Dynamic Fixation in an Active Visual Agent

Mårten Björkman and Jan-Olof Eklundh

Abstract

Stereo is an important cue in robotics and in order for an autonomous robot to work under different conditions, dynamic vergence can be used to overcome limitations in image resolution and field of view. In this paper a binocular stereo system capable of dynamic vergence is presented. The external calibration is performed continuously taking temporal consistency into consideration, greatly simplifying the process. The essential matrix, which is estimated in real-time, is used to describe the epipolar geometry. It will be shown how outliers can be identified and excluded from the calculations. An iterative approach based on a differential model of the optical flow, as commonly used in structure from motion, is also presented and tested towards the essential matrix. The iterative method will show to be superior in terms of both computational speed and robustness, when the vergence angles are less than about 15°. For larger angles, the differential model will show to be insufficient and the essential is used instead. Both methods have been integrated into a system able to compute dense disparity maps in real-time. These maps will finally be used for a number of visually guided robot behaviours.

I. Background

Stereo vision can be used for several tasks in the area of robotics. For applications, such as visually guided manipulation, stereo is a valuable cue in order to recognize shapes and orientations. Numerous groups [11], [23] have also been using stereo for visual navigation and obstacle detection. These systems typically consists of two or more small cameras located in parallel with relatively short baselines. Since the systems are expected to work for various distances, small baselines become a necessity. Objects located close to the observer might otherwise only be seen by one of cameras, greatly complicating depth estimation and making the system useless for things such as obstacle avoidance. However, for manipulation accuracy is crucial and a wider baseline is desired.

This discussion leads us to believe that a mobile platform would benefit from an active stereo vision system with dynamic vergence. Based on the activity at hand, an active system can control its gaze in such a way that, the visual system is always used at its full potential. Ballard [2] gave a number of reasons how active vision can be used to overcome limitations in resolution and field of view, making the visual computations much less expensive, than that of a passive system. Using fixation many visual operations, such as visual navigation, 3D motion estimation and figure-ground segmentation, can further be simplified. Fermüller and Aloimonos [6] used normal-flow in conjunction with fixation to solve problems such as ego-motion recovery, 3D motion and time-to-impact estimation. Daniilidis [5] decoupled the optical flow and found the translation of an observer through two one-dimensional searches, radically simplifying the ego-motion estimation process.

One of the reasons why dynamic vergence has seldom been used in practical robotics, is the problem of external calibration. The relative orientation of the cameras have to be known, in order for the stereo vision system to compute accurate estimates of depths in metric space. Even if the mechanics of the system is highly advanced and the position of every joint can be measured at any given point in time, it might not be enough. An initial calibration typically has to be performed anyway, making sure that nothing has happened to the system and all position measurements are correct. A system based on the external calibration being performed continuously or at least every now and then, may be cheaper

to manufacture and more likely to handle disturbances the robot might suffer from.

In this work the essential matrix, as described by Longuet-Higgins [14], is used to establish the correspondence between two projections of a single scene. The fact that a stereo head system in fixation constrains the number of possible configurations, the number of degrees of freedom can be kept low. The essential matrix used here has earlier been presented in a review of stereo head configurations by Brooks et al [3]. The essential matrix is found using linear optimization and compared to results that of nonlinear optimization, as well as to a method [25] based on the full set of degrees of freedom. Robust statistics is utilized to make the results less sensitive to outliers in the dataset. Typically [7], [22] some kind of support measure is used to evaluate the sufficiency of estimates, but calculating such a measure can be relatively costly on a constrained system like a fixating stereo head system. Instead a simple analysis of eigenvectors is performed, reducing the computational cost without affecting the robustness of estimates.

Commonly used in structure from motion is differential motion as described by Longuet-Higgins and Prazdny [13]. The use of differential motion in stereo is limited, because it is expected that the motion of the camera from one frame to another is small compared to the distances of objects in the scene. However, in this paper it will be shown that these requirements in fact hold for a stereo head system used under normal conditions. This is due to the fact that typical vergence angles are small and the baseline short compared to distances in 3D space. A simple iterative method for estimation of structure and geometry, similar to that of Heel [10], is presented and evaluated towards approaches based on the essential matrix. It will be shown that not just the speed, but also the robustness, will be superior to methods using the essential matrix. However, the method is based on the assumption that the median to objects in 3D space is approximately known. In most real application this is not a serious limitation.

The remainder of this paper is organized as follows. In Section II the essential matrix of a typical binocular stereo-head will be derived and an explanation of its parameters given. Section III describes how the parameters can be estimated and outliers identified in an efficient manner. Simulations will be performed in Section IV and the performance of the alternative methods evaluated. A very simple and efficient iterative approach is presented in Section V and tested towards methods based on the essential matrix. In Section VI, the stereo system is integrated in a complete system in which dense disparity maps are created and used for mobile navigation. Finally, the system will be tested through a series of real-time experiments as described in Section VII. The article ends with conclusions and a discussion about future research.

II. Epipolar geometry

A typical binocular stereo system consists of two different cameras separated by a fixed baseline, as shown in Figure 1. Before we are able to say anything about metric distances to objects in 3D space viewed by the two cameras, we have to know the orientation of both cameras. In this section the essential matrix is used to describe such a relation. Since we are interested in a system where the cameras are in constant fixation, we may use this fact to keep the structure of the essential matrix as simple as possible.

In order to describe the geometry, a coordinate system is needed. We may use the baseline to define a x-axis, as seen in Figure 1. The stereo-head being a fixating system implies that the stereo-head is constrained, such that the two optical axes always will intersect somewhere in-front of the cameras. This point together with the two camera centers define a plane and we may the use the normal of this plane as y-axis, while the z-axis is located on the plane and perpendicular to the baseline.

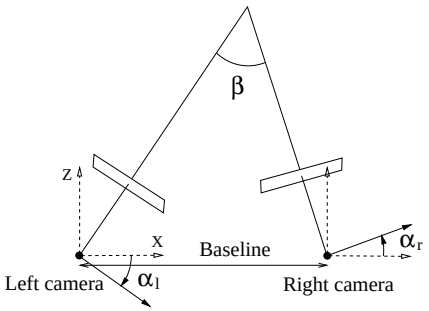

Fig. 1. Stereo-head system

For an unconstrained system we might expect as many as six degrees of freedom, three rotational degrees for each camera. However, we easily see that the full flexibility of the system might be kept, even if we only use two degrees. We do not need any rotations around the optical axes, because such rotations will not change the visual data, only its orientation. Since the cameras are kept fixated, the optical axes have to lie on the same plane and there is no relative tilt between the cameras. A joint tilt of both cameras around the baseline, will not change the nature of the problem and can be ignored.

Consequently, we only have to consider two degrees of freedom, that is the rotations around the y-axis of each camera, represented by the angles α_l and α_r in Figure 1. These rotations can be described by the corresponding rotation matrices

$$\mathbf{R}_{\alpha_l} = \begin{pmatrix} \cos(\alpha_l) & 0 & \sin(\alpha_l) \\ 0 & 1 & 0 \\ -\sin(\alpha_l) & 0 & \cos(\alpha_l) \end{pmatrix} \quad \text{and} \quad \mathbf{R}_{\alpha_r} = \begin{pmatrix} \cos(\alpha_r) & 0 & -\sin(\alpha_r) \\ 0 & 1 & 0 \\ \sin(\alpha_r) & 0 & \cos(\alpha_r) \end{pmatrix}. \tag{1}$$

Using the rotation matrices we derive the projection matrices $\mathbf{P}_{\alpha_l}$ and $\mathbf{P}_{\alpha_r}$. These two matrices describe how an object point $\mathbf{u} = (x, y, z, w)^T$, shown here in homogeneous coordinates, is being projected onto image planes of the left and right cameras. The image point in the left image, $\mathbf{x_l} = (x_l, y_l, w_l)^T$, is a result of the projection $\mathbf{x_l} = \mathbf{P}_{\alpha_l}\mathbf{u} = \mathbf{R}_{\alpha_l}(\mathbf{I}|\mathbf{0})\mathbf{u}$. For the right camera the projection also includes a translation along the baseline, $\mathbf{t} = (k, 0, 0)^T$, which means that the right image point $\mathbf{x_r} = (x_r, y_r, w_r)^T$ can be expressed as $\mathbf{x_r} = \mathbf{P}_{\alpha_r}\mathbf{u} = \mathbf{R}_{\alpha_r}(\mathbf{I}| - \mathbf{t})\mathbf{u}$.

