Robust Near-Isometric Matching via Structured Learning of Graphical Models

Julian J. McAuley
NICTA/ANU
julian.mcauley@nicta.com.au

Tibério S. Caetano
NICTA/ANU
tiberio.caetano@nicta.com.au

Alexander J. Smola
Yahoo! Research*
alex@smola.org

Abstract

Models for near-rigid shape matching are typically based on distance-related features, in order to infer matches that are consistent with the isometric assumption. However, real shapes from image datasets, even when expected to be related by "almost isometric" transformations, are actually subject not only to noise but also, to some limited degree, to variations in appearance and scale. In this paper, we introduce a graphical model that parameterises appearance, distance, and angle features and we learn all of the involved parameters via structured prediction. The outcome is a model for near-rigid shape matching which is *robust* in the sense that it is able to capture the possibly limited but still important scale and appearance variations. Our experimental results reveal substantial improvements upon recent successful models, while maintaining similar running times.

1 Introduction

Matching shapes in images has many applications, including image retrieval, alignment, and registration [1, 2, 3, 4]. Typically, matching is approached by selecting features for a set of landmark points in both images; a correspondence between the two is then chosen such that some distance measure between these features is minimised. A great deal of attention has been devoted to defining complex features which are robust to changes in rotation, scale etc. [5, 6].[1]

An important class of matching problems is that of *near-isometric* shape matching. In this setting, it is assumed that shapes are defined up to an isometric transformation (allowing for some noise), and therefore distance features are typically used to encode the shape. Recent work has shown how the isometric constraint can be exploited by a particular type of graphical model whose topology encodes the necessary properties for obtaining optimal matches in polynomial time [11].

Another line of work has focused on *structured learning* to optimize graph matching scores, however no explicit exploitation of the geometrical constraints involved in shape modeling are made [12].

In this paper, we combine the best of these two approaches into a single model. We produce an exact, efficient model to solve near-isometric shape matching problems using not only isometry-invariant features, but also appearance and scale-invariant features. By doing so we can learn the relative importances of variations in appearance and scale with regard to variations in shape *per se*. Therefore, even knowing that we are in a near-isometric setting, we will capture the eventual variations in appearance and scale into our matching criterion in order to produce a *robust* near-isometric matcher. In terms of learning, we introduce a two-stage structured learning approach to address the speed and memory efficiency of this model.

*Alexander J. Smola was with NICTA at the time of this work.

[1] We restrict our attention to this type of approach, i.e. that of matching landmarks between images. Some notable approaches deviate from this norm – see (for example) [7, 8, 9, 10].

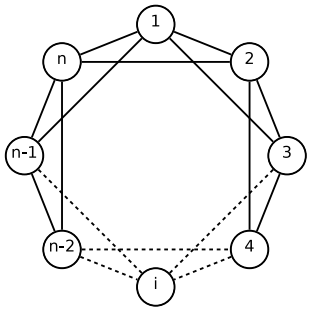

Figure 1: The graphical model introduced in [11].

2 Background

2.1 Shape Matching

'Shape matching' can mean many different things, depending on the precise type of query one is interested in. Here we study the case of identifying an instance of a template shape ($\mathcal{S} \subseteq \mathcal{T}$) in a target scene ($\mathcal{U}$) [1].[2] We assume that we *know* $\mathcal{S}$, i.e. the points in the template that we want to query in the scene. Typically both $\mathcal{T}$ and $\mathcal{U}$ correspond to a set of 'landmark' points, taken from a pair of images (common approaches include [6, 13, 14]).

For each point $t \in \mathcal{T}$ and $u \in \mathcal{U}$, a certain set of *unary features* are extracted (here denoted by $\phi(t)$, $\phi(u)$), which contain local information about the image at that point [5, 6]. If $y : \mathcal{S} \to \mathcal{U}$ is a generic mapping representing a potential match, the goal is then to find a mapping $\hat{y}$ which minimises the aggregate distance between corresponding features, i.e.

$$\hat{y} = f(\mathcal{S},\mathcal{U}) = \operatorname*{argmin}_{y} \sum_{i=1}^{|\mathcal{S}|} c_1(s_i, y(s_i)), \text{ where } c_1(s_i, y(s_i)) = \|\phi(s_i) - \phi(y(s_i))\|_2^2. \quad (1)$$

(here $\|\cdot\|_2$ denotes the L_2 norm). For injective y eq. (1) is a linear assignment problem, efficiently solvable in cubic time. In addition to unary or first-order features, pairwise or second-order features can be induced from the locations of the unary features. In this case eq. (1) would be generalised to minimise an aggregate distance between pairwise features. This however induces an NP-hard problem (quadratic assignment). Discriminative structured learning has recently been applied to models of both linear and quadratic assignment in [12].

