitrary window of a given size into the grid and tally up the features, thus creating a histogram, or simply, a bag of features. Do this many times. Given all the derived bags, and without knowledge of which parts of the grid they came from, is it possible to reconstruct the original grid, or at least a grid that shares lots of spatial characteristics with the original? In [1,2] it is illustrated that spatial patterns indeed emerge when the process of bag mapping and grid refinement is iterated. This is especially the case in vision applications, as a lot of the variation we see in bags of image-derived features is due to the camera movement, while the traditional componential models focus on mixing sources (e.g. topics in text or different objects in the scene). Motion of the camera results in losing a set of features on one side of the image, and gaining new ones on the other. To model such variation, componential models require a large

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part VI, LNCS 7577, pp. 837–851, 2012.
© Springer-Verlag Berlin Heidelberg 2012

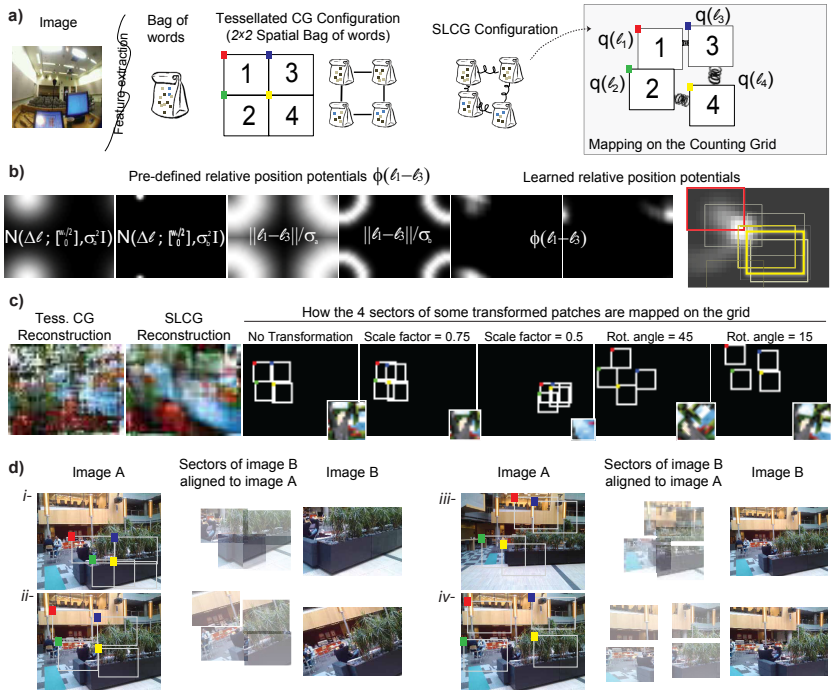

Fig. 1. The spring lattice CG (SLCG) model a), allows for flexible mapping of image sections onto the counting grid. The relative position constraints are captured by manually set or learnable potentials b). Analogous toy data experiment to [2], where the windows sampled from the ground truth tree image were rotated and scaled c). The basic tesselated CG model fails to fit the data well, but the SLCG model achieves a decent reconstruction, due to its ability to approximate the geometric deformations by moving sections wrt each other in the mapping. The learned relative position potential is shown in b). In d) we illustrate the use of the spring lattice in mapping close up images to an image of a broader scene acquired at a larger distance.

number of components, and thus a large training set. But these constraints are naturally captured by the counting grid model, and so it tends to require much less training data and is less prone to local minima problems.

Despite these appealing properties, esp. for vision applications, in their basic form CGs directly capture only translational camera motion, while other geometric transformations in the image have to be captured by feature mapping within the best fitting window, where the distributions will be a bit blurred. And while this blur still helps CGs beat other generative models of bags of words in scene classification on standard datasets, CGs, as well as other bag of words models we have today, are likely to be less of a great fit for "free form" visual streams, such as the ones obtained by a moving robot or a wearable camera [13,4]. This type of imagery is what the biological vision has evolved for, and, arguably, analysis of this type of visual stream is the ultimate goal of the field

of computer vision. Consider a set of images a human (or their wearable camera) will see as the subject is moving through their daily lives. The features will indeed leave the field of view and then be replaced by new features occurring on the opposite side, making CGs a good choice for a basic model, but there will also be a lot of compression of certain areas and expansion of others due to angle of view and scale changes. To better model such effects, we extend the CG model by mapping sections of the image, and allowing them to be mapped semi-independently to the counting grid as illustrated in Fig. 1a. The dependencies in mapping are modeled by the novel lattice of adaptive springs model, which has applications beyond CG mapping, as discussed in Sec. 2. The resulting *Spring Lattice Counting Grid* (SLCG) model was tested here on standard scenes datasets, and most importantly, on a wearable camera dataset consisting of 3 weeks of a subject's life [4], demonstrating that it is indeed possible to analyze realistic visual streams with numerous practical applications.