Using the fact that the two image points, originating from a certain point in space, and the camera centers have to be located on the same plane, it is possible to derive a constraint known as the epipolar constraint. If an image point is known in one window, the corresponding point in the other image will be constrained to a line as defined by the following equation:

$$\mathbf{x_l^T}\underline{\mathbf{E}}\mathbf{x_r} = \mathbf{x_l^T}(\mathbf{R}_{\alpha_l}\mathbf{T}\mathbf{R}_{\alpha_r}{}^{\mathbf{-1}})\mathbf{x_r} = \mathbf{0}, \tag{2}$$

where $\mathbf{T}$ is the skew-symmetric matrix associated to the translation $\mathbf{t}$, that is

$$\mathbf{T} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & k \\ 0 & -k & 0 \end{pmatrix}. \tag{3}$$

After performing some simple calculations, we find that the essential matrix describing the epipolar constraint can be

expressed by

$$\underline{\mathbf{E}} = k \begin{pmatrix} 0 & -\sin(\alpha_l) & 0 \\ -\sin(\alpha_r) & 0 & \cos(\alpha_r) \\ 0 & -\cos(\alpha_l) & 0 \end{pmatrix}. \tag{4}$$

This constraint has got one major advantage; the stereo problem has efficiently been decoupled from the structure problem. We can find the relation between the two cameras, without knowing the exact depth for each and every point seen by the cameras. The essential matrix can further be decomposed into a product $\mathbf{RS}$ of two separate components, a rotational component and a translational one. The component $\mathbf{R} = \mathbf{R}_{\alpha_l}{\mathbf{R}_{\alpha_r}}^{-1}$, is a rotation matrix that describes the relative orientation between the two cameras. This factor is important in order to be able to relate the disparities, that is the differences in image positions between the left and the right camera, to actual distances to points in space. The skew-symmetric translational component $\mathbf{S} = \mathbf{R}_{\alpha_r}\mathbf{T}{\mathbf{R}_{\alpha_r}}^{-1}$ can be considered as the position of the right camera relative to the left.

A. Linear optimization

As we saw earlier the true essential matrix can be found in a two-dimensional space of possible epipolar constraints. This space is spanned by the two angles α_l and α_r, and an optimal estimation of the true matrix $\underline{\mathbf{E}}$ could possibly be found solving a nonlinear optimization problem. However, such a problem typically relies on a number of iterations and an initial guess close to the true solution is critical. An alternative linear approach is finding the matrix $\mathbf{E}$ that minimizes the error $\mid \mathbf{x_l^T E x_r} \mid$, where

$$\mathbf{E} = \begin{pmatrix} 0 & f_1 & 0 \\ f_2 & 0 & f_4 \\ 0 & f_3 & 0 \end{pmatrix}. \tag{5}$$

After finding an optimum in the four-dimensional space spanned by the parameters f_j, $j = 1 \ldots 4$, a projection to the two-dimensional manifold of possible epipolar constraints is done, using an approach described by Hartley [9]. This method is based on the fact that a matrix can only be factored into $\mathbf{R}$ and $\mathbf{S}$, a characteristic feature of essential matrices as described earlier, if and only if the matrix has two equal non-zero singular values and one singular value equal to 0. Following Hartley, the singular value decomposition of $\mathbf{E}$ can be shown to be $\mathbf{UDV^T}$, where

$$\mathbf{U} = \frac{1}{\sigma_1}\begin{pmatrix} f_1 & 0 & f_3 \\ 0 & \sigma_1 & 0 \\ f_3 & 0 & -f_1 \end{pmatrix}, \qquad \mathbf{V} = \frac{1}{\sigma_2}\begin{pmatrix} 0 & f_2 & -f_4 \\ \sigma_2 & 0 & 0 \\ 0 & f_4 & f_2 \end{pmatrix} \tag{6}$$

and $\mathbf{D}$ is a diagonal matrix with the singular values $\sigma_1 = \sqrt{f_1^2 + f_3^2}$, $\sigma_2 = \sqrt{f_2^2 + f_4^2}$ and 0. A new essential matrix is found, substituting the epipolar geometry $\mathbf{E}$ with an approximation $\hat{\mathbf{E}} = \mathbf{U}\hat{\mathbf{D}}\mathbf{V^T}$, where $\hat{\mathbf{D}} = diag(\sigma, \sigma, 0)$ and $\sigma = (\sigma_1 + \sigma_2)/2$. This matrix is the one closest to the estimated matrix $\mathbf{E}$ under the Frobenius norm, that satisfies our requirements. We finally end up with an essential matrix that looks like the following:

$$\hat{\mathbf{E}} = \begin{pmatrix} 0 & \frac{f_1}{\sigma_1} & 0 \\ \frac{f_2}{\sigma_2} & 0 & \frac{f_4}{\sigma_2} \\ 0 & \frac{f_3}{\sigma_1} & 0 \end{pmatrix}. \tag{7}$$

As mentioned earlier a valid epipolar constraint can be divided into two different components, one rotational component and one skew-symmetric translational one. If we factorize the essential matrix in Equation 7 into two such matrices, we get the rotation matrix

$$\mathbf{R} = \begin{pmatrix} c_\beta & 0 & -s_\beta \\ 0 & 1 & 0 \\ s_\beta & 0 & c_\beta \end{pmatrix}, \tag{8}$$

with the elements $c_\beta = -(f_1 f_2 + f_3 f_4)/\sigma_1\sigma_2$ and $s_\beta = (f_1 f_4 - f_2 f_3)/\sigma_1\sigma_2$. Considering the fact that the determinant equals $c_\beta^2 + s_\beta^2 = 1$, we conclude that $\mathbf{R}$ really is a rotation and that c_β and s_β can be seen as the cosine and sine of the vergence angle β in Figure 1. The corresponding translational component

$$\mathbf{S} = \frac{1}{\sigma_2} \begin{pmatrix} 0 & -f_2 & 0 \\ f_2 & 0 & f_4 \\ 0 & -f_4 & 0 \end{pmatrix}, \tag{9}$$

is the second factor of the essential matrix $\hat{\mathbf{E}}$. In the same manner, the elements f_2/σ_2 and f_4/σ_2 can be identified as $\sin(\alpha_r)$ and $\cos(\alpha_r)$.

B. Additional parameters

In the previous section we described how to find the epipolar geometry with only two degrees of freedom taken into consideration. It can be argued whether more parameters ought to be added to the optimization problem. We have chosen to minimize the number of parameters out for two different reasons. The primary reason is speed. The more often we are able to update our estimated epipolar geometry, the easier it will be to take advantage of likely temporal consistency. Secondly, too many parameters added to the problem might seriously affect the robustness of the estimations. Oliensis and Govindu showed in [20] that a projective optimization, where internal camera parameters are treated as unknowns, does not necessarily result in more robust results, than an Euclidean case where the "known" parameters are in fact just approximate.

In case of a varying focal length, the fact that the essential matrix only can be estimated up to a constant means that there will be one parameter missing, in order to determine the focal lengths of both cameras. However, if the lengths are known to be equal, the same parameters f_j, $j = 1 \ldots 4$ can be used to estimate the focal lengths, but at the cost of a considerably higher variance, especially in the rotational component. It can be shown that the errors may be extremely large if the camera configuration is close to symmetric, such as in our case. Due to this fact we have chosen not to include such an estimation in the process. In fact, a moderately erroneous focal length does not hurt the robustness to a great extent, but the results will be slightly biased. More critical to the robustness is a relative difference between the focal lengths of the two cameras. However, this difference can be robustly estimated in a separate operation, simply by assuming that one of the cameras has a known focal length and searching the other. Even if the known focal length is just approximately known, the calculated relative difference will be very close to the true result.

