A Family of Visual Homing Algorithms for Mobile Robots

Anatoly Gurevich, Dept. of Mathematics, tola@tx.technion.ac.il
Roman Klovsky, Dept. of Computer Science, c1193391@t2.technion.ac.il
Ehud Rivlin, Dept. of Computer Science, ehudr@cs.technion.ac.il
Ilan Shimshoni, Dept. of Ind. Eng. and Mgmt., ilans@ie.technion.ac.il
The Technion – Israel Inst. of Technology
Haifa 32000 Israel

Abstract

In this paper we present a family of visual homing algorithms for mobile robots. The algorithms are characterized by the non-general motion the robot can perform and projection model of the camera used. The task to be solved is to find corresponding features in current and target image and use them to recover the transformation between the current and target camera poses. This transformation is then used to guide the robot to the target pose. The algorithms solve these two problems together, exploiting the degenerate motion of the camera. This motion requires less corresponding pairs and puts constraints on where the corresponding points will be in the image. We present experimental results for the case of a mobile robot moving and rotating in the plane which is the most common application.

1 Introduction

In [3, 5, 4], we proposed a different approach to the problem of guiding a robot to desired positions and orientations. In our method the target pose is specified by an image taken from that pose (the *target image*). The task given to the robot is to move to a position where an image taken by a camera mounted on the robot will be identical to the target image. During the execution of this task the robot is allowed to take pictures of the environment, compare them with the target image and use the result of this comparison to determine its subsequent steps. We refer to the use of images to guide a robot to desired positions and orientations by *visual homing*. The problem has also been addressed by [8].

Schematicly speaking, a visual homing algorithm has the following steps.

- Take the current image.
- Find image features in the current and target images.
- Find corresponding pairs of features in the two images.
- Compute from them the transformation between the two camera poses (e.g. using [6]).
- Take a step towards the goal, tracking the features to the next image [4], and maintaining them in the image [8].
- Converge to the goal pose.

The most problematic step in this general algorithm is the one in which corresponding feature pairs are found in the two images. This problem is especially hard when the two images are very different due to the large difference in pose. This problem has been addressed by [9, 13]. In [9], the problem was attacked by building the transformation incrementally starting from a small number of matches of nearby points assumed to be co-planar and using the transformation found to find additional matches which improve the quality of the transformation. In [13], corresponding features are found using local affine invariants.

We take a different approach. Instead of solving the general visual homing problem we concentrate on a family of visual homing problems. Each algorithm is characterized by the non-general motion the robot can perform and projection model of the camera used. A typical example of one of the problems, is of a mobile robot translating and rotating in the plane. These degenerate motions which are very common in mobile robots can be exploited to find matching features and the relative pose of the camera more easily and more robustly then in the general case. Algorithms will be presented for two standard camera models: The full perspective projection model and the weak perspective model which is a good approximation to full perspective projection when the objects in the scene are relatively far from the camera with respect to their size. One of the cases which we discuss here of a robot translating in the plane has been studied in [12].

The paper is organized as follows. In the next section, we describe the different homing algorithms, for the weak and full perspective projections. In Section 3, we concentrate on the most common case, of a perspective camera fixed on a mobile robot rotating and translating in the plane. Experimental results of running our algorithm on real images are presented in Section 4. Finally, future research directions are discussed in Section 5.

2 Special Cases

In previous papers we have considered the general case of visual homing. We now discuss some special common cases in which the homing process can be simplified due to the special context in which the process takes place. In these cases the number of degrees of freedom is reduced from 6 to some context dependent value. In addition, given the position of a point in the first image, the position of its corresponding point in the second image might be constrained. And finally, in these restricted cases when there is no rotation in the image plane, correlation between the regions around feature points in the two images can be used to make a first guess as to which feature points in the first image match feature points in the second image. In the general case more general feature matching techniques must be used.

In each of these cases the algorithm will have the following steps: Feature points will be found using a corner detector (e.g. [11]). Pairs of features which are most similar (e.g. using the sum of squared distances (SSD)) will then be found. Usually, a large majority of these matches will be incorrect. In the final step, the correct transformation will be found and most incorrect matches will be discarded. This step will require robust methods as standard techniques are very intolerable to outliers.

