# View Matching with Blob Features

Per-Erik Forssén and Anders Moe
Computer Vision Laboratory
Department of Electrical Engineering
Linköping University, Sweden

## Abstract

*This paper introduces a new region based feature for object recognition and image matching. In contrast to many other region based features, this one makes use of colour in the feature extraction stage. We perform experiments on the repeatability rate of the features across scale and inclination angle changes, and show that avoiding to merge regions connected by only a few pixels improves the repeatability. Finally we introduce two voting schemes that allow us to find correspondences automatically, and compare them with respect to the number of valid correspondences they give, and their inlier ratios.*

## 1. Introduction

The mathematics behind conic sections in projective geometry is fairly well explored, see e.g. [9, 4, 11]. There are two main applications to conic correspondence: one is making use of the *absolute conic* for self-calibration [6, 4], and the other one is to detect actual conics in images, and match them [9, 11]. While conic matching works quite well, its application is limited to special situations, such as projective OCR [5] (where the printed letter ‘o’ can be used as a conic) or scenes containing human artifacts such as mugs and plates [9, 11].

This paper demonstrates that the mathematics behind conic matching can also be used to match *blob features*, i.e. regions that are approximated with ellipses (which are conic sections). This works well for a wide range of regions such as rectangles and other ellipse-like regions, and less well for some other region shapes. Thus we also derive (and test) a measure that can be used to judge whether a certain region is sufficiently well approximated by an ellipse.

Regions have also been used in wide baseline matching and 3D object recognition using affine invariants [8, 13]. As we will demonstrate, the use of homography transformation rules for conics allows us to match planar regions under quite severe inclination angles, i.e. the part of the homographic transformation that is not part of the affine transformation. This means that our results are relevant for both wide baseline matching and 3D object recognition.

A somewhat related approach to region based matching is presented in [1], where tangents to regions are used to define linear constraints on the homography transformation, which can then be found through linear programming. The connection is that a *line conic* describes the set of all tangents to a region. By matching conics, we thus implicitly match the tangents of the ellipse-approximated regions.

## 2. Blob features

We will make use of blob features extracted using a clustering pyramid built using robust estimation in local image regions [2]. Each extracted blob is represented by its average colour $\mathbf{p}_k$, area $a_k$, centroid $\mathbf{m}_k$, and inertia matrix $\mathbf{I}_k$. I.e. each blob is a 4-tuple

$$\mathcal{B}_k = \langle \mathbf{p}_k, a_k, \mathbf{m}_k, \mathbf{I}_k \rangle \ . \tag{1}$$

Since an inertia matrix is symmetric, it has 3 degrees of freedom, and we have a total of $3 + 1 + 2 + 3 = 9$ degrees of freedom for each blob. Figure 1 gives a brief overview of the blob detection algorithm.

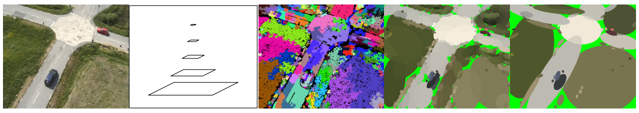


**Figure 1. Steps in blob detection algorithm.**

Starting from an image the algorithm constructs a clustering pyramid, where pixels at coarser scales are computed as robust averages of 12 pixel regions at the lower scale. Regions where the support of the robust average is below a threshold $c_{\text{min}}$ have a confidence flag set to zero. The algorithm then creates a label image using the pyramid, by traversing it top-down, assigning new labels to points which have their confidence flag set, but don't contribute to any robust mean on the level above. The labelling produces a set of compact regions, which are then merged in the final step by agglomerative clustering. Regions with similar colour, and which fulfil the condition

$$M_{ij} > m_{\text{thr}}\sqrt{\min(a_i, a_j)} \qquad (2)$$

are merged. Here $M_{ij}$ is the count of pixels along the common border of blobs $\mathcal{B}_i$ and $\mathcal{B}_j$, and $m_{\text{thr}}$ is a threshold. For more information about the feature estimation, please refer to [2], and the implementation [14], both of these are available on-line.

The current implementation [14] processes $360 \times 288$ RGB images at a rate of 0.22 sec/frame on a AMD Athlon 64, 3800+ CPU 2.4GHz, and produces relatively robust and repeatable features in a wide range of sizes.

The blob detection algorithm could probably replaced by a conventional segmentation algorithm, as long as the detected regions are converted to the form (1). As we will show later, the option to avoid merging regions which are only connected by a few pixels, according to (2) seems to be advantageous though. After estimation, all blobs with approximating ellipses partially outside the image are discarded. Figure 2 shows two images where $107$ and $120$ blobs have been found.