Related Work in Scene classification. Previous probabilistic approaches to scene modeling differ in their treatment of spatial arrangement of image features. In bag of words models, spatial relationships of features are ignored in a way that leads to computational efficiency and high level of generalization. Examples include topics models [5], which represent each feature bag as a mixture of topics, and Counting Grids [1], described above. As only the feature counts, not the coordinates they come from, are modeled, these models are invariant to feature rearrangements in image. To capture some spatial constraints, it is possible to model separately the bag of words originating in different (predefined) regions of the image. Examples include the *tessellated Counting Grids* [2], the Spatial Bow (*SpaBow*) model [19], and spatial pyramid matching [9]. In these approaches, feature rearrangement is tolerated within each region, but the regions themselves cannot move, thus requiring the modeled images to be approximately aligned. Recent approaches that relax this assumption are [19,20]. The former, *RecBow* [19], represents a scene as a collection of parts arranged in a reconfigurable pattern. Each image is divided into pre-defined regions and a latent variable specifies which "region model" (e.g., sky, grass...) is assigned to each image region. In contrast [20] represents scenes using deformable parts: A lower-resolution root filter is placed in the center of the image and a set of higher-resolution part filters arranged in a flexible spatial configuration.

The Spring Lattice Counting Grid model lies at the intersection of [19],[20] and [1]. As in [19] images are partitioned in pre-defined sectors, which, as in [1,2] are then mapped on bigger space (the counting grid), to get a "panoramic" reconstruction (see Fig. 1a, Fig. 4b). As in [20], the sectors are mapped so that they can move with respect to each other, thus capturing, to some extent, the geometric distortions of images which should otherwise be mappable to a 2D grid like illustrated in Fig. 1d or in Fig. 4b. However, the deformation model is learnable and more general then in previous approaches, and allows for efficient position constraint propagation, as discussed in the next section.

Summarizing, in this paper, we make three important contributions to the field: *i)* we introduce a novel generative model, the spring lattice counting grids *ii)* we introduce an efficient approach to propagating relative position constraints that allows for more general position constraints than the ones previously used in computer vision and *iii)* we show how the combination of the counting grid representation and relative position map inference leads to flexible models that can deal with "All-I-Have-Seen" type data [4], consisting of images acquired by wearable cameras in completely unconstrained settings.

2 Lattice of Adaptive Nonlinear Springs

Modeling relative position constraints for image parts is not a novel notion. In the constellation models [15] the locations of different parts are jointly Gaussian. On the other hand, constraints on feature/part locations are also captured using Gaussian graphical models, most often in the form of Markov random fields in which nearby locations are connected through potentials [16,17,21]. As the latter is a more general model, we start the discussion with a piece of a Gaussian graphical model that captures two 2-D position variables ℓ_1 and ℓ_2 and their relative position potential, $f_1(\ell_1)f_2(\ell_2)\phi(\ell_1,\ell_2)$.

Functions f denote the quality of the fit of each location, while the pairwise potential $\phi(\ell_1,\ell_2)$ captures the relative location constraints. If the total number of possible locations is N, then storing ϕ as a lookup table would require storage of N^2 elements, unless ϕ is parameterized as a real-valued distribution. In most applications, e.g. [18], the latter approach is adopted and ϕ is a Gaussian distribution over the difference between position vectors. This can be thought of as having locations connected with linear springs, possibly in an anisotropic way, as the potential is the function of the square of the distance, or more generally, a quadratic form with the 2-D difference vector $\ell_1-\ell_2$ as its argument. Here we use just the fact that the potential can be written in the form $\phi(\ell_1-\ell_2)$ (Fig. 1b), which means that the updates, in either variational or loopy belief propagation inference, reduce to computing convolutions. For example, the message passing algorithm would need to pass the following message from node ℓ_2 to node ℓ_1:

$$m_{\ell_2\to\ell_1}(\ell_1)=\sum_{\ell_2} m(\ell_2)\phi(\ell_1-\ell_2), \tag{1}$$

where $m(\ell_2)$ represents the product of all messages coming into node ℓ_2 from nodes other than ℓ_1, including the term $f_2(\ell_2)$, the message from the observation node. On the other hand, ignoring links to variables other than ℓ_2, the variational update for the approximate distribution over ℓ_1 has the form,

$$q_1(\ell_1)\propto exp\left(\log f_1(\ell_1)+\sum_{\ell_2} q_2(\ell_2)\log\phi(\ell_1-\ell_2)\right) \tag{2}$$

where q denotes the appropriate variational (approximate posterior) distribution. Thus, in both belief propagation and variational inference, summation over one

variable reduces to convolution of the belief m, or posterior q,with either the potential $\phi(\Delta\ell)$ or the logarithm of it. Thus while the discrete treatment of the general pairwise potential $\phi(\ell_1, \ell_2)$ requires at least $O(N^2)$ operations, dealing with $\phi(\Delta\ell)$ requires $O(N \log N)$ operations, as convolutions can be computed using FFTs. Discrete representation of $\phi(\Delta\ell)$ has three advantages:

