April 2014 Technical Report: KUL/ESAT/PSI/1402

An Elastic Deformation Field Model for Object Detection and Tracking

Marco Pedersoli, Radu Timofte,
Tinne Tuytelaars, Luc Van Gool

PSI

Kasteelpark Arenberg 10 – box 2441
B-3001 Heverlee, Belgium

KU LEUVEN

Telephone: +32-(0)16-32.17.13 Fax: +32-(0)16-32.17.23 http://www.esat.kuleuven.be/psi

An Elastic Deformation Field Model for Object Detection and Tracking

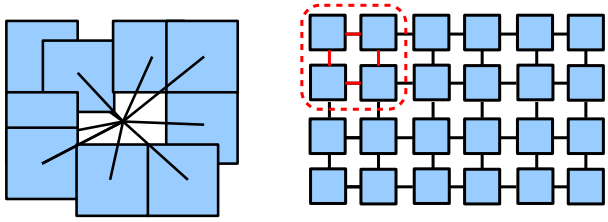
Marco Pedersoli Radu Timofte Tinne Tuytelaars Luc Van Gool

Abstract

Deformable Parts Models (DPM) are the current state-of-the-art for object detection. Nevertheless they seem sub-optimal in the representation of deformations. Object deformations are often continuous and not confined to big parts. Therefore we propose to replace the DPM star model based on big parts by a deformation field. This consists of a grid of small parts connected with pairwise constraints which can better handle continuous deformations. The naive application of this model for object detection would consist of a bounded sliding window approach: for each possible location of the image the best part configuration within a limited bound around this location is found. This is computationally very expensive. Instead, we propose a different inference procedure, where an iterative image-level search finds the best object hypothesis. We show that this approach is faster than bounded sliding windows yet produces comparable accuracy. Experiments further show that the deformation field can better approximate real object deformations and therefore, for certain classes, produces even better detection accuracy than state-of-the-art DPM. Finally, the same approach is adapted to model-free tracking, showing improved accuracy also in this case.

1 Introduction

Current state-of-the-art object detection methods allow for some deformations, in particular the successful deformable parts models (DPM) [1] introduced in [15]. DPM is based on a star model (see Fig. 1a), that connects each part to a center. It imposes a prior on the displacement of a part w.r.t. its resting position or "anchor point" and, given the object location, assumes that parts move independently. This does not really hold for most real objects and only works if parts can be chosen strategically. In fact, in most of the cases the moving parts only approximate an underlying continuous deformation, for instance due to a perspective distortion. To obtain a more precise representation of the object deformation it is then necessary to split the object into many smaller parts. Yet, this would not work in a star model, as small parts would have little discriminative power and consequently little capacity to be appropriately driven away from their anchor point.



(a) (b)

Figure 1: **Star Model vs. Elastic Field.** (a) The Star model should be composed of big parts as they move independently and they can lose their structural meaning; (b) the elastic field can be composed of small parts because the global regularization is enough to maintain the structure coherence. Notice that if some small parts have strong connections (red lines) then they are equivalent to a big part.

Thus, we investigate the use of a different deformation structure when using many small parts: an *elastic deformation field* . This elastic deformation field model (EDFM) imposes a prior on the displacement of each part

[1] In this work when referring to DPM we consider the specific implementation of [15].

w.r.t. its local neighborhood. Each part has pairwise connections with the neighboring parts, that determine the local stiffness of the model (see Fig. 1b). The net displacement of each part now depends on the global configuration of the model. Fig. 1b shows how EDFM can still represent big parts if needed. Setting the pairwise costs between some of the small parts to infinity yields bigger rigid parts. As these costs are learned during training, the model *'knows' where parts should move together and where independently*. The deformation field can represent a wide range of deformations, like those represented in Fig. 3.

An intuition of the difference between the star model and the deformation field when using many small parts is given in Fig. 2. For a uniform stretching, the deformation field has a uniform distribution of the deformation over the parts (Fig. 2d). In contrast, in the star model (Fig. 2c), parts that are far from the object center should take into account the deformations of the previous parts, such that the pairwise displacement is higher. Now, considering that in our models the deformation costs are learned during training, in order to be able to represent the same samples, the higher pairwise displacement of the star model induces a lower deformation cost. Thus, this implies a looser model that is more prone to false positive detections.

To apply this model for the problem of object detection a standard solution would consist in using a bounded *sliding window approach*. That is, consider each possible location of the image and for each one find the best part configuration within a bounded region. In practice, as parts are connected with a loopy graph for an EDFM, for each location a CRF optimization over the parts locations is needed. This is too expensive for practical use. For instance, in [28] a similar deformation field was studied, but locally affine constraints had to be introduced to the object part locations to simplify the inference and make the model computationally feasible.

In contrast, here we propose to use a "real" loopy graph CRF model where each part (node) can be placed virtually in any possible location of the image (labels). Instead of using a sliding window procedure, we consider an *iterative image-level approach* where we repeatedly search for the best parts placement on the entire image. As our CRF optimization is based on alpha expansion, its computational cost is linear in the number of part locations. Under these conditions we show that the iterative image-level approach is much faster and more flexible than bounded sliding window and, equally important, does not compromise the detection quality.

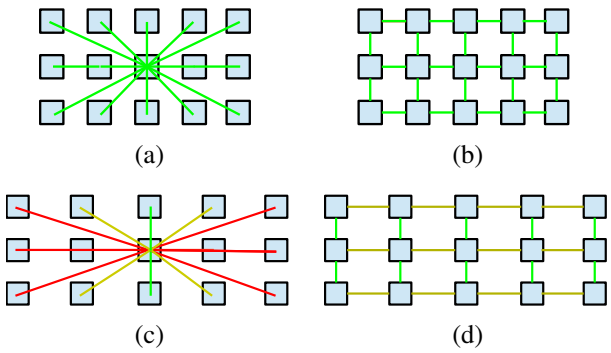


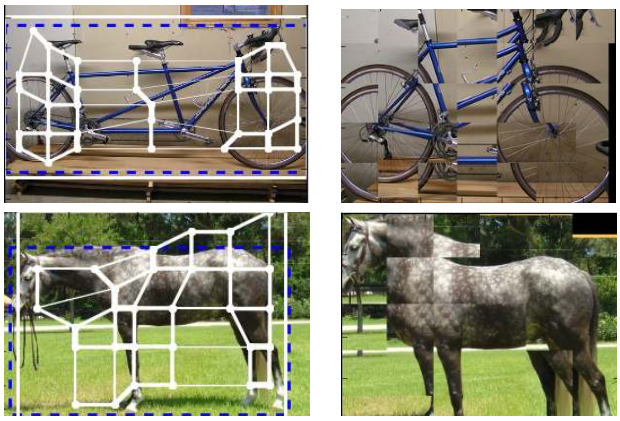
Figure 2: **Cost of Deformation.** We compare the cost of a horizontal uniform stretching in case of a star model and an elastic field with the same number and location of parts. Colors of the pair-wise connections represent the amount of deformation. (a) Star model at rest. (b) Deformable field at rest. (c) Star model with deformation; parts that are further from the center have a larger displacement. (d) Deformable field with deformation; the deformation cost is distributed uniformly over the edges. In general, for any kind of perspective deformation a similar behavior is expected.

To summarize, the main contributions of this paper are the following:

1. we propose the use of an elastic deformation field model for object detection, where parts are small and connected with a loopy graph which can better represent typical object deformations;

2. we show that by using a image-level optimization strategy the new deformation model, even if more complex than the commonly used star model, can be used for detection with a comparable computation time;

3. we experimentally prove that our deformation field in most of the cases outperforms the star model; and



(a) (b)

Figure 3: **Elastic Deformation Field.** (a) Detection, parts center placement and pairwise distortion; (b) Reconstructed object. As the elastic deformation field encodes deformations locally, it is also able to explain unusual and highly deformed object configurations.

4. we improve model-free tracking with the use of our deformation field.

The paper is structured as follows. In section 2 we discuss how our work relates to various topics of computer vision. Then, in section 3, we define our model and introduce a fast inference for it. Finally, section 4 reports on experiments comparing our model with the state of the art, while in section 5 we draw conclusions.

2 Related work

The idea of using an elastic deformation field is not new in vision. In this section we first compare our method with other deformable detectors and second, we give a broader overview of different methods that apply similar deformation models to different tasks of computer vision. Finally we also present the related work for the application of our deformable model to tracking.

