On Line-based Homographies in Urban Environments

Nils Hering[1], Lutz Priese[1] and Frank Schmitt[2]
[1]*Institute of Computational Visualistics, University of Koblenz Landau, Universitätsstraße 1, Koblenz, Germany*
[2]*TOMRA Sorting GmbH, Otto-Hahn-Str. 6, Mülheim-Kärlich, Germany*

Keywords: Building Contours, Line Homography, Registration.

Abstract: This paper contributes to matching and registration in urban environments. We develop a new method to extract only those line segments that belong to contours of buildings. This method includes vanishing point detection, removal of wild structures, sky and roof detection. A registration of two building facades is achieved by computing a line-based homography between both. The known crucial instability of line homography is overcome with a line-panning and iteration technique. This homography approach is also able to separate single facades in a building.

1 INTRODUCTION

The location and identification of buildings from 2D images is a task under much research. There are many applications, e.g. in Google Street View and similar concepts of other companies. A combination of GPS and a compass in a cellular phone can give you rough information on the landmarks in front of you. The approach of pose (position and direction) estimation by matching markers is also well known. We will contribute here to an approach to markerlessly match and identify a building by its contours.[1] We start with two images I_1, I_2 of the same urban environment from different points of view. To match both images we have two problems:

• Firstly to detect contours of buildings. We present a rather refined technique to locate and identify the line segments that solely belong to buildings. This implies to suppress even strong straight contours not coming from buildings, even if they intersect orthogonally. We will call the resulting set of contours a "blueprint", simply because of its visual similarity to a buildings blueprint.

• Secondly to compute a homography between a building b in I_1 and I_2. Point features, such as SIFT or Harris corners, e.g., are usually quite similar at different locations of b and thus cannot be easily used for corresponding pairs of points in I_1 and I_2 to calculate a homography. The same holds for the endpoints of corresponding line segments: usually line segments of the same line end in two different images in different visible or detectable endpoints. Therefore, we use corresponding straight lines in a line-based homography. It is known that a line-based calculation of a homography is highly unstable. We present a new algorithm *lineHomography* to overcome this instability that operates with an iterated panning and iteration technique.

We approach the task to match buildings from image I_1 and I_2 by the following process:

1. Blueprint reconstruction
2. Detection of initial line segment correspondences
3. Initial homography estimation
4. Homography refinement
5. Isolation of single facades
6. Decision: building (facade) matched or not

In this paper we examine the steps 1,3,4,5, and 6. We omit the second step and assume the initial correspondences to be known. Several methods for the creation of initial line segment correspondences are known in literature like (Guerrero and Sags, 2003) and (Bay et al., 2005), see section 2.

2 PREVIOUS WORK

The analysis of aerial images of urban environments is widely studied, see, e.g. (Mayer, 1999) and (Baltsavias, 2004) for an overview. However, detection, analysis and identification of buildings in ground level

[1]This work was supported by the DFG under grant PR161/12-2 and PA599/7-2

Hering N., Priese L. and Schmitt F..
On Line-based Homographies in Urban Environments.
DOI: 10.5220/0004293603580369
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2013), pages 358-369
ISBN: 978-989-8565-47-1

images has been studied much less and is only recently gaining more and more attention with the advent of cameras in mobile computing devices like smartphones and the global availability of ground level images from services like Google Street View.

A general approach for detection of man-made objects in images has been introduced in (Kumar and Hebert, 2003). The algorithm is based on a causal multiscale random field where the distribution of multiscale feature vectors is modeled as a mixture of Gaussians and a set of features typical for man-made structures is presented. However, this approach seems more suitable for natural scenes than for urban environments.

In (Iqbal and Aggarwall, 1999) an algorithm which detects whether an image contains a building or not is presented. For this, the authors search for typical line junctions in the image. However, they don't detect the location of the buildings in the images.

Our approach is inspired by the works of David (David, 2008) who searches for clusters of lines with consistent orientation. However, he uses a less involved algorithm for edge extraction and orientation analysis.

For facade identification an algorithm is presented in (Zhang and Košecká, 2007), where the currently analyzed facade is matched against a set of known facade images. First a rough preselection is done by matching color histograms of the facades, whereby simple 1D hue value histograms are used. Afterwards, the best matching reference facade is found by the comparison of SIFT-features (Lowe, 2003).

In (Schmid and Zisserman, 1997) the authors introduce a method which matches line segments between images by the usage of the known F-matrix. The aim of this method is to create a basis for the 3D reconstruction of scenes containing planar surfaces.

In (Guerrero and Sags, 2003) the authors present a technique for simultaneous line matching and homography estimation. They start with matching lines from one image to their nearest neighbor in another image using a brightness and a geometric criterion. These first matching candidates are used to compute a homography which is in turn used to reject wrong matches and to gain new matches. They use their method mainly for robot homing. In a continuative work (López-Nicolás et al., 2005) a method to compute the motion between two views without knowing 3D structure is presented. In this work the basic matching entities have changed from lines to points.

In (Bay et al., 2005) a method for matching line segments is introduced. Their scope are uncalibrated wide-baseline images. They first find matching candidates by a color histogram based assginment which

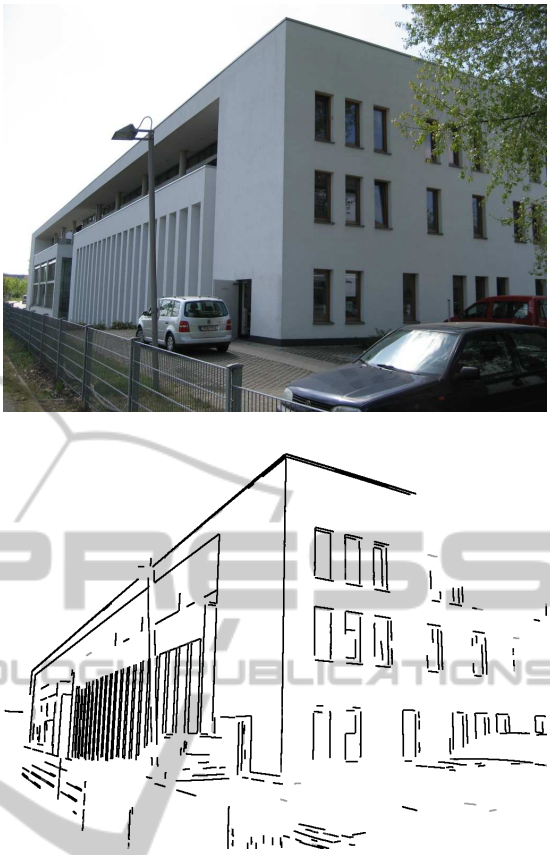

Figure 1: Pair of original and processed image.

is improved by a topological filter. Afterwards homographies are computed on the base of the initial matches to gain more matches and to estimate the epipolar geometry.