1. *Computational efficiency*: The $O(N \log N)$ complexity is not a major stumbling block given that treating location as a real-valued variable may still lead to significant computational cost. In case of loopy BP, passing messages in form of mixtures of Gaussians requires reparameterization to avoid combinatorial explosion[16], and this step is of order $O(M^2)$, where M is the number of particles (typically around 100). In some cases, using the discrete representation may in fact be faster[1].
2. *Representational flexibility*: Treating ϕ as a lookup table over all possible 2-D location differences $\Delta\ell$ does away with the need for assumptions on the parametric form of ϕ. Fig. 1b illustrates various types of relative position constraints that can be enforced, some of them very non-Gaussian.
3. *Adaptation of the relative position potential*: Finally, the potential ϕ can be fit to the data. For example, variational EM algorithm would iterate the q update with an update on ϕ.

$$\phi(\Delta\ell) \propto \sum_t \sum_{\ell_1} q_1^t(\ell_1) q_2^t(\ell_1 + \Delta\ell), \tag{3}$$

which, as a correlation computation, can also be computed in $O(N \log N)$ time. In Fig. 1b, we show an example of the learned relative position potential for neighboring image quadrants estimated from a dataset of windows into an image, subjected to various scaling and rotational deformations.

Therefore, if we connect a set of locations $\ell_1, \ell_2, ..., \ell_S$ with some pairwise potentials acting like springs that keep (probabilistically) these locations at certain relative positions, we get a spring lattice

$$\prod_{(i,j)\in\mathcal{C}} \phi_{i,j}(\ell_i - \ell_j), \tag{4}$$

where $\mathcal{C} \subset [1..S]^2$ is the set of constrained pairs. The spring lattice potential can multiply any graphical model's joint probability (or density function) to enforce relative position constraints. As discussed above, due to their discrimination these relative position constraints come from a much wider set of functions than Gaussian potentials, and can in fact be estimated from the data so as to best fit the application (see Fig. 1b). Thus we refer to them as the lattice of adaptive springs in this paper. This idea can be directly applied to previously studied models, such as constellation models [15] or the MRFs over feature/part locations

[1] This applies only to modeling locations, or other variables where the use of a distribution over the variable difference is suitable; nonparametric BP may have an advantage elsewhere

[16,17]. The locations may represent positions of various kinds, e.g., locations in the image, or in a 3-D space (with use of 3D FFTs), or, as in this paper, a mapping to a derived representations, such as epitomes [7,4] or counting grids [1,2].

3 The Spring Lattice Counting Grid

We extend here the basic 2-D CG model by allowing sections of the image to be mapped semi-independently to the counting grid. The dependencies in mapping are modeled by the lattice of adaptive springs model, discussed above.

3.1 The Basic Counting Grid Model

Formally, the basic 2-D counting grid $\pi_{\mathbf{i},z}$ is a set of normalized counts of words/features indexed by z on the 2-dimensional discrete grid indexed by $\mathbf{i} = (i_x, i_y)$ where each $i_d \in [1 \dots E_d]$ and $\mathbf{E} = (E_x, E_y)$ describes the extent of the counting grid. Since π is a grid of distributions, $\sum_z \pi_{\mathbf{i},z} = 1$ everywhere on the grid. A given bag of words/features, represented by counts $\{c_z\}$ is assumed to follow a count distribution found somewhere in the counting grid. In particular, using windows of dimensions $\mathbf{W} = (W_x, W_y)$, each bag can be generated by first averaging all counts in the hypercube window $W_{\mathbf{k}}$ starting at 2-dimensional grid location $\mathbf{k}$ and extending in each direction d by W_d grid positions to form the histogram $h_{\mathbf{k},z} = \frac{1}{\prod_d W_d} \sum_{\mathbf{i} \in W_{\mathbf{k}}} \pi_{\mathbf{i},z}$, and then generating a set of features in the bag. In other words, the position of the window $\mathbf{k}$ in the grid is a latent variable given which the probability of the bag of features $\{c_z\}$ is

$$p(\{c_z\}|\mathbf{k}) = \prod_z (h_{\mathbf{k},z})^{c_z} = \frac{1}{\prod_d W_d} \prod_z \Big(\sum_{\mathbf{i} \in W_{\mathbf{k}}} \pi_{\mathbf{i},z} \Big)^{c_z}, \tag{5}$$