2.2 Graphical Models

In isometric matching settings, one may suspect that it may not be necessary to include all pairwise relations in quadratic assignment. In fact a recent paper [11] has shown that if only the distances as encoded by the graphical model depicted in figure 1 are taken into account (nodes represent points in $\mathcal{S}$ and states represent points in $\mathcal{U}$), exact probabilistic inference in such a model can solve the isometric problem optimally. That is, an energy function of the following form is minimised:[3]

$$\sum_{i=1}^{|\mathcal{S}|} c_2(s_i, s_{i+1}, y(s_i), y(s_{i+1})) + c_2(s_i, s_{i+2}, y(s_i), y(s_{i+2})). \quad (2)$$

In [11], it is shown that loopy belief propagation using this model converges to the optimal assignment, and that the number of iterations required before convergence is small in practice.

We will extend this model by adding a unary term, $c_1(s_i, y(s_i))$ (as in (eq. 1)), and a third-order term, $c_3(s_i, s_{i+1}, s_{i+2}, y(s_i), y(s_{i+1}), y(s_{i+2}))$. Note that the graph topology remains the same.

[2] Here $\mathcal{T}$ is the set of *all points* in the template scene, whereas $\mathcal{S}$ corresponds to those points in which we are interested. It is also important to note that we treat $\mathcal{S}$ as an *ordered* object in our setting.

[3] s_{i+1} should be interpreted as $s_{(i+1) \mod |\mathcal{S}|}$ (i.e. the points form a loop).

2.3 Discriminative Structured Learning

In practice, feature vectors may be very high-dimensional, and which components are 'important' will depend on the specific properties of the shapes being matched. Therefore, we introduce a parameter, θ, which controls the relative importances of the various feature components. Note that θ is parameterising the matching criterion itself. Hence our minimisation problem becomes

$$\hat{y} = f(\mathcal{S},\mathcal{U};\theta) = \underset{y}{\operatorname{argmax}} \langle h(\mathcal{S},\mathcal{U},y),\theta\rangle \tag{3}$$

$$\text{where } h(\mathcal{S},\mathcal{U},y) = -\sum_{i=1}^{|\mathcal{S}|} \Phi(s_i, s_{i+1}, s_{i+2}, y(s_i), y(s_{i+1}), y(s_{i+2})). \tag{4}$$

(y is a mapping from $\mathcal{S}$ to $\mathcal{U}$, Φ is a third-order feature vector – our specific choice is shown in section 3).[4] In order to measure the performance of a particular weight vector, we use a *loss function*, $\Delta(\hat{y}, y^i)$, which represents the cost incurred by choosing the assignment $\hat{y}$ when the correct assignment is y^i (our specific choice of loss function is described in section 4). To avoid overfitting, we also desire that θ is sufficiently 'smooth'. Typically, one uses the squared L_2 norm, $\|\theta\|_2^2$, to penalise non-smooth choices of θ [15].

Learning in this setting now becomes a matter of choosing θ such that the empirical risk (average loss on all training instances) is minimised, but which is also sufficiently 'smooth' (to prevent overfitting). Specifically, if we have a set of training pairs, $\left\{\mathcal{S}^1 \dots \mathcal{S}^N\right\}$, $\left\{\mathcal{U}^1 \dots \mathcal{U}^N\right\}$, with labelled matches $\left\{y^1 \dots y^N\right\}$, then we wish to minimise

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N} \Delta(f(\mathcal{S}^i,\mathcal{U}^i;\theta), y^i)}_{\text{empirical risk}} + \underbrace{\frac{\lambda}{2}\|\theta\|_2^2}_{\text{regulariser}} . \tag{5}$$

Here λ (the regularisation constant) controls the relative importance of minimising the empirical risk against the regulariser. In our case, we simply choose λ such that the empirical risk on our validation set is minimised.