An extension of the DLT-algorithm (see (Hartley and Zisserman, 2004)) by lines is presented in (Dubrofsky and Woodham, 2008). The authors use both point and line correspondences in this point-centric-normalized DLT to match images from hockey videos to a hockey rink model.

3 BLUEPRINT RECONSTRUCTION

To clearly understand a blueprint reconstruction we start with the results first. The pair of images in figure 1 present in color a 2D image from an architectural environment (the *original* image). Below of the original image the result of our contour extraction algorithm is presented (the *blueprint* image).

In the blueprint image the essential parts of the building contours are clearly visible while almost all

other contours are suppressed. If one has some knowledge on the geometry of the buildings - say, there occur only angles of 90 between adjacent straight lines - even a partial 3D reconstruction from the blueprint images should become possible.

A freshman in computer vision may ask, where is the difficulty? Use some straight line detector and the processing is mainly done. Such straight line detectors are known for decades. They usually transform the image into grayscale, apply a Canny edge detector and a Hough transformation. Finally, match the lines from the Hough transformation with the original image to get the contours. However, every expert who has some experience with straight lines knows that this will lead to much poorer results, not even close to the quality in our examples shown e.g. in figure 1. In principle we follow the mentioned approach but have to add a whole series of new techniques that we present now.

3.1 Extraction of Edge Points and Straight Lines

We regard an image I as a mapping $I : Loc_I \rightarrow Val_I$ and a pixel $P \in I$ as a pair $P = (p,v)$ of a position in Loc_I and a value in Val_I. Of course, Loc_I is usually a 2-dimensional and Val_I a 1- (gray values) or 3-dimensional (color values) array of integers. $val(p)$ is the value of the pixel at position p.

3.1.1 Extraction of Edge Points

In the first step of our processing chain, we strive to extract prominent edge points from the image. For this we use the classic Canny edge detector (Canny, 1986). The result of the canny detector is a set of straight and curved lines in the image corresponding to edges. This set can be used to generate a binary image I^e of all edges as shown in figure 2.

3.1.2 Straight Line Detection

For a straight line detection one usually applies a Hough transform on the edges detected by the Canny algorithm. A *Hough transformation* of I is a function $h_I : A \rightarrow \mathbb{N}$. A is called the *accumulator*, an element $b \in A$ a *bin*. Usually A is of some higher dimension and a bin is a formal description of an object or a set of objects in I. $h_I(b)$ counts how often the object described by b appears in the image I. A straight line is represented in the *Hesse normal form* and the origin $(0,0)$ of $\mathcal{R}^2$ is thought to be in the middle of Loc_I. The Hesse normal form describes a straight line l by two parameters: α, the *angle* between the normal of the line and the x-axis, and d, the *distance* between line and image origin. Thus, the accumulator A is 2-dimensional with one coordinate for α and the other for d.

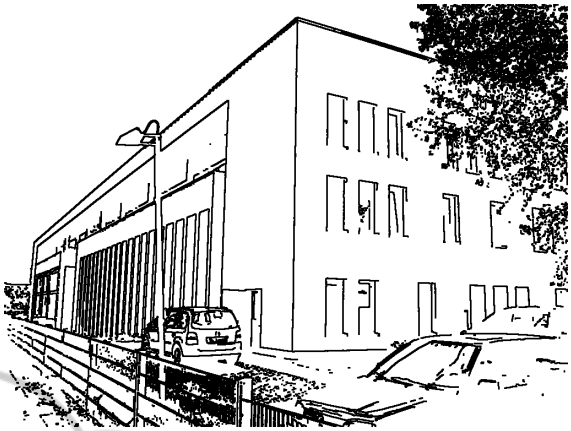

Figure 2: Result of Canny edge detector.

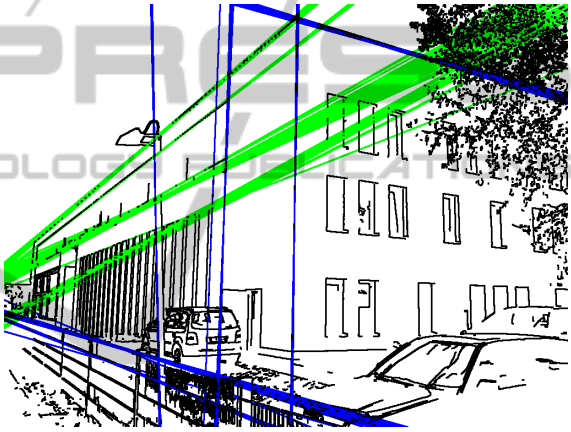

Figure 3: Result of unsophisticated Hough transformation.

Figure 3 shows the 120 strongest lines found by an unsophisticated Hough transformation in the canny output of figure 2. Clearly, such a result is not sufficient for further analysis.

3.1.3 Thinning

We firstly thin out straight lines closely neighbored in the accumulator. If na is the number of angles and nd the number of distances representable with the chosen accumulator we apply a non-maxima-suppression with a square window of width $\frac{\sqrt{na^2+nd^2}}{100}+1$. Thinning clearly improves results, however it is not sufficient as frequently many prominent lines in the image are not detected and too often a single line in the original becomes a bundle of lines in the transformation.

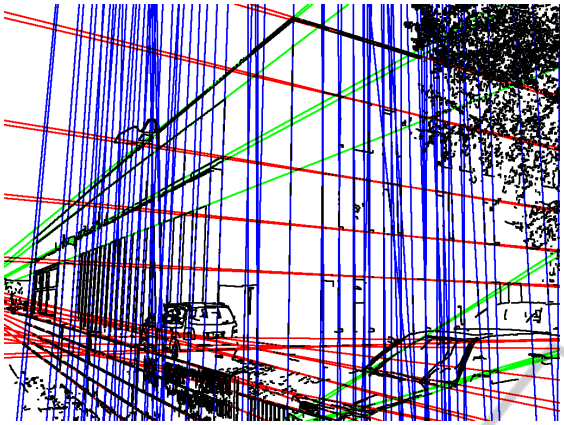

Figure 4: Result of Hough transformation with thinning and butterfly filtering.