Relaxing the terminology, $\mathbf{E}$ and $\mathbf{W}$ are referred to as, respectively, the counting grid and the window size. The ratio of the window volumes, κ is called the capacity of the model in terms of an *equivalent number of topics*, as this is how many non-overlapping windows can be fit onto the grid. Finally, with $W_{\mathbf{k}}$ we indicate the particular window placed at location $\mathbf{k}$. To make this less abstract, we can keep an example of the data and the grid in mind. A set of images, of resolution, say 320×240 capture various narrow fields of view into a larger scene. Each image is turned into a matrix of features, typically with a stride of 8 pixels yielding a 40×30 matrix. The features could be color, SIFT, GIST, or something else entirely. The spatial arrangement of the features is dropped and the 1200 features from each image in this example form a disorder bag of features with counts $\{c_z\}$. We map these onto a CG of larger size $\mathbf{E}$, and using the mapping window of size $\mathbf{W}$, controlling in this way the possible extent of bag overlap. One natural choice in this example would be $\mathbf{W} = (40, 30)$ since the features indeed came from a window of that size, and the counting grid could be, say $\kappa = 4$ times bigger, e.g. $\mathbf{E} = (160, 120)$. The distributions in each location of the counting grid are estimated so as to fit the bags as well as

possible, and in the process, some of the lost spatial configuration of the features in 40×30 miraculously emerges in the counting grid, and if we are lucky, the entire broad panoramic scene is reconstructed. But the exact reconstruction is rarely the goal: In recognition tasks, it is usually enough to reconstruct enough local spatial patterns to properly constrain the variation over the observed bags of features, thus making π a good probability model for recognition tasks. For details of the learning algorithm and on its (linear) computational cost see [1,2]. In summary, the E-step aligns all the bags of features to to the grid, inferring $q^t_{\mathbf{k}} \propto \exp \sum_z c^t_z \cdot \log h_{\mathbf{k},z}$, based on the KL distance between the bag's and CG window's feature counts. In the M-step the features from each training bag are laid out in the mapped window $\mathbf{k}$ of π so as to best match the distribution that is already there ($h_{\mathbf{k}}$), and the counting grid is reestimated by summing up the mapped features and normalizing,

$$\hat{\pi}_{\mathbf{i},z} \propto \pi_{\mathbf{i},z} \cdot \sum_t c^t_z \sum_{\mathbf{k}|\mathbf{i}\in W_{\mathbf{k}}} \frac{q^t_{\mathbf{k}}}{h_{\mathbf{k},z}}. \tag{6}$$

3.2 Mapping Image Sections with Spring Lattice Constraints

In [1,2], the CG model is mostly considered as a good density model for classification task, but in [2], the authors also studied the original thought experiment: Starting from some ground truth CG (in their work, the "Tree image", can it be reconstructed using only the observed bags? It turns out that the EM algorithm gets stuck in local minima so that the estimated π exhibits many of the local spatial patterns of the original, thus being a good density model, but fails to find the globally correct reconstruction. However, if the bags of features come broken into $B_x \times B_y$ sectional bags, then the EM does discover globally fairly accurate CG with as coarse an image tessellation as 2×2.

Here we use similar coarse tessellation of input images, but not with the sole goal of avoiding local minima or enforcing coarse spatial layout. Instead, we also wish to capture image deformation beyond simple translational motion of the camera (Fig. 1d). In [2], the sections are mapped to CG in the exact same spatial configuration they have in the image, e.g., the second quadrant should be mapped below the first quadrant, at a distance exactly half the window size $\frac{W_y}{2}$ (we will refer to this model as Tessellated CG). Instead, we allow the different sections to map anywhere on the CG, but impose soft constraints on relative positions of the CG locations to keep the sections loosely grouped. In this way, some stretching of the mapped locations away from each other would model approximate scaling of the images, and a variety of other geometric deformations can be captured as well. In Fig. 1c we illustrate this by extracting, and then rotating and/or scaling patches from the "Tree" image[2] [2], and learning a Spring Lattice CG. The learned ϕ potentials reflect the transformational tendencies in the data.

Adding the spring lattice to the counting grid is straightforward. The feature bags in S different image sections, indexed by s are denoted by $p(c_{z,s}|\mathbf{k}_s)$ and

[2] Matlab load trees

play the role of the observation fit function f in the previous section. Together with the spring lattice distribution over sectional (4) the joint distribution over the image sections becomes,

$$\left(\prod_s p(c_{z,s}|\mathbf{k}_s)p(\mathbf{k}_s)\right)\left(\prod_{(i,j)\in\mathcal{C}} \phi_{i,j}(\mathbf{k}_i - \mathbf{k}_j)\right), \tag{7}$$

where $p(\mathbf{k}_s)$ can again be omitted if the prior is kept fixed to uniform, and $\mathcal{C} \subset [1..S]^2$ is the set of constrained section pairs we choose to model. When these constraints are tree structured, then the above is a normalized probability distribution, as long as potentials ϕ are normalized. Otherwise, the above is an unnormalized factor graph (or an MRF). Either way, summing over the hidden variables $\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_S$ and taking the logarithm to evaluate the probability of the observed set of S sectional bags for one image can be performed in similar fashion as in [1], by introducing the posterior distributions $q(\mathbf{k}_s)$, and approximating the true posterior as $Q(\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_S) = \prod_s q(\mathbf{k}_s)$. With this approximation, the bound on the log likelihood of the t-th training image broken into sections indexed by s becomes,