We first discuss these cases under the weak perspective projection. We discuss the case of full perspective next. We will first address the most degenerate transformations and move to the less degenerate cases modifying the algorithm accordingly to deal with it.

2.1 Weak Perspective Projection

In the general weak perspective projection case, four matching points are required to recover all the parameters of the transformation which can be recovered from two images. These parameters are: the scale s between the images, the rotation in the image plane θ, the great circle on the viewing sphere on which the viewing direction $\mathbf{v}_z$ lies and the translation component in the image orthogonal to the direction of the great circle. The position of the viewing direction on the great circle and the other component of translation can not be found from two images (for details see for example [10]).

2.1.1 Homing with 2 DOF

We assume that the camera is mounted on a mobile (or Cartesian) robot whose only permitted motion is translation in the plane. In this case under weak perspective projection translation towards the target affects the scale parameter and translation in the orthogonal direction affects t_x. The viewing direction, $\mathbf{v}_z$, the rotation θ and the translation component t_y, are invariant to this type of constrained motion. Thus the effect of the motion is to scale the image and to move the points parallel to the x axis (on the epipolar lines). The order of the feature points sorted according to their position component that is orthogonal to the epipolar lines is invariant to motion. In this special case because we know the orientation of the epipolar lines, the feature points can be sorted according to their y coordinate. From the source and target images the feature points are extracted, and then sorted. They are then matched by a process which takes into account that not all points found in one image will be found in the other image. The ratio between the distances in the y direction between corresponding points is the scale and this constraint can be used for efficiently finding the matched pairs.

In the case where the optical center x_0, y_0 is known the line $y = y_0$ is invariant to motion in this case and a virtual feature point on this line in both images can be used to help the matching process. Each pair of feature points votes for a candidate scale and the scale with the most votes is returned.

After a successful match has been found we now turn to recover the motion parameters. The scale is recovered from the ratio of the distances between two pairs of corresponding points. After the image has been scaled, the distance between corresponding points is only due to t_x which can now be recovered easily.

2.1.2 Homing with 3 DOF

In this case we assume that the robot may also rotate in the plane, hence the viewing direction is a unit vector which lies in the $x-z$ plane (has a pan component only). This case is similar to the previous one except that the translation in the x direction in the image is due to two factors, change in the viewing direction $\mathbf{v}_z$, and the translation t_x, which as described for the general case can not be recovered from two images. The procedure for solving the correspondence between the feature points is the same as for the previous case. Thus we can only recover the scale between the two images. The viewing direction and translation can only be decoupled using an additional image.

2.1.3 Homing with 4 DOF

Up down

We assume that the camera can be elevated (e.g. periscope) as well. The case is similar to the previous case, only that a vertical translation component t_y has been added to the unknown motion parameters. Here we can not assume that the line going through $y = y_0$ is invariant to the motion, and the virtual feature on this line can not be used anymore in the matching process. Thus we need to use two pairs of matching points to compute the scale instead of only one as before. Once the scale has been compensated for, changes in the vertical direction are due to t_y and these can be easily recovered.

Tilt

When the robot head can be tilted the viewing direction can be an arbitrary unit vector on the viewing sphere. We therefore apply the general algorithm which recovers four of the six unknowns. However, these are not the four unknowns which exist in our problem (scale,viewing direction, and translation in the x direction). We will now show how to extract the missing parameters using the constraints we have on the transformation. We shall assume that the current image was taken at tilt $= 0$, or at least that the tilt is known which is usually the case as the camera's orientation is controlled by the robot. Then the image with tilt=0 can be computed from the current image (as explained in the next section). As the x coordinate is parallel to the floor in both images, the image rotation can be used to find where on the great circle is the actual viewing direction. This is different from the general case where the viewing direction can not be recovered. After finding the component of the translation in the direction orthogonal to the epipolar lines one can take advantage of the fact that the overall image translation is along the x axis and deduce it from its recovered component. Thus in this case, we are able to recover the complete transformation which is not true in general.

2.1.4 Homing with 5 DOF

Tilt combined with up down

This case is the same as the previous one only that in this case we have a general translation. In this case the component of the translation in the direction parallel to the epipolar lines can be recovered using the fact the we have recovered the viewing direction.