Solving (eq. 5) exactly is an extremely difficult problem and in practice is not feasible, since the loss is piecewise constant on the parameter θ. Here we capitalise on recent advances in large-margin structured estimation [15], which consist of obtaining convex relaxations of this problem. Without going into the details of the solution (see, for example, [15, 16]), it can be shown that a convex relaxation of this problem can be obtained, which is given by

$$\min_{\theta} \ \frac{1}{N}\sum_{i=1}^{N} \xi_i + \frac{\lambda}{2}\|\theta\|_2^2 \tag{6a}$$

subject to

$$\langle h(\mathcal{S}^i,\mathcal{U}^i,y^i) - h(\mathcal{S}^i,\mathcal{U}^i,y),\theta\rangle \geq \Delta(y,y^i) - \xi_i$$

$$\text{for all } i \text{ and } y \in \mathcal{Y} \tag{6b}$$

(where $\mathcal{Y}$ is the space of all possible mappings). It can be shown that for the solution of the above problem, we have that $\xi_i^* \geq \Delta(f(\mathcal{S}^i,\mathcal{U}^i;\theta), y^i)$. This means that we end up minimising an upper bound on the loss, instead of the loss itself.

Solving (6) requires only that we are able, for any value of θ, to find

$$\underset{y}{\operatorname{argmax}} \left(\langle h(\mathcal{S}^i,\mathcal{U}^i,y),\theta\rangle + \Delta(y,y^i)\right). \tag{7}$$

In other words, for each value of θ, we are able to identify the mapping which is consistent with the model (eq. 3), yet incurs a high loss. This process is known as 'column generation' [15, 16]. As we will define our loss as a sum over the nodes, solving (eq. 7) is no more difficult than solving (eq. 3).

[4] We have expressed (eq. 3) as a maximisation problem as a matter of convention; this is achieved simply by negating the cost function in (eq. 4).

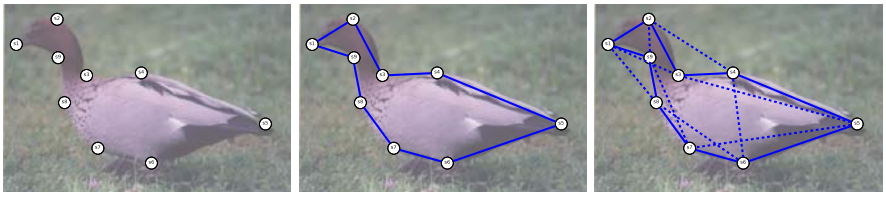

Figure 2: Left: the (ordered) set of points in our template shape ($\mathcal{S}$). Centre: connections between immediate neighbours. Right: connections between neighbour's neighbours (our graphical model).

3 Our Model

Although the model of [11] solves isometric matching problems optimally, it provides no guarantees for *near*-isometric problems, as it only considers those compatibilities which form cliques in our graphical model. However, we are often only interested in the boundary of the object: if we look at the instance of the model depicted in figure 2, it seems to capture exactly the important dependencies; adding additional dependencies between distant points (such as the duck's tail and head) would be unlikely to contribute to this model.

With this in mind, we introduce three new features (for brevity we use the shorthand $y_i = y(s_i)$):

$\Phi_1(s_1, s_2, y_1, y_2) = (d_1(s_1, s_2) - d_1(y_1, y_2))^2$, where $d_1(a, b)$ is the Euclidean distance between a and b, scaled according to the width of the target scene.

$\Phi_2(s_1, s_2, s_3, y_1, y_2, y_3) = (d_2(s_1, s_2, s_3) - d_2(y_1, y_2, y_3))^2$, where $d_2(a, b, c)$ is the Euclidean distance between a and b scaled by the average of the distances between a, b, and c.

$\Phi_3(s_1, s_2, s_3, y_1, y_2, y_3) = (\angle(s_1, s_2, s_3) - \angle(y_1, y_2, y_3))^2$, where $\angle(a, b, c)$ is the angle between a and c, w.r.t. b.[5]

We also include the unary features $\Phi_0(s_1, y_1) = (\phi(s_1) - \phi(y_1))^2$ (i.e. the pointwise squared difference between $\phi(s_1)$ and $\phi(y_1)$). Φ_1 is exactly the feature used in [11], and is invariant to isometric transformations (rotation, reflection, and translation); Φ_2 and Φ_3 capture triangle similarity, and are thus also invariant to scale. In the context of (eq. 4), we have

$$\Phi(s_1, s_2, s_3, y_1, y_2, y_3) := \big[\Phi_0(s_1, y_1), \Phi_1(s_1, s_2, y_1, y_2) + \Phi_1(s_1, s_3, y_1, y_3), \\ \Phi_2(s_1, s_2, s_3, y_1, y_2, y_3) + \Phi_2(s_1, s_3, s_2, y_1, y_3, y_2), \Phi_3(s_1, s_2, s_3, y_1, y_2, y_3)\big]. \quad (8)$$