For a typical stereo-head system, it is sometimes hard to full-fill the requirement that there is no relative tilt between the two cameras. Due to the mounting of the cameras and the placement of the actual CCD chips inside the cameras, the center of the images can easily be separated by as much as 10 pixels. Without adjusting the extracted feature

positions to such a separation, the final estimated geometry will either be useless or seriously biased. In our system, we use the fact that no vertical disparities should be expected along the x-axis of the images, that is the horizontal line in the vertical center of the images. As an initialization, features are extracted from a sequence of pairwise images. Using the feature correspondences close to the x-axis, the linear relation between the x-positions and the vertical disparities is found, fitting a straight line to the data. Least median squares were used as objective function in our system. Once the epipolar estimation process has stabilized, rectified data can be used instead. Through the same procedure, data may also be adjusted to small erroneous relative rotations around the optical axes.

III. Parameter estimation

As described in Section II-A the epipolar geometry can be found through a linear optimization in the four-dimensional space spanned by the parameters f_j, $j = 1 \ldots 4$. When estimating these parameters, it is expected that we have a number of extracted corner features. Each point in 3D space will be projected onto the two camera image planes, producing a pair of image features $(\mathbf{x}_\mathbf{l}^\mathbf{i}, \mathbf{x}_\mathbf{r}^\mathbf{i})$, as shown in Section II. In our system, we use Harris' corner detector [8] to locate corners in the left and right images. These features have then to grouped into pairs, before the actual epipolar geometry estimation start.

In practise, this matching is far from trivial and a good matching is most critical for the performance of the system. Since the corners are viewed from two different angles by two different physical cameras, they might look slightly different from the two cameras. If transparencies and specularities exist in the scene, the matching procedure is further complicated. It is also possible that due to occlusions some points may only be seen by one of the cameras and the corresponding features can not be matched at all. A final reason why erroneous pairs might be created is because repeatable patterns exist in the scene and many points actually look much the same.

In order to match features from image to image, we use normalized cross-correlation of 9×9 pixel image patches around the extracted corners. Each feature in one image is tested towards all possible corners in the other image and for each combination of two features, a corresponding matching strength is calculated. Based on matching strength a favourite, for each corner in one image, is found among the features in the other image. To make sure that the number of erroneous pairs, known as outliers, is as low as possible, we only use pairs with two corners that are eachothers favourites. If $\mathbf{x_r}$ is the favourite of $\mathbf{x_l}$, then $\mathbf{x_l}$ must be the favourite of $\mathbf{x_r}$, in order for $(\mathbf{x_l}, \mathbf{x_r})$ to be accepted as a good feature pair. Only such valid pairs will be used in the epipolar geometry estimations and they represent about 40% of the total number of corner features. Empirically, we have seen that the vast majority, 80%-90%, of those pairs that are considered as valid pairs are in fact valid, in that they originate from the same object point in 3D space.

A. Least square optimization

A natural way to recover the epipolar geometry, could be finding the optimal estimate in a least square sense, using all the available correspondences found during matching process. The more data we use, the better we may suppress the influence of noise. However, since the errors used in the optimization are squared, even single outliers in the data set may make the estimate useless. A common way of reducing the effect of such outliers, is the use of random sampling. Instead of using as many features as possible in the calculations, we use a minimum set of randomly selected feature pairs, so as to minimize the likelihood that the set includes one or more outliers. For such a set, $M(j)$, the corresponding

parameters are being calculated, minimizing the objective function

$$\sum_{i \in M(j)} (f_1 x_l^i y_r^i + f_2 y_l^i x_r^i + f_3 w_l^i y_r^i + f_4 y_l^i w_r^i)^2. \tag{10}$$

The procedure is repeated numerous times, and for each set the estimated epipolar geometry is being evaluated. A commonly used method known as the random sampling consensus paradigm (RANSAC) [7], uses the complete set of correspondences in order to perform such evaluations. The estimate that the highest number of feature pairs can agree on, is selected as the best estimate of the true epipolar geometry. However, since all image correspondences are being used for each and every evaluation, this process can be time-consuming and might hurt the performance of the complete system.

Unlike ordinary least square optimization as given by Equation 10, where the errors are considered in one coordinate only, an often more successful approach is orthogonal least square minimization. We try to fit a hyper-plane $\mathbf{f} = (f_1, f_2, f_3, f_4)^T$, located in a four-dimensional space, to a set $M(j)$ of points given by $\mathbf{z_i} = (x_l^i y_r^i, y_l^i x_r^i, w_l^i y_r^i, y_l^i w_r^i)^T$ and minimize the sum of squared orthogonal errors from the points to the hyper-plane, that is minimizing the expression

$$\sum_{i \in M(j)} (\mathbf{f^T z_i})^2 = \sum_{i \in M(j)} \mathbf{f^T (z_i z_i^T) f} = \mathbf{f^T M f}, \tag{11}$$

subject to $\mathbf{f^T f} = 1$. The second moment matrix $\mathbf{M}$ is a 4×4 matrix with eigenvalues λ_1, λ_2, λ_3 and λ_4 in increasing order, and the corresponding eigenvectors $\mathbf{u_1}$, $\mathbf{u_2}$, $\mathbf{u_3}$ and $\mathbf{u_4}$. The eigenvector $\mathbf{u_1}$ is equivalent to the vector $\mathbf{f}$ that minimizes Equation 11 and λ_1 is the smallest sum of squared errors that we are able to get. The corresponding epipolar geometry, after performing a projection down to the two-dimensional manifold of possible essential matrices, is given by Equation 7.

B. Inverse Power method

Since the speed is of major concern in our system, it is important that computations can be made as fast as possible. Since $\mathbf{u_1}$ is the only eigenvector we are really interested in, we do not bother to find all the other eigenvectors. Hence we may use an iterative approach such as the Inverse Power method, if we can make sure that convergence will be achieved. We begin calculating the adjoint of the matrix $\mathbf{M}$, that is

$$\mathbf{M_a} = adj(\mathbf{M}) = det(\mathbf{M}) \cdot \mathbf{M^{-1}}. \tag{12}$$

The reason why we use the adjoint, instead of the inverse of $\mathbf{M}$, is simply because the system might be ill-conditioned and we do not like to divide the adjoint by the determinant, when calculating the inverse. Since the final eigenvector will be normalized anyway, this scaling does not change the final result. Using an arbitrarily chosen initial vector $\mathbf{v_0}$, we iterate a number of times and calculate

$$\mathbf{v_k} = \frac{\mathbf{M_a v_{k-1}}}{|\,\mathbf{M_a v_{k-1}}\,|}. \tag{13}$$

It can be shown that upon convergence $\mathbf{v_k}$ will converge towards $\mathbf{u_1}$, as $k \to \infty$, and the error will decrease at a rate of λ_1/λ_2. The question is what this actually means.

Analyzing the eigenvalues of $\mathbf{M}$ shows that there is a strong dependency between the scales of the eigenvalues and different aspects of selected points, such as the noise level of extracted corner features and how the points are being distributed in image space and depth. The smallest eigenvalue λ_1 increases quadratically as the level of noise is doubled, but does not change very much as the distribution of points is varying. For the second smallest eigenvalue λ_2 the dependencies are reversed. As the points are spread twice as much over the image space or in depth, the eigenvalue doubles for each dimension. The same eigenvalue is, however, not very sensitive to noise. In conclusion, a low value of λ_1/λ_2 is equivalent to the selected feature points being widely spread in space and suffering from a limited amount of

noise. Thus if convergence is achieved using the Inverse Power method, one can assume that a proper set feature points has been selected.