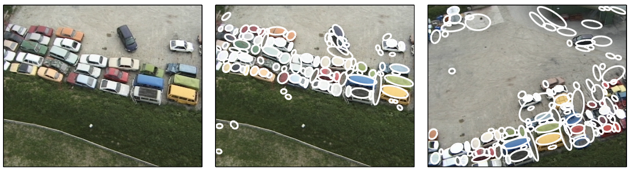


**Figure 2. Blob features in an aerial image. Left to right: Input image 1, Input image 1 with detected features ($107$ blobs), Input image 2 with detected features ($120$ blobs).**

The blob estimation [14] has two main parameters: a colour distance threshold $d_{\text{max}}$, and a propagation threshold $c_{\text{min}}$ for the clustering pyramid. For the experiments in this paper we have used $d_{\text{max}} = 0.16$ (RGB values in interval $[0, 1]$) and $c_{\text{min}} = 0.5$. Additionally, we remove blobs with an area below $a_{\text{min}} = 20$.

### 2.1. Shape ratio

From the eigenvalue decomposition $\mathbf{I}_k = \lambda_1\hat{\mathbf{e}}_1\hat{\mathbf{e}}_1^T + \lambda_2\hat{\mathbf{e}}_2\hat{\mathbf{e}}_2^T$ we can find the axes of the approximating ellipse as $2\sqrt{\lambda_1}\hat{\mathbf{e}}_1$ and $2\sqrt{\lambda_2}\hat{\mathbf{e}}_2$ [2]. Thus the approximating ellipse area is given by $4\pi\sqrt{\det \mathbf{I}_k}$. By comparing the approximating ellipse area, and the actual region area, we can define a simple measure of how ellipse-like a region is:

$$r_k = \frac{a_k}{4\pi\sqrt{\det \mathbf{I}_k}}\,. \qquad (3)$$

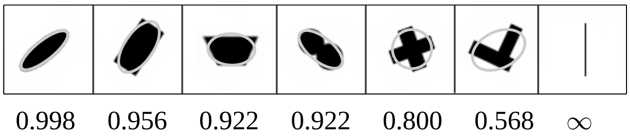


**Figure 3. A selection of shapes, and their shape ratios. Approximating ellipses are drawn in grey.**

Figure 3 shows a selection of shapes, their approximating ellipses, and their shape ratios. In the continuous case, the ellipse is the most compact shape with a given inertia matrix $\mathbf{I}_k$. Thus the shape ratio will be near 1 for ellipses, and decrease as shapes become less ellipse-like. The sole exception is the case where all region pixels lie on a line (see figure 3, right). Such regions are approximated by a degenerate ellipse with a zero length minor axis, and thus zero area. Such regions are automatically eliminated by the blob detection algorithm, by requiring that $\det \mathbf{I}_k > 0$.

## 3. Homography transformation of ellipses

We will now derive a homography transformation of an ellipse. Assume that we have estimated blobs in two images of a planar scene. The images will differ by a homography transformation, and the blobs will obey this transformation more or less well, depending on the shape of the underlying regions.

A blob $\mathcal{B}$ in image 1, represented by its centroid $\mathbf{m}$ and inertia $\mathbf{I}$ approximates an image region by an ellipse shaped region with the outline

$$\mathbf{x}^T\mathbf{C}\mathbf{x} = 0 \;\text{ for }\; \mathbf{C} = \frac{1}{4}\begin{pmatrix} \mathbf{I}^{-1} & -\mathbf{I}^{-1}\mathbf{m} \\ -\mathbf{m}^T\mathbf{I}^{-1} & \mathbf{m}^T\mathbf{I}^{-1}\mathbf{m} - 4 \end{pmatrix} \qquad (4)$$

see [2]. This equation is called the *point conic* form of the ellipse, and $\mathbf{x} = \begin{pmatrix} x_1 & x_2 & 1 \end{pmatrix}^T$ are image point coordinates in homogeneous form. To derive the mapping to image 2, we will express the ellipse in *line conic* form [4]. The line conic form defines a conic in terms of its tangent lines $\mathbf{l}^T\mathbf{x} = 0$. For all tangents $\mathbf{l}$ in image 1 we have

$$\mathbf{l}^T\mathbf{C}^*\mathbf{l} = 0\,, \quad \mathbf{C}^* = \begin{pmatrix} 4\mathbf{I} - \mathbf{m}\mathbf{m}^T & -\mathbf{m} \\ -\mathbf{m}^T & -1 \end{pmatrix} \qquad (5)$$

where $\mathbf{C}^*$ is the inverse of $\mathbf{C}$ in (4). A tangent line $\mathbf{l}$ and its corresponding line $\mathbf{l}'$ in image 2 are related according to $\mathbf{l} = \mathbf{H}^T\mathbf{l}'$, where $\mathbf{H}$ is a $3 \times 3$ homography mapping. This gives us

$$\mathbf{l}^{'T}\mathbf{H}\mathbf{C}^*\mathbf{H}^T\mathbf{l}' = 0 \text{ and we set } \mathbf{H}\mathbf{C}^*\mathbf{H}^T = \begin{pmatrix} \mathbf{B} & \mathbf{d} \\ \mathbf{d}^T & e \end{pmatrix} . \tag{6}$$