3.1.4 Butterfly Filtering

A second filter from (Leavers and Boyce, 1987) removes accumulator maxima resulting from edge points which are not part of longer lines in the image (in outdoor images, such edge points typically occur, e.g., in trees) by increasing accumulator values in the middle of the typical butterfly formed structures resulting from true lines while lessening other accumulator values. When no such filter is applied, the numerous edges in strongly textured structures often yield artificial straight lines in the Hough transformation. Figure 4 shows the best 120 lines after thinning and butterfly filtering. All prominent lines are detected successfully.

3.1.5 Removal of Wild Structures

Heavily textured areas tend to produce "false" lines in the Hough transformation, even if a thinning and butterfly filter is applied. To avoid this, we have developed in our group a "wilderness map" which characterizes these parts of an image and masks them.

Our algorithm calculates for each pixel a binary measure classifying the pixel either as wild or non-wild using binary morphology on the edge image from the Canny edge detector I^e. We apply binary closing with a circular structuring element of radius 4 on I^e to get the image I^e_r. In I^e_r all gaps of a width of less than 4 pixels are closed. Afterwards an erosion with a circular structuring element of radius 1 is applied on I^e_r which removes all outer edge points from I^e_r. Finally we remove all foreground (e.g. edge) pixels from I^e which are also foreground in I^e_r. The radius 4 for the closing operation is suitable for our application scenario of urban environments with images of size 1024×768 pixels but might have to be adapted otherwise.

After wild structures are thus removed from the Canny result I^e, a Hough transformation is applied. Figure 5 shows the effect of wilderness removal on I^e, figure 6 the effect on the Hough transform. Many false lines resulting from random Moire structures (e.g., in the window shutters) are successfully suppressed. Let I^l denote the binary image of all lines detected by those combined techniques.

3.2 Vanishing Point Detection

Our improved Hough transform reliably detects the most prominent straight lines in the image. However, it remains to choose from these lines those which are actually part of the favored structures. As we are mainly searching for structures planar in 3D all parallel lines in those structures will point towards a common point, the vanishing point. In images showing architectural environments there are typically two or three vanishing points corresponding to lines pointing upwards, left, and right. We use the vanishing point detector introduced in (Schmitt and Priese, 2009a). This new detector uses several ideas from literature and combines them with a new *intersection point neighborhood*.

The size of this neighborhood depends on the number of possible intersections of measurable lines, and, thus, depends on the way one actually computes the straight lines.

A canonical next step is a cluster analysis of the measured intersection points. Unfortunately, the intersection neighborhood defines no topology in the mathematical sense and a cluster analysis technique without a concept of a distance is required. For this an adaption of the AGS algorithm introduced in (Priese et al., 2009) is used that computes an intersection rate of two sets as a substitute for their distance.

Finally, geometric considerations on the candidates generated by the cluster analysis give the final vanishing points. Those detected lines pointing to a vanishing point are further candidates for contours of buildings.

3.3 Back Projecting into the Image

3.3.1 Straightforward Approach

The lines in I^l are without endpoints. To get the contours of the desired structures one may match those lines in I^l that point to one of the detected vanishing points with lines segments in the original image, say, with the edge points generated by the Canny detector. Let I^{vl} be the resulting binary image of all lines pointing to a vanishing point in their correct length, i.e.,

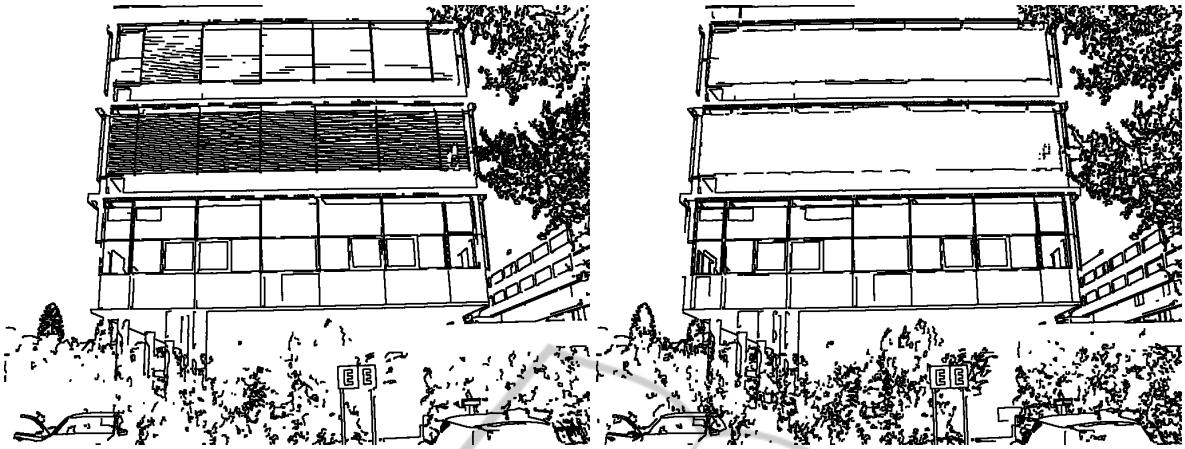

Figure 5: Result of Canny edge detector and result after removal of wild structures.