$$B_t = \sum_{s,\mathbf{k}_s^t} q_{\mathbf{k}_s^t} \log q_{\mathbf{k}_s^t} + \sum_{s,\mathbf{k}_s^t} q_{\mathbf{k}_s^t} \sum_z c_{z,s}^t \log h_{\mathbf{k}_s^t,z} + \sum_{(i,j)\in\mathcal{C}} \sum_{\mathbf{k}_i^t,\mathbf{k}_j^t} q_{\mathbf{k}_i^t} q_{\mathbf{k}_j^t} \log \phi_{i,j}(\mathbf{k}_i^t - \mathbf{k}_j^t)$$

where, as before, $h_{\mathbf{k}_s^t,z} = \sum_{\mathbf{i}\in W_{\mathbf{k}^t}} \pi_{\mathbf{i},z}$. Optimizing for variational posterior over a single section's location, we get

$$q_{\mathbf{k}_s^t} \propto \exp\left(\sum_z c_{z,s}^t \log h_{\mathbf{k}_s^t,z} + \sum_{j|(s,j)\in\mathcal{C}} \sum_{\mathbf{k}_j^t} q_{\mathbf{k}_j^t} \log \phi_{s,j}(\mathbf{k}_s^t - \mathbf{k}_j^t)\right), \tag{8}$$

where we assume that potentials $\phi_{i,j}$ are all flipped if necessary to make section under consideration be the first in the nomenclature. As discussed in Section 2, the computations in the second term are convolutions and can be dealt with efficiently in the FFT domain, as long as ϕ's are treated as discrete maps of the size of the CG. The bound can be computed using estimated q distribution with the same order of complexity and used as a proxy for the likelihood under the model in classification algorithms. Considering a set of training examples indexed by t, we can also analyze the bound on joint likelihood of the training set, $B = \sum_t B_t$, and find the parameters π and ϕ that best fit the data. The counting grid estimate π is computed as in [1], and optimizing the bound B wrt relative position potentials and under the normalization constraints that these potentials have to sum to one, we get

$$\phi_{i,j}(\Delta\mathbf{k}) \propto \sum_t \sum_{\mathbf{k}} q_j^t(\mathbf{k}) q_i^t(\mathbf{k} + \Delta\mathbf{k}), \tag{9}$$

which as a set of cross-correlations is of the same complexity as the E step.

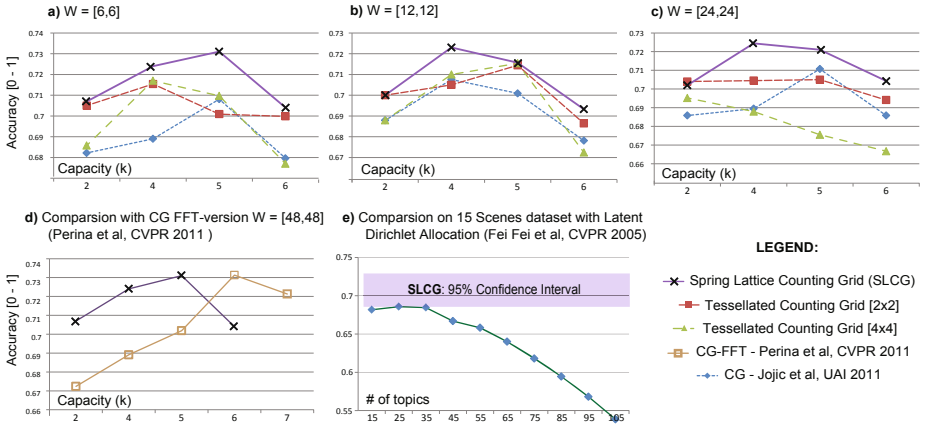

Fig. 2. Scene Classification Results

4 Experiments

In the following experiments, we used SIFT features [10], extracted from 16×16 patches spaced 4 pixels apart, clustered in Z=200 visual words or GIST descriptors [8]. The image resolution was typically 640×480. Except where explicitly indicated, we randomly picked half of images for training and used the remaining half for testing. The classification rates were computed over 3 random train/test splits. In most datasets, the series of 1-vs-All tests indicated that the 2×2 tessellation is statistically better than 3×2, 3×3, 4×3, 1×2 and 2×1 with p-values lower than 0.01. The 2×3 did not differ statistically significantly from 2×2 case. In general this will depend on the amount of training data (very low in our experiments) and the feature extraction density.

4.1 Scene Classification

We performed classification on the standard Scene Dataset presented in [5,9] and the most recent indoor scene recognition dataset [12]. We compared with previous work on Counting Grids [1,2] and Lda [5]. The Counting Grids evaluated here used window sizes $\mathbf{W}$ ranging from 6×6 to 24×24, with capacities (ratios of the grid size to the window size) $\kappa \in \{2, 3, 5, 6\}$.

In Fig. 2a-c we report comparisons with [1] and with the tessellated version of basic CG (2×2 and 4×4) varying the window size: spring lattice CG outperforms both with a large margin, for every choice of $\mathbf{W}$ and k. Accuracy drops for all methods when $k \leq 9$.

