Fast Wide Baseline Matching for Visual Navigation

Toon Goedemé[1], **Tinne Tuytelaars**[1], **and Luc Van Gool**[1,2]

[1]PSI-VISICS, University of Leuven, Belgium [2]BIWI, ETH Zürich, Switzerland
[toon.goedeme, tinne.tuytelaars, luc.vangool]@esat.kuleuven.ac.be

Abstract

A new and fast way to find local image correspondences for wide baseline image matching is described. The targeted application is visual navigation, e.g. of a semi-automatic wheelchair. Such applications pose some additional requirements, like the need to work with natural landmarks rather than artificial markers, and the need to recognize locations fast. The restricted motion of the camera can be exploited to simplify the feature extraction. These features should support their identification from different, but nevertheless restricted viewing directions, and under variable illumination conditions. The paper proposes a specialization of so-called affine invariant regions for these particular conditions, which in this case simplifies to column segments. Their applicability is wider than robot navigation, and includes localization for wearable computing and scene recognition for automatic movie indexing.

1 Introduction

Traditionally, visual navigation for mobile robots has been based on the construction of metric, 3D models of the scene and the matching of 3D data against this model. An alternative, which seems to come closer to human behavior [1], would be to build a scene model based on images organized in graph-like structures and to relate one's position to that at which the most similar model image(s) have been taken. We have discussed the automatic construction of such models elsewhere [2]. Here, we focus on how to match the current view (during navigation) to the set of model images.

In order to maximize the applicability of the approach, the use of special markers should be avoided. Our main intended application is automatic wheelchair navigation for severely disabled people. Covering the walls of people's homes with special markers is hardly acceptable. Similarly, outdoor applications would render the use of such markers highly impractical. Hence, the goal is to base the localization on the recognition of natural landmarks. As the model cannot be built from images taken from all possible viewing positions and under all possible illumination conditions, the recognition of the most similar images should be based on landmarks that can be extracted irrespective of such variations, i.e. they should be invariant. These features should also be sufficiently local, so that they are robust against occlusions and scene clutter. Recently, so-called 'affine invariant regions' [3] have been proposed as features with these properties. We propose a more specific type of features, dedicated to navigation tasks, that are easier to extract and that allow for fast matching. A preview of the results of our fast wide baseline matching algorithm is displayed in fig. 1.

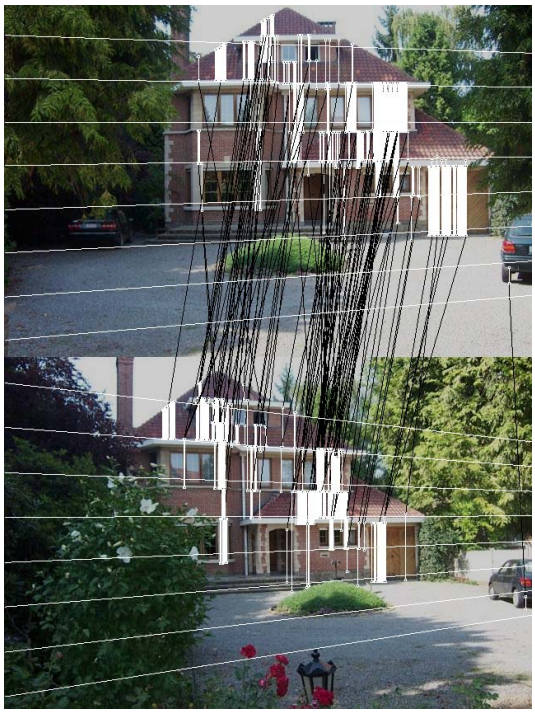

Figure 1: *Matching result. Matching column segments are in white. Black connection lines show corresponding column clusters. White lines show the epipolar geometry found.*

The remainder of the paper is organized as follows. Section 2 discusses related work in the field of wide baseline matching. Section 3 explains the constraints on the camera and scene that our method presumes. Section 4 describes the extraction and matching of invariant features. Experiments follow in section 5, while section 6 examines generalizations of the method and gives a birds-eye view on a few possible applications. We conclude with section 7.