A general 5 DOF manipulator

Usually when given a general 5 DOF manipulator special case homing algorithms can not be derived and the general homing algorithm needs to be invoked. The derived motion instructions for the manipulator will yield (almost always) illegal configurations because the algorithm works in a general 6 DOF configuration space. In this case some simple method has to be devised to find the closest point to the solution in the 5 DOF configuration space.

As our homing algorithm is memoryless, it does not matter if the motion is in the exact trajectory given by our general homing algorithm as long as continuous progress is guaranteed. For an example of an algorithm for this type of manipulator see [3, 5].

2.2 Full Perspective Projection

In the full perspective case we find corresponding points and use them to recover the essential matrix E. The essential matrix enables us to recover R, the rotation matrix, and the orientation of the translation vector $\mathbf{t}$. In the general case 8 points are needed to recover the matrix using a linear method [6] or 5 for a non-linear method [7]. In the special cases we will exploit the limited possible motions of the robot to reduce the number of correspondences needed.

2.2.1 Homing with 2 DOF

In this case the rotation matrix R is the identity matrix. The translation vector is $\mathbf{t} = (t_x, 0, t_z)$. Then

$$E = \begin{pmatrix} 0 & -t_z & 0 \\ t_z & 0 & -t_x \\ 0 & t_x & 0 \end{pmatrix}.$$

One point correspondence gives t_x/t_z. In this special case the epipole is in the same place, $(t_x/t_z, 0)$, in the two images and lies on the x axis. The line that connects two corresponding points will pass through the epipole. Every pair of suspected corresponding points votes for a position of the epipole. The participant points should be matched using a matching process which can evaluate the goodness of the match. One should also take into account the uncertainty involved in the exact positioning of a feature point. This case is discussed in detail in the next section and was addressed also in [12].

2.2.2 Homing with 3 DOF

In this case the rotation matrix R which is only due to a pan by θ is

$$R = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix}.$$

The translation vector is $\mathbf{t} = (t_x, 0, t_z)$. Then

$$E = \begin{pmatrix} 0 & -\cos(\theta)t_z + \sin(\theta)t_x & 0 \\ t_z & 0 & -t_x \\ 0 & \sin(\theta)t_z + \cos(\theta)t_x & 0 \end{pmatrix}.$$

Three points correspondences are needed to solve for E linearly. For the non-linear case 2 points are sufficient. For each tested θ, the second image can be derotated, eliminating the effect of the pan by θ, leaving us with the previous case. When the correct θ is chosen all lines connecting all correct matches will intersect the x axis at the same place. We will discuss this case in detail in the next section.

2.2.3 Homing with 4 DOF

Up down

The rotation matrix R in this case is the same as in the previous case. The translation vector is $\mathbf{t} = (t_x, t_y, t_z)$. Then

$$E = \begin{pmatrix} -\sin(\theta)t_y & -\cos(\theta)t_z + \sin(\theta)t_x & \cos(\theta)t_y \\ t_z & 0 & -t_x \\ -\cos(\theta)t_y & \sin(\theta)t_z + \cos(\theta)t_x & -\sin(\theta)t_y \end{pmatrix}.$$

Five points correspondences are needed to solve for E linearly due to the degeneracy of E. The robust algorithm described for the previous case can be generalized for this case. The only difference is that the epipole can lie in any place in the plane and not only on the x axis. All lines connecting correct matches must intersect at that point for a correct θ. Thus at least 3 point matches are needed to recover the transformation.

Tilt

The rotation matrix R in this case is the multiplication of the two rotation matrices around the X and Y axes where the rotations were θ_1 and θ_2 respectively. The translation vector is $\mathbf{t} = (t_x, 0, t_z)$.

As in the weak-perspective case, we must assume that the pan and tilt of the current image are known. Six point correspondences are needed to solve for E linearly due to two zero elements in E. For the non-linear case one can use the following geometric argument to show that 3 point matches are sufficient. For each tested θ_1, the angle θ_2 is computed such that lines from two matching pairs intersect at the same point on the x axis. A third pair is needed to verify that this is indeed the correct (θ_1, θ_2).

2.2.4 Homing with 5 DOF

Tilt combined with up down

This case is the same as the previous one only that in this case we have a general translation.