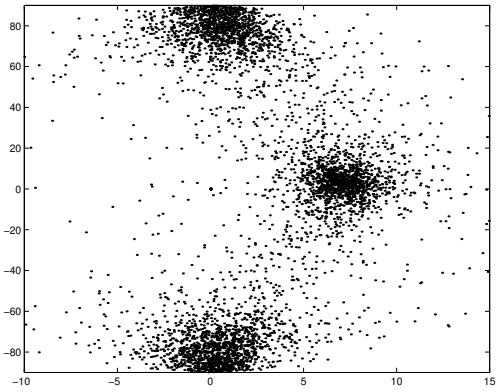

Fig. 2. Distribution of sample points when estimating the epipolar geometry. The x-axis represents the rotational and the y-axis the translational component. The data set includes 500 feature pairs, out of which 20% are outliers.

C. Quality testing estimates

Unfortunately, convergence is not enough to guarantee that the estimate is actually a good estimate not too far away from the real epipolar geometry. Figure 2 show a number of sample points found, when estimating the epipolar geometry according to the scheme described above. Further experiments will be presented in greater detail later in Section IV. As seen in the figure, the points are clustered around more than one center in the two-dimensional space of epipolar geometries. The cluster center near $(7, 3)$ corresponds to the true epipolar geometry. The existence of the two other cluster can be understood considering the following. In case the selected feature points are being influenced by a high level of noise or if the feature set includes one or more outliers, the two eigenvectors $\mathbf{u_1}$ and $\mathbf{u_2}$ may change order. This problem has been earlier been reported by Ma et al [15]. In case of the eigenvalues changing orders, the iteration in Equation 13 will converge towards the wrong eigenvector, represented by the two other cluster centers in Figure 2.

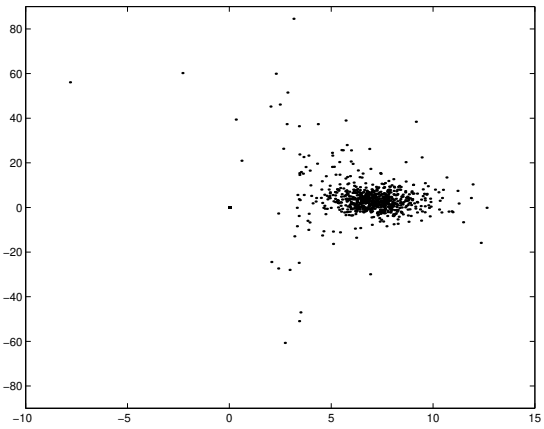

Fig. 3. Distribution of sample points, after filtering out samples that do not pass the two quality tests. The simulation was performed in the same manner as for the points in Figure 2.

In order to separate such false samples from real ones, we employ two additional quality tests before we decide whether to accept a sample point as a good estimate of the epipolar geometry or not. The first test ensures that the sum of

squared errors is reasonably small, that is the expression in Equation 11 has to be lower than a predefined threshold, which is usually set to a value in a neighbourhood of the variance of the corner detector. The other test involves a check whether every point in $M(j)$ is actually located in front of the cameras, when the three-dimensional positions have been reconstructed. Unlike what is to be expected, the left and right optical axes seldom coincide in a single 3D point. This is because there will always be some noise in the system and the calculations can not be done totally free from small truncation errors, but in many cases this fact can simply be ignored. From the derivations in Section II we know that the two projections can be described by $\mathbf{x_l} = \mathbf{R}_{\alpha_l}(\mathbf{I}|\mathbf{0})\mathbf{u}$ and $\mathbf{x_r} = \mathbf{R}_{\alpha_r}(\mathbf{I}|-\mathbf{t})\mathbf{u}$. Combining these two equations and dropping the y coordinates, we get the following equation:

$$\begin{pmatrix} x_r & -\cos(\beta)x_l - \sin(\beta) \\ 1 & \sin(\beta)x_l - \cos(\beta) \end{pmatrix} \begin{pmatrix} \omega_r \\ \omega_l \end{pmatrix} = \begin{pmatrix} \cos(\alpha_r) \\ \sin(\alpha_r) \end{pmatrix}.$$

In order for the in-front test to be successful, the distances from an object point to the two cameras, ω_r and ω_l, both have to be positive. After performing these two quality tests on the sample points in Figure 2 and omitting those points that do not pass the tests, we end up with epipolar estimates clustered around one center one, as shown in Figure 3.

D. Nonlinear optimization

The method described in the previous section relies on a minimum being found in a four-dimensional space spanned by the parameters f_j, $j = 1 \ldots 4$. However, what we are really looking for is the minimum of the objective function in a two-dimensional manifold in this space. We try to find a point on the manifold hopefully close enough to the real minimum, performing an approximation as described in Section II. A problem with this approach is the fact that we really do not know whether we really end up close to the correct minimum.

We let the two angles $\alpha = (\alpha_l, \alpha_r)$ define the coordinate system of the manifold, with the corresponding point in the four-dimensional space given by

$$\mathbf{f}(\alpha) = (f_1(\alpha), f_2(\alpha), f_3(\alpha), f_4(\alpha))^T = (-\sin(\alpha_l), -\sin(\alpha_r), \cos(\alpha_r), -\cos(\alpha_l))^T. \tag{14}$$

To find the minimum of the objective function

$$E(\alpha) = \frac{1}{2}\mathbf{f}(\alpha)^T\mathbf{M}\mathbf{f}(\alpha) \tag{15}$$

on the manifold, a possible solution is using Newton's method. For this iterative method the optimal updating direction is given by the vector $\mathbf{p_k}$ that satisfies

$$\nabla^2 E(\alpha^{(k)})\mathbf{p_k} = -\nabla E(\alpha^{(k)}). \tag{16}$$

As an initial estimate $\alpha^{(0)}$, we may use the point we get from the linear optimization as explained in Section III-A. Since the function $E(\alpha)$ typically is well-behaved around the minimum, the estimate can be updated without any additional backtracking operation. Hence the *(k+1)*th estimate of α is given by

$$\alpha^{(k+1)} = \alpha^{(k)} + \mathbf{p_k}. \tag{17}$$

The gradient and Hessian of the function to be minimized can be shown to be

$$\nabla E(\alpha) = (\mathbf{f^T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{l}}, \mathbf{f^T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{r}})^T \quad \text{and} \tag{18}$$

$$\nabla^2 E(\alpha) = \begin{pmatrix} \mathbf{f^T}\mathbf{M}\frac{\partial^2 \mathbf{f}}{\partial \alpha_\mathbf{l}^2} & 0 \\ 0 & \mathbf{f^T}\mathbf{M}\frac{\partial^2 \mathbf{f}}{\partial \alpha_\mathbf{r}^2} \end{pmatrix} + \begin{pmatrix} \frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{l}}^\mathbf{T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{l}} & \frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{l}}^\mathbf{T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{r}} \\ \frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{r}}^\mathbf{T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{l}} & \frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{r}}^\mathbf{T}\mathbf{M}\frac{\partial \mathbf{f}}{\partial \alpha_\mathbf{r}} \end{pmatrix}. \tag{19}$$

Since nonlinear optimization is relatively computationally costly, it is important for us to discard bad estimates as soon

as possible. In our system we test the sum of squared errors directly after performing the linear optimization. This turned out to be a quick way of filtering out most sets of image correspondences that include one or more outliers, before we continue with the nonlinear optimization. The convergence test of the Inverse Power procedure described earlier ensures that the function $E(\alpha)$ is well-behaved and few iterations are required for the nonlinear optimization. As seen in Figure 4 the spread of sample points is significantly reduced, especially in the translational component.