Deformable Parts Models. We consider EDFM as a natural evolution out of DPM. Besides the original DPM of Felzenszwalb et al. [17], there are multiple refinements [15, 33, 39]. Vedaldi and Zisserman [33] proposed an alternative representation as a predefined set of part configurations that can yield global affine deformations. However deformations are often local, so global transformations may be less effective than the more general DPM. Another interesting extension of DPMs is presented in [39], where the star model is extended to a real tree with a deeper structure. This hierarchical structure can better approximate real object deformations because nearby parts have the same parent and follow its displacements. Yet, this structure has previously been used for image restoration and optical flow, and there it is known to produce artifacts due to the spatially varying neighborhood structure. That is, inside the 'big' parts there tends to be over-smoothing, whereas their boundary regions tend to be under-smoothed. A partial remedy is proposed in [30], where the hierarchy of parts is enriched with local loops of pairwise connections that dynamic programming can solve efficiently. Although locally more consistent, there are still disconnected parts that move independently. Finally, if the deformation structure of the model is known *a priori* (e.g. for human detection), a simpler tree model can suffice [36]. However, we deal with the more general case where the deformation structure should also be learned. The deformable model work most similar to ours is probably that of Ladicky et al. [28], who propose a locally affine deformation field to detect humans. After introducing a complete CRF model for deformation, they claim that it is too deformable and a per instance tuning of the pairwise costs would be necessary. We show that, by learning the pairwise costs during training a complete CRF model can be used and yields better accuracy (see section 4). The authors of [28] also claim that a CRF optimization sliding window style is computationally too expensive. We came to the same conclusion, but propose an alternative that obtains the same detection accuracy without sliding window and has a much smaller computational cost (see section 4).

Random fields have been widely used for object detection and segmentation. Usually, the MRF/CRF is used to model the compatibility between object parts [20, 23, 31], or to reason about multiple detections and context [11, 27]. In the context of modeling object deformations, [9] uses a generalized star model (k-fans) and shows that for certain classes it can improve the recognition accuracy. However, that model is more limited than our array of parts and has a slower inference. In object

recognition, loopy graphs with pairwise connections similar to ours have been previously used to model the spatial relationship among parts [5, 18, 26]. However, to the best of our knowledge, our work is the first that can use a loopy graph CRF to model object part deformations for object detection, where multiple objects are to be found at multiple scales and locations. Considering the task of detecting multiple objects at multiple scales produces a multiplication of the computational cost such that loopy graphs have so far generally been considered too expensive to use.

Graph matching kernel. Another recent paper with some similarities is [12]. The authors use a deformable model similar to EDFM for object classification. Whereas in EDFM the deformation field is considered a latent variable, they use it to build a kernel that measures a distance between examples. Apart from the fact that this kernel is not positive definite and the problems this can entail, the approach is computationally costly. To evaluate an example one has to evaluate the kernel with every learned support vector (possibly thousands). This is feasible for classification, where the number of samples is low, but not for detection where for each training image there can be many samples to evaluate. Instead of evaluating each sample with the learned support vectors, our EDFM compares each sample with the learned model. This is thousands of times faster and can therefore also be used for detection. Another advantage of EDFM is that pairwise costs can be learned, which is difficult when using a kernel.

Image registration. Image registration or alignment has also some similarities with our approach. In [19] image registration is accomplished with similar pairwise deformation constraints and an inference based on a generalization of alpha expansion to the primal-dual formulation [25]. Even if the inference is quite similar to ours, the final task of image registration is to find a one to one correspondence between images. In contrast our task is to find a class model that is able to recognize and align possibly multiple objects in an image at any possible location and scale. Also in this case the order of the computational costs is very different because for detection we need to align every possible location and scale to a model, which makes a direct use of a CRF in a sliding window manner computationally challenging, as will be shown in the experimental results.

Optical flow. There is an interesting analogy between optical flow techniques and our adaptation of DPM. The Lucas-Kanade extraction of optical flow [29] splits the initial image into patches and finds for each the displacement that minimizes the least squares distance to the new image. This is quite similar to the DPM approach [15], where object parts are searched independently to best fit the object model. In contrast, Horn and Schunck [21] use a global approach, enforcing optical flow smoothness to solve the aperture problem. This is similar to what we do in EDFM, where we enforce global coherence on part displacements. As a matter of fact, the global smoothness of the optical flow often is too strong assumption, since the objects in a scene can move independently. For object deformation this assumption makes more sense, especially as we can learn the pairwise constraints of the field.

Model-free tracking. In model-free tracking, the only information about the object to track is the location (generally a bounding box) of the object in the first frame of the video. The most successful methods learn a discriminative on-line classifier that separates the object from the background. The classifier is updated to adapt to the local changes of the scene. Examples of this scheme are [3, 22]. However, updating the classifier is not always the best strategy because it can introduce a drift in the model representation. Instead, by modeling the variations that the appearance of the object can assume, for instance by considering an invariant or co-variant representation, can reduce this drifting. This idea has been widely applied for illumination changes by using features that are invariant to linear or affine light changes, e.g. SIFT. The same concept of invariance can be used also for the object structure. In [38] for instance, the deformable structure of multiple objects or object parts is used to improve the tracking accuracy. We go further in this direction, considering a deformable representation based on the EDFM, which can better represent local and perspective deformations.

3 Elastic Deformation Field Model

We define our deformable model, EDFM, as a set of parts, each with their position (x,y) and scale s, yielding triples in the location space $\mathscr{L} \subset \mathbb{N}^3$. The appearance score A_i generated by the part i at location $l_i = (x,y,s) \in \mathscr{L}$, is

given by:

$$A_i(l_i) = \left\langle w_i^A, H(l_i) \right\rangle, \quad (1)$$

where H is a function that extracts the local features (e.g. HOG features) of the part at l_i and w_i^A is a vector of learned weights. Applying a linear model, we define the total model score as the sum of the N part scores:

$$A(\mathbf{l}) = \sum_{i=1}^{N} A_i(l_i), \quad (2)$$

where $\mathbf{l} = [l_1, l_2, ..., l_N]$. Spatial coherence comes from the assumption that each part location l_i is displaced by i) a fixed displacement a_i relative to an object location p and ii) a further distortion/deformation d_i. This yields $l_i = p + a_i + d_i$, where $p + a_i$ is the anchor position (including the scale) for the part i. Setting the distortions d_i to 0, but varying p, would correspond to sliding around a rigid template (e.g. the Dalal and Triggs detector [10]). Yet, we allow for the distortions as just introduced. In order to avoid parts moving independently – thereby increasing the risk that they do so incoherently – we need to set a cost D on the part distortions from their fixed anchor position a. This then yields the following overall score for a hypothesis on the position of an object and its parts:

$$S(p, \mathbf{l}) = A(\mathbf{l}) - D(\mathbf{l} - \mathbf{p} - \mathbf{a}), \quad (3)$$

where $\mathbf{p} = [p, p, .., p]$, i.e. all parts are displaced w.r.t. the same object center and $\mathbf{a} = [a_1, a_2, ..., a_N]$. The final detection (maximal score) thus yields a trade-off between deformation and appearance.

In **DPM** [15] the prior is set as:

$$D^S(\mathbf{d}) = \sum_{i=1}^{N} \left\langle w_i^D, [d_i, d_i^2] \right\rangle, \quad (4)$$

where w_i^D is a vector of positive weights applied to the absolute values of the displacements $d_i = l_i - p - a_i$ of the part i and its squared values d_i^2. This prior says that the larger a distortion is, the larger its penalty. It limits the distance a part can move away from its anchor, but allows parts to move independently given the object center p. This model works properly when 1) *parts are big*, so that each part's appearance is discriminative enough to not get confused by local distractors (other parts provide no localization support), and/or 2) *part displacements are small* so that their independence is almost true.

Our EDFM aims at large deformations, which can be gradual/non-rigid over the object. Therefore parts need to be small. With those goals in mind, we do not penalize part distortions *per se*, but the differences with the distortions of their neighbors. Thus, the elastic stretching of a pattern comes at a cost, while parts moving together is for free. The new deformation cost is therefore:

$$D^F(\mathbf{d}) = \sum_{(i,j) \in \mathscr{E}} \left\langle w_i^D, [|d_i - d_j|, (d_i - d_j)^2] \right\rangle, \quad (5)$$

where $\mathscr{E}$ are the edges of a graph defining the neighborhood (in our experiments a 4-connected neighborhood as shown in Fig. 1b).

3.1 Inference

In this section we analyze the different methods that are generally used for finding the best "deformable" detection in an image and compare their corresponding costs. Finally we present our approach based on an iterative image-level optimization.