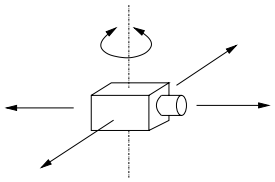

Figure 2: *The allowed movements of the camera.*

2 Related work

The matching problem that we have to solve, in order to find the images in the database that are most similar to the ones taken (repeatedly) by the mobile robot, is one of so-called wide baseline matching. Being able to match under such conditions allows one to keep the number of reference images in the scene model to a minimum.

Most wide baseline matching approaches are based on so-called *invariant regions*. These are constructed around interest points, such as corners, in a way as to adapt their shapes to the viewpoint and keep the part of the scene they enclose fixed. These regions are then described by a descriptor vector, the elements of which are invariant under combinations of geometric and photometric changes. They can be matched efficiently between views taken from different viewpoints and under different illumination. The crux of the matter is that these affine invariant regions are determined solely on the basis of a single image, i.e. no information about the other view(s) is necessary during extraction.

The differences between approaches lie in the way in which interest points, local image regions and descriptor vectors are extracted. An early example is the work of Schmid and Mohr [4], where geometric invariance was still under image rotations only. Scaling was handled by using circular regions of several sizes. Lowe *et al.* [5] extended these ideas to real scale-invariance. They have also achieved recognition at high speeds. More general affine invariance has been achieved in the work of Baumberg [6], that uses an iterative scheme and the combination of multiple scales, and in the more direct, constructive methods of Tuytelaars & Van Gool [3], Matas *et al.* [7], and Mikolajczyk & Schmid [8]. For tasks like navigation, full affine invariance is often not really required, and the level of geometric invariance needed lies in between those offered by the original and the later methods that we have just mentioned. Indeed, the camera tends to move in a constrained way. This is the subject of the next section. As a result, we will propose a type of feature that shares several properties with the earlier ones (local, combining geometric and photometric invariance), but that comes closer to the linear features introduced by Tell & Carlsson [9]. They look at line segments between two Harris corners. Fourier coefficients of the intensity values along the segments serve as their descriptors. Their method is also quite efficient, but a caveat is the large number of corner pair combinations that may have to be considered. Their features have also not been designed for the particular conditions we will consider and do not include color and geometry information.

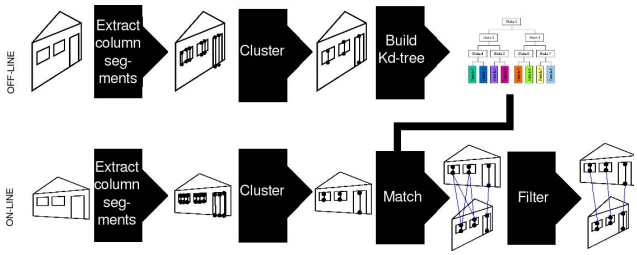

Figure 3: *Overview of the algorithm. Top row: off-line process, bottom row: on-line process.*

3 Constraints

As mentioned before, our technique has been developed for use on a mobile robot. Imagine a robot with a fixed mounted camera, moving on a horizontal plane. Hence, the camera may (1) translate in the horizontal plane and (2) rotate around a vertical axis, as shown in fig. 2. We also assume that the camera is oriented horizontally, with its optical axis parallel to this plane (although these conditions can be relaxed, as discussed later).

Furthermore, we assume that the scene contains several vertical elements. This is certainly the case for human-made environments (walls, doors, windows, furniture, etc.), but also natural scenes tend to contain several such structures (e.g. trees). These structures don't have to be planar, so e.g. cylindrical elements like pillars do comply just the same.

4 Feature extraction and matching

Under the specified constraints, invariant features can be extracted that are simpler than the affine invariant regions and therefore can be extracted and matched faster, an important condition for on-line navigation. Under these constraints, a vertical line in the world always projects to a vertical line in the image plane. The image content of this line will only be scaled around the point where it intersects the horizon.

Fig. 3 gives an overview of the algorithm. In our target application, like in many other wide baseline applications, a new image must be matched with one or more prerecorded reference image(s). That is why we split up the algorithm in an off-line part and an on-line part, depicted respectively as the top and bottom row in the figure. While the off-line part is not time-critical, we try to achieve high-speed execution of the on-line part, where the localization takes place.