In practice, landmark detectors often identify several hundred points [6, 17], which is clearly impractical for an $O(|\mathcal{S}||\mathcal{U}|^3)$ method ($|\mathcal{U}|$ is the number of landmarks in the target scene). To address this, we adopt a two stage learning approach: in the first stage, we learn only unary compatibilities, exactly as is done in [12]. During the second stage of learning, we collapse the first-order feature vector into a single term, namely

$$\Phi'_0(s_1, y_1) = \langle \theta_0, \Phi_0(s_1, y_1) \rangle \quad (9)$$

(θ_0 is the weight vector learned during the first stage). We now perform learning for the third-order model, *but consider only the p 'most likely' matches for each node*, where the likelihood is simply determined using $\Phi'_0(s_1, y_1)$. This reduces the performance and memory requirements to $O(|\mathcal{S}|p^3)$. A consequence of using this approach is that we must now tune *two* regularisation constants; this is not an issue in practice, as learning can be performed quickly using this approach.[6]

[5] Using features of such different scales can be an issue for regularisation – in practice we adjusted these features to have roughly the same scale. For full details, our implementation is available at (*not included for blind review*).

[6] In fact, even in those cases where a single stage approach was tractable (such as the experiment in section 4.1), we found that the two stage approach worked better. Typically, we required much less regularity during the second stage, possibly because the higher order features are heterogeneous.

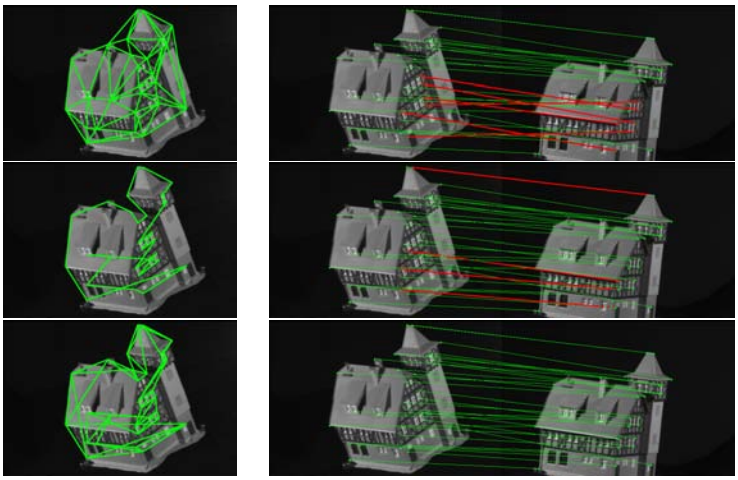

Figure 3: Left: The adjacency structure of the graph (top); the boundary of our 'shape' (centre); the topology of our graphical model (bottom). Right: Example matches using linear assignment (top, 6/30 mismatches), quadratic assignment (centre, 4/30 mismatches), and the proposed model (bottom, no mismatches). The images shown are the 12^{th} and 102^{nd} frames in our sequence. Correct matches are shown in green, incorrect matches in red. All matches are reported *after* learning.

4 Experiments

4.1 House Data

In our first experiment, we compare our method to those of [11] and [12]. Both papers report the performance of their methods on the CMU 'house' sequence – a sequence of 111 frames of a toy house, with 30 landmarks identified in each frame.[7] As in [12], we compute the Shape Context features for each of the 30 points [5].

In addition to the unary model of [12], a model based on *quadratic* assignment is also presented, in which pairwise features are determined using the adjacency structure of the graphs. Specifically, if a pair of points (p_1, p_2) in the template scene is to be matched to (q_1, q_2) in the target, there is a feature which is 1 if there is an edge between p_1 and p_2 in the template, *and* an edge between q_1 and q_2 in the target (and 0 otherwise). We also use such a feature for this experiment, however our model only considers matchings for which (p_1, p_2) forms an edge in our graphical model (see figure 3, bottom left). The adjacency structure of the graphs is determined using the Delaunay triangulation, (figure 3, top left).