Eight points correspondences are needed to solve for E linearly. For the non-linear case a generalization of the previous argument can be used to show that 4 point matches are needed. This case is addressed in [1].

A general 5 DOF manipulator

Usually in the general case with 5 DOF special case homing algorithms can not be derived and here too, as in the the weak-perspective case, the general homing algorithm needs to be invoked. The constraints on the transformation can be used to choose only a permitted transformation which leads to the goal pose.

3 Planar Rotation and Translation

In this section we will describe in detail an algorithm which deals with a stationary camera mounted on a mobile robot which is translating and rotating in the plane. This algorithm can be used to help localize a mobile robot or be the first and hardest step in a homing algorithm whose remaining steps have been described in [4]. The algorithm is given as input the two images. It has to extract image features in the two images attempt to find corresponding features (usually with poor results) and use the results of this step to recover the rotation angle of the robot and the direction in the plane towards the position from which the target image was taken.

We will first deal with the simpler problem of a robot which only translates in the plane which has also been addressed by [12] and then extend this solution to deal also with rotation.

3.1 Robot Translating in the Plane

Given two images taken by a camera whose center of projection is (x_0, y_0) and whose focal length is f. Assuming that the first image was taken at $O = (0,0,0)$ and the second at $O' = (X, 0, Z)$, all lines connecting corresponding points should intersect the line $y = y_0$ (the x axis) at the point (x, y_0) such that $(x - x0, y_0 - y_0, f) \cong (X, 0, Z)$ ($\mathbf{p} \cong \mathbf{p}'$ means that $\mathbf{p} = k\mathbf{p}'$ for some k). Thus the direction of motion toward the target pose is $\alpha = \tan^{-1}((x - x_0)/f)$. In order to determine if the motion should be forward or backward, use the fact that when we move forward, points in the image move further away from the x axis. We use all these facts in designing our algorithm.

We first use the SUSAN corner detector [11] to detect point features in the two images. Each of the corners in the first image is matched with all of the corners in the second image and vice-versa using the SSD criterion. When a pair of corners receives the best score in the two tests it is added to the list of correspondences. From this list we remove all pairs of points who lie on both sides of the x axis (Figure 1(a)). We then remove all point pairs whose direction of motion with respect to the x axis is different than the majority (Figure 1(b)). These two measures remove a large percentage of the incorrect matches and determine if the motion is forward or backward.

Each of the remaining matches votes for the point x_e on the x axis which lies on the infinite line connecting the two points.

$$x_e = (x_2(y_0 - y_1) + x_1(y_2 - y_0))/(y_2 - y_1), \qquad (1)$$

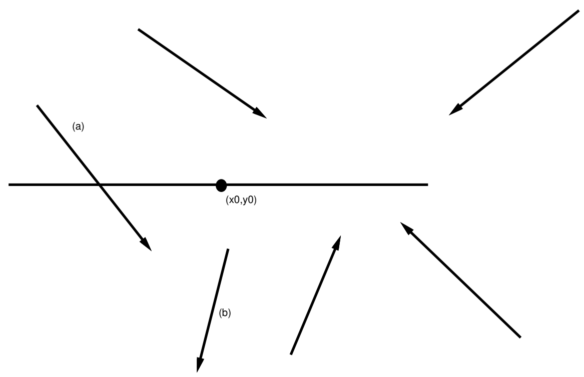

Figure 1: An illustration of a typical group of pairs of matching points and pairs which can be discarded. (a) A pair of points where one of the points is above and the other is below the x axis; (b) A pair of points where the first point is closer to the x axis than the second one unlike the majority of point pairs.

where (x_1, y_1) and (x_2, y_2) are the matching pair of points in the image. The point with the largest number of votes is chosen as the epipole.