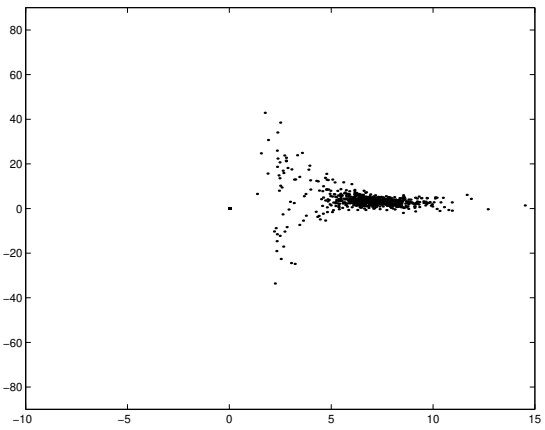

Fig. 4. Distribution of sample points, after nonlinear optimization and filtering out samples of low quality. The simulation was performed in the same manner as for the points in Figure 2.

IV. Experiments

To evaluate the alternative methods, a number of simulations are performed. Series of around 500 randomly generated points are spread in 3D around the fixation point, for a number of different camera configurations. The points are spread such that they evenly cover the 45° field of view. The closest points are located at about half the distance to the fixation point, whereas the maximum distance is around twice the same distance. Each point is projected onto the 360×288 pixel image planes of the left and right cameras. For each projection noise with a standard deviation of about a pixel is added, in order to simulate errors of the feature extraction. The negative effect of outliers in real data is tested, adding an additional 30 pixel noise source, to the left feature point position of about 20% of the 500 feature pairs.

The first approach to be tested is ordinary linear optimization (**linear**) as previously describe in Section III. Using random sampling a series of 10000 sets, $M(j)$, of six image correspondences each are being generated. For each set the corresponding essential matrix is calculated. The quality of the results are then tested as shown earlier and bad estimates are discarded. Typically only about 10% of the estimates are left after this procedure, which might seem as a low figure. However, based on the number of outliers in the data set, it can be expected that about $1-(1-0.20)^6 = 74\%$ of the sets include at least one outlier and their corresponding estimates are to be ignored. Additional estimates are thrown away simply because the feature points are not properly spread in 3D space.

What we finally end up with is a number of epipolar constraints, that are expected to be good estimates of the true epipolar constraint. As shown in Figure 3, these estimates can be seen as sample points in a two-dimensional space Ω_F, one coordinate representing the rotational component and another the translational one. The good estimates are spread around an ellipsoid in this space and we calculate the mean point position, which is an estimate the center of the ellipsoid, and use this as our final epipolar constraint. The size and shape of the ellipsoid vary depending on the feature

noise level and the camera configurations.

As explained earlier the linear optimization procedure does not necessarily lead to an optimum in the two-dimensional domain of possible epipolar constraints. To make sure that a local minimum really is found, one may add an extra nonlinear optimization operation (**nonlinear**). This approach is tested using the same randomly generated sets of points and evaluated towards the linear method.

Instead of just using the mean sample point position as the final estimate of the epipolar constraint, one may also test in what sense a certain estimate is supported by the complete set of image correspondence. In this approach (**support**), the epipolar estimate that has got the highest number of supporting feature pairs wins and will be used as the final estimate. We say the an estimate $\hat{\mathbf{E}}$ is supported by a pair of points $(\mathbf{x_l}, \mathbf{x_r})$, if $|\mathbf{x_l^T}\hat{\mathbf{E}}\mathbf{x_r}| < \sigma_E$. The constant σ_E is a predefined threshold covering the expected noise of the corner extractions.

A. Least Median of Squares

There are primarily two reasons for keeping the degrees of freedom of our binocular system as few as possible. The first reason is the speed and the other one is robustness. We test the robustness by comparing it to runs made with the full set of six degrees of freedom. The method used is a Least Median of Squares (**LMedS**) approach proposed by Zhang et al [25]. Random sampling is used as described above and for each set of image correspondences an estimate, $\mathbf{E_{LMedS}}$, of the epipolar geometry is being calculated and for every feature pair in the whole data set the squared error is obtained. The median of the squared errors is used as a quality measure for the set $M(j)$. The set with the lowest median will be considered as the winner and its corresponding epipolar estimate will be used as the final epipolar geometry estimate.

In order to be able to compare the different methods, the results of the **LMedS** approach has to be translated into two parameters only, one translational and one rotational parameter. First the essential matrix $\mathbf{E_{LMedS}}$ is decomposed into one rotational matrix and one skew-symmetric one. We then try to find the matrices of the forms of Equation 8 and 9, closest to those of the estimation process under the Frobenius norm. The corresponding two parameters are then used in the evaluations.

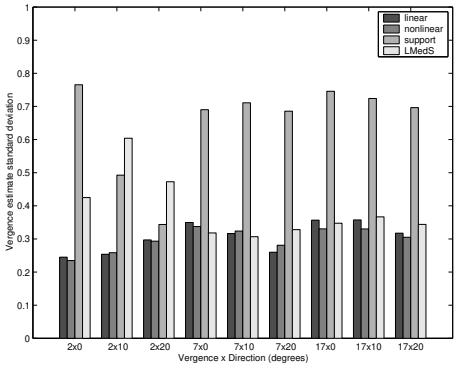

Fig. 5. Standard deviation of the rotational component, for different combinations of true rotational and translational components.

B. Experimental results

Simulations were performed using the four different methods mentioned above. The results of the rotational component can be seen in Figure 5 and 6, which show the standard deviation and mean of the estimated rotational parameter

respectively. Different groups of bars represent the results from different combinations of true rotational and translational components. Rotational components of 2°, 7° and 17° correspond to the minimum, typical and maximum vergence angles of a stereo-head system under normal working conditions. The thin horizontal lines in Figure 6 show the true rotational parameter values and can be used as references.

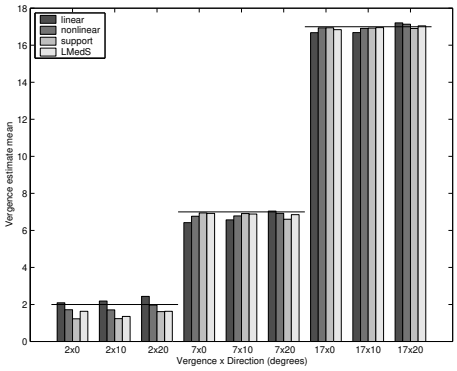

Fig. 6. Mean of the rotational component

We see that all methods come close to the true configuration, since the mean error never exceeds one degree and the standard deviations are reasonable small. In general the **support** method is not as robust as the other methods. It is also worth noting that, contrary to what the higher number of degrees of freedom might suggest, the **LMedS** method is just as robust as the first two methods, unless the vergence angle is too small. We may also note a slight improvement of the bias, when nonlinear optimization has been added to the ordinary linear one, but it is questionable if the relatively small improvements justify the increased complexity of the method.

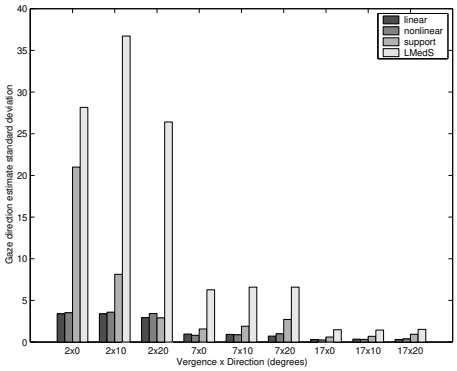

Fig. 7. Standard deviation of the translational component, for different combinations of true rotational and translational components.

Considering the graphs showing the mean and standard deviation of the translational component in Figure 7 and 8, we may conclude that this component is much harder to estimate. Here the methods based on the estimations of two parameters show a much better robustness, than the **LMedS** approach. Systems where the cameras are close to parallel are especially hard and even the best methods show a radically reduced robustness. However, a standard deviation of about three degrees, which is equivalent to the worst results of the **linear** and **nonlinear** methods, is probably acceptable in most applications. The method based on **support** does not come close to the robustness of the first to methods, even if the results are considerably more stable than that of **LMedS**.

In conclusion, the translational component of the camera configuration is far easier estimated using the **linear** and **nonlinear** methods, than the other methods. However, for the rotational component all methods show similar results and there is no real factor excluding one method from the others. In most cases, **nonlinear** is slightly less biased than **linear**, which is true especially for the translational parameter.