We recognise the result of (6) as a new line conic form

$$-e\mathbf{l}^{'T} \begin{pmatrix} 4\tilde{\mathbf{I}} - \tilde{\mathbf{m}}\tilde{\mathbf{m}}^T & -\tilde{\mathbf{m}} \\ -\tilde{\mathbf{m}}^T & -1 \end{pmatrix} \mathbf{l}' = 0\,. \tag{7}$$

This allows us to identify $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{I}}$ as

$$\tilde{\mathbf{m}} = \mathbf{d}/e \quad \text{and} \quad \tilde{\mathbf{I}} = (-\mathbf{B}/e + \tilde{\mathbf{m}}\tilde{\mathbf{m}}^T)/4\,. \tag{8}$$

Note that since this mapping only involves additions and multiplications it can be implemented very efficiently.

# 4. Blob comparison

To decide whether a potential blob correspondence $\mathcal{B}_i \leftrightarrow \mathcal{B}'_j$ is correct or not we will make use of colour, position and shape distances, which we will now define. For this to be meaningful, we assume that all elements in $\mathcal{B}'_j$ have been mapped through the appropriate transformations, and signify this by a '$\sim$' symbol on each mapped element e.g. $\tilde{\mathcal{B}}'_j$. For colour comparison the mapping could be a *colour correction*[12] according to known illumination, or estimated from image statistics. For position and shape comparison the mapping will be the homography transformation derived in section 3.

## 4.1. Colour distance

We define the distance between two colour vectors $\mathbf{p}_i$ and $\mathbf{p}'_j$ as:

$$d(\mathbf{p}_i, \mathbf{p}'_j)^2 = \left(\mathbf{p}_i - \tilde{\mathbf{p}}'_j\right)^T \mathbf{W} \left(\mathbf{p}_i - \tilde{\mathbf{p}}'_j\right)\,. \tag{9}$$

The matrix $\mathbf{W}$ defines the colour space metric. We have used $\mathbf{W} = \mathbf{T}^T\text{diag}^{-2}[\mathbf{d}]\mathbf{T}$ where

$$\mathbf{T} = \frac{1}{255}\begin{pmatrix} 65.4810 & 128.5530 & 24.9660 \\ -37.7970 & -74.2030 & 112 \\ 112 & -93.7860 & -18.2140 \end{pmatrix} \quad \text{and}$$
$$\mathbf{d}^T = \begin{pmatrix} 0.18 & 0.05 & 0.05 \end{pmatrix}^T\,. \tag{10}$$

The matrix $\mathbf{T}$ is the standard mapping from RGB to the YCbCr colour space (as defined in ITU-R BT.601) for RGB values in interval $[0, 1]$. The vector $\mathbf{d}$ thus contains scalings for the Y, Cb, and Cr components respectively. The purpose of this scaling is mainly to reduce the sensitivity to differences in illumination.

## 4.2. Position and shape distance

For two blobs with centroids $\mathbf{m}_i$ and $\mathbf{m}'_j$, we define their position distance as:

$$\begin{aligned} d(\mathbf{m}_i, \mathbf{m}'_j)^2 = & (\mathbf{m}_i - \tilde{\mathbf{m}}'_j)^T(\mathbf{m}_i - \tilde{\mathbf{m}}'_j) \\ & + (\tilde{\mathbf{m}}_i - \mathbf{m}'_j)^T(\tilde{\mathbf{m}}_i - \mathbf{m}'_j)\,. \end{aligned} \tag{11}$$

For two blobs with inertia matrices $\mathbf{I}_i$ and $\mathbf{I}'_j$, we define their shape distance as:

$$d(\mathbf{I}_i, \mathbf{I}'_j) = \frac{||\mathbf{I}_i - \tilde{\mathbf{I}}'_j||}{||\mathbf{I}_i|| + ||\tilde{\mathbf{I}}'_j||} + \frac{||\tilde{\mathbf{I}}_i - \mathbf{I}'_j||}{||\tilde{\mathbf{I}}_i|| + ||\mathbf{I}'_j||}\,. \tag{12}$$

Note that we apply the transformation both ways in order to remove possible bias favouring one direction of the transform.

# 5. Repeatability test

We will now demonstrate how the repeatability of the blobs is affected by view changes. To do this in a controlled fashion, we make use of a large aerial photograph ($3125 \times 5468$ pixels) of a city. We place a synthetic camera with focal length $f = 200$ above the image, at varying angles and distances (scale). We then compute blob representations of both the synthesised view ($200 \times 200$ pixels), and a cropped version of the aerial photograph, see figure 4.