This approach fails to account for the uncertainty in the position of the feature points. In order to deal with in the uncertainty in the position of x_e, we find the first order contribution of the error in the coordinates of the two points to x_e by differentiating (1) with respect to the respective coordinate. The error due to x_1 and y_1 are:

$$dx_{x1} = (y_2 - y_0)/(y_2 - y_1)dx1$$
$$dx_{y1} = (y_2 - y_0)/(y_2 - y_1)^2(x_2 - x_1)dy1$$

where $dx1$ and $dy1$ are the errors in x_1 and y_1 respectively. Similar expressions are obtained for x_2 and y_2. Thus, if the uncertainty of each coordinate has a normal distribution with variance σ^2, x_e can be approximated as a normal distribution with variance

$$\sigma_x^2 \approx (dx_{x1}^2 + dx_{y1}^2 + dx_{x2}^2 + dx_{y2}^2)\sigma^2.$$

Thus with very high probability the correct x should have a value within $x_e \pm 2\sigma_x$. In normal distribution points closer to x_e have a higher probability then points further away and should therefore get a higher score. In addition pairs with smaller variances have higher probabilities then pairs whose distribution has a larger variance. This is true when all correspondence pairs are correct, however when outlier pairs are involved a single incorrect pair can have a tragic effect. For example, very long segments connecting the two corresponding points or a pair where one of the points is very close to the x axis can yield distributions with very small variances giving a disproportional score for an incorrect x_e. Thus we decided to give score 1 for each point in the $x_e \pm 2\sigma_x$ region. This produces a slightly less accurate result when all corresponding pairs are correct but is robust to incorrect matches.

In the final step of the algorithm, x_e with the maximal score is chosen. Pairs of features which did not vote for the selected x_e and are therefore probably outliers are discarded and the remaining pairs are outputted to be used in the future steps of the homing algorithm.

3.2 Robot Translating and Rotating in the Plane

Extending the algorithm to deal also with rotation can be achieved using the following fact. Given an image taken by a perspective projection camera, an image that could have been taken by that camera but from a different orientation can be computed from the image with no knowledge of the scene, by simply applying the rotation matrix R of the new orientation to the points in the original image yielding the points in the new image.

$$\mathbf{p}' \cong R\mathbf{p}.$$

Thus for each suspected rotation angle θ we can derotate the second image by its respective rotation matrix $R(\theta)$ and apply the algorithm described above to the derotated image. The highest score should be achieved for the correct θ at the correct epipole. The only question which has to be answered is whether the algorithm which worked satisfactorily when the search was one dimensional (x_e) will also work when the search is two dimensional (x_e, θ). To answer this question we ran experiments on real images which are described in the next section.

4 Experimental Results

We have implemented the algorithm described above and present here results of it tested on several real images of an unknown scene shown in Figure 2.

For each of the runs of the program we display in Table 1 the following information.

- How many features were found by the corner detector in each of the images.
- How many pairs of features were found using the SSD matcher.
- How many pairs of features remained after impossible pairs were removed.
- The final number of pairs outputted by the algorithm.

Image Pair	Features	Pairs	Pairs Left	Outputted Pairs
1-4	(358,301)	143	73	22
3-2	(304,370)	152	106	62
3-4	(304,301)	146	81	34

Table 1: Pairs.

In Figure 3 we show the accumulator for the translation and rotation angles. The brightest point in the accumulator represents the best translation, rotation pair which is outputted by the algorithm.

Figure 3: The accumulator of the results of running the algorithm on one of the image pairs. The horizontal axis represents the translation angle and the vertical axis the translation. The brightest pixel is chosen as the answer.

In addition, to check the quality of the results we present in Table 2 for each run of the algorithm the following three results: The results obtained by running our algorithm, the results obtained by running our algorithm on only correct matches, and the results obtained by running a general transformation finding algorithm which does not assume that we have a degenerate transformation. This algorithm is also run only on correct matches.

The difference between the first and second run show the small effect of several outliers that were left within the matched pairs and which voted for candidates close to the correct result and caused the small change in the result. The third result is as close as we can measure to the ground truth. The relatively large error in orientation (2.7°) in the third experiment is mainly due to the fact that the translation is virtually in the x direction with no z component. In these cases it has been shown in [2] that the translation can not be found accurately even by the general algorithm which also affects the accuracy of the recovered rotation angle.