In every image, we first extract *invariant column segments*, the local image features we use. Section 4.1 explains this. For each column segment, a descriptor vector is computed based on geometrical, color and intensity information (section 4.2). Because of repetitive elements in the image, the column segments need to be clustered, as discussed in section 4.3. To speed up the execution, the data of the ref-

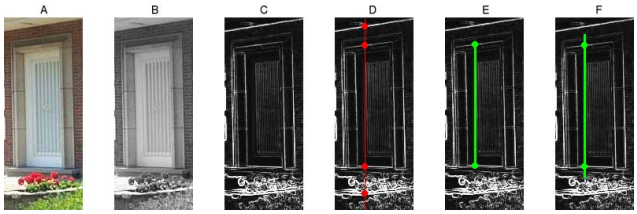

Figure 4: *Column segment extraction algorithm: (A) original image, (B) intensity image, (C) gradient magnitude image, (D) local maxima of the gradient for one pixel column, (E) one pair of maxima defines a column segment, (F) expanded segment.*

erence image(s) is stored in a Kd-tree, in support of fast matching (section 4.4). The RANSAC-based outlier rejection scheme used for filtering is explained in section 4.5.

4.1 Column segment extraction

In this section we describe the affine invariant column segments that we use as features. As a matter of fact, every step in their extraction is invariant to the allowable changes in viewpoint and illumination, where the latter amount to a scaling and offset. The extraction process is illustrated in fig. 4. From the intensity image (B) the gradient magnitude image (C) is computed using the Sobel operator. For every pixel column in the image we look for points having a local maximum gradient value (D). Every consecutive pair of gradient maxima on a pixel column demarcate a new column segment (E). A first selection on the column segments is made by comparing the average gradient value between the end points and the gradient value of the end points. The column segment with end points a and b is eliminated if $avg(grad(]a,b[)) > \alpha_{th} min(grad(a), grad(b))$, which is the case if it contains too much noise. To increase the discriminative power of the descriptor vector, the column segment is expanded on both sides with a constant fraction of the segment length. Fig. F shows this.

It is important to see that we do *not* use *edges* as features, as in the work of Schmid and Zisserman [10]. The column segment features we extract are just vertical columns of pixels between two points with large gradient value.

4.2 Descriptor vector calculation

In order to characterize the different invariant column segments, a total of 11 descriptors are used, which combine geometrical, color and intensity properties:

- A *geometrical invariant*: As explained before, column segments are scaled around the horizon, i.e. the vanishing line of the plane of motion. Because the end points give two measurements in this 1-parameter transformation, at least one invariant can be expected. The proof immediately follows from similar triangles, as shown in fig. 5. Because of the parallelism of the vertical world segment ab and the image plane segment $a'c'$, $\triangle afc \sim \triangle a'fc'$ and $\triangle bfc \sim \triangle b'fc'$, and therefore $\frac{\|a'c'\|}{\|b'c'\|} = \frac{\|ac\|}{\|bc\|}$.

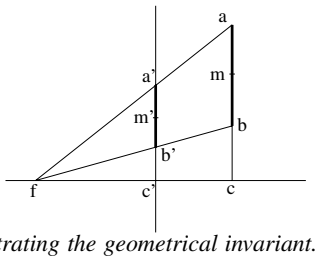

Figure 5: *Illustrating the geometrical invariant. ab is a vertical element in the world, f is the focal point of the camera and $a'b'$ is the projection of ab on the image plane. fc is the optical axis.*

As length ratios along the vertical lines are invariant under image projection, we can also use the midpoints of the segments m and m', and compare their distances to the horizon with respect to the segment length: $\frac{\|m'c'\|}{\|a'b'\|} = \frac{\|mc\|}{\|ab\|}$.

This ratio can be used as the geometric invariant in our descriptor vector, which is numerically more stable than the former. If the camera constraints are fully satisfied, the horizon can be found as the horizontal line through the image center. Section 6 discusses how the horizon can be found when these constraints do not apply anymore.