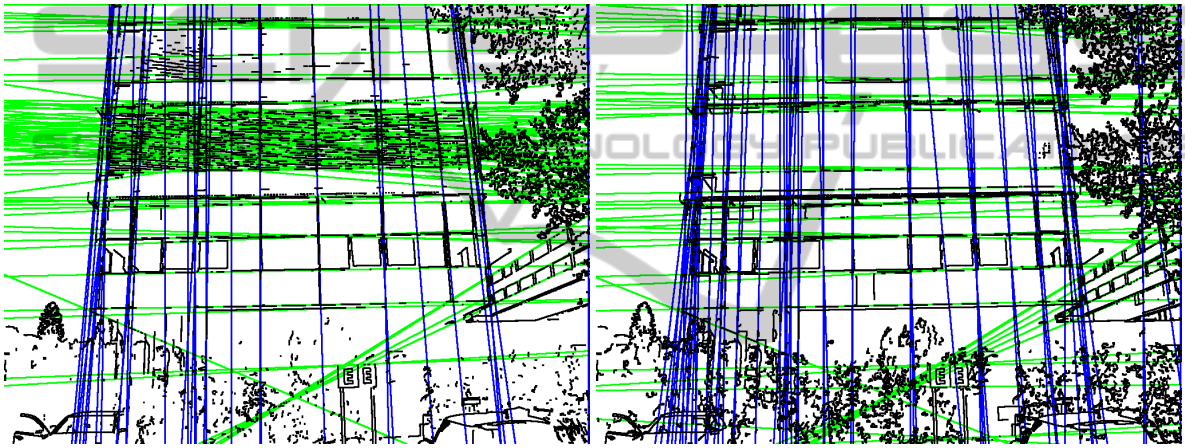

Figure 6: Result of Hough transformation on normal edge image and result after removal of wild structures.

with endpoints. Figure7 presents I^{vl} of an example. At a first glimpse the results look promising, but there are two problems.

1. Often the roofs of buildings are not parallel to the main directions of the building and, thus, do not point to any detected vanishing point.
2. Not all edges that are part of a contour of a building appear in a straight line in the Hough transform at all. E.g. window frames often do not contain enough edge points to generate a Hough line.

   We therefore do not follow this route.

3.3.2 Refined Approach

To solve problem 1 we compute roof line segments with a very different method. We firstly apply the sky detector introduced in (Schmitt and Priese, 2009b). This sky detector finds the sky rather reliably, no matter whether the sky is cloudless, partially clouded or overclouded, and works in short as follows. I is enhanced by a Kuwahara filter (M. Kuwahara, K. Hachimura, S. Eiho, and M. Kinoshita, 1976) that smoothens in rather homogeneous regions and simul-

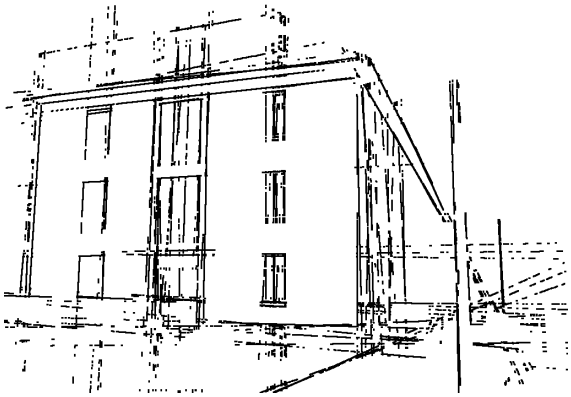

Figure 7: Insufficient result after simple back projection of Hough lines into Canny image.

taneously sharpens edges. After this a CSC segmentation (Priese and Rehrmann, 1993) is applied. The CSC divides those parts of the sky where the color is smoothly changing into regions with irregular bounds. However, parts of a building touching the sky are segmented into regions with a very regular boundary towards sky segments, even if the color of the building is similar to the sky. This distinction between regular and irregular bounds allows to find a rather stable sky line. Therefore, this technique would not work with a split-and-merge segmentation technique (Horowitz and Pavlidis, 1976). Roof lines are now found by simply looking for straight parts of the sky line.

To solve problem 2 we do not combine the detected vanishing points with the results from the Hough transformation but directly with the Canny image. Note, this doesn't imply that Hough is not required as a Hough transformation is essential for the vanishing point detection. However, after knowing the vanishing points the Hough lines are no longer needed. The Canny image $I^C : Loc_I \to \mathcal{R}$ [2] gives for any location a gradient strength and gradient direction (of the detected edge), where too weak edge points are suppressed and small gaps in longer edges are closed. The tangent at an image position $p \in Loc_I$ is the straight through p with direction orthogonal to the gradient direction at p. A *C-line* in I^C is a connected line in the Canny image, not necessarily straight. We now process all C-lines in the following way:

- Remove all C-lines with a length of less then 10 pixels.
- Split all C-lines s into sub-C-lines $s_1,..,s_n$ such that for each sub-C-line s_i there holds:
  - the length of s_i is at least 10,
  - the tangents at all pixels in s_i pass one vanishing point in a close distance.

The generated sub-C-lines and the roof lines together form a first blueprint.

3.4 Enhancement

We now enhance this first version of the blueprint and erase all rather short line segments that are parallel to a close-by longer line segment. Then we close gaps between line segment partners. Two lines segments l_1, l_2 are partners if one endpoint of l_1 and one of l_2 are in close proximity and l_1 and l_2 point in the same direction.

4 HOMOGRAPHY BASICS

The following chapter shall give a short summary of the homography estimation. Especially the line based homography shall be introduced. Dubrowsky et al. describe the line-based homography estimation more precisely in (Dubrofsky and Woodham, 2008). $\mathbb{P}^2$ denotes the *projective plane*. A *homography* is a linear invertible mapping $h : \mathbb{P}^2 \to \mathbb{P}^2$ where the following holds: three points p_1,p_2 and p_3 lie on one line if and only if $h(p_1)$,$h(p_2)$ and $h(p_3)$ lie on one line.

The general problem is to find a homography h which maps points $p_i \in \mathcal{R}, 1 \leq i \leq N$, from a plane π in image I to their corresponding points $p_i^{'} \in \mathcal{R}, 1 \leq i \leq N$, on the plane $\pi^{'}$ in image $I^{'}$. The correspondences $(p_i, p_i^{'})$ are known and the homography h that satisfies $p_i^{'} = Hp_i$ for all correspondences is in demand.

4.1 The DLT Algorithm

Let $p = (x,y)$ be a point in a two-dimensional image. This point p may be represented as a three-dimensional vector $p_v = (a_1, a_2, a_3)$ with $x = \frac{a_1}{a_3}$, $y = \frac{a_2}{a_3}$ and $a_3 \neq 0$. The vector p_v is called a *homogenous representation* of p and lies on the projective plane $\mathbb{P}^2$.

To compute the projective transformation, the *homography matrix* H has to be found. H can be changed by a multiplication with a constant $c \neq 0$ without changing the transformation. Hence, H is a *homogenous matrix* and has eight degrees of freedom while it has nine elements. Thus to find H eight unknowns have to be determined. We usually identify a homography h and its matrix H.