In Fig. 2d we compare SLCG with the FFT-version of [2], which uses the CG step in inference, but an epitome M step in feature mapping. The results are comparable, but note that the FFT-CG [2] is forced to use a window size as big as the "feature image" (48x48 in this case), making the approach 50-100 times slower than our method. On the same dataset, in Fig. 2e we compared

SLCG with the supervised Latent Dirichlet Allocation (LDA) version of [5]. The complexities of the two models are not directly comparable, and so we report the accuracies of [5] across a wide choice of number of topics *vs* the 95% confidence interval containing all the accuracies obtained by SLCG, i.e., all the SLCG curves from (Fig. 2a-c). For all tested choices of $\mathbf{W}$ and κ, our model outperforms the optimal complexity of LDA.

Finally we also ran our algorithm on the 67 categories of the indoor scene classification dataset on the standard partition [12]. We tested SLCG with $\mathbf{W} = [12, 12]$, and k=5 and obtained classification rate of 27,1%, outperforming [12] (no annotation) and other generative approaches tested on this dataset (LDA [5], the most of the complex LDA variations of [11]) and nearly matching the performance of generative RecBow [19].

We obtained a further boost in classification when we used the latent structure in an additional discriminative training step [6]: We simply described each sample by a vector of its generative class-likelihoods and trained SVMs on this representation. In this SLCG-derived features lead to a new state of the art on Torralba artificial images (95,3%) (a subset of the 15 scenes dataset), while the results on other datasets (92,3% on Torralba's Natural Images, 80,1% on [5,9]) where very close to the state of the art.

4.2 SenseCam Dataset

In this section we introduce the new set of 32 labeled classes of images acquired with SenseCam and we compare our approach with the state of the art on scene/place recognition and image clustering[3]

SenseCam dataset : In 2009 we recorded 21 days-worth of images with a SenseCam, a wearable camera which automatically shoots a photo every 20 secs [4]. This collection of images has been shared along with a very reduced number of labeled images. These images were chosen so as to represent some 50-80 % of imagery acquired by the camera on an average, non-eventful day when the subject was mostly sedentary and so most of the within class variation is in illumination and more or less translational camera motion. However, the subject's life does contain a variety of other events and scenes that are fairly hard to model. Starting from this dataset, we labeled a random 10% of the whole corpus. We highlighted 32 recurrent place/scenes in the subject's life. Some images for each class are shown in Fig. 3. This new dataset is much more challenging than one presented in [4] as the labeled images are a random subset of the whole visual input of a subject, and therefore images present dramatic viewing angle, scale and illumination variations.

Place classification : We compare our method with [5,3,13,4,1,2,9,20,19]. Since some categories have much more images than others, we used 15 images per

[3] The dataset is available upon request,
`http://research.microsoft.com/en-us/people/alperina/`

Fig. 3. From top left to bottom right, we show three images from each class of *SenseCam-32*. The 32 classes are: *Bathroom Home, Bedroom, Biking, Cafeteria, Car, Classroom, Conference Room, Corridors Work, Dining Room, Bakery, Garage, Atrium, Entry, Hiking Trail, Ice Palace, Kids Bedroom, Game Room, Kitchen, Living Room, Lounge, Home Office, Campus, Parking Work, Patio, Playground, Restroom Work, Small Bathroom Home, Small Home Office, Tennis Court, Food Court, Grocery Store, Work Office*.

Table 1. Place Classification on SenseCam-32. For [5,3] we reported the best result varying the number of topics from 5 to 125, for [19,9,20] we obtained the code from the authors and we run it using 2 components for [20], and 2 levels for [9]. In the table, "Stel Ep." is the stel epitome, "MoD" stands for Mixture of Dirichlet distribution.

Discriminative		Generative		Eptiomes		CGs / Tess.(2×2) CGs		**SLCGs**	
Method	Acc.	Method	Acc.	Method	Acc.	Method	Acc.	Method	Acc.
SVM lin.	49,94%	MoD [13]	49,19%	Stel Ep. [4]	15,2%	W=[6,6]	50,93%	W=[6,6]	54,55%
[3]	48,65%	RecBow [19]	57,47%	CG-fft [2]	39,4%	W=[12,12]	53,4%	W=[12,12]	**60,12%**
DPM [20]	43,65%	LDA [5]	58,05%			Tess. W=[6,6]	53,43%	W=[18,18]	58,64%
SPK [9]	52,76%	SpaBow [19]	44,11%			Tess. W=[12,12]	54,43%	W=[24,24]	56,73%

class for training and at most 15 images of each class as test set. Fig. 4a shows that despite the reduced number of images available, learning the ϕ potential helped recognition with SLCG as each place is characterized by different set of viewpoint changes and other transformations. For example, in an office a strong potential that keeps the four sectors is expected, as most of the variation in mapping can be modeled with translation motion, while in a corridor we expect a variation in scale. Images taken in hard classes like kitchen, are characterized by so many viewpoints and object movement that nearly flat potentials are learned (i.e., the four sectors quite free to move in the grid). In Tab. 1 we compare the recognition accuracies of several methods; the lattice of adaptive springs outperforms the competitors with large margin. The poor accuracies of discriminative methods [9,3,20], are clearly do to overtraining but with this type of datasets we must expect scarce labeled data. Interestingly, the methods based on pixelwise comparisons, like the epitomes [4], fail on this extended dataset. Being a hybrid between epitomes and counting grids, [2] reaches decent results, but it also finds it hard to generalize with so little labeled data.