Given an image, we are interested in finding the maximum a posteriori (MAP) solution of the probabilistic problem associated to our scoring function, i.e. the best detection. In DPM the score of a configuration of parts depends on both the position of the object p and the locations $\mathbf{l}$ of its parts. The best detection corresponds to the best scoring location:

$$p^*, \mathbf{l}^* = \arg\max_{p, \mathbf{l}} (A(\mathbf{l}) - D(\mathbf{l} - \mathbf{p} - \mathbf{a})). \quad (6)$$

If the deformation connections form a star, as defined by D^S in Eq.(4), then the problem can be solved using dynamic programming with a computational cost that is $O(N|\mathscr{L}|^2)$, which is in general too expensive.

In [16], the use of particular deformation costs that only depend on the deformation $(\mathbf{l} - \mathbf{p} - \mathbf{a})$ and not on the absolute object location p is promoted such that a *generalized distance transform* can be used. In this case the optimization is first computed at each p and for each part l_i independently with a linear cost and then the best location p is found as the best sum of the parts scores:

$$\arg\max_{p} (\sum_{i=1}^{N} \arg\max_{l_i} (A_i(l_i) - D_i(l_i - p - a_i))). \quad (7)$$

As the distance transform can be computed in linear time, the optimization has a cost that is proportional to $N|\mathscr{L}|$. In contrast to the star model, the deformation field defined in Eq.(5) composes a loopy graph. The solution based on a distance transform can therefore not be applied.

Another way to optimize it is to use a *bounded sliding window* approach. It is based on the assumption that the deformation of the parts is limited and therefore we can search for the best parts configuration in a local region $\mathscr{P}$ around the object center (predefined in [28, 30] or learned in [14]):

$$\arg\max_{p}(\arg\max_{\mathbf{l}\in\mathscr{P}}(A(\mathbf{l})-D(\mathbf{l}-\mathbf{p}-\mathbf{a}))). \tag{8}$$

As the optimization of each local region has a bounded cost $|\mathscr{P}|$, the total computational cost is now $O(N|\mathscr{P}||\mathscr{L}|)$, which is affordable if $|\mathscr{P}|$ is small enough. Unfortunately, as we will show in section 4, this approach still is computationally too expensive when using a deformation field. To reduce this cost, in [28] additional constraints (local affine deformations) are considered. This allows for the use of bounded sliding windows together with a loopy CRF, as the corresponging optimization can be solved with efficient dynamic programming. Instead we want to be able to use the complete deformation capabilities of the EDFM model without limiting them to be locally affine. We therefore use a different approach that allows us to solve "real" CRFs in a faster manner.

Recent combinatorial methods for CRF optimization such as alpha expansion [8] come with guarantees about their optimality and at the same time have a computational cost that is linear with the number of labels. Considering the linear cost of the method, we can compute directly the MAP solution over the entire label space $\mathscr{L}$ as:

$$\arg\max_{\mathbf{l}\in\mathscr{L}}(A(\mathbf{l})-D^{F}(\mathbf{l}-\mathbf{p}-\mathbf{a})). \tag{9}$$

As D^F computes differences of deformations, the solution is independent of p and we do not need to optimize for it. The MAP solution of the *image-level optimization* defined in Eq.(9) has a cost that is proportional to $N|\mathscr{L}|$, while using the bounded sliding window approach has a cost proportional to $N|\mathscr{P}||\mathscr{L}|$. Thus, the computation of the MAP solution is $|\mathscr{P}|$ times faster than the sliding window approach. On the other hand, it provides only the MAP solution whereas a (bounded) sliding window approach provides a solution for each p. Below, we describe how an iterative application of the image-level optimization allows to cope with multiple objects in an image.

3.2 Multiple hypotheses

In detection it is not really necessary to evaluate the score of every possible window location p in the image. What really matters is to find all the objects present in the scene. In this sense, we can reshape our problem as finding the K-best detections in the image. If K is large enough, this can lead to the detection of all the objects in the image. For finding the K-best detections, we add an extra scoring term that penalizes configurations $\mathbf{l}$ with parts placed similarly to a previous detection $\mathbf{l}^*$:

$$\Delta^{g}(l_i,l_i^*)=\begin{cases}-\infty & if \quad |l_i-l_i^*|<\mu_P\\ 0 & otherwise\end{cases} \tag{10}$$

This term forbids part i to be placed closer than μ_P to a location l_i^* that has already been occupied by the same part in a previous detection. Considering now a set of detections $\mathscr{D}_K=\{\mathbf{l}_1,\cdots,\mathbf{l}_K\}$, the total potential is $\Delta^G(\mathbf{l},\mathscr{D})=\sum_{\mathbf{l}^*\in\mathscr{D}}\sum_{i=1}^{N}\Delta^g(l_i,l_i^*)$. This is a particular instance of the general problem proposed in [4]. As in our case Δ^G depends only on the part location, its score can be added directly to the data term of Eq. (6). In this way, solving for the K-best solutions boils down to solving the same optimization K times, with varying data terms. Because during the K iterations the generated graph is quite similar (only the data terms are modified), reusing the previously computed flows in the graph-cut optimization (as explained in [24]) can help to further speed-up the inference procedure.

Reconsidering now the computational costs, we can see that computing the K best solutions takes $O(NK|\mathscr{L}|)$ whereas the bounded sliding window approach takes $O(N|\mathscr{P}||\mathscr{L}|)$. Therefore in the bounded sliding window approach, to speed-up the method we should consider a reduced deformation $|\mathscr{P}|$. In contrast, in our image-level optimization (similarly to generalized distance transforms) we do not need to limit a priori the object deformation. Instead, we need to fix the number of detections K that we want to obtain, knowing that the computational cost of the method is linear in it. This allows for more flexibility, and combined with additional considerations it

can lead to quite high speed-ups. This is the topic of the next section.

3.3 Multiscale detections

So far we have assumed the feature space $\mathscr{L}$ to be uniform over the three dimensions (x,y,s). However our real space of features (e.g. HOG features), for reasons of speed, is not uniform over scales. This corresponds to moving from a uniform grid of features to the usual pyramid of features. As we want our model to be scale invariant (i.e. the detection score should not depend on the scale), the distances between parts are computed with respect to the size of the parts: the scale. In this way the space of the parts $\mathscr{L}$ becomes the space $(x/s, y/s, s)$. In this space the usual concept of distance is lost. The space is not metric and, therefore, alpha expansion cannot be used. In order to still optimize the CRF with fast alpha expansion we force the parts to be all at the same scale. Thus, the difference between the two deformation vectors $d_i - d_j$ of Eq.(5) on s are always set to zero. In this way we can optimize each scale independently and fixing the scale $s = \hat{s}$ the 2D space $(x/\hat{s}, y/\hat{s})$ is again a valid metric. With these assumptions, we can split our image-level inference into an independent inference for each scale.

In a naive implementation, to compute the K-best detections it is then necessary to apply the CRF inference at every scale K times. When considering S scales, this would produce KS detections, whereas we are only interested in the best K. This is clearly sub-optimal. In order to speed-up the procedure without losing detections, we use a greedy search over the natural splits defined by scales.

For the first detection we apply the optimization of Eq. (9) at each scale s and maintain a priority list of scores v_s. The best score is selected as best detection. Then, to select the second-best detection we penalize the first detection as explained in the previous section and we repeat the optimization at the same scale. The new best detection is the maximum score of the updated list of scores. Again, the found detection is penalized and a new detection at the same scale is found. The procedure is repeated until the desired number of detections has been obtained. Assuming that the optimization is exact [1] (i.e. it always gives the best scoring configuration), this procedure gives the K-best results computing only $K+S$ optimizations. In Table 1 we formally define the algorithm for finding the K-best hypotheses. To further speed-up the algorithm, the priority list is initialized by computing an upper-bound of the maximum score as sum of the unary terms $\max_{\mathbf{l}} \sum_{i=1}^{N} A_i(l_i)$. This is a safe bound because the pairwise terms are forced to be negative.

Find the K best detections $\{det_d\}$.

```
0   for   s ∈ [1..S]
1         v_s ← max_{l∈L_s} A(l) − D(l)
2   d ← 0
3   while   d < K
4         h ← arg max_i v_i
5         det_d ← v_h
6         d ← d + 1
7         A(l) ← A(l) + Δ^G(l, l_h)
8         v_h ← max_{l∈L_h} A(l) − D(l)
```

Table 1: Algorithm for finding the K best detections in an image.

At this point we can see that the image-level optimization defined in Eq. (9), in case of multiscale search, is much faster than the bounded sliding window approach. Now, the total cost for detecting at multiple scales with the bounded sliding windows is proportional to $\sum_{s=1}^{S} |\mathscr{P}||\mathscr{L}_s|$ where $|\mathscr{L}_s|$ is the number of locations at scale s. Instead, for the image-level optimization with our greedy search the computational cost (without considering the additional speed-up of the previously defined bound) is $\sum_{k=1}^{K} |\mathscr{L}_{s_k}| + \sum_{s=1}^{S} |\mathscr{L}_s|$ where s_k is the selected scale for the k-th detection. In practice, while sliding windows has to search uniformly over all image scales, our image-level inference can focus only on the scales where the object is more likely to be.