- Three *color invariants*: To include color information in the descriptor vector, we compute the following color invariants, based on 'generalized color moments' [11], calculated over the entire column segment: C_{RB}, C_{RG} and C_{GB}, with

$$C_{PQ} = \frac{\int PQ\,\mathrm{d}x \int \mathrm{d}x}{\int P\,\mathrm{d}x \int Q\,\mathrm{d}x}, \qquad (1)$$

where R, G and B represent the red, green and blue channels, centralized around their means. The inclusion of this color information, which is invariant to the illumination, makes the vector more discriminative.

- Seven *intensity invariants*. To characterize the intensity profile along the column segment, good features could be obtained through the Karhunen-Loève transform. But because all the data is not known beforehand this is not practical. We have used the *discrete cosine transform* (DCT) instead, as it offers a good alternative in such cases and because it is fast to compute and yields real values. We compute 7 DCT coefficients on the normalized intensity profile along the column segment. These DCT computations in our algorithm are executed fast using the 8-point 1D method of Loeffler *et al.* [12].

Every step in the extraction of the column segments and their descriptors is invariant to changes in viewpoint, if these satisfy the camera motion constraint. In contrast to the invariant regions that are typically used, these features also *perfectly* withstand the effect of perspective projection. Invariance to scene illumination is also guaranteed if it affects each color band by a scaling and an offset.

For matching and clustering column segments, a distance measure for the description vectors must be defined. In our algorithm, we use the Mahalanobis distance r, with $r^2 = (\mathbf{X} - \mu)^T {\mathbf{C_X}}^{-1} (\mathbf{X} - \mu)$. In this formula μ and $\mathbf{C_X}$

are the mean and intra-group correlation matrix of $\mathbf{X}$, which contains the descriptor vectors. We measured the real intra-group $\mathbf{C_X}$ on single-group training images, rather than using the overall correlation.

4.3 Intra-image Clustering

In many cases there is horizontal repetition of similar column segments in the scene, originating from horizontally constant elements. An example is the rightmost garage gate of the house in fig. 1. To avoid matching over and over again very similar column segments during the on-line navigation stage, on both the reference image and the query image clustering of the column segments is performed first. This clustering can be implemented very efficiently, because the column segments are already sorted by their order of extraction. As clustering measure we use the Mahalanobis distance of the descriptor vectors, together with the horizontal distance between the line segments. Each cluster is represented by a prototype column segment, with the average descriptor vector and centered in the middle of the cluster.

4.4 Matching

The goal of the on-line segment matching is to quickly find corresponding column segments between two (or more) images. Here, we define 'corresponding' as having a Mahalanobis distance between the descriptor vectors smaller than a fixed threshold. To avoid having to compare each column segment of one image with each column segment in the other image, a Kd-tree of the reference image data is built of the cluster prototypes. We used the on-line available package ANN by Mount *et al.* [13] for this. Searching in this kind of database is very fast. Also, the database can contain data of more than one reference image, so we can match one image in parallel with several images.

4.5 Rejecting false matches

Apart from the classic causes of mismatches like image noise or geometric and photometric changes beyond the allowable range, repetitions are another source of mismatches. These are quite frequent in the type of architectural scenes we work with. In the schematic example of fig. 3, each window would match any similar window. An additional rejection of such mismatches is needed. We apply RANSAC to impose compliance with the dominant epipolar geometry [14].

But, because of our camera motion constraint, the F-matrix has fewer degrees of freedom than in the general case, as 2 of its elements are always zero. Indeed, because

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R} = \begin{bmatrix} 0 & -Z_1 & 0 \\ Z_1 & 0 & -X_1 \\ 0 & X_1 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2}$$

we see that, for a CCD camera (i.e. no skew),

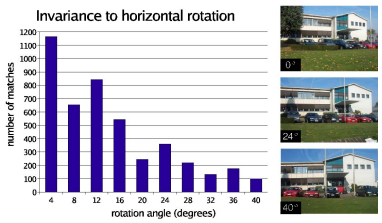

Figure 6: *Viewpoint invariance. Number of matches between unrotated view ($0°$) and image taken after rotation with different horizontal angles.*

$$\mathbf{F} = \mathbf{K}^{-T}\mathbf{E}\mathbf{K}^{-1} = \begin{bmatrix} 0 & f_{12} & f_{13} \\ f_{21} & 0 & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}. \tag{3}$$

Therefore, we developed an adapted F-matrix estimation which takes these constraints into account, and used that within RANSAC. With a random sample of six points, the F-matrix can be computed. Also, because the result is restricted, the outcome is more probable to be correct.