We then transform the blobs in the synthetic view to the coordinate system of the aerial photograph, and vice versa, using the relations derived in section 3, and compute the position, shape and colour distances between blobs. Correspondences falling below a threshold in colour, position and shape are counted, and are used to determine the *repeatability rate* of the detector.

The *repeatability rate* [10] concept was originally used to compare interest point detectors, by computing the frequency with which an interest point detector gives repeated detection of the same 3D point in different 2D projections of the scene. We now extend the repeatability rate definition to regions, by also considering the shape and colour of the detected features

$$r(\sigma_s, \sigma_p, \sigma_c) = \frac{N_c(\sigma_s, \sigma_p, \sigma_c)}{\min(N_p, N_s)}\,. \tag{13}$$

Here $N_p$ and $N_s$ are the number of detected blobs in the overlapping parts of the photo and the synthesised view respectively, and $N_c$ is the number of correspondences found. The explicit formula for $N_c$ is

$$\begin{aligned} N_c(\sigma_s, \sigma_p, \sigma_c) = |\,& \left\{ \langle \mathcal{B}_i, \mathcal{B}'_j \rangle \,|d(\mathbf{p}_i, \mathbf{p}'_j) < \sigma_c \right\} \cap \\ & \left\{ \langle \mathcal{B}_i, \mathcal{B}'_j \rangle \,|d(\mathbf{m}_i, \mathbf{m}'_j)^2/\sigma_p^2 + d(\mathbf{I}_i, \mathbf{I}'_j)^2/\sigma_s^2 < 1 \right\} |\,. \end{aligned} \tag{14}$$

The parameters $\sigma_s$, $\sigma_p$, and $\sigma_c$ are shape, position and colour distance thresholds. We have used $\sigma_p = 7$ and

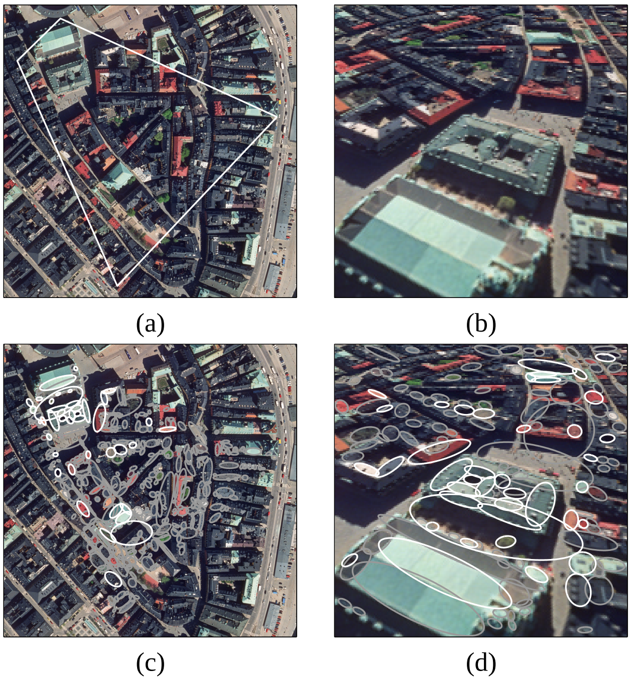
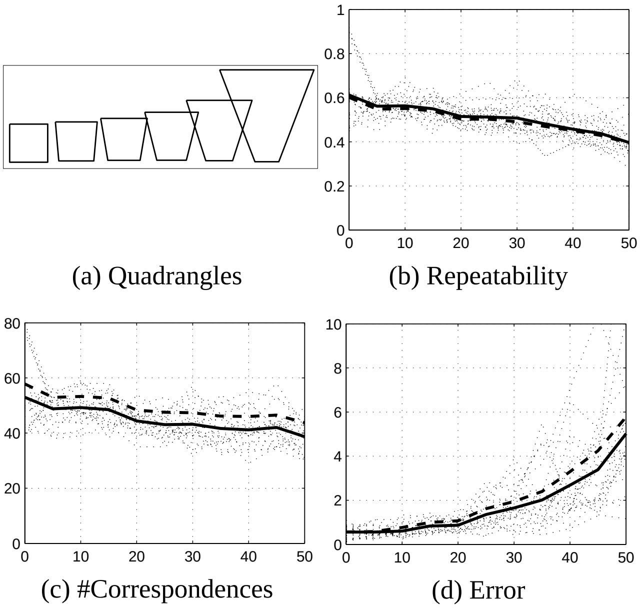

(a) (b) (c) (d)

**Figure 4. Experiment setup. (a) Cropped photograph with homography quadrangle overlaid. (b) Synthesised view. (c) Photograph with detected blobs painted in. (d) Synthesised view with detected blobs painted in. White blobs are valid correspondences, grey blobs are discarded.**



(a) Quadrangles (b) Repeatability

(c) #Correspondences (d) Error

**Figure 5. (a) Homography quadrangles for inclination angles $0 \ldots 50°$. (b) Repeatability as function of inclination angle. (c) Number of correspondences as function of inclination angle. (d) Error in homography estimate from correspondences, as function of inclination angle. Blobs with shape ratios below $0.7$ have been excluded. Solid curves are averages over all in-plane rotations. Thick dashed curves are averages without thresholding on shape ratio.**

$\sigma_s = 0.3$, which gives visually quite similar ellipses after transformation, and for colour we have simply set $\sigma_c = 1$.