Note, however, that the number of hypotheses K is not exactly equal to the number of objects expected in an image. In fact, the same object can be detected at multiple scales, and therefore multiple hypotheses could be associated to it. Experimental results comparing detection accuracy and speed-ups for the two methods are presented

[1] Typically, using alpha expansion the solution will not be exact. However, we experimentally found that still the algorithm works well, which is a indirect observation that the found solution is generally very close to the exact one. See Fig. 4 in the experimental results.

in section 4.

3.4 Optimization Variants

As solving Eq. (9) boils down to solving a CRF problem, standard techniques for CRF optimization can be used. We have tested different algorithms for optimizing Eq. (9) and found that a graph-cut with alpha expansion [8] works best. Alpha expansion decomposes the difficult multi-label problem into a sequence of simpler binary subproblems [7]. Each subproblem iteration randomly chooses a new label, and determines for each node to either accept the new label or to stay with the old one. Such alpha expansion has completed a cycle when all possible labels have been evaluated. Each binary subproblem is solved with a graph-cut, so that its solution is exact. The energy of the multi-label problem can only decrease throughout the alpha expansion. As deformation costs are defined as the application of a convex function over the part displacements $S^D(\mathbf{l})$, this function is a valid metric and therefore we are sure that alpha expansion can be used and that every subproblem is sub-modular.

Alpha expansion is also efficient. With the maximum number of cycles fixed, the optimization is linear in the number of labels (i.e. locations where to place a part). If the complexity were more than linear, a image-level optimization would make little sense because a sliding windows approach using a subset of all possible labels would at some point become faster. The worst-case computational cost is polynomial in the number of nodes, but our experiments have shown a linear behavior in practice. This allows us to use models with a relatively high number of parts. Finally, the algorithm can be speeded up using dynamic graph-cut optimization in the alpha expansion, as shown in [1].

We also tried to optimize our score function with other methods, but the results were inferior in terms of both speed and quality. Loopy belief propagation constantly gets stuck in poor local minima, that depend on the label initialization. Sequentially reweighted trees usually give the correct solution, but are still based on belief propagation on subtrees of the graph. Therefore the computational time is of the order of $O(|\mathscr{L}|^2)$ and the image-level optimization is very slow for normally sized images. One can combine sequentially reweighted trees with a distance transform, reducing the computational time to $O(|\mathscr{L}|)$, but in practice the constant factor for this inference is much higher than for graph cuts with alpha expansion. Neither would the fast method of [12] work, because it searches for the best expansion only in orthogonal directions. In general the best detection is not orthogonal to the initial solution. Therefore also this method tends to get stuck in a poor local minimum. Increasing the number of search directions can mitigate the problem, but slows down the algorithm.

3.5 Learning

Given a set of positive and negative images and the object bounding boxes $\mathscr{B}$, we want to learn a vector of weights $\mathbf{w}^*$ such that:

$$\begin{aligned}\mathbf{w}^* = \arg\min_{\mathbf{w}} \{ & \frac{1}{2}|\mathbf{w}|^2 \\ & + C \sum_{n=1}^{M} \sum_{k=1}^{K} \max(0, 1 + \max_{\mathbf{l}} (S(\mathbf{l}, \mathbf{x}_{n,k}, \mathbf{w}) + \Delta^G(\mathbf{l}, \mathscr{D}_{k-1}^{(n)}))) \\ & + C \sum_{n=1}^{|\mathscr{B}|} \max(0, 1 - \max_{\mathbf{l}} (S(\mathbf{l}, \mathbf{x}_n, \mathbf{w}) + \Delta^H(\mathbf{l}, B_n)))\}.\end{aligned} \tag{11}$$

This minimization is an instance of the latent SVM problem [15]. The locations of the object parts $\mathbf{l}$ are the latent variables. For negative examples, as we are interested in ranking detections, we select the first K best detections generated from M negative images. For doing that we use the potential Δ^G defined in section (3.2) that forces the optimization to search for different solutions. As the hinge loss has zero loss for correct classifications with margin higher than one, the search for the best K detections can safely stop when the detection score is smaller than -1. This speed-ups the training significantly while maintaining the same accuracy. For positive examples we want them to be inside the bounding box $B_n \in \mathscr{B}$. For this we add to $S(\mathbf{l}, \mathbf{x}, \mathbf{w})$ [2] another potential $\Delta^H(\mathbf{l}, B_n) = \sum_{i=1}^{N} \Delta^h(l_i, B_n)$ that assigns a score of $-\infty$ to parts outside the object bounding box B_n.

As opposed to binary SVMs, here the problem is not symmetric. Due to the maximization of the latent variables, the loss for the negative samples is convex, while

[2] $S(\mathbf{l}, \mathbf{x}, \mathbf{w})$ is the maximization defined in Eq. (9), where we make explicit the dependency on the image $\mathbf{x}$ and $\mathbf{w}$.

the loss for the positive samples is concave. This is solved using an iterative procedure. Given an initial $\mathbf{w}$ we find the latent values $\mathbf{l}$ for the positive samples. Then, fixing those, we find a new $\mathbf{w}$ optimizing the convex problem with standard SVM solvers. In practice, as we want to enforce positiveness to the deformation parameters, we solve the problem in the primal by using stochastic gradient descent [32] and re-projecting the solution in the feasible space of solutions with positive deformation coefficients.

In the ideal case, when we can find the optimal solution for Eq. (9), the loss of the positive samples can only decrease at each iteration and, hence, the algorithm converges [37]. Unfortunately, in our case the alpha expansion algorithm puts only a weak bound on the quality of the solution [8]. To keep the convergence, we slightly modify the assignment of the positive latent variables. We build a buffer with the previously assigned values. When the new assignment is effectuated, we maintain it only if it produces a lower loss (higher score); otherwise the old assignment is restored.

3.6 Tracking

Improving the deformation model is not only useful for detection, also other tasks can benefit. In this section we show how to apply our deformation model for tracking. Using a more precise deformation model for tracking leads to a more accurate and invariant object representation, which in turn can better follow the object of interest.

In tracking, the object of interest is annotated only at the first frame of a video sequence. Then, the object should be tracked throughout the rest of the video. This is a challenging task because during the video the object is seen from different points of view, there can be partial or total occlusions, similar objects can confuse the tracker and also light changes cause the loss of the object of interest.

We argue that in tracking most of the changes in appearance, apart from occlusions and illumination changes are due to changes of viewpoint, which correspond to small object deformations due for instance to out of plane object rotations. Here we make the object observations more robust to viewpoint changes by using our deformation field. As changes in the point of view often correspond to perspective distortions, our grid of small parts can approximate them properly and better than other approaches.

The tracker is initialized learning a model of the object of interest. The model is learned using Eq. (11), where there is only one positive sample, the object annotated in the first frame of the video, and a set of negatives that are all the other bounding boxes that do not overlap with the annotated one. For localizing the object we consider the image-level approach defined in Eq. (9). In this case the problem is more constrained however, because the model has only one component, and we know a priori the number of objects that should be found. In this way we can reduce the number of hypotheses to generate and the algorithm takes less than 1 second to compute a frame. The update of the model is effectuated using an online version of Eq. (11), where the object model is recomputed at each new frame using stochastic gradient descent. The new observation of the target in a new frame is the one that produces the highest score and has at least 10% overlap with the previous one. This enforces a minimum smoothness on the target displacements. In all the experiments for training we maintain a buffer with the last 100 best positive detections and 1000 negative detections.

4 Experiments

After first introducing the used datasets and giving some technical details of our implementation of the EDFM, we then present several experiments that evaluate the different contributions of our method. In our first experiment we compare the proposed image-level inference with bounded sliding windows in terms of speed and accuracy. Then we compare the performance of the field and the star model varying the number and size of the parts. Finally, we compare our method, based on a set of small parts connected with a field, with the state-of-the-art.

Datasets. We compare our method to various baselines on Pascal VOC 2007 [13]. This dataset contains $9,963$ images separated into 20 different classes of objects. It is well recognized for its difficulty and is the de-facto standard benchmark for object detection. Additionally, considering that the deformation field can also improve the more specific task of human detection, we train and evaluate our method also on two challenging human datasets, where people can be found in very different positions and poses. The first one is the Buffy dataset [28] that contains 748 images of humans performing many different actions,

which makes it a good benchmark to evaluate deformable models. The second one is the UIUC people dataset [35] that contains 593 images of people playing badminton and other sports. As the Buffy and UIUC people datasets do not provide images to collect negative examples, we use the negative images from the INRIA dataset [10]. Finally, to test the EDFM for tracking we compare it to several state-of-the-art trackers using the publicly available collection of videos of [3].