5 Experimental results

5.1 Matching and recognition experiments

We have analyzed the computation times of the different parts of the algorithm for the example in fig. 1. It contains two 640×480 color images of the same scene. In the reference image (above) 1871 column segments were found, and clustered into 1292 prototypes. In the second image the 1706 column segments found were clustered into 995 prototypes. Before RANSAC filtering, 161 matches between prototypes were found for these two images, after filtering 141 remained, of which only 6 are wrong. The algorithm has been implemented in C++ and currently runs on a 1GHz machine. Although not yet optimized using features such as MMX, the total on-line execution time was as low as $0.966s$ in this case, and these times are typical.

Fig. 6 illustrates the viewpoint invariance. This small experiment gives a feel for the survival of features under increasing rotation about the vertical axis. Even at $40°$ there are still more than 100 matching clusters.

The previous example was based on the matching of a pair of images, that were known to contain correspondences. In reality, an incoming query image will have to be matched against several reference images of the site model, even if the number can often be reduced based on context information. Therefore, we have tested the algorithm on the *ZuBuD* database, containing 1005 images of buildings in Zürich [15]. It should be noted that these images have not been taken with our camera motion constraint in mind. This already hints at the rather wide applicability of the approach in practice. An example of two matched ZuBuD images is given in fig. 7, top right.

In this case all reference images are processed off-line and put together in a large Kd-tree database, which took

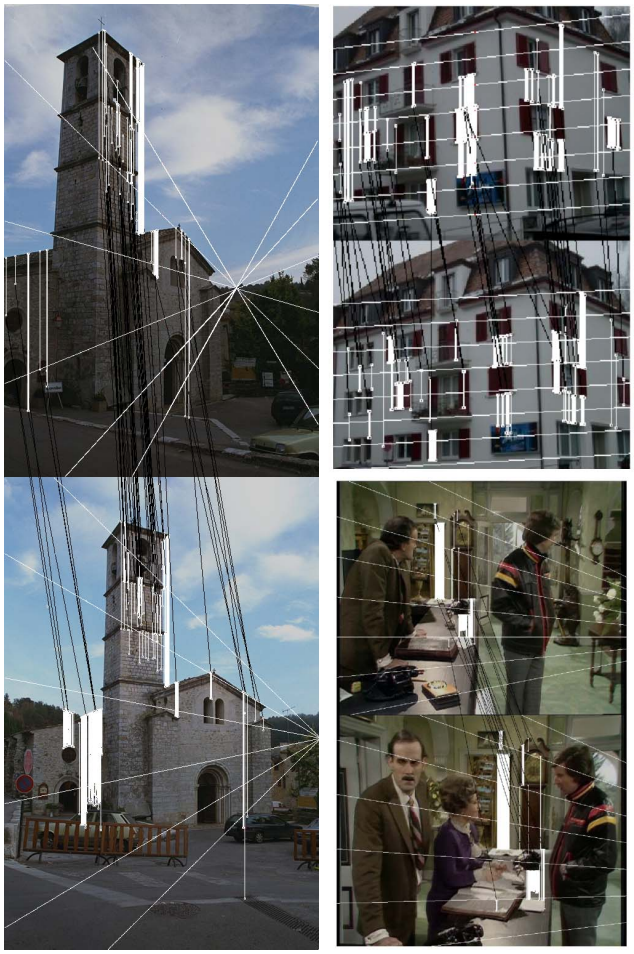

Figure 7: *Left: Matching results on the Valbonne image pair (image numbers 9 and 13), 208 matches are found. Right: Example matching of two images: ZuBuD (top), Fawlty Towers (bottom).*

only about 4 minutes of computation time. The 115 query images were taken from different viewpoints and under different weather conditions than the database images. The average total recognition time was $1074ms$, with a standard deviation of $280ms$. For 92% of the query images the retrieval was correct, which is 6% better and about three times faster than the results reported by Shao *et al.* [16].