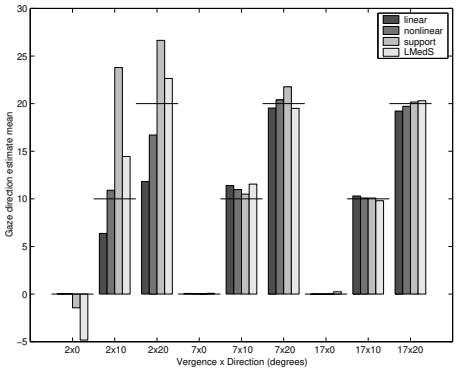

Fig. 8. Mean of the translational component

V. An iterative approach

In the previous section we tried to estimate the epipolar geometry of a stereo-head system using the essential matrix. As described in Section II, the epipolar constraint can be seen as decoupling the structure from the stereo problem. Without knowing anything about structure, the relative location and orientation of the cameras can be extracted. However, in many applications the depths are in fact approximately known and the question is whether this knowledge can my used to simplify the problem. We might not have an accurate depth value for each individual image point, but at least the distribution of depths may be given. This is often the case for autonomous robots that are typically equipped with range sensors such as lasers and sonars.

Instead of using the essential matrix, one may view the problem as a differential motion problem. The left and right images can be considered as two images taken by the same cameras at two different instances in time, while the camera is moving in 3D space. The vertical and horizontal disparities can be used as an approximation of the optical flow. It can be argued whether such a treatment of the problem is feasible, since such approximations are known to break down if the motion from frame to frame is too large and in our problem we are in fact working with stereo, rather than motion. However, in this paper it will be shown that for a stereo head in fixation, where objects in space are located relatively far away from the platform and the vergence angles are small, one will stay within the bounds of acceptable results.

Considering a stereo-head as shown in Section II and following the derivations by Longuet-Higgins and Prazdny [13], the relation between an image point (x_i, y_i) in the left and right image is given by:

$$\begin{pmatrix} dx_i \\ dy_i \end{pmatrix} = \begin{pmatrix} 1 + x_i^2 \\ x_i y_i \end{pmatrix} \beta + \frac{1}{Z_i} \begin{pmatrix} T_x - x_i T_z \\ -y_i T_z \end{pmatrix} \tag{20}$$

The values dx_i and dy_i represent the horizontal and vertical disparities respectively, that is the difference in image position between the left and right images. Unlike the essential matrix, the depth Z_i of each point appears in the equation. The translational component, the position of the right camera relative to the left, is given by T_x and T_z, while β represents a rotation around the y-axis, as seen in Figure 1.

One major advantage of the system we are using, is the fact that the horizontal disparities of different points in space viewed by the two cameras, always have the same ordering [24], as the cameras are verged. This means that if the distribution of depths is known in advance, we can easily find an approximate depth value for the individual image points. This will be shown in the algorithm below. A similar approach has been presented by Heel [10], even if his system was dealing with the full set of motion parameters and based on Kalman filters in order to track features in space.

Our approach is a two-stage iterative algorithm, that first calculates the geometry parameters β, T_x and T_z, using available estimates of the depths. In the second stage the depth values Z_i are updated using the new geometry parameters. As seen in Equation (20), the depths and translation are related and can only be estimated up to a constant. Letting $t = T_z/T_x$ and denoting the inverted and scaled depth $\lambda_i = T_x/Z_i$, we rewrite the equation into

$$\begin{pmatrix} dx_i - \lambda_i \\ dy_i \end{pmatrix} = \begin{pmatrix} 1 + x_i^2 & -x_i\lambda_i \\ x_i y_i & -y_i\lambda_i \end{pmatrix} \begin{pmatrix} \beta \\ t \end{pmatrix}. \tag{21}$$

A. The algorithm

Using Equation (21) we come up with an **iterative** algorithm that can be summarized as follows:

1. Initialize $\beta^{(0)}$ based on expected median distance to object points in space and distribution of horizontal disparities of extracted feature pairs, and let $t^{(0)} = 0$.
2. Update the inverted and scaled depths using the current estimated geometry.

$$\lambda_i^{(n)} = \frac{\mathbf{e_i^T b_i}}{\mathbf{e_i^T e_i}}, \quad \text{where} \tag{22}$$

$$\mathbf{b_i} = \begin{pmatrix} dx_i - (1 + x_i^2)\beta^{(n)} \\ dy_i - x_i y_i \beta^{(n)} \end{pmatrix} \quad \text{and} \quad \mathbf{e_i} = \begin{pmatrix} 1 - x_i t^{(n)} \\ -y_i t^{(n)} \end{pmatrix}. \tag{23}$$

3. Calculate the geometry parameters based on estimated depths, in a least square framework with all valid feature pairs included.

$$\begin{pmatrix} \beta^{(n)} \\ t^{(n)} \end{pmatrix} = (\sum_i \mathbf{A_i^T A_i})^{-1} \sum_i \mathbf{A_i^T d_i}, \quad \text{where} \tag{24}$$

$$\mathbf{A_i} = \begin{pmatrix} 1 + x_i^2 & -x_i\lambda_i^{(n-1)} \\ x_i y_i & -y_i\lambda_i^{(n-1)} \end{pmatrix} \quad \text{and} \quad \mathbf{d_i} = \begin{pmatrix} dx_i - \lambda_i^{(n-1)} \\ dy_i \end{pmatrix}. \tag{25}$$

4. Return to 2. until convergence is achieved.

B. Reducing the effect of outliers

As mentioned earlier in Section III, we always have to expect that a considerable fraction of the data set suffers from outliers, that is feature pairs that do not correspond to a single point in 3D space. Just as for the other methods described previously, outliers might seriously hurt the convergence of the algorithm and a couple of adjustments have to be made to improve the robustness.

The most obvious and dominating outliers can be identified by making sure that the expression

$$|dy_i| \leq |x_i y_i| \max(\beta) + |y_i| \max(T_z/Z_i) + \sigma_y \tag{26}$$

is satisfied for all the accepted feature pairs. The constants $\max(\beta)$, $\max(T_z/Z_i) = \min(Z_i/T_z)^{-1}$ and σ_y are predefined and represent the maximum possible vergence angle, the inverse of the minimum depth relative to the z-component of the translation and the expected noise of the feature extractor respectively. Image correspondences that do not pass this test will be omitted from the data set and not used in the iterations. The test can either be performed before step 1 or directly after. If the test is done afterwards we may take advantage of the fact that $\beta^{(0)}$ can be expected to be close to the real result and a more conservative limit than $\max(\beta)$ can be used, making it possible to identify also less critical outliers.

In order to further suppress the effect of outliers, we assign a weight w_i for each correspondence and use these weights in the least square optimization of step 3. We use the errors of step 2, $\varepsilon_i = \mathbf{d_i} - \lambda_i^{(n)}\mathbf{e_i}$, to define the weights. The weights reflect in what sense the horizontal and vertical disparities can agree on a single depth value. Letting σ_e be a predefined value covering the noise of the feature extractor, one might use the following definition of weights:

$$w_i = \left\{ \begin{array}{ll} 1 - (\frac{\varepsilon_i}{\sigma_e})^2, & \text{if } |\varepsilon_i| < \sigma_e \\ 0, & \text{otherwise} \end{array} \right\}. \tag{27}$$

In conclusion, after introducing the weights given above, we substitute Equation (24) with

$$\begin{pmatrix} \beta^{(n)} \\ t^{(n)} \end{pmatrix} = (\sum_i w_i \mathbf{A_i^T A_i})^{-1} \sum_i w_i \mathbf{A_i^T d_i} \tag{28}$$

and get a new estimate, where points that do not match the current estimate will be suppressed, reducing the effect of outliers.

It should be noted that the success of this procedure depends on the camera configuration being within bounds previously defined. In cases of unusual configurations or when the initial depths values are far away from the truth, correct feature correspondences may be eliminated by mistake and outliers falsely kept in the data set. The effect of this will be seen in the next section.