Implementation Details. In this section we provide a detailed description of the most critical parts of our implementation of the EDFM. For the initialization of the deformation field we initially learn a rigid model (i.e. pairwise costs $w^D = -\infty$). Then we use the appearance of the rigid model and relax the pairwise constraints. In this way we are able to iteratively and jointly refine the pairwise deformation costs and the model appearance in the latent SVM optimization.

We notice that for the robust learning of SVMs it is very important to reach convergence during the harvesting of negative examples. If not, the learned deformations are too loose so that the next relabeling of the positive examples can be partially wrong and the model rapidly drifts to a wrong configuration without any possibility to recover. The detection time of the EDFM depends linearly on the number of alpha expansion cycles. In general, as noticed in [8], few cycles are already enough for convergence. In our experiments we found 3 cycles to be sufficient. In all experiments as inference strategy we use the image-level CRF method specified in section 3.3, searching for the first 200 detections.

We use a model with 2 components and their vertical mirror for VOC 2007, and 3 for UIUC people and Buffy. For VOC 2007, more than 2 components did not improve the global performance. The number of parts for each model is then established by the model resolution which is estimated based on the size of the bounding boxes of the training images as in [15]. The neighborhood size μ_P for penalizing an already detected location is set to 2 HOG cells. For evaluation, we estimate the final bounding box, as the minimum bounding box containing all the object parts. The local penalties Δ^G of each part location allow the method to find multiple detections at the same location. This can be useful in very specific situations (e.g. two distinct but highly overlapping objects), but in general it can produce multiple detections of the same object. How to use these hypotheses in a better way is left as future work. For the moment, after the bounding box estimation a standard bounding box based non-maximal suppression is applied.

Image-level Inference vs. Sliding Window. First we compare the EDFM with image-level optimization (with different numbers of hyphotheses) vs. the bounded sliding window scheme introduced in section 3. In Fig. 4 we draw the precision-recall curves for the two configurations, reporting also in the legend average precision and average time per image. As the bounded sliding window approach is very slow, we only used Pascal VOC 2007 images containing bicycles for the test. For bounded sliding window approach we define the search region $\mathscr{P}$ to be half the size of the object. The model used for this experiment has two components each with 24 parts and each part composed of 3×3 HOG cells. By reducing the number of parts the inference time can be further reduced (with a cost in terms of AP, as shown in the next subsection), but the relative speedups of the different configurations are similar. The bounded sliding window approach has a higher recall but lower precision and it has a computational time that is almost 2 orders of magnitude higher than our approach. For the image-level optimization, varying the number of hypotheses corresponds to a different trade-off between precision and recall. Notice that already with as few as 10 hypotheses our image-level inference obtains results very close to sliding windows at just a fraction of the computation time needed for the bounded sliding window approach. The best results are obtained using 500 hypotheses. More slightly reduce the AP due to additional false positives.

The fact that the bounded sliding window approach yields a score inferior to our approach is an indication that: i) our image-level inference works well and does not get stuck in poor local minima (otherwise its AP would be much worse than sliding windows), and ii) the bounded sliding window, searching everywhere, commits to many more high-scoring errors. In the rest of the experiments we always use image-level inference with $K = 200$ since it is a good trade-off between speed and accuracy. EDFM with image-level inference has a detection time that is comparable with other state of the art methods like DPM [15], running at around 8 seconds per image, or LADF [28] that takes around 50 seconds to run with 10 components.

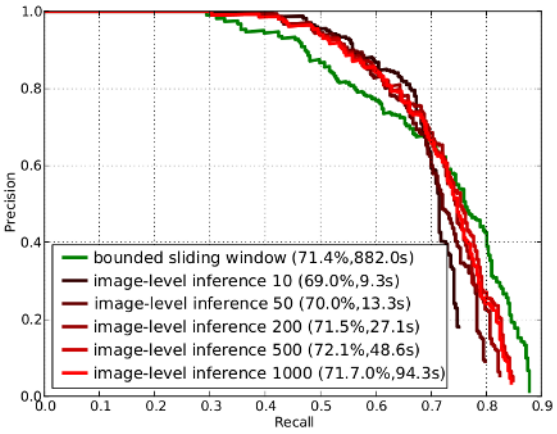
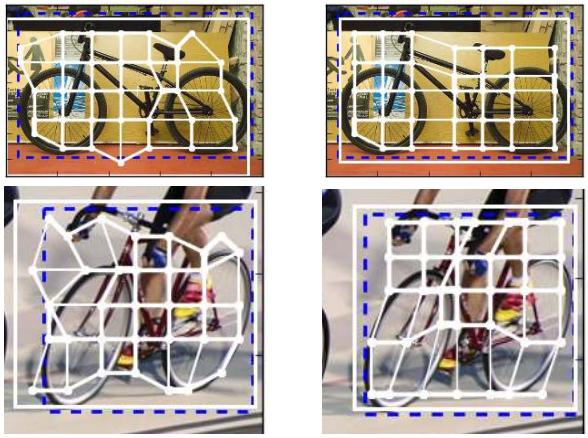


Figure 4: **Image-level Inference vs. Bounded Sliding Window.** We compare EDFM with image-level inference (with different number of hypotheses) with bounded sliding window in terms of AP and average computational time on Pascal VOC 2007 bicycles. In the image-level inference, when varying the number of hypotheses we obtain a different trade-off between precision and recall. In terms of time, our method is always much faster than the bounded sliding window.



(a) (b)

Figure 6: **Star vs. Elastic Field detection.** (a) Star model detection. (b) Deformation field detection. Nodes represent the location of the parts center while edges represent the local distortion between two neighbor parts. The deformation field tends to maintain more coherence among neighbor parts and therefore can represent global distortions better than the star model.

Elastic Field vs. Star Model. To contrast the elastic field with a star model we compare both of them using exactly the same parameters for the two configurations, but varying the size of the parts (and thus the number of such parts in each component).

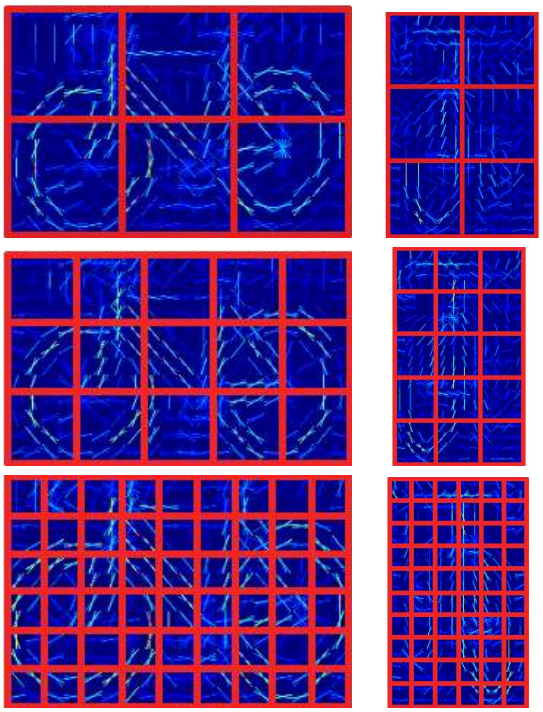
In Fig. 5 we plot the performance (in terms of AP) of detectors trained with different part sizes using either a star model or a deformation field. Visualizations of the EDFM with different part sizes for the class bicycle are presented in Fig. 7. In general, when using big parts the two models have a similar performance. When reducing the part size the deformation field improves its performance while the star model remains the same or becomes worse. This is due to the global coherence enforced by the pairwise connections that permits to better represent deformations with many small parts. For the deformation field the best trade-off between part size and performance is in general found for parts of 3×3 HOG cells, which is the part size used in the rest of the experiments.

In Fig. 6 we show two examples of detections with the relative position of the parts for a star model (a) and a deformation field (b). Whereas in the star model each part moves independently, for the deformation field the locations of the parts follow a global structure enforced by the pairwise constraints.

Human detection on Buffy. Next, we evaluate the EDFM on the Buffy dataset. Based on the validation set we build a model with 3 components, each one composed of parts of 3×3 HOG cells. For the evaluation we use the same protocol as in [28]. In Table 2 we compare our EDFM with the standard DPM and the locally affine deformation field (LADF) introduced in [28].

Our model outperforms the DPM with more than 5%. This shows that deformations are important for detection and that a finer grid of parts and a better prior on the deformations are effective ways to improve detection.