Figure 5b shows how the repeatability is affected by the inclination angle. The angle is varied in the range $0 \ldots 50°$ which results in homography quadrangles according to figure 5a. As can be seen in the graphs, using the shape measure to remove non-elliptical shapes makes a minor improvement on the repeatability rate, at the price of reducing the number of correspondences (see figure 5c).

### 5.1. Homography error

To be able to compare an estimated homography with a known correct one we will now define a homography error measure. We do this by comparing where transformed points end up. The distance between two points $\mathbf{x}$ and $\mathbf{y}$, in homogeneous coordinates is

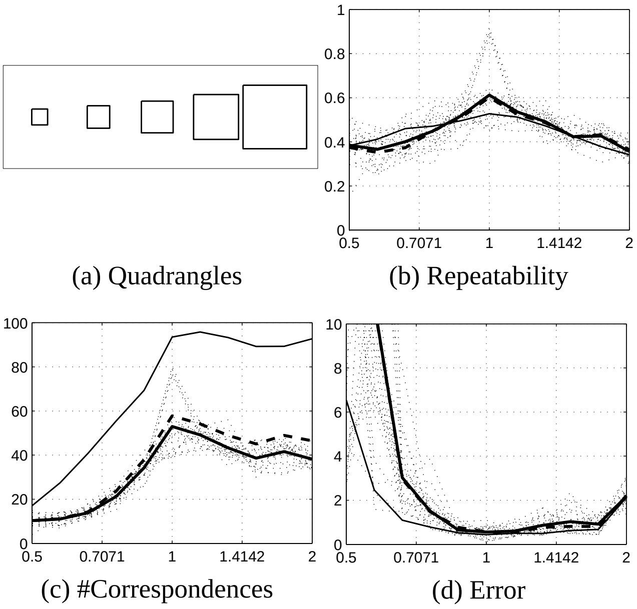
$$d(\mathbf{x},\mathbf{y})^2 = (x_1/x_3 - y_1/y_3)^2 + (x_2/x_3 - y_2/y_3)^2 \,. \quad (15)$$

For the synthesised image, we denote the four corner points by $\mathbf{x}_1 \ldots \mathbf{x}_4$. We can now write the error of the estimated homography $\mathbf{H}_{\text{est}}$ wrt. the correct one $\mathbf{H}$ as

$$\epsilon(\mathbf{H}_{\text{est}},\mathbf{H})^2 = \frac{1}{4}\sum_{k=1}^{4} d(\mathbf{H}_{\text{est}}\mathbf{x}_k, \mathbf{H}\mathbf{x}_k)^2 + \frac{1}{4}\sum_{k=1}^{4} d(\mathbf{H}_{\text{est}}^{-1}\mathbf{x}_k, \mathbf{H}^{-1}\mathbf{x}_k)^2 \,. \quad (16)$$

We have used the centroids of the correspondences found using (14) to estimate a homography (see e.g. [4]), and computed the error (16) with respect to the correct homography. The result is shown in figure 5d.

As can be seen, the error increases with the inclination angle. This is to be expected, since the centroids are merely affine invariants, and thus don't correspond to the same image points when we change the inclination angle. The purpose of this plot however, was to show that removal of non-ellipse-like shapes reduces the error slightly. This is despite the fact that fewer correspondences are used, and thus the



(a) Quadrangles (b) Repeatability

(c) #Correspondences (d) Error

**Figure 6. (a) Homography quadrangles for scales $0.5 \ldots 2.0$. (b) Repeatability as function of scale. (c) Number of correspondences as function of scale. (d) Error in homography estimate from correspondences, as function of scale. Blobs with shape ratios below $0.7$ have been excluded. Solid curves are averages over all in-plane rotations. Thick dashed curves are averages without thresholding on shape ratio. Solid thin curves used $300 \times 300$ synthetic views instead of $200 \times 200$.**