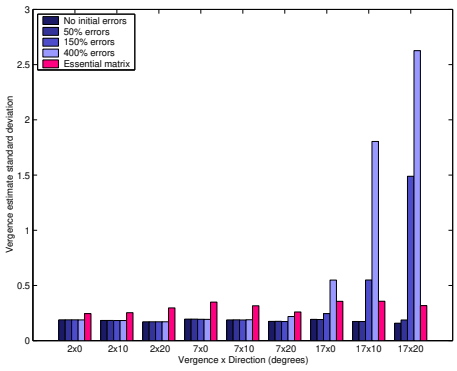

Fig. 9. Standard deviation of the rotational component, for different combinations of true rotational and translational components.

C. Simulations

Simulations were performed using the same set of random points as for the experiments described in Section IV. In order to test the influence of an erroneous initial median depth, the parameter s, which is the fraction between the

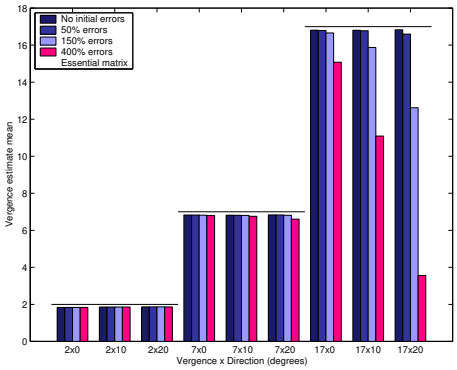

Fig. 10. Mean of the rotational component

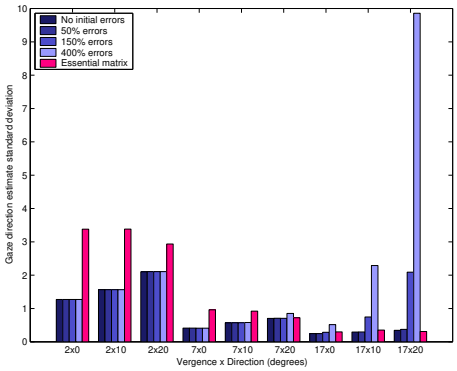

Fig. 11. Standard deviation of the translational component, for different combinations of true rotational and translational components.

guessed median depth and the true depth, is varied. The different sets of bars in Figure 9 and 10 represent estimated rotational parameters for different combinations of true rotational and translational components. As seen in Figure 9, the standard deviation of the rotational component is considerably lower than that of earlier presented methods, if the initial guess is close to the true median depth. However, when the initial guess is not as good, the algorithm collapses for combinations of large vergence angles and asymmetric camera settings.

The translational component, as seen in Figure 11 and 12, also seems to be more robustly estimated, than for methods based on the essential matrix. Even if small vergence angles are still critical, the standard deviations stay within about 2°, which ought to be acceptable in most cases. Unlike the earlier presented methods, the robustness is slightly dependent on the true translational component.

The iterations converge at a rate remarkably independent from the camera configuration. Typically 20 to 30 iterations are needed for the relative errors of the depth values to decrease to 10% of the original errors. The rate is slightly higher for near symmetric camera configurations and small vergence angles. In situations where outliers have not be properly eliminated the convergence rate can be much worse, even if the procedure never seems to diverge. Based on these observations we decided to use forward prediction to improve the speed of convergence. However, data presented in this paper are not based on such predictions.

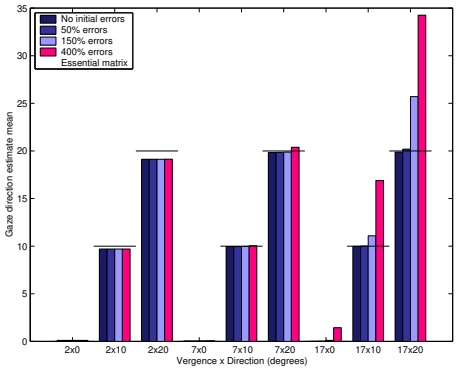

Fig. 12. Mean of the translational component

VI. System integration

As was told in the beginning, the intention is to use the geometry estimation process in order to facilitate dynamic vergence based behaviours of a robotic system. The only viable way of testing such a process is putting it into a context, where it is supposed to be used. For that reason a complete binocular system was created with the geometry estimation process as one of its fundamental bases. Using information about the stereo-head geometry, the system creates a dense disparity map as will be described below in Section VI-A. Such a map can then be used for behaviours, such as visual path planning and avoidance. The system includes the following six components:

1. Corner feature extraction
2. Matching of corner features
3. Estimation of geometry
4. Rectification of images
5. Calculation of dense disparity map
6. Update of occupancy grid

As mentioned in Section III corner features are extracted using a Harris corner detector [8]. In order to suppress noise the images are low-pass filtered before features are extracted. Unlike some extractors, such as SUSAN [21], Harris detector suffers from the fact that the initial filtering procedure may affect the positioning of the corner points and move them in the direction of the center of curvature. However, in this study the Harris detector has been used anyway, since the operations of the detector are more easily parallelized, using hardware such as the MMX capabilities of recent Intel and AMD processors. In fact, even if the extracted corner positions are erroneous, the relative positions between the two images are of greater importance and these will not change as much.

In order to avoid a combinatorial explosion when matching the corner features, care has been taking to minimize the number of matching operations. Based on earlier estimates of depths and by constraining the possible camera configurations, the regions where to look for possible matches are kept as small as possible. A more detailed description of the actual matching operation was described above in Section III.

In a complete system it might be hard to decide which geometry estimation algorithm to use. The essential matrix is preferable if one does not know anything about the depths of feature points and if the cameras are not close to parallel. The increased complexity of nonlinear optimization is hardly justified in most situations. The speed of the iterative approach is attractive, but it relies on the median depths of feature points in space being approximately known.

However, for a mobile robot such as the one used in this study, sonar sensors can be be used in give a rough estimate of such a median depth. In fact, just knowing the dimensions of the environment in which the system is supposed to work, turned out to be enough in our study. Consequently, in our complete system we decided to use a combination of the iterative approach and the linear method based on essential matrices. As long as the vergence angles stay within about $10°$, the convergence rate of the **iterative** approach will be acceptable and the system keeps using this the results from this method. The calculated median depth from one update is used as an input in the next update. Upon failure the system switches to the **linear** approach and stays there until the iterative approach can be resumed.

Rectification of images is necessary, since most disparity algorithms rely on the epipolar lines being located parallel to the scan-lines, in order to be run in real-time. The complexity of the algorithms can then be greatly reduced and a parallel implementation is easier to achieve. The projective transformation of the rectification process involves a bilinear filtering of the images. Since such a filtering might influence the correlations done when calculating the disparities, it is preferable to use the same kind of filtering for both images. If the vergence angles are relatively small, it ought to be possible only to transform one of the two images, reducing the computational cost of the rectification. However, we noticed in our study that such an approach considerably hurts the performance of the disparity calculations. The best results where achieved if both images where transformed, using approximately the same rotation around the y-axes of the images. A benefit of the geometry estimation process is that one gets a good idea of the range of available disparities. During the rectification images can be shifted, so that the range of disparities matches that of the disparity calculation algorithm.

A. Binocular disparity

In order to calculate disparities we use a method based on correlations of grey-level image pixels. This approach turned out to be more flexible and robust than other more sophisticated methods that was tested during our study. Around each pixel a small window, with a typical size of 7×7 pixels, is defined. Matches are searched in the other image by scanning the epipolar lines, correlating these windows of pixel intensity values. The location in the other image that corresponds to the maximum correlation is used to calculate the estimated disparity for that particular pixel. Sub-pixel accuracy can be achieved by fitting the correlation values around the maximum of a second-degree polynomial.

Numerous groups have been using correlation, but the implementations look slightly different [12], [11], [25]. The most commonly used correlation-criteria is probably sum of absolute differences and the reason for this is often its speed. However, since we are using two different cameras we have to make sure that they produce images of the same contrast and average light intensity. Slight variations of these parameters make sum of absolute difference more of less useless. If accuracy is more important than speed, one better uses some kind of normalized cross-correlations instead, but this at the cost of a considerable slower system.