As to LADF, both methods use a field of small parts. However, there are several differences between the two methods. First, in LADF, to better adapt to the differ-



(a) (b)

Figure 7: **Bicycle models with different parts size.** Models used in Fig. 5. The models have a similar resolution but a different number of parts (Top 6×6, middle 4×4, and bottom 2×2). (a) Lateral model. (b) Frontal Model.

ent poses in the dataset, the data is split into 10 different clusters (components) using a distance based on the deformable field. In contrast, our EDFM uses a much simpler split into 3 components based on the aspect ratio as in DPM, because we believe that our elastic deformation field can cope with the deformations introduced by different human poses without the need of many components. Despite the smaller number of components our EDFM outperforms LADF with a margin of almost 2%. We believe that our model performs better than LADF due to the different deformation field used. In LADF the deformation field is restricted to locally affine deformations with a predefined and uniform cost which limits the amount of configurations that can be represented. In our model instead, we use a complete CRF model, where the more probable deformations are learned in a discriminative way through the joint learning of the pairwise costs and the model appearance in the latent SVM procedure.

implementation	def. model	components	AP(%)
DPM [15]	star model	3	72.3
LADF [28]	loc. aff. field	10	76.0
EDFM(ours)	elastic field	3	**77.8**

Table 2: **Human detection on Buffy**. We compare DPM, LADF and our EDFM on the Buffy dataset. Because of the high variation of poses of the humans in the dataset EDFM clearly outperforms DPM. Also, our EDFM with only 3 components outperforms the LADF trained with 10 components.

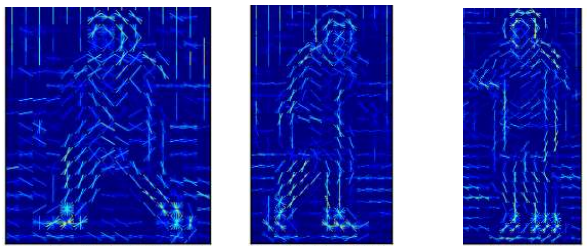
	Ours	HPos[35]	PS rev.[2]	DPM[15]	Poselet[6]
Acc.	**78.0**	66.8	50.6	48.6	45.8

Table 3: **Human detection on UIUC people**. As is standard for this dataset we compute the accuracy of the best detection in each image. A detection is considered valid if it has an overlap higher than 50% using the Pascal overlap criteria [13].

Human Detection on UIUC people. As for Buffy, also here we built a human detector based on 3 components, each one containing parts of 3×3 HOG cells. We compare our detector based on the deformation field with other state-of-the-art methods, some of them ([6],[2],[35]) also using parts annotations for training (see Table 3). Also in this case our method outperforms the other approaches. In Fig. 8 we show the 3 components of the model learned by our approach (parts are omitted for clarity).

Object detection on PASCAL VOC 2007. We also evaluate our model on the 20 object classes of the challenging PASCAL VOC 2007. For all the classes we use 2-component models and parts of 3×3 HOG cells. In Table 4 we compare our EDFM with DPM based methods. It outperforms the others on 10 out of 20 classes.

From the table we see that our model outperforms the



(a) (b) (c)

Figure 8: **UIUC people model components.** The three components represent the main poses that a person assumes in the dataset.

	Ours		MIL[3]		TLD[22]		Star[38]		MST[38]	
	Err.	Prec	Err.	Prec	Err.	Prec	Err.	Prec	Err.	Prec
Sylvester	**5.5**	**0.99**	10.9	0.73	20.0	0.91	9.3	0.90	7.1	0.93
David	5.6	**1.00**	22.9	0.61	4.5	**1.00**	4.5	**1.00**	**3.5**	**1.00**
Cola Can	9.3	**0.76**	20.9	0.22	16.3	0.52	21.4	0.37	**7.1**	0.75
Occl. Face 1	16.6	0.85	27.2	0.78	16.8	0.99	5.5	**1.00**	**4.6**	**1.00**
Occl. Face 2	14.6	0.87	20.2	0.82	22.1	0.77	12.1	0.85	**7.4**	**1.00**
Surfer	**5.5**	0.78	9.2	0.76	7.9	**0.84**	189.2	0.26	13.4	0.43
Tiger1	**5.7**	**0.93**	15.3	0.58	28.7	0.13	22.1	0.37	6.1	0.89
Tiger2	**6.5**	**0.90**	17.1	0.64	37.5	0.27	26.5	0.39	7.6	0.88
Dollar	4.5	**1.00**	14.8	0.95	**3.9**	**1.00**	4.5	**1.00**	5.5	**1.00**
Cliff bar	**6.7**	0.76	11.6	0.77	12.3	0.36	67.6	0.35	12.1	**0.79**
Tea Box	19.0	0.48	**10.2**	**0.86**	39.0	0.18	28.6	0.43	41.9	0.40
Girl	18.8	0.71	32.0	0.57	24.7	0.78	10.5	**1.00**	**10.4**	**1.00**
Avrg. Rank	**1.92**	**1.83**	3.5	3.08	3.66	3.00	3.25	2.91	2.16	**1.83**

Table 5: **Performance of different trackers** in terms of average distance in pixels (Err) and precision (Prec). Our method is markedly better than the others for those videos where the object deformation is the main difficulty.

others in 6 classes. For classes like "bird", "cat", "dog", the appearance is extremely variable due to the very different poses these animals can assume. In these case, as there is no distinctive appearance, it becomes very difficult to learn a proper deformation model and therefore simpler assumptions about the deformations, as in a star model, can work better. Another interesting case is "person". Here our performance is quite inferior to the other DPM models in spite of the good results obtained in the other datasets. This is probably due to the high variability in appearance present in these images, that again does not let the model to properly learn the correct deformations. Also, it is important to mention that our model is composed of a single layer of HOG features, while all the others use two or more HOG resolutions. Fig. 9 shows some examples of detections with the inferred deformations for different object classes.

Tracking. Finally, we evaluate the extension of our deformation field for tracking, as explained in section 3.6, on the publicly available collection of videos introduced in [3]. Table 5 reports the performance of our deformation field tracker compared with other methods such as the MILboost tracker [3], the TDL tracker [22] and the structure preserving object tracker [38]. The evaluation is presented in terms of average distance in pixels between the predicted bounding box and the ground truth annotations and as precision. For precision we consider a detection as correct if its overlap with ground truth is more than 50% as in [38].

In the videos where the main challenge is a change of view or an out of plane rotation, as in Sylvester or Tiger, our detector clearly outperform the previous methods because it can represent the appearance distortion better. This shows that the deformations induced by the deformation field are also useful for tracking. Note that [38] is based on DPM models where parts are connected in a star (Star in Table 5) or with the minimum spanning tree (MST in Table 5), based on the initialization frame. In contrast to these models where the structure of the connected parts should be defined as a tree to optimize it with dynamic programming, with our model we can connect each adjacent part avoiding the problem of defining a priori the best deformation structure.

In the last row of Table 5 we report a global estimation of the quality of each method as average ranking over the 12 videos. Our method obtains the lowest pixel error and (on par with MST) the best precision. Finally, in Fig. 10 the detections obtained by our tracker (dotted red line) are compared with the ground truth (green line) on several frames.

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
DPM [15]	29.0	54.6	0.6	13.4	26.2	39.4	46.4	16.1	16.3	16.5	24.5	5.0	43.6	37.8	35.0	8.8	17.3	21.6	34.0	39.0	26.8
HDPM [39]	29.4	55.8	9.4	14.3	**28.6**	44.0	51.3	**21.3**	20.0	19.3	25.2	12.5	50.4	38.4	36.6	**15.1**	19.7	25.1	36.8	39.3	29.6
Vedaldi [34]	30.6	58.1	10.0	12.7	21.2	52.1	55.0	20.1	19.4	22.1	20.5	11.3	56.4	43.6	38.8	10.8	14.8	25.6	43.8	**43.8**	30.5
DPM V5.1	33.2	60.3	**10.2**	**16.1**	27.3	**54.3**	**58.2**	**23.0**	20.0	24.1	**26.7**	**12.7**	58.1	**48.2**	**43.2**	12.0	21.1	**36.1**	**46.0**	43.5	33.7
EDFM(ours)	**36.4**	**60.5**	9.5	15.7	25.7	47.8	54.4	12.2	**20.5**	**31.2**	22.5	10.5	**58.9**	45.5	28.4	11.7	**27.9**	28.0	44.1	41.1	31.6

Table 4: **Object detection on PASCAL VOC 2007.** Average precision results for each class for different DPM star-based models and our EDFM. Our model with a single resolution HOG features is able to outperform previous DPM models on 10 out of 20 classes.