The **D**irect **L**inear **T**ransform algorithm is an algorithm for the determination of homographies. $p_i^{'} = Hp_i$ can also be written as $c\binom{u}{v}_1 = H\binom{x}{y}_1$ with $c \neq 0$ and

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}.$$

With division of line one by line three and line two by line three one gets the following two equations:

$$-h_{11}x - h_{12}y - h_{13} + (h_{31}x + h_{32}y + h_{33})u = 0$$

$$-h_{21}x - h_{22}y - h_{23} + (h_{31}x + h_{32}y + h_{33})v = 0$$

One can shape this into $Ah = 0$ with

$$A = \begin{pmatrix} -x & -y & -1 & 0 & 0 & 0 & ux & uy & u \\ 0 & 0 & 0 & -x & -y & -1 & vx & vy & v \end{pmatrix}$$

and

$$h = (h_{11}\ h_{12}\ h_{13}\ h_{21}\ h_{22}\ h_{23}\ h_{31}\ h_{32}\ h_{33})^T.$$

To be able to solve the eight unknowns in H the matrix A has to consist of at least eight lines. That means for the number of correspondences that $N \geq 4$ has to hold. Every point correspondence produces two lines in A. Hence, four point correspondences are necessary. The null space of A contains the solutions for h. In the case of digital images there is no solution for $Ah = 0$ because of discretization and noise. Therefore, an optimized solution that minimizes Ah has to be found. The state of the art optimization method in this case is *singluar value decomposition (SVD)*.

In (Hartley and Zisserman, 2004) the authors reveal that the results of the DLT-algorithm depend on the origin, scale and orientation of the image. Hence, they advise a data normalization step to improve the DLT's results. This normalization steps consists basically of a coordinate transformation to accomplish that the new origin is the centroid of the points and the average distance from the origin is $\sqrt{2}$.

Consequences for Blueprints. The first idea might be to use the end points of the corresponding line segment sets of blueprints as point correspondences to compute the homography H. If one regards the pair of blueprints shown in figure 8 it is easy to identify the main problem with that. Especially the line segments which form the left and right border of the building facade appear clearly different.

It is a rather common effect in our blueprints that the endpoints of corresponding line segments are nor corresponding themselves. Thus, using the end points of the line segments as correspondences would cause wrong homographies and we decided to use the straight lines defined by their segments instead.

4.2 The DLT with Lines

In the case of a line-based homography the point correspondences are replaced or assisted by line correspondences. These are lines l_j in π and their corresponding lines $l_j^{'}$ in $\pi^{'}$.

A line l in the two-dimensional space can be represented in *main form* by $ax + by + c = 0$. Therefore, it can be also understood as a vector $(a\ b\ c)^T$. This is a *homogenous representation* with two degrees of freedom as $f \cdot (a\ b\ c)^T$ represents the same line as $(a\ b\ c)^T$ for any factor $f \neq 0$.

A point x lies on a line l if $x^T l = 0$ and $l^T x = 0$ respectively. Let $x^{'}$ be the point corresponding to x and $l^{'}$ the line corresponding to l. With $x^{'} = Hx$ follows from $l^{'^T} x^{'} = 0$ that $l = H^T l^{'}$. Consequently, it follows: $c\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = H^T \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$ Transformed analogously follows for lines:

$$A_i = \begin{pmatrix} -u & 0 & ux & -v & 0 & vx & -1 & 0 & x \\ 0 & -u & uy & 0 & -v & vy & 0 & -1 & y \end{pmatrix}$$

The matrix A can be build of (restricted) combinations of point and line correspondences accordingly. Of course a matrix consisting of only of line correspondences is possible, too.

In (Dubrofsky and Woodham, 2008) Dubrofsky et. al. point out that they know no corresponding normalization step for line based homographies. In their application with mixed point and line correspondences they obtain the point-based coordinate transformation and apply it to both point and line correspondences.

The example in figure 9 clarifies the instability of line-based homographies. The first image shows the same polygon with four corners on two corresponding planes π (red polygon p) and $\pi^{'}$ (blue polygon p') resulting from different points of view in 3D. Using perfect correspondences (four points or four lines, respectively) leads to a homography that maps p perfectly onto p', both with point- or line-based DLT. Now we have changed the upper left corner of the red polygon p on plane π_1 by just by one pixel in x- and in y-direction to a new polygon p_1. We have applied DLT with the corners of both polygons as corresponding points to compute a homography H . The middle image shows both $H(p_1)$ and p', still an almost perfect match with just a slight error. However, if we compute a homography H' with the corresponding four lines of p_1 and p' with DLT H' becomes completely corrupt. The right image shows $H'(p_1)$ and p'.

Thus, a straight forward application of line-based DLT must fail in practice. Maybe this is the reason why some authors working with line-based homography have switched to point-based.

Consequences for Blueprints. Our blueprint algorithm detects line segments based on edges. In general those edge points are an imprecise and incomplete sampling of the real image edge. This problem is paired with the general discretization error in digital images.

Partially or wrong covered image line segments in the blueprint can cause clearly different underlying lines as shown in figure 10. Thus, a straight forward application of a line-based DLT must fail even for our rather sophisticated blueprints.

Figure 8: Two blueprints of the same building from slightly different views. The different occurrence of line segments is obvious.

Figure 9: The left image shows two corresponding planes π and π'. The middle and the right image show the homography-based transformation results with slightly wrong correspondence information. The point-based homography (middle) performs still well while the line-based homography (right) fails.

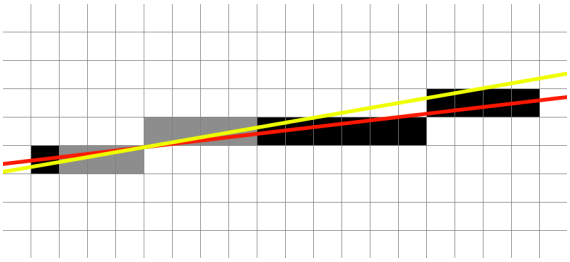

Figure 10: A schematic representation of a line segment (black) which is only covered partially by detected edge points (gray). Obviously the underlying line for the black line segment (red/dark) runs clearly different from the underlying line of the gray line segment (yellow/bright).

5 THE PANNED ITERATION LINE HOMOGRAPHY ALGORITHM

We want to detect a facade F in both images I_1, I_2 with a homography. This can only work for plane facades. We transform I_i into two blueprints B_i and apply the following iteration algorithm *lineHomography* based on panned line DLT.