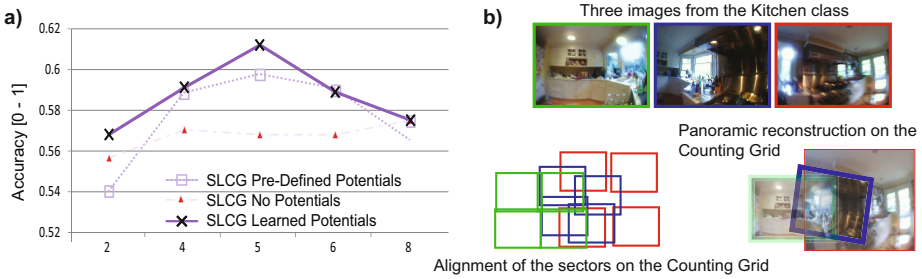

Fig. 4. a) Classification results on SenseCam-32 for three choices of the positional constraints. b) Some exemplars from the Kitchen class and SLCG modeling. Image sectors on the left-bottom figure have been placed by hand in agreement with the mapping on a counting grid of the relative sectors.

We can also draw some conclusions on which types of scenes are most appropriately modeled using different approaches to spatial reasoning. Bag-of-words models [5,13,1], which allow full rearrangement of features within an image, are robust to large amount of clutter and extreme geometric transformations. Method based on a fixed partition, like SpaBow, and [9,20] encounter problems when images are not aligned. In particular in [20] (DPM) it is assumed that a scene may be described by a constellation model with a fixed "root" encompassing the salient part of a scene (e.g., the kitchen range for a kitchen scene, the altar for a church) and moveable regions of interest. While this assumption is satisfied for some scenes, it does not work well for places where at least some alignment is necessary. A DPM with multiple components becomes better able to deal with intra-class variability, but it still fails to capture non-aligned images from a free-form stream captured by a wearable camera. Counting Grids map bags onto a panorama with larger real-estate, exploiting the fact that bags come from bigger scenes, thus better fitting the data in the Sensecam dataset. However, the basic version of [1] discards all the spatial information within the mapped window, while the tessellated version is too rigid in following the exact sector configuration. The SLCG is able to recover better panoramas and it often captures typical 3D transformations of dataset acquired with a wearable camera (see Fig. 4b). Finally, RecBow, [19] can model SenseCam images as each region is free to model each "concept" regardless of the position, but the model does not explicitly recover or exploit their panoramic nature.

Day images classification using SLCG HMMs : We manually labeled the images from 2 days (1800 images ca. each day) and we asked how many of them we could correctly guess using the SLCG trained above and an hidden Markov model (hmm) over possible classes, capturing transition constraints such as living room - kitchen, or office - corridor - atrium, etc.

During each day, the camera bearer visited roughly 20 of the labeled locations and the two days share only 12 locations. Nevertheless we trained models with all the 32 classes as a-priori we cannot know the locations visited during a day.

Table 2. Classification of all the places visited in a day. *SenseCam Day-1/2*. We wrote ~~hmm~~ when we turned off the hidden Markov model.

Mixture of Dirichlet [13]				LDA [14]				**SLCGs**			
Day 1		Day 2		Day 1		Day 2		Day 1		Day 2	
~~hmm~~	hmm	~~hmm~~	hmm	~~hmm~~	hmm	~~hmm~~	hmm	~~hmm~~	hmm	~~hmm~~	hmm
54,68%	70,37%	45,5%	60,79%	n.a.	76,80%	n.a.	67,23%	68,76%	**86,35%**	60,21%	**82,65%**

Our goal is to compute the place posterior probabilities at the instant t, given all the previous images $P(c_t = k|x_{1:t})$. We used the forward-backwards procedure to recursively compute it, in formulae

$$P(c_t = k|x_{1:t}) \propto p(x_t|c_t = k) \sum_c P(c_t = k|c_{t-1} = c) P(c_{t-1}|x_{1:t-1}) \qquad (10)$$

We fixed the observation loglikelihood to the negative free energy given by our model (bound B_t, while we used EM estimate the transition matrix $A_{k|c} = P(c_t = k|c_{t-1} = c)$ and the place (c_t) posteriors Eq.10 in an unsupervised way, simply fitting the likelihood to the day's images. Since this approach is similar to [13], the reference provides a natural point of a comparison. Besides a similar use of the hmm the idea of [13] is to learn a mixture of model for each class to eventually compute the observation likelihood. To use the very same features and spatial information, we learned a Dirichlet mixture model using the feature histograms that come from each of the four sectors of the image.