5 Conclusions

In this paper we have presented a method for the detection of deformable objects. This method is inspired by previous work on optical flow and image registration where for a good estimation of the displacement field it is necessary to enforce a global regularization. Following this path, we have introduced an object model composed of a regular grid of small parts the location of which depends on their distance from neighboring parts. We have shown that, in spite of the complexity of the problem, this can be optimized in a reasonable time avoiding a sliding window search. Finally we have empirically proved that our model is able to better represent complex deformations showing comparable or better performance than the state-of-the-art for different applications and multiple datasets.

References

[1] Alahari, K., Kohli, P., Torr, P.H.S.: Dynamic hybrid algorithms for map inference in discrete mrfs. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**, 1846–1857 (2010)

[2] Andriluka, M., Roth, S., Schiele, B.: Pictorial structures revisited: People detection and articulated pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1014–1021 (2009)

[3] Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(8), 1619–1632 (2011)

[4] Batra, D., Yadollahpour, P., Guzman, A., Shakhnarovich, G.: Diverse m-best solutions in markov random fields. In: Proceedings of the European Conference on Computer Vision (2012)

[5] Bergtholdt, M., Kappes, J., Schmidt, S., Schnörr, C.: A study of parts-based object class detection using complete graphs. International Journal of Computer Vision **87**(1-2), 93–117 (2010)

[6] Bourdev, L.D., Maji, S., Brox, T., Malik, J.: Detecting people using mutually consistent poselet activations. In: Proceedings of the European Conference on Computer Vision, pp. 168–181 (2010)

[7] Boykov, Y., Veksler, O., Zabih, R.: Markov random fields with efficient approximations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 648–656 (1998)

[8] Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(11), 1222–1239 (2001)

[9] Crandall, D., Felzenszwalb, P., Huttenlocher, D.: Spatial priors for part-based recognition using statistical models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10–17 (2005)

[10] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 886–893 (2005)

[11] Desai, C., Ramanan, D., Fowlkes, C.: Discriminative models for multi-class layout. In: Proceedings of the IEEE International Conference on Computer Vision (2009)

[12] Duchenne, O., Joulin, A., Ponce, J.: A graph-matching kernel for object categorization. p. 1056. Barcelona, Spain (2011)

[13] Everingham, M., Zisserman, A., Williams, C., Van Gool, L.: The pascal visual obiect classes challenge 2007 (voc2007) results (2007)

[14] Felzenszwalb, P.F., Girshick, R., McAllester, D.: Cascade object detection with deformable part models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2241–2248 (2010)

[15] Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1627–1645 (2010)

[16] Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transforms of sampled functions. Tech. rep. (2004)

[17] Felzenszwalb, P.F., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)

[18] Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 264–271 (2003)

[19] Glocker, B., Komodakis, N., Tziritas, G., Navab, N., Paragios, N.: Dense image registration through mrfs and efficient linear programming. Medical Image Analysis **12**(6), 731–741 (2008)

[20] Hoeim, D., Rother, C., Winn, J.M.: 3d layout crf for multi-view object class recognition and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2008)

[21] Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence **17**, 185–203 (1981)

[22] Kalal, Z., Matas, J., Mikolajczyk, K.: P-n learning: Bootstrapping binary classifiers by structural constraints. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 49–56 (2010)

[23] Kapoor, A., Winn, J.: Located hidden random fields: Learning discriminative parts for object detection. In: Proceedings of the European Conference on Computer Vision, pp. 302–315 (2006)

[24] Kohli, P., Torr, P.H.S.: Dynamic graph cuts for efficient inference in markov random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(12), 2079–2088 (2007)

[25] Komodakis, N., Tziritas, G., Paragios, N.: Performance vs computational efficiency for optimizing single and dynamic mrfs: Setting the state of the art with primal-dual strategies. Computer Vision and Image Understanding **112**(1), 14–29 (2008)

[26] Lades, M., Vorbruggen, J.C., Buhmann, J., Lange, J., Malsburg, C.v.d., Wurtz, R.P., Konen, W.: Distortion invariant object recognition in the dynamic link architecture. IEEE Transactions on Computers **42**(3), 300–311 (1993)

[27] Ladicky, L., Sturgess, P., Alahari, K., Russell, C., Torr, P.: Where, what and how many?: Combining object detectors and crfs. In: Proceedings of the European Conference on Computer Vision, pp. 424–437 (2010)

[28] Ladicky, L., Torr, P.H.S., Zisserman, A.: Latent svms for human detection with a locally affine deformation field. In: Proceedings of the British Machine Vision Conference, pp. 10.1–10.11 (2012)

[29] Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the international joint conference on Artificial intelligence, pp. 674–679 (1981)

[30] Pedersoli, M., Vedaldi, A., Gonzàlez, J.: A coarse-to-fine approach for fast deformable object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1353–1360 (2011)

[31] Quattoni, A., Collins, M., Darrell, T.: Conditional random fields for object recognition. In: Advances in Neural Information Processing Systems, pp. 1097–1104. MIT Press (2004)

[32] Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A.: Pegasos: primal estimated sub-gradient solver for svm. Math. Program. **127**(1), 3–30 (2011)

[33] Vedaldi, A., Zisserman, A.: Structured output regression for detection with partial occulsion. In: Advances in Neural Information Processing Systems, pp. 1928–1936 (2009)

[34] Vedaldi, A., Zisserman, A.: Sparse kernel approximations for efficient classification and detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2320–2327 (2012)

[35] Wang, Y., Tran, D., Liao, Z.: Learning hierarchical poselets for human parsing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1705–1712 (2011)

[36] Yang, Y., Ramanan, D.: Articulated human detection with flexible mixtures-of-parts. IEEE Transactions on Pattern Analysis and Machine Intelligence **99**(PrePrints), 1 (2012)

[37] Yuille, A., Rangarajan, A., Yuille, A.L.: The concave-convex procedure (cccp). In: Advances in Neural Information Processing Systems, pp. 1033–1040 (2002)

[38] Zhang, L., van der Maaten, L.: Structure preserving object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2013)

[39] Zhu, L., Chen, Y., Yuille, A., Freeman, W.: Latent hierarchical structural learning for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2010)