We tested the speed of different wide baseline matching techniques to compare with our method. Fig. 8 shows the average execution times of the feature extraction and descriptor calculation stage on a 1 GHz machine on 640×480 color images. We see that our method is more than ten times faster than our closest competitor.

Comparing matching quality over different approaches is not straightforward. On one image pair, one can count the number of matches found with different techniques and compare these numbers. But this comparison lacks information about the accuracy of the matches, the size and shape of the matching regions, the number of extracted regions, the scene contents and constraints on the set-up. That

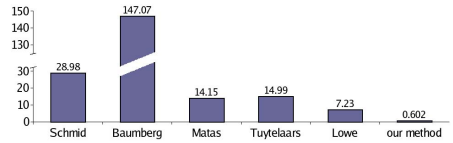

Figure 8: *Average computation time (seconds) needed for feature extraction and descriptor calculation for 640×480 color images with different methods on 1GHz PC. From left to right, the methods of: Schmid [8], Baumberg [6], Matas [7], Tuytelaars [3], Lowe [5], and the method of the paper.*

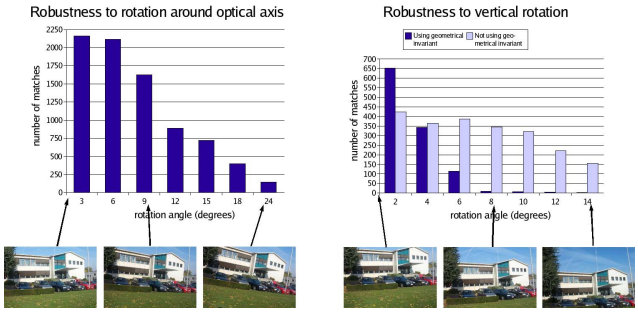

Figure 9: *Robustness to rotation around optical axis (left) and vertical rotations (right). Number of matches between unrotated view (0°) and view from different rotation angles.*

is why we present the matching results on a Valbonne image pair. These images are widely used by other researchers to demonstrate wide baseline matching results, so can give a visual indication of the relative matching quality. On this pair of images, we found 208 column segment matches, displayed left in fig. 7. On the very same image pair, Tell [17] reports 127 point matches and Mikolajczyk & Schmid [8] found 22 region matches.

5.2 Robustness to constraint violations

In theory, this wide baseline matching technique is only valid if the camera motion constraints are satisfied and if the scene elements are vertical. In some applications these preconditions can be met precisely, like e.g. in indoor mobile robot navigation. But other applications, like vision in a wearable computing context, will fullfill these conditions only approximately. Fortunately, the proposed technique is also usable in such conditions, as is corroborated by the experiments in this section. As a matter of fact, this is already suggested by the additional matches that are found on the slanted roof of fig. 1.

To examine allowable camera position constraint violations, we did several tests. In fig. 9, left, the result of rotations around the optical axis is shown. We see that, due to the horizontal width of some scene elements the number of matches stays reasonably high, even at a relatively large rotation. If the rotation is too large, the height of the column segments near the side of the image deviates too much from the actual value.

In fig. 9, right, the effect of vertical rotation (like if the camera is nodding 'yes') is shown. We see (dark bars) that

the number of matches rapidly decreases with increasing angle. That is because the geometrical invariant uses the distance of the column segments to the center line of the image. If the camera rotates vertically, this center line is not equivalent to the horizon anymore.

6 Generalizations and applications

In order to use this system in applications only approximately satisfying the camera motion constraint, some solutions can be worked out. We can drop the geometric descriptor from the feature vector, which results in the light bars on the left of fig. 9. Unfortunately, as then also its discriminative power decreases, we find slightly less matches. In other applications, it will be possible to mount the camera in a way that it levels itself using gravity.

The approach can be easily generalized toward more general camera motions. As long as the horizon line of the planar motion can be detected as well as the vanishing point of parallel column segments in the scene, the resulting images can be transformed into the kind needed here. A projective transformation can be applied to move the segment line intersection to infinity and to put the horizon as a horizontal line through the center. This means that the camera need not really be positioned horizontally on the robot in such case, for instance. An approach to find the horizon and the vertical vanishing point – which is fully compatible with wearable computing systems – would be to add a tilt sensor. A popular example of wearable computing is a virtual tourist guide that provides location-based information. The alternative, an image-based horizon detection based on vanishing point detection, has been discarded in our implementation because of the high computational load and the extra scene constraints needed.