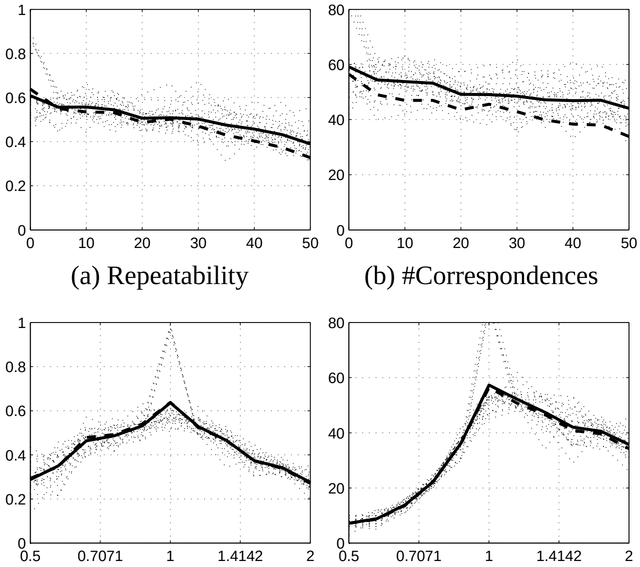
removed blobs (on the average 5 out of between 55 and 40, see figure 5c) contribute even more to the error than a first glance at these curves would suggest.

Note that all thick curves in figure 5 are averages over in-plane rotations (in steps of $15°$), thus we also demonstrate that the features are repeatable across rotations.

The result of changing scale instead of inclination angle is shown in figure 6. As can bee seen here, the repeatability drops to $0.4$ after a one octave scale change, in either direction. The homography error on the other hand increases slowly as we move away from the image, but increases dramatically as we move closer. The reason for this is that there simply aren't enough large features in an image of the chosen size. To demonstrate that this is an issue of image size, we redid the experiment with $300 \times 300$ synthetic views, instead of $200 \times 200$. As shown in the graphs, using the larger image size gives basically the same results, except for the error at small scales which is now smaller. The reason is that for $200 \times 200$ images there simply aren't enough features to compute the homography reliably.

### 5.2. Changing the merger threshold

To some extent it is possible to constrain which region mergers are allowed in the region merging step of the blob detection algorithm (see section 2). Figure 7 shows a comparison of setting the merger threshold to $0.5$ and $0$, where the latter corresponds to always merging connected regions (i.e. the strategy used in conventional segmentation algorithms). As can be seen, setting $m_{\text{thr}} = 0.5$ is clearly preferable when the perspective is changed. When only the scale changes however, the difference is negligible. The value $m_{\text{thr}} = 0.5$ has been selected by making similar comparisons.



(a) Repeatability (b) #Correspondences

(c) Repeatability (d) #Correspondences

**Figure 7. (a) Repeatability as function of inclination angle. (b) Number of correspondences as function of angle. (c) Repeatability as function of scale. (d) Number of correspondences as function of scale. Solid thick $m_{\text{thr}} = 0.5$, Dashed thick $m_{\text{thr}} = 0$.**

## 6. Solving the correspondence problem

Automatic matching of two views that differ by a homography is often accomplished by RANSAC sampling of an initial set of tentative correspondences [4]. The minimal number of corresponding conics needed to estimate a homography is 2, when the entire shape of the conic is used. See [5] for a description of how to do this estimation. A less advanced (but easier to implement) approach is to pick $4$ correspondences, and estimate the homography using the conic centroid points alone. Note that this requires more RANSAC samplings, and has a built in bias, since the centroids are only affine invariants.

An algorithm for automatic solving of the correspondence problem would be the following (see [3] for an earlier publication of this scheme):

1. Compute blob features in the two images to match.
2. Find a set of tentative correspondences using a voting scheme.
3. Apply RANSAC sampling to the set of tentative correspondences to eliminate outliers.
4. Re-estimate homography using the inliers (the consensus set). This step may be repeated a few times until convergence.

Ideally, step 4 should be replaced by an iterative optimisation that considers the entire conic shape, and not just the centroid.

### 6.1. Voting scheme 1

For the voting scheme we use a correspondence matrix $\mathbf{M}$ where each cell $M_{ij}$ estimates the likelihood that blob $\mathcal{B}_i$ in image 1 and blob $\mathcal{B}'_j$ in image 2 correspond. As a first screening, we only allow votes to cells where the colour distance is below a threshold $\sigma_c$.