We identify a blueprint B with its set of line segments. To a line segment $s = p_1p_2$ with endpoints p_1, p_2 let $l_s = \{(x,y) \in \mathcal{R}^2 \mid x\cos\alpha + y\sin\alpha = d\}$ be the line through p_1 and p_2 in *Hesse form*. To any line segment s we annotate its endpoints p_1, p_2 and the parameters α, d of l_s. We thus can easily switch from line segments to lines. We measure the distance $d(s_1,s_2)$ of two line segments by the distance of their lines $d(l_{s_1}, l_{s_2})$ that is given as the differences of their parameters α_i and of d_i. Since α and d have different ranges we adapt these ranges, weight them equally and scale the difference to $[0,1]$. This is achieved by $d(s_1,s_2) = \frac{|\alpha_1-\alpha_2|\cdot f_r + |d_1-d_2|}{f_s}$ with a range factor f_r and a scale factor f_s which depend on the image size. Thus, even two disjoint line segments on different parts of the same line are in distance 0. d is a metric on lines but not on line segments.

For a given homography H and threshold t we denote with C_H the set of H-correspondences defined as

$$C_H := \{(s,s') \in B_1 \times B_2 | d(H(s),s') < t\}.$$

An H-correspondences is thus a pair (s,s') of line segments in B_1 and B_2 such that H maps s to a line segment close in distance to $s_2 \in B_2$.

5.1 Initial Line Segment Correspondences

The first aim is to find an initial set $\mathbf{C}_{init}$ of pairs of line segment correspondences. This means to identify pairs of lines segments $l_i \in B_1$ and $l'_j \in B_2$ which describe the same line of the pictured object and lie all on the same facade F in B_1 and B_2. An automatic detection of candidates of line on one facade is in progress using of the known vanishing point but not finished and implemented yet. Therefor, we manually present $\mathbf{C}_{init}$. However, we also add some wrong correspondences (l,l') to $\mathbf{C}_{init}$ where l and l' present different line segments in B_1 and B_2 on different facades on different planes. Simply because no automated detection of corresponding lines can avoid mistakes.

5.2 Panned DLT

We present the algorithm *pannedDLT* that will be used frequently.

The input is a set C of 4 pairs of line segments from $B_1 \times B_2$.

They posses 16 pairs of endpoints that are now homogeneously panned. Homogeneously means that the endpoints of the same line are panned in different directions and that different line segments with a similar direction are panned in the same way. This panning leads to 49 sets $C_1,...,C_{49}$ where each C_i consists of supposed corresponding pairs of lines segments.

For any set C' of four corresponding pairs (s_1,s_2) of line segments we regard the set $\hat{C}'$ of the four pairs (l_{s_1},l_{s_2}) of corresponding lines in their main form. We now apply the line-based DLT to any set $\hat{C}_i, 1 \leq i \leq 49$, where we use the parameters a,b,c for each line l_s in $\hat{C}_i$. The resulting 49 homographies $H_1,...,H_{49}$ form the output of *pannedDLT* with input C.

5.3 Line RANSAC

Hartley and Zisserman describe in chapter 4.8 of (Hartley and Zisserman, 2004) the automatic computation of a (point-based) homography. An important part of their proposed algorithm is the robust homography estimation based on RANSAC which was published by Fischler and Bolles in (Fischler and Bolles, 1981). We present here a RANSAC algorithm *lineRANSAC* for lines based on panned DLT.

The input is a set $\mathbf{C}$ of $n \geq 4$ pairs of corresponding line segments in $B_1 \times B_2$.
Set $j := 1$,
repeat
create C_j as a set of four randomly chosen pairs from $\mathbf{C}$,
compute 49 homographies $H_{j_1},...,H_{j_{49}}$ from *panned-DLT* with input C'_j,
compute the H_{j_k}-correspondences $C_{j_k} := C_{H_{j_k}}$ for $1 \leq k \leq 49$,
$j := j+1$
until break (where break is a criterion as used usually in RANSAC, depending on the average of the found sizes of the sets C_{j_k}).
Let j_0 be some index with

$$|C_{j_0}| = \max\{|C_{i_k}| \mid 1 \leq i < j, 1 \leq k \leq 49\}.$$

{ **RANSAC-step**
Compute H by DLT with **all** correspondences in $\hat{C}_{j_0}$.}

Obviously, C_{j_k} consists of all inliers of the homography H_{j_k}. C_{j_0} is a maximal set of found inliers in the process. All its correspondences are now used to compute a new homography H in the RANSAC-step.

5.4 RANSAC Line Homography

The input is $\mathbf{C}_{init}$ as described in subsection 5.1.
Apply *lineRANSAC* with input $\mathbf{C}_{init}$ to get the output H and $\mathbf{C}$.
$\mathbf{C}_{new} := \emptyset$,
repeat
$\mathbf{C}_{new} := \mathbf{C}_{new} \cup \mathbf{C}$,
apply *lineRANSAC* with input $\mathbf{C}_{new}$ to get the output H and $\mathbf{C}$
until a chosen number of iterations is reached or $\mathbf{C} - (\mathbf{C} \cap \mathbf{C}_{new}) = \emptyset$.

The output of this *ransacLineHomography* algorithm is the final homography H.

5.5 Line Homography

However, *ransacLineHomography* failes to give reasonable homographies based on line correspondences. It turns out that the RANSAC-step in 5.3 is the reason: the DLT-algorithm with $N > 4$ line correspondences is too unstable. We therefore have to drop the RANSAC-step.

By *lineHomography* we denote the *ransacLine-Homography* algorithm without the RANSAC-step in 5.3. The result of 5.3 is now the homography H_{i_0} that gives reason to the maximal set C_{i_0} of correspondences.