Moreover, although these constraints seem quite limiting, a strikingly large portion of everyday imagery obeys the constraints right away, or at least comes very close. To corroborate this, we analyzed two TV productions, *Flikken* and *Fawlty Towers*, and found 95.1% and 98.9%, resp., of the keyframes to satisfy the camera motion constraint. Figure 7, bottom right, illustrates this. Hence, the proposed techniques also provide a means to detect similar locations in such media productions, as part of a multimedia indexing system.

7 Conclusion

We have presented a wide baseline matching algorithm which is based on novel, local features and constraints that are put on the allowable camera motion. This and the use of efficient database indexing, result in an algorithm that is suited for quasi-real time applications like visual localization. The speed is more than ten times faster than the closest competing algorithm for the moment, and further code optimizations are possible.

Acknowledgements

The authors thank Hao Shao at ETH Zürich for providing us with his ZuBuD image database [15]. This research has been carried out with the financial support of the Inter-University Attraction Poles, IUAP-AMS, the Flemish Institute for the Advancement of Science in Industry, IWT, and the Fund for Scientific Research Flanders, FWO.

References

[1] M. O. Franz and H. A. Mallot, *Biomimetic robot navigation*, Robotics and Auton. Systems, vol. 30, pp. 133-153, 1998.

[2] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, *Vision Based Intelligent Wheelchair Control: the role of vision and inertial sensing in topological navigation*, Journal of Robotic Systems, 21(2), pp. 85-94, 2004.

[3] T. Tuytelaars and L. Van Gool, *Wide baseline stereo based on local, affinely invariant regions*, BMVC, pp. 412-422, 2000.

[4] C. Schmid, R. Mohr, C. Bauckhage, *Local Greyvalue Invariants for Image Retrieval*, PAMI, Vol. 19-5, pp. 872-877, 1997.

[5] D. Lowe, *Object Recognition from Local Scale-Invariant Features*, ICCV, pp. 1150-1157, 1999.

[6] A. Baumberg, *Reliable feature matching across widely separated views*, CVPR, pp. 774-781, 2000.

[7] J. Matas, O. Chum, M. Urban and T. Pajdla, *Robust wide baseline stereo from maximally stable extremal regions*, BMVC, pp. 384-396, 2002.

[8] K. Mikolajczyk and C. Schmid, *An affine invariant interest point detector*, ECCV, vol. 1, 128–142, 2002.

[9] D. Tell and S. Carlsson, *Wide baseline point matching using affine invariants computed from intensity profiles*, ECCV, pp. 814-828, 2000.

[10] C. Schmid and A. Zisserman, *Automatic line matching across views*, Proceedings Conference on Computer Vision and Pattern Recognition, pp. 666-671, 1997.

[11] F. Mindru, T. Moons, and L. Van Gool, *Recognizing color patters irrespective of viewpoint and illumination*, CVPR vol. 1, pp. 368-373, 1999.

[12] C. Loeffler, A. Ligtenberg and G. Moschytz, *Practical Fast 1-D DCT Algorithm with 11 Multiplications*, ICASSP, pp. 988-991, 1989.

[13] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu, *An optimal algorithm for approximate nearest neighbor searching*, J. of the ACM, vol. 45, pp. 891-923, 1998, http://www.cs.umd.edu/∼mount/ANN/.

[14] M.A. Fischler, R.C. Bolles, *Random Sampling Consensus - a paradigm for model fitting with applications to image analysis and automated cartography*, Commun. Assoc. Comp. Mach., vol. 24, no. 6, pp. 381-395, 1981

[15] H. Shao, T. Svoboda, L. Van Gool, *ZuBuD Image Database*, http://www.vision.ee.ethz.ch/showroom/zubud.en.html

[16] H. Shao, T. Svoboda, T. Tuytelaars, and L. Van Gool, *HPAT indexing for fast object/scene recognition based on local appearance*, CIVR, pp. 71-80, 2003.

[17] D. Tell, *Wide Baseline Matching with Applications to Visual Servoing*, Doctoral Dissertation, Stockholm, 2002.