As a second stage in the voting, we generate pairs of neighbouring blobs in both images by joining together spatially adjacent blobs. Each blob gets to form ordered pairs with its three nearest neighbours. Thus, if we had $N_1$ and $N_2$ blobs in the two images, we now have $3N_1$ and $3N_2$ blob pairs. We will now try to find correspondences of such blob pairs, i.e. $\langle\mathcal{B}_i, \mathcal{B}_k\rangle \leftrightarrow \langle\mathcal{B}'_j, \mathcal{B}'_l\rangle$. For each such correspondence, we first check if the colours match, using the colour distance. This excludes most candidates. For the correspondences that match, we then calculate a similarity mapping

$$\mathbf{x}' = \begin{pmatrix} s\mathbf{R} & t \\ \mathbf{0} & 1 \end{pmatrix} \mathbf{x} \,. \tag{17}$$

using the blob centroids. We then transform both blobs in the pair through the mapping, and compute their shape distances (12). Both distances are summed and added in a new correspondence matrix $\mathbf{S}$ according to

$$S_{ij} + e^{-(d_{ij}^2 + d_{kl}^2)/\sigma_s^2} \mapsto S_{ij} \tag{18}$$

$$S_{kl} + e^{-(d_{ij}^2 + d_{kl}^2)/\sigma_s^2} \mapsto S_{kl} \tag{19}$$

where $\sigma_s = 0.25$ is a shape distance scaling. This implements a soft voting scheme, where very few constraints on the image structure have been imposed. A set of tentative correspondences $\mathcal{B}_i \leftrightarrow \mathcal{B}'_j$ are now extracted from $\mathbf{S}$ by requiring that the position $S_{ij}$ should be a maximum along both row $i$ and column $j$, and above a threshold $S_{\text{thr}} = 0.5$.

### 6.2. Voting scheme 2

Since the similarity transform is quite a strong assumption, which does not hold for large inclination angles, we will also try an alternative voting scheme. We now replace the similarity transform with a projective invariant using a pair of coplanar conics, see e.g. [7], chapter 3. We proceed in the same way as for the similarity transform, by generating pairs of neighbouring blobs. For correspondences $\langle\mathcal{B}_i, \mathcal{B}_k\rangle \leftrightarrow \langle\mathcal{B}'_j, \mathcal{B}'_l\rangle$ that match in colour, we then calculate the projective invariants

$$I_1(i,j) = \text{tr}(\hat{\mathbf{C}}_i^{-1}\hat{\mathbf{C}}_j) \text{ and } I_2(i,j) = I_1(j,i)\,. \tag{20}$$

Note that the conics must be normalised according to $\hat{\mathbf{C}}_i = \mathbf{C}_i/(\det \mathbf{C}_i)^{1/3}$ for (20) to be invariants. To evaluate a correspondence pair, we then use the absolute distances between these invariants

$$\begin{aligned} d((i,k),(j,l)) = &|I_1(i,k) - I_1(j,l)| \\ &+ |I_2(i,k) - I_2(j,l)| \end{aligned} \tag{21}$$

and fill in the correspondence matrix $\mathbf{S}$ according to

$$S_{ij} + e^{-d((i,k),(j,l))/\sigma_s^2} \mapsto S_{ij} \tag{22}$$

$$S_{kl} + e^{-d((i,k),(j,l))/\sigma_s^2} \mapsto S_{kl} \tag{23}$$

where $\sigma_s = 1.0$ is a shape distance scaling. Correspondences are then extracted from $\mathbf{S}$ in the same way as before.

### 6.3. RANSAC cleanup

When applying RANSAC to the initial sample set, we simply pick four of our tentative correspondences, compute a homography $\mathbf{H}$ using the centroid positions, and count the number of inliers wrt. $\mathbf{H}$. This is repeated until a sufficiently large number of inliers is found. A potential correspondence is counted as an inlier if the colour distance is below a threshold $\sigma_c$, and the shape and position is inside an ellipse with radii $\sigma_p$ and $\sigma_s$, just like in the repeatability test, (cf. (14)). Note that since all potential correspondences are considered, this allows us to find correspondences that were not found in the voting step.

When sufficiently many inliers are found (here $N_{\text{min}} = 15$), we re-estimate $\mathbf{H}$ using the inliers (the consensus set). This step is repeated until $\mathbf{H}$ stops changing.

## 7. Evaluation

We have tried the algorithm above on the data-set used for the repeatability test (see section 5). For most views the algorithm ends up with the correspondences found in the repeatability test (it fails for 2 of the images at $50°$ inclination, (using projective invariant voting)). More interestingly however this experiment allows us to compare the two voting strategies. As can be seen in the graphs in figure 8, the similarity transform voting does far better for scale changes, and is also better for moderate inclination changes. The projective invariant is however better at high inclination angles (above $35°$ if number of inliers is considered.).

A high inlier fraction $1-\epsilon$ is quite important since the expected number of RANSAC steps varies dramatically with $\epsilon$. As a rule of thumb, the number of required samples to draw to obtain an uncontaminated one, for sample size $K = 4$ and likelihood $m = 0.99$ is given by [4]:

$$N = \frac{\log(1-m)}{\log(1-(1-\epsilon)^K)} \,. \tag{24}$$

Some examples relating to figure 8 are given below.