5.6 Matching Decision

There are line segments in B_1 and in B_2 that we suppose to lie on the same facade F presented as facade F_1 in B_1 and F_2 in B_2. We choose an initial defective set $\mathbf{C}_{init}$ of pairs of corresponding line segments, apply *lineHomography* and result in a homography H_{final}. The correspondences which fit eminently well to H_{final} are used to hypothesize the locations of the main facades $F(B_1)$ and $F(B_2)$. This is $\mathbf{C}_{best} := \{(s,s') \in B_1 \times B_2 |\ d(H_{final}(s),s') < t_0\}$ for a new threshold $t_0 < t$. $\mathbf{C}^1_{best}$ consists of all line segments in B_1 that appear as a first coordinate in $\mathbf{C}_{best}$, $\mathbf{C}^2_{best}$ analogously are the line segments in B_2 appearing in $\mathbf{C}_{best}$. CH_1 is the convex hull of $\mathbf{C}^1_{best}$ and CH_2 that of $\mathbf{C}^2_{best}$. We map CH_2 by H^{-1}_{final} in B_1 and set $CH_1^{\cap} := CH_1 \cap H^{-1}_{final}(CH_2)$. $CH_1^{\cap}$ is used to process the final evaluation.

Every line segment s of B_1 that lies inside $CH_1^{\cap}$ is projected to $H_{final}(s)$ in B_2. We now compute a line segment $s' \in CH_2^{\cap} := H(CH_1) \cap CH_2$ with a minimal distance $d(H_{final}(s),s')$. If this distance is below a third threshold t_m we regard s and s' a s a match. The sum of all these distances per match is divided by the number of all the matches to get the final error $\varepsilon \in [0,1]$.

If ε is smaller than a final fourth threshold $t_f := 0.011$ and a percentage ρ of at least 25% of all segments of B_1 within $CH_1^{\cap}$ have found a match, F_1 and F_2 are considered as a match of one and the same facade from different point of views. The two polygons $CH_1^{\cap}$ and $CH_2^{\cap}$ form the assumptions of the facades $F(B_1)$ and $F(B_2)$ in B_1 and B_2.

6 EVALUATION

To evaluate our approach, we use 15 pairs of building images. Each pair shows among others the same building facade from a different point of view. All the images are recorded with the same SLR. By camera calibration radial distortion is reduced in the images.

After applying the blueprint reconstruction algorithm we posses a pair of blueprints B_1 and B_2 for each image pair. In an evaluation set E_{l_4} four correct line segment correspondences (s,s') are annotated to each pair of blueprints and furthermore, two randomly generated incorrect line segment correspondences. These six correspondences form the initial correspondence set $\mathbf{C}_{init}$. In addition, we use a second and third evaluation set E_{l_6} and E_{p_4}. In E_{l_6} six correct line segment correspondences and three incorrect line segment correspondences are used for each pair of images. In E_{p_4} only four correct point correspondences are used.

Table 1: The results of evaluation set E_{l_4} with four correct and two incorrect line segment correspondences. ε and ρ are explained in chapter 5.6.

pair	matched lines	ε	ρ	match
1	42	0.003681	35	1
2	17	0.006531	11	0
3	125	0.002412	34	1
4	50	0.001998	33	1
5	123	0.003005	27	1
6	100	0.002769	38	1
7	42	0.003102	45	1
8	19	0.002632	33	1
9	81	0.002694	35	1
10	61	0.004254	38	1
11	174	0.002942	37	1
12	87	0.003375	30	1
13	154	0.005112	41	1
14	25	0.006121	24	0
15	62	0.01051	59	1
mean	**77**	**0.0041**	**35%**	**86.7%**

As explained in 4.1 it is impossible in most instances to find corresponding points in two blueprints. Because of this the original image had to be used for the annotation of the corresponding points in E_{p_4}. It should be noted that a creation of E_{p_4} is much more involved than of E_{l_4} and E_{l_6}.

We have run our algorithm *lineHomography* on each $\mathbf{C}_{init}$ of all the blueprint pairs of both evaluation sets E_{l_4} and E_{l_6} and used the described matching decision. On E_{p_4} the *DLT* has been run. In all three cases the results are gained as described in chapter 5.6. Tables 1, 2 and 3 present the results.

The results for E_{l_4}, E_{l_6}, and E_{p_4} are shown in table 1, 2, and 3, respectively. Because of the random steps in our algorithm *lineIteration* the results per blueprint pair of E_{l_4} and E_{l_6} vary in their quality and accuracy. The evaluation runs presented in the tables are representative but better and worse results are possible. As all the image pairs contain in each case the same building facade a 100% matching rate is desired.

E_{p_4} produces the best results. As the DLT for points is known to be stable and only correct correspondences are used this is as expected and the results may be considered as a standard. The desired matching rate of 100% is reached, the absolute number of matched line segments is the highest and ε is the lowest.

Compared with these E_{p_4}-results we consider the results of E_{l_4} and E_{l_6} to be acceptable. The bad mean ε of 0.0097 when using E_{l_6} is caused only by one mis-

Table 2: The results of evaluation set E_{l_6} with six correct and three incorrect line segment correspondences. ε and ρ are explained in chapter 5.6.

pair	matched lines	ε	ρ	match
1	2	0.1018	12	0
2	69	0.003355	43	1
3	168	0.003023	40	1
4	44	0.001875	28	1
5	127	0.002635	28	1
6	95	0.00247	35	1
7	50	0.002869	48	1
8	34	0.002931	52	1
9	81	0.002098	35	1
10	62	0.003997	43	1
11	199	0.003061	45	1
12	67	0.004469	32	1
13	19	0.003671	19	0
14	101	0.002671	66	1
15	76	0.004361	46	1
mean	**80**	**0.0097**	**38%**	**86.7%**

Table 3: The results of evaluation set E_{p_4} with four correct point correspondences. ε and ρ are explained in chapter 5.6.

pair	matched lines	ε	ρ	match
1	51	0.004077	38	1
2	61	0.004433	31	1
3	154	0.00282	35	1
4	51	0.002521	33	1
5	130	0.002739	27	1
6	97	0.002782	37	1
7	44	0.006547	49	1
8	34	0.003582	38	1
9	68	0.002414	30	1
10	68	0.003504	41	1
11	206	0.002802	40	1
12	111	0.003496	31	1
13	185	0.004673	48	1
14	85	0.003846	45	1
15	39	0.00458	60	1
mean	**92**	**0.0037**	**39%**	**100.0%**

match with a very high ε. E_{l_6} has the same number of mismatches, but both are only caused by few missing percentage points in ρ. However, the computation time of E_{l_4} and E_{l_6} is much higher than of E_{p_4}.

Figure 11 shows the two blueprints (upper image) and the matched result (lower image) of image pair no. 11 of E_{l_4}. This is one example for the quality of the results which *lineIteration* is able to produce.