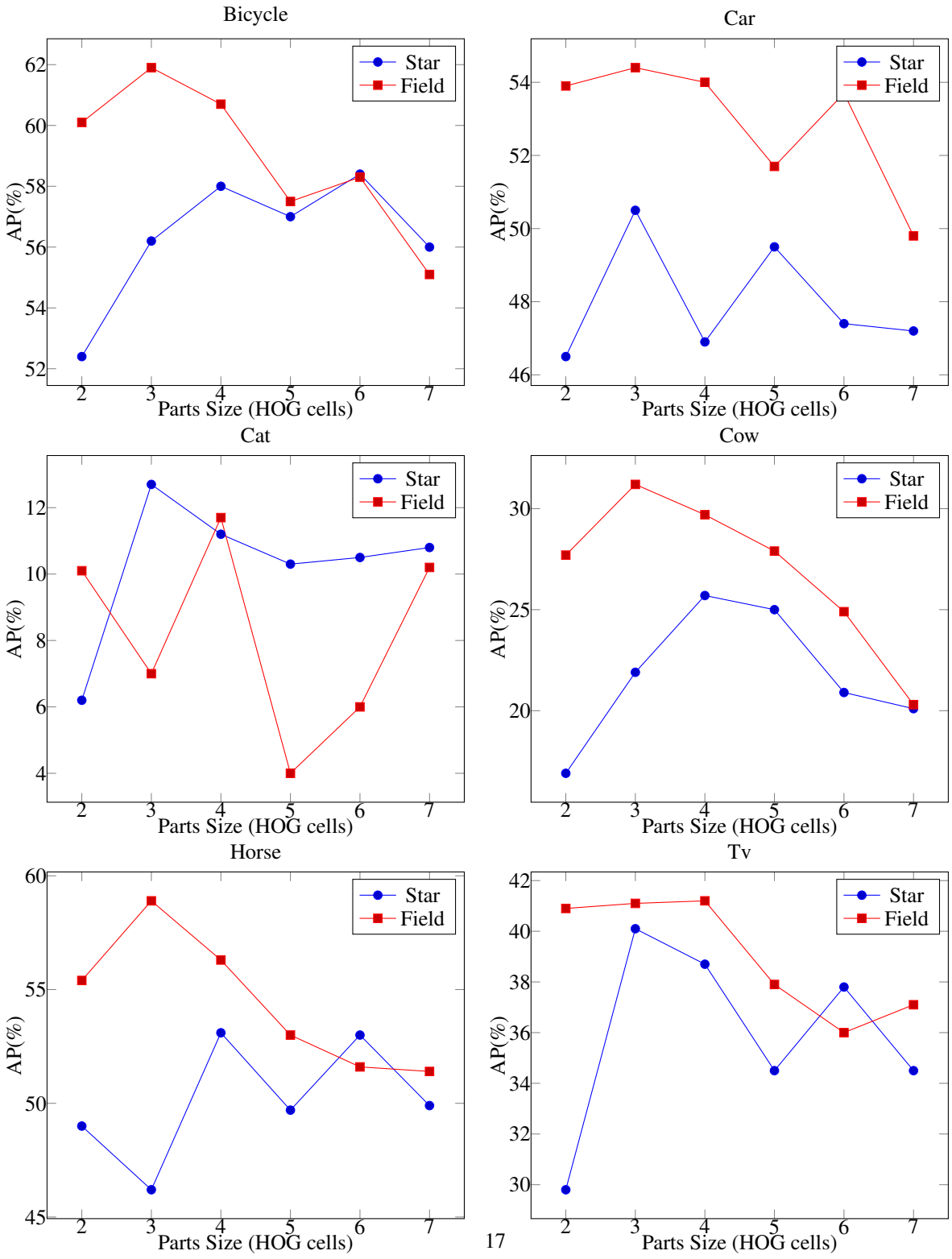


Figure 5: **Parts size vs. AP**.We compare the performance of the star model and the deformation field varying the parts size in HOG cells for some representative classes of Pascal VOC. With few and big parts, the two models tend to perform similarly. However, when the parts are smaller the deformation field performs markedly better than the star model. A special case is cat, where the learned appearance is quite poor and star or field deformation model do not make any difference.

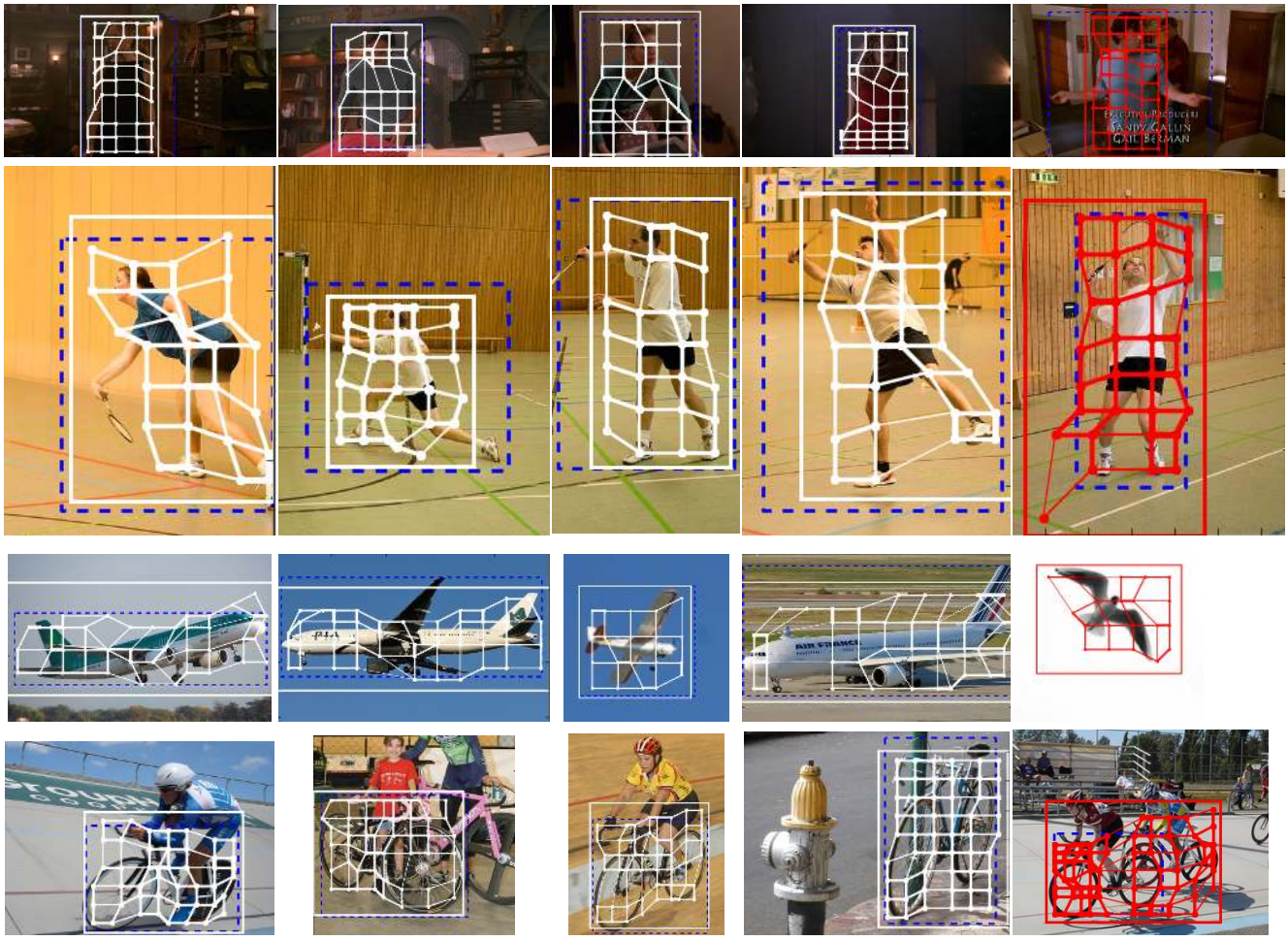


Figure 9: **Detections on the different datasets.** The deformation field is used to handle perspective and point of view distortions as well as intra-class variability. Red detections (last column) are false positives. Top row: Buffy; second row: UIUC people; third and forth row: Pascal VOC 2007.

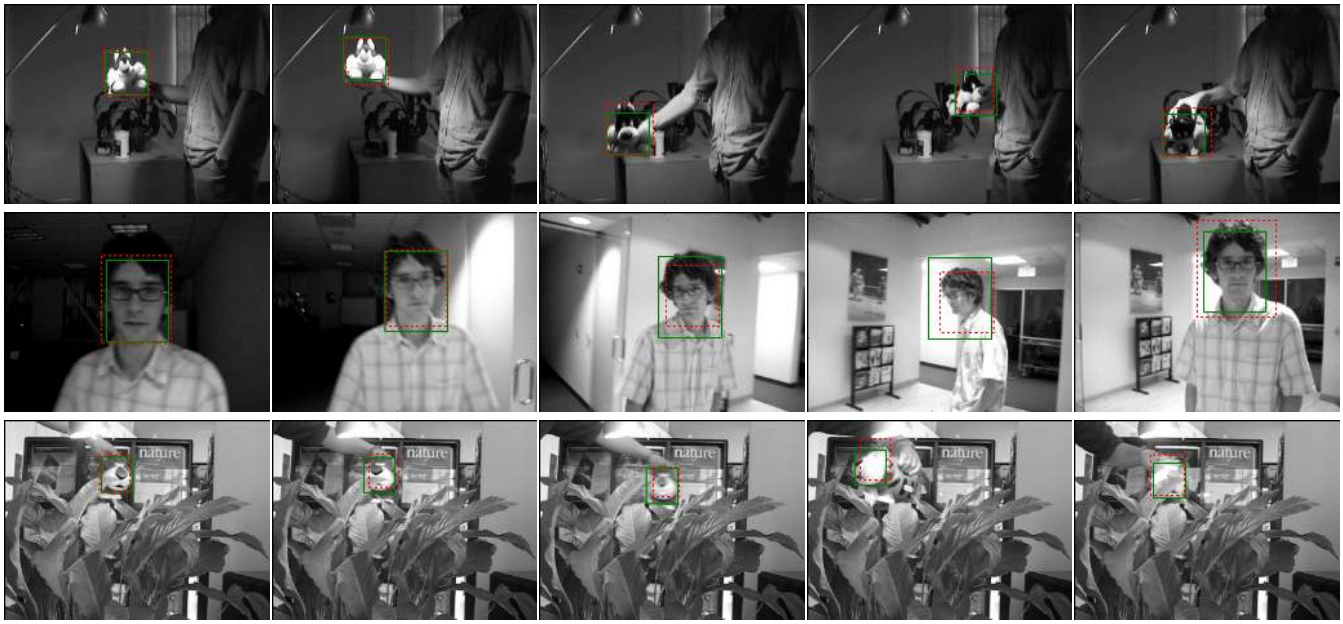


Figure 10: **Tracking.** Bounding box obtained by our method (dashed red) and ground truth (green) on several frames of Sylvester, David and Tiger1.