As in [11], we compare pairs of images with a fixed baseline (separation between frames). For our loss function, $\Delta(\hat{y}, y^i)$, we used the normalised Hamming loss, i.e. the proportion of mismatches. Figure 4 shows our performance on this dataset, as the baseline increases. On the left we show the performance without learning, for which our model exhibits the best performance by a substantial margin.[8]

Our method is also the best performing after learning – in fact, we achieve almost zero error for all but the largest baselines (at which point our model assumptions become increasingly violated, and we have less training data). In figure 5, we see that the running time of our method is similar to the quadratic assignment method of [12]. To improve the running time, we also show our results with $p = 10$, i.e. for each point in the template scene, we only consider the 10 'most likely' matches, using the weights from the first stage of learning. This reduces the running time by more than an order of

[7] `http://vasc.ri.cmu.edu/idb/html/motion/house/index.html`

[8] Interestingly, the quadratic method of [12] performs *worse* than their unary method; this is likely because the *relative* scale of the unary and quadratic features is badly tuned before learning, and is indeed similar to what the authors report. Furthermore, the results we present for the method of [12] after learning are much *better* than what the authors report – in that paper, the unary features are scaled using a pointwise exponent ($-\exp(-|\phi_a - \phi_b|^2)$), whereas we found that scaling the features linearly ($|\phi_a - \phi_b|^2$) worked better.

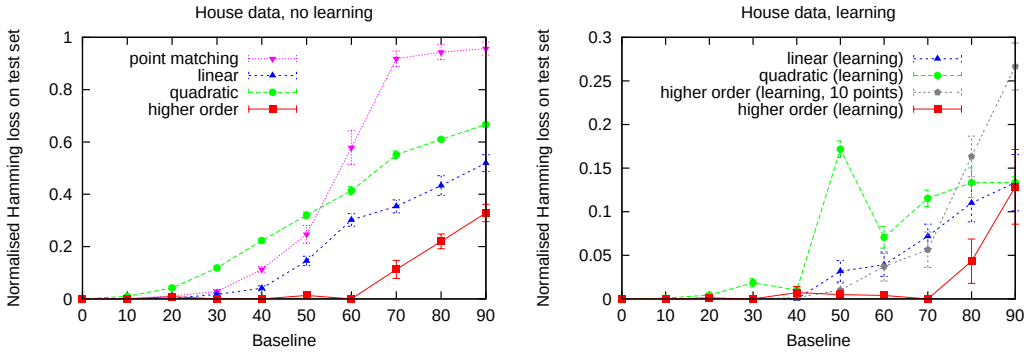

Figure 4: Comparison of our technique against that of [11] ('point matching'), and [12] ('linear', 'quadratic'). The performance before learning is shown on the left, the performance after learning is shown on the right. Our method exhibits the best performance both before and after learning (*note the different scales of the two plots*). Error bars indicate standard error.

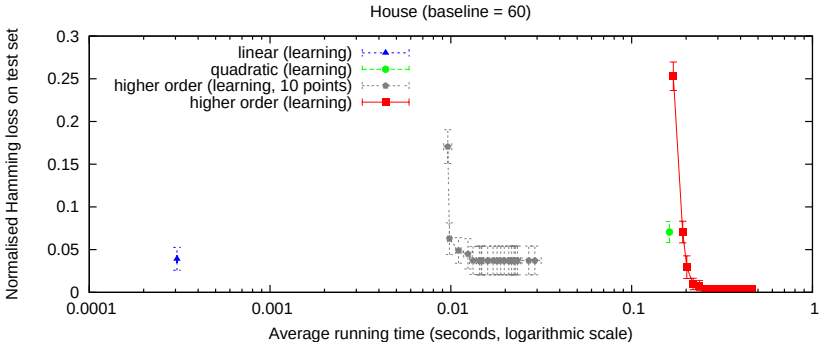

Figure 5: The running time and performance of our method, compared to those of [12] (note that the method of [11] has running time identical to our method). Our method is run from 1 to 20 iterations of belief propagation, although the method appears to converge in fewer than 5 iterations.

magnitude, bringing it closer to that of linear assignment; even this model achieves approximately zero error up to a baseline of 50.