Our blueprint approach provides a robust detection of line segments even in image regions with poor textures. The results show that our approach states a practical matching alternative in the absence of usable or distinctive point features.

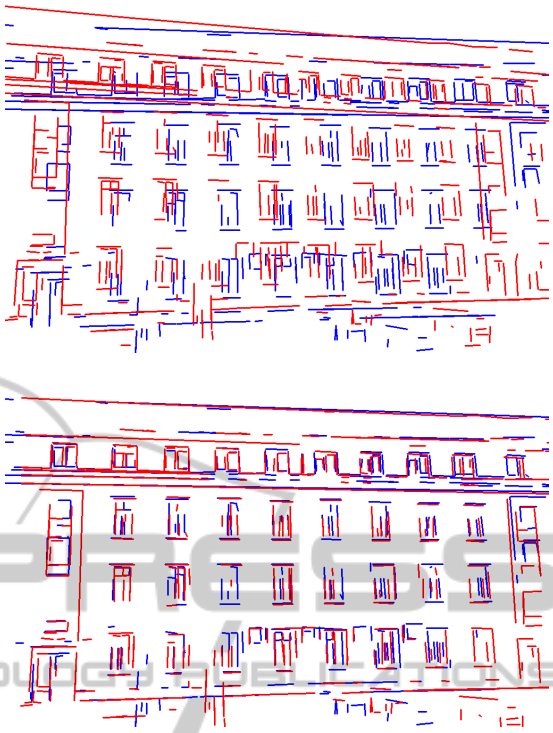

Figure 11: Two different views of a building (upper image) and the matching result (lower image) using line homography with 4 correct and 2 incorrect line pairs initially.

7 CONCLUSIONS

We have introduced a reconstruction of blueprints from building images. We are quite content with this algorithm and regard it as widely completed. The blueprint forms a rather complex feature for image analysis. We have used this feature for line-based homographies that may be used for facade detection e.g. The instability of line-based homographies is well-known in the literature. Nevertheless, we succeed with our panned iterated algorithm in reasonable results. These results are compareable with but not as good as point-based results. On the other hand, it seems to be easier to find initial line correspondences with the help of blueprints than initial point correspondences from the original image.

We still have to integrate a method for finding initial line segment correspondences automatically. Further, we will include the Levenberg-Marquardt optimization algorithm as described in Appendix 6 in (Hartley and Zisserman, 2004).

REFERENCES

Baltsavias, E. P. (2004). Object extraction and revision by image analysis using existing geodata and knowledge: current status and steps towards operational systems. *ISPRS Journal of Photogrammetry and Remote Sensing*, 58(3-4):129 – 151. Integration of Geodata and Imagery for Automated Refinement and Update of Spatial Databases.

Bay, H., Ferrari, V., and Van Gool, L. (2005). Wide-baseline stereo matching with line segments. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 329–336, Washington, DC, USA. IEEE Computer Society.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698.

David, P. (2008). Detection of building facades in urban environments. In Rahman, Z.-u., Reichenbach, S. E., and Neifeld, M. A., editors, *Visual Information Processing XVII*, volume 6978 of *Proceedings of the SPIE*, pages 69780P–69780P–11.

Dubrofsky, E. and Woodham, R. J. (2008). Combining line and point correspondences for homography estimation. In *Proceedings of the 4th International Symposium on Advances in Visual Computing, Part II*, ISVC '08, pages 202–213, Berlin, Heidelberg. Springer-Verlag.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395.

Guerrero, J. and Sags, C. (2003). Robust line matching and estimate of homographies simultaneously. In Perales, F., Campilho, A., Blanca, N., and Sanfeliu, A., editors, *Pattern Recognition and Image Analysis*, volume 2652 of *Lecture Notes in Computer Science*, pages 297–307. Springer Berlin Heidelberg.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.

Horowitz, S. L. and Pavlidis, T. (1976). Picture segmentation by a tree traversal algorithm. *J. ACM*, 23(2):368–388.

Iqbal, Q. and Aggarwall, J. K. (1999). Applying perceptual grouping to content-based image retrieval:building images. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, Fort Collins, CO, USA.

Kumar, S. and Hebert, M. (2003). Man-made structure detection in natural images using a causal multiscale random field. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages 119–126.

Leavers, V. F. and Boyce, J. F. (1987). The radon transform and its application to shape parametrization in machine vision. *Image Vision Comput.*, 5(2):161–166.

López-Nicolás, G., Sagüés, C., and Guerrero, J. J. (2005). Automatic matching and motion estimation from two views of a multiplane scene. In *Proceedings of the Second Iberian conference on Pattern Recognition and Image Analysis - Volume Part I*, IbPRIA'05, pages 69–76, Berlin, Heidelberg. Springer-Verlag.

Lowe, D. (2003). Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 20, pages 91–110.

M. Kuwahara, K. Hachimura, S. Eiho, and M. Kinoshita (1976). Processing of ri-angiocardiographic images. In Preston, K. and Onoe, M., editors, *Digital Processing of Biomedical Images*, pages 187–202.

Mayer, H. (1999). Automatic object extraction from aerial imagery–a survey focusing on buildings. *Computer Vision and Image Understanding*, 74(2):138 – 149.

Priese, L. and Rehrmann, V. (1993). A fast hybrid color segmentation method. In S.J. Pppl, H. Handels, editor, *Proc. DAGM Symposium Mustererkennung, Informatik Fachberichte, Springer Verlag*, pages 297–304.

Priese, L., Schmitt, F., and Hering, N. (2009). Grouping of semantically similar image positions. In Salberg, A.-B., Hardeberg, J. Y., and Jenssen, R., editors, *16th Scandinavian Conference, SCIA 2009, Oslo, Norway, June 15-18, Proceedings*, volume 5575, pages 726–734.

Schmid, C. and Zisserman, A. (1997). Automatic line matching across views. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 666 –671.

Schmitt, F. and Priese, L. (2009a). Intersection point topology for vanishing point detection. In *Accepted for publication in: Discrete Geometry for Computer Imagery 2009, Montreal, Canada*.

Schmitt, F. and Priese, L. (2009b). Sky detection in csc-segmented color images. In *Fourth International Conference on Computer Vision Theory and Applications (VISAPP) 2009, Lisboa, Portugal*, volume 2, pages 101–106.

Zhang, W. and Košecká, J. (2007). Hierarchical building recognition. *Image Vision Comput.*, 25(5):704–716.