In conclusion, these experiments show that the algorithm is able to produce quite accurate results auto-

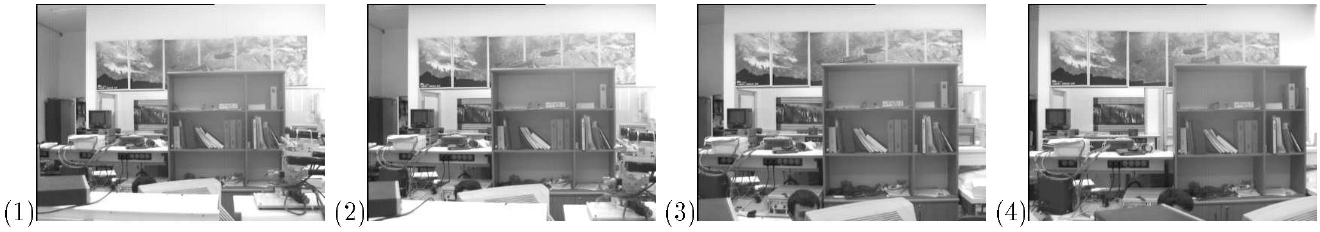

Figure 2: Four images of a scene.

matically when only the two images are given as input. These results enable the robot to start progressing towards the goal. Then new images are taken, correct matches are tracked, incorrect matches are discovered and discarded and the algorithm is rerun guiding the robot to the goal as described in [3, 5].

Image Pair	Algorithm Result	Clean Input	General Algorithm
1-4	(0,-9.5,0) (-56,0)	(0,-9.1,0) (-54,0)	(0.12,-8.0,-0.22) (-41,-1)
3-2	(0,2.9,0) (151,0)	(0,3.1,0) (148,0)	(0,2.7,0) (149,-2)
3-4	(0,-3.5,0) (-80,0)	(0,-3.5,0) (-80,0)	(0,-6.2,-0.2) (-85,-13)

Table 2: Results of running the algorithm on several image pairs: each result is represented by the axis of rotation and its magnitude in degrees (which should only have a y component in our model) and a translation direction (which should only have an x/z component in our model). The third algorithm which was run did not make that assumption and found small values in the components which should be zero.

5 Conclusions

In this paper we have introduced several classes of visual homing algorithms which deal with cases where the sought after transformation was limited. These cases are very common when the camera is mounted on a mobile robot. We have shown ways to solve these problems more easily and more robustly then the general homing algorithm. To demonstrate the applicability of these algorithms we implemented the algorithm for the most common problem of a fixed camera mounted on a mobile robot rotating and translating in the plane and presented results of running it on real images.

Future work will concentrate on implementing other classes of algorithms described here and incorporating the homing algorithm as part of a mobile robot navigation and localization system.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust motion estimation. In preparation.

[2] A. Adam, E. Rivlin, and I. Shimshoni. Computing the sensory uncertainty field of a vision-based localization sensor. Submitted to *IEEE Int. Conf. on Robotics and Automation*, 2000.

[3] R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. In *Proc. Int. Conf. Comp. Vision*, pages 863–869, 1998.

[4] R. Basri, E. Rivlin, and I. Shimshoni. Image-based robot navigation under the perspective model. In *IEEE Int. Conf. on Robotics and Automation*, pages 2578–2583, 1999.

[5] R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. *Int. J. of Comp. Vision*, 1999, to appear.

[6] R. Hartley. In defence of the 8-point algorithm. In *Proc. Int. Conf. Comp. Vision*, pages 1064–1070, 1995.

[7] E. Kruppa. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 122:1939–1948, 1913.

[8] E. Malis, F. Chaumette, and S. Boudet. 2-1/2-D visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, April 1999.

[9] P. Pritchett and A. Zisserman. 3D motion recovery via affine epipolar geometry. In *Proc. Int. Conf. Comp. Vision*, pages 754–760, 1998.

[10] I. Shimshoni, R. Basri, and E. Rivlin. A geometric interpretation of weak-perspective motion. *IEEE Trans. Patt. Anal. Mach. Intell.*, 21(3):252–257, March 1999.

[11] S.M. Smith and J.M. Brady. SUSAN - a new approach to low level image processing. *Int. Journal of Computer Vision*, 23(1):45–78, May 1997.

[12] D. Tell and S. Carlsson. View based visual servoing using epipolar geometry. In *Proc. Scandinavian Conf. Image Analysis*, pages 63–70, 1999.

[13] T. Tuytelaars, L. Van Gool, L. D'haene, and R. Koch. Matching of affinely invariant regions for visual servoing. In *IEEE Int. Conf. on Robotics and Automation*, pages 1601–1606, 1999.