Finally, figure 6 (left) shows the weight vector of our model, for a baseline of 60. The first 60 weights are for the Shape Context features (determined during the first stage of learning), and the final 5 show the weights from our second stage of learning (the weights correspond to the first-order features, distances, adjacencies, scaled distances, and angles, respectively – see section 3). We can provide some explanation of the learned weights: the Shape Context features are separated into 5 radial, and 12 angular bins – the fact that there are peaks around the 16^{th} and 24^{th}, features indicates that some particular radial bins are more important than the others; the fact that several consecutive bins have low weight indicates that some radial bins are unimportant (etc.). It is much more difficult to reason about the second stage of learning, as the features have different scales, and cannot be compared directly – however, it appears that all of the higher-order features are important to our model.

4.2 Bikes Data

For our second experiment, we used images of bicycles from the Caltech 256 Dataset [18]. Bicycles are reasonably rigid objects, meaning that matching based on their shape is logical. Although the images in this dataset are fairly well aligned, they are subject to reflections as well as some scaling and shear. For each image in the dataset, we detected landmarks automatically, and six points on the frame were hand-labelled (see figure 7). Only shapes in which these interest points were not occluded were used, and we only included images that had a background; in total, we labelled 44

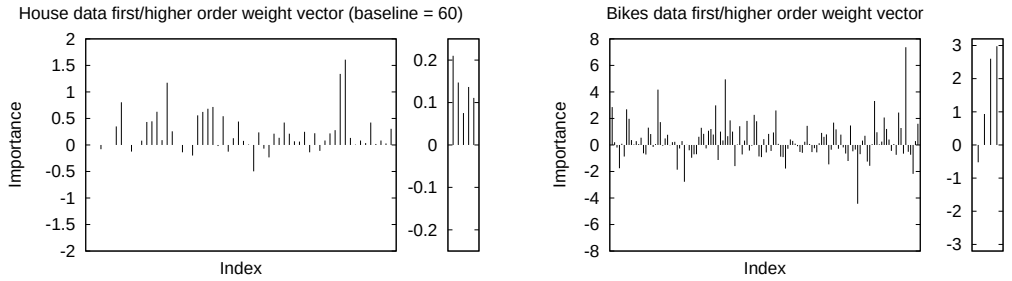

Figure 6: Left: The weight vector of our method after learning, for the 'house' data. The first 60 weights are for the Shape Context features from the first stage of of learning; the final 5 weights are for the second stage of learning. Right: The same plot, for the 'bikes' data.

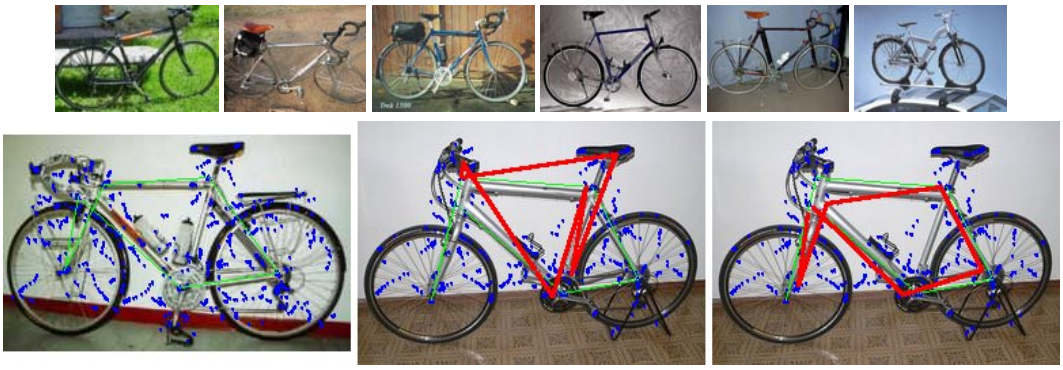

Figure 7: Top: A selection of our training images. Bottom: An example match from our test set. Left: The template image (with the shape outlined in green, and landmark points marked in blue). Centre: The target image, and the match (in red) using unary features with the affine invariant/SIFT model of [17] after learning (endpoint error = 0.27). Right: the match using our model after learning (endpoint error = 0.04).

images. The first image was used as the 'template', the other 43 were used as targets. Thus we are learning to match bicycles similar to the chosen template.

Initially, we used the SIFT landmarks and features as described in [6]. Since this approach typically identifies several hundred landmarks, we set $p = 20$ for this experiment (i.e. we consider the 20 most likely points). Since we cannot hope to get exact matches, we use the endpoint error instead of the normalised Hamming loss, i.e. we reward points which are close to the correct match.[9] Table 1 reveals that the performance of this method is quite poor, even with the higher-order model, and furthermore reveals no benefit from learning. This may be explained by the fact that although the SIFT features are invariant to scale and rotation, they are not invariant to reflection.