We used at most 30 images per class to learn the models. For the lattice of adaptive springs we set W=[12,12] and $k = 5$, while for [13] we tried several values for the number of mixture centers, reporting the value that gives the best accuracy. Without using the temporal information, our model correctly classifies the 68,76% of the images while Torralba's method stops at 54,68% (Day 1). As expected, the recognition accuracy rises respectively to 86,35% and 70,37% when we "turn on" the hmm (Eq. 10). Similar results were obtained for the second day (see Tab. 2). We have also implemented the original method of [13] using their descriptors from the whole images and within the four sectors. Their performances were below 50%.

5 Conclusions

We have introduced the Lattice of Adaptive Spring Model, which with its learned relative position potentials seems to model well the geometric transformations present in scenes. Our experiments demonstrated a superior performance over LDA and previous work on CGs. We also set the state of the art on some datasets and are close to it on the others. The Spring Lattice component of the model can be used without CG modeling in all other models that need to constrain the relative location of parts. Another important contribution is the introduction and analysis of a new labeled large dataset which resembles the input to biological vision systems much more than most previously published datasets. The analysis of this dataset lead to several important conclusions. First, it is

possible to disambiguate over dozens of visual scenes (locations) encountered over the course of several weeks of a human life with accuracy of over 80%, and this opens up possibility for numerous novel vision applications, from early detection of dementia to everyday use of wearable camera streams for automatic reminders, and visual stream exchange. Second, our experimental results indicate that, while on average, SLCG is a better fit to this dataset than the componential models, such as LDA, LDA still outperforms other models on some types of scenes, esp. the ones with relatively random clutter and a mix of objects (e.g. kitchen, or a grocery store). CG-derived models have an advantage at explaining sedentary situations or movement through larger scenes (living room, office, corridors, atrium, outdoor categories). Thus, combining the aspects of both models is likely to lead to further increase in classification, as well as models that allow for scene segmentation and object/people or activity recognition.

References

1. Jojic, N., Perina, A.: Multidimensional counting grids: Inferring word order from disordered bags of words. In: UAI 2011, pp. 547–556 (2011)
2. Perina, A., Jojic, N.: Image analysis by counting on a grid. In: CVPR 2011, pp. 1985–1992 (2011)
3. Bosch, A., Zisserman, A., Muñoz, X.: Scene Classification Via pLSA. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part IV. LNCS, vol. 3954, pp. 517–530. Springer, Heidelberg (2006)
4. Jojic, N., Perina, A., Murino, V.: Structural Epitome: a way to summarize one's visual experience. In: NIPS 2010, pp. 1027–1035 (2010)
5. Li, F.-F., Perona, P.: A Bayesian Hierarchical Model for Learning Natural Scene Categories. In: CVPR (2), pp. 524–531 (2005)
6. Perina, A., Cristani, M., Castellani, U., Murino, V., Jojic, N.: Free Energy score space. In: NIPS 2009, pp. 1428–1436 (2009)
7. Jojic, N., Frey, B.J., Kannan, A.: Epitomic analysis of appearance and shape. In: ICCV 2003, pp. 34–43 (2003)
8. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. Int. Jrn. of Computer Vision 42 (2001)
9. Lazebnik, S., Schmid, C., Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In: CVPR (2), pp. 2169–2178 (2006)
10. Lowe, D.: Distinctive Image Features from Scale-Invariant Keypoints. Int. Jrn. of Computer Vision 60 (2004)
11. Zhu, J., Li, L.-J., Li, F.-F., Xing, E.P.: Large Margin Learning of Upstream Scene Understanding Models. In: NIPS 2010, pp. 2586–2594 (2010)
12. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: CVPR 2009, pp. 413–420 (2009)
13. Torralba, A., Murphy, K.P., Freeman, W.T., Rubin, M.A.: Context-based vision system for place and object recognition. In: ICCV 2003, pp. 273–280 (2003)
14. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3(5) (2003)
15. Fergus, R., Perona, P., Zisserman, A.: Object Class Recognition by Unsupervised Scale-Invariant Learning. In: CVPR 2003 (2003)

16. Sudderth, E., Ihlerl, A., Isard, T., Freeman, W., Willsky, A.: Non Parametric Belief Propagation. In: CVPR 2003 (2003)
17. Isard, M., Pampas, M.: Real-Valued Graphical Models for Computer Vision. In: CVPR 2003 (2003)
18. Sudderth, E., Mandel, M., Freeman, W., Willsky, A.: Visual Hand Tracking Using Nonparametric Belief Propagation. In: CVPR 2004 Workshop on Generative Model Based Vision (2004)
19. Parizi, S.N., Oberlin, J., Felzenszwalb, P.F.: Reconfigurable models for scene recognition. In: CVPR 2012 (2012)
20. Pandey, M., Lazebnik, S.: Scene recognition and weakly supervised object localization with deformable part-based models. In: ICCV 2011 (2011)
21. Krahenbuhl, P., Koltun, V.: Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In: NIPS 2011 (2011)