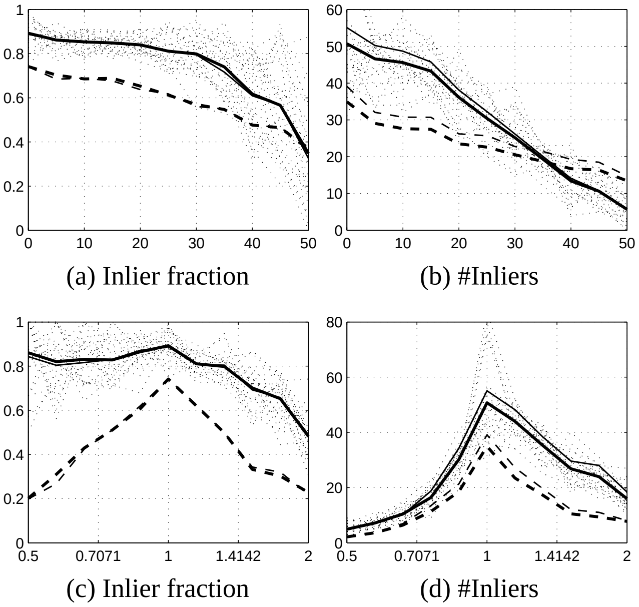
| $\epsilon$ | 0.1 | 0.25 | 0.5 | 0.7 | 0.8 |
|---|---|---|---|---|---|
| $N$ | 5 | 13 | 72 | 567 | 2876 |

## 8. Concluding remarks

In this paper we have introduced a new region based feature, and demonstrated its stability across scale and plane projective view changes. We have shown that avoiding to merge regions connected by only a few pixels improves repeatability (and at the same time gives more correspondences per image pair). To judge whether a region is ellipse-like (and thus transform as a conic) we have introduced a measure of how ellipse-like a region is. Removing non-ellipse like regions gives a small improvement in repeatability, and the remaining regions gives a slightly better homography estimate. We have also introduced two voting schemes, that allow region correspondences to be found automatically in a robust fashion. When comparing the voting strategies, it was found that similarity transform voting was superior under scale, and moderate inclination angle changes. For large inclination angles, however the projective invariant voting did better. Thus, the similarity transform voting is to be preferred whenever we know that inclination angles are small, but without any a priori information we should probably use the projective invariant voting.

## Acknowledgements

The work presented in this paper was performed in the scope of the VISCOS project (funded by SSF) and the EU 6:th framework programme COSPAL.



(a) Inlier fraction (b) #Inliers

(c) Inlier fraction (d) #Inliers

**Figure 8. (a,b) Perspective change. (c,d) Scale change. (a,c) Fraction of correspondences from voting that were inliers. (b,d) Total number of inliers from voting. Solid curves are averages when the similarity transform voting is used. Dashed curves show result from the projective invariant voting. Thick curves show the result when blobs with shape ratios below $0.7$ have been excluded, thin curves show results without thresholding. Dotted curves are single view measurements corresponding to the solid thick curves.**

## References

[1] R. Basri and D. W. Jacobs. Projective alignment with regions. *IEEE PAMI*, 23(5):519–527, May 2001.

[2] P.-E. Forssén. Low and medium level vision using channel representations, March 2004. Dissertation No. 858, ISBN 91-7373-876-X.

[3] P.-E. Forssén and A. Moe. Blobs in epipolar geometry. In *Proceedings SSBA'04 Symposium on Image Analysis*, pages 82–85, Uppsala, March 2004. SSBA.

[4] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

[5] M. P. Kumar, C. V. Jawahar, and P. J. Narayanan. Geometric structure computation from conics. In *Proceedings of the Indian Conference on Vision, Graphics and Image Processing (ICVGIP)*, Kolkata, India, 2004.

[6] Q.-T. Luong and O. D. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *IJCV*, 3(22):261–289, 1997.

[7] J. L. Mundy and A. Zisserman, editors. *Geometric Invariance in Computer Vision*. The MIT Press, Cambridge, MA. USA, 1992. ISBN 0-262-13285-0.

[8] S. Obdrzálek and J. Matas. Object recognition using local affine frames on distinguished regions. In *13th BMVC*, pages 113–122, September 2002.

[9] L. Quan. Conic reconstruction and correspondence from two views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):151–160, February 1996.

[10] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *Int. Journal of Computer Vision*, 37(2):151–172, 2000.

[11] C. Schmid and A. Zisserman. The geometry and matching of lines and curves over multiple views. *International Journal of Computer Vision*, 4(30):194–234, 2000.

[12] L. V. Tran. *Efficient Image Retrieval with Statistical Color Descriptors*. PhD thesis, Linköping University, 2003. Thesis No. 810.

[13] T. Tuytelaars and L. V. Gool. Matching widely separated views based on affinely invariant neighbourhoods. *International Journal of Computer Vision*, 1(59), August 2004. To appear.

[14] A. web page. `http://www.isy.liu.se/∼perfo/software/`.