In [17], the authors report that the SIFT features *can* provide good matches in such cases, as long as landmarks are chosen which are locally invariant to affine transformations. They give a method for identifying affine-invariant feature points, whose SIFT features are then computed.[10] We achieve much better performance using this method, and also observe a significant improvement after learning. Figure 7 shows an example match using both the unary and higher-order techniques.

Finally, figure 6 (right) shows the weights learned for this model. Interestingly, the first-order term during the second stage of learning has almost zero weight. This must not be misinterpreted: during the second stage, the response of each of the 20 candidate points is so similar that the first-order features are simply unable to convey any new information – yet they are still very useful in determining the 20 candidate points.

[9] Here the endpoint error is just the average Euclidean distance from the correct label, scaled according to the width of the image.

[10] We used publicly available implementations of both methods.

Table 1: Performance on the 'bikes' dataset. The endpoint error is reported, with standard errors in parentheses (note that the second-last column, 'higher-order' uses the weights from the *first* stage of learning, but not the second).

Detector/descriptor		**unary**	**+ learning**	**higher-order**	**+ learning**
SIFT [6]	Training:	0.335 (0.038)	0.319 (0.034)	0.234 (0.047)	0.182 (0.031)
	Validation:	0.343 (0.027)	0.329 (0.019)	0.236 (0.031)	0.257 (0.033)
	Testing:	0.351 (0.024)	0.312 (0.015)	0.302 (0.045)	0.311 (0.039)
Affine invariant/SIFT [17]	Training:	0.322 (0.018)	0.280 (0.016)	0.233 (0.042)	0.244 (0.042)
	Validation:	0.337 (0.015)	0.298 (0.019)	0.245 (0.028)	0.229 (0.032)
	Testing:	0.332 (0.024)	0.339 (0.028)	0.277 (0.035)	**0.231 (0.034)**

5 Conclusion

We have presented a model for near-isometric shape matching which is robust to typical additional variations of the shape. This is achieved by performing structured learning in a graphical model that encodes features with several different types of invariances, so that we can directly learn a "compound invariance" instead of taking for granted the exclusive assumption of isometric invariance. Our experiments revealed that structured learning with a principled graphical model that encodes both the rigid shape as well as non-isometric variations gives substantial improvements, while still maintaining competitive performance in terms of running time.

Acknowledgements: We thank Marconi Barbosa and James Petterson for proofreading. NICTA is funded by the Australian Government's *Backing Australia's Ability* initiative, and the Australian Research Council's *ICT Centre of Excellence* program.

References

[1] Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. PAMI **24** (2002) 509–522

[2] Mori, G., Belongie, S., Malik, J.: Shape contexts enable efficient retrieval of similar shapes. In: CVPR. (2001) 723–730

[3] Mori, G., Malik, J.: Estimating human body configurations using shape context matching. In: ECCV. (2002) 666–680

[4] Frome, A., Huber, D., Kolluri, R., Bulow, T., Malik, J.: Recognizing objects in range data using regional point descriptors. In: ECCV. (2004)

[5] Belongie, S., Malik, J.: Matching with shape contexts. In: CBAIVL00. (2000) 20–26

[6] Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV. (1999) 1150–1157

[7] Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV **61** (2005) 55–79

[8] Felzenszwalb, P.F., Schwartz, J.D.: Hierarchical matching of deformable shapes. In: CVPR. (2007)

[9] LeCun, Y., Huang, F.J., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. CVPR (2004) 97–104

[10] Carmichael, O., Hebert, M.: Shape-based recognition of wiry objects. PAMI **26** (2004) 1537–1552

[11] McAuley, J.J., Caetano, T.S., Barbosa, M.S.: Graph rigidity, cyclic belief propagation and point pattern matching. PAMI **30** (2008) 2047–2054

[12] Caetano, T., Cheng, L., Le, Q., Smola, A.: Learning graph matching. In: ICCV. (2007) 1–8

[13] Canny, J.: A computational approach to edge detection. In: RCV. (1987) 184–203

[14] Smith, S.: A new class of corner finder. In: BMVC. (1992) 139–148

[15] Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: ICML. (2004)

[16] Teo, C., Le, Q., Smola, A., Vishwanathan, S.: A scalable modular convex solver for regularized risk minimization. In: KDD. (2007)

[17] Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. **60** (2004) 63–86

[18] Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology (2007)