One of the major benefits of using correlation is the fact that you are able to identify most occlusions. This is done by performing the matching in both directions, from the left image to the right and then from the right image to the left. The calculated disparities in both directions have to agree and in case they do not agree one possibly has an occlusion. Another benefit is the fact that regions where the estimated disparity is of low confidence can easily be identified, using the variance of calculated correlation values. When the variance of low, the amount of structure in the region is limited and the confidence of the calculated disparity is lower.

In the system presented here, dynamic programming is used in a probabilistic framework to further improve the

results of the calculated disparities [19], [4]. A maximum likelihood cost function is used to find the most probable combination of disparities along a scan-line. This cost function is able to take the ordering and uniqueness constraints into consideration. The complexity of finding a unique maximum is greatly reduced by dynamic programming. The strength of the method is the ability to cover areas of limited texturing creating very dense disparity maps. Thus a combination of correlation-based stereo and dynamic programming, makes it possible to achieve both sub-pixel accuracy and high density. For areas of low texturing dynamic programming efficiently increases the smoothness of the resulting disparities, while regions of high texturing still includes results of sub-pixel accuracy.

Since speed is one of great concern, we wanted to keep the range of possible disparities low. This was done using a 3-level pyramid, with calculations starting from the coarsest scale and moving to finer scales. Each level only covers disparities of up to 16 pixels. The data used as the final result at a particular location come from the finest possible scale. Consequently, the smaller the disparities, the finer scale is being used and the more accurate one can expect the estimates to be. Since disparities of 16 pixels in the coarsest scale is equivalent to 64 pixels in the finest scale, we can get a good balance between disparity coverage and speed.

B. Application

As mentioned in the beginning, the final aim of this study is to apply the system to a real mobile platform. Nice-looking disparity maps do not really prove anything, unless one is able to show that they actually can be used in a real application. In our case we wish the stereo system to be used for obstacle avoidance and path planning, and thus we need a way of interpreting the disparity maps and come up with proper commands to control the movements of the robot. It is important to consider the fact the disparity maps might include a considerable amount of errors. Due to specularities, occlusions and mismatches from repetitive patterns in the images, one easily gets small patches of erroneous disparities, that could be interpreted as objects located close to the cameras.

It may also be hard to get a good estimate of distances to objects located rather close to the observer, even if the cameras are properly verged. Since the cameras are located on top of the robot, distances from objects to cameras might not vary very much, compared to the distances parallel to the ground level. An optimal positioning of the stereo head system, in order to suppress this effect of Pythagoras, is of course close to the floor, but that is usually not practically achievable. In order for the platform to move around in the environment, a short-term memory becomes a necessity.

We use a grid-based local representation of the nearby environment as our short-term memory. Such a representation, often referred to as an occupancy grid [18], [16], consists of a 2D array of cells, where each cell includes a value, presenting the probability that the cell is occupied. In the beginning all these probabilities are set to 0.5, which represents the state *unknown*. The robot is always located in the center of the grid, which means that as the platform moves, the cells are shifted in the array correspondingly.

The disparity map is divided into a number of sectors and for each sector the shortest distance to a possible obstacle is calculated by analyzing its disparity values. In order to suppress the effect of possible erroneous depth values, a histogram is calculated for each sector. The distance equivalent to the 5% closest depth value of such a histogram is used to represent the nearest obstacle of the corresponding sector. For each sector the occupancy grid is then updated drawing an arc of probability values of 1.0 (*occupied*). All cells located in the sector between the center of the platform and the possible obstacle are decreased, with a lowest value of 0.0 (*empty*).

What showed to be a serious limitation using stereo-based vision only, is that it is hard to tell anything about the

Fig. 13. Occupancy grid

surroundings, unless there is enough of structure visible by the cameras. In our study we noticed that this may seriously constrain the movability of the platform. In fact, it is easy to get stucked in areas that you do not dare to leave, because you are not sure there is a safe passage out of it. This has led us to the conclusion that visual depth information is hardly enough and we have to rely on other sensors as well. For example, using sonars as complementing sensor may be advantageous, since sonars have a high accuracy in depth, whereas stereo vision has a relatively high angular resolution.

Figure 13 show an occupancy grid from a typical run in our lab. The *empty* space is black in the image, while occlusions are drawn in lighter shades of grey. To the left of the image one clearly sees a number of grey spikes, that represent an area lacking enough of structure. The noisy top-left and bottom-right areas, include objects located close to the floor. As the robot approaches these areas, the objects disappear below the field-of-view of the cameras. A serious amount of noise could also be seen from areas that include a high degree of specularities.

C. Implementation

The system was implemented on both a SGI Octane based on a two 195 MHz MIPS R10K processors and on a 400 MHz Pentium III powered Nomad200 platform. Everything was programmed in standard C++, which meant that the code was relatively easily ported from one system to the other. Both systems were running at approximately 6 Hz, but the distributions of computational costs were different. The speed of floating point operations of the R10K processor made the epipolar geometry code about 50% quicker on the Octane, than on the Pentium.

In order to maximize the speed of the second implementation, the MMX capabilities were used for the Pentium implementation to extract corner features as well as calculating the dense disparity map. The feature extractor requires about 15 ms for each 360×288 pixel image and the disparity routine has a computational cost of about 20 ms for half the resolution. The dynamic programming routine was not used in the Pentium implementation for two different reasons. First of all, the occupancy grid is better created using data of high confidence and areas that are only covered when using the dynamic programming extension can not be considered confident enough. Secondly, the complexity of dynamic programming makes it difficult to implement using MMX.

On the SGI machine both processors were used, one processors for feature extraction, matching and epipolar geometry estimation, while the other processor was taking care of the remaining operations. Due to an almost perfect balancing between the loads of the processors, the speed could be improved, even if none of the available hardware acceleration

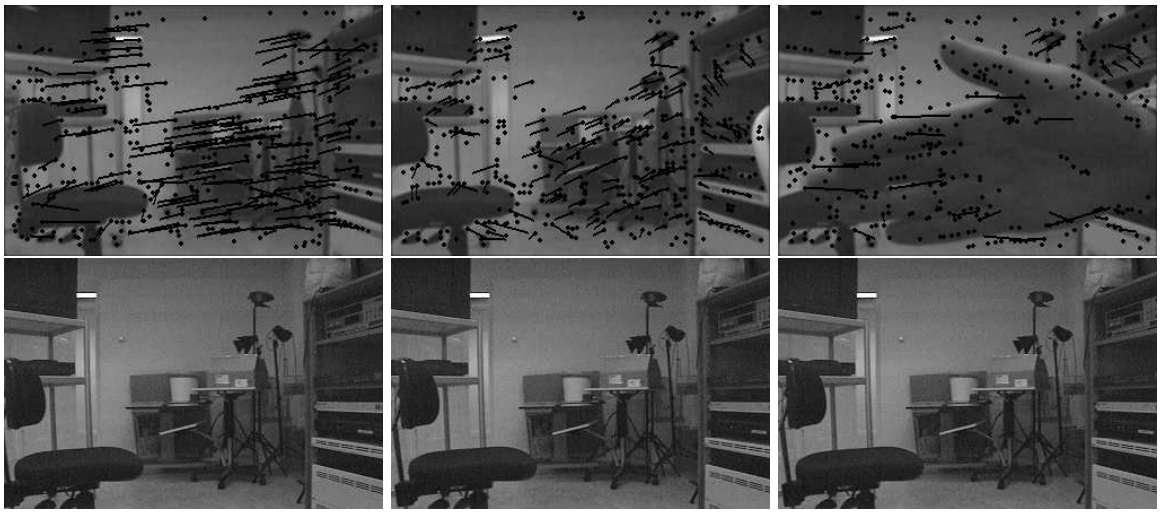

Fig. 14. On the upper row the images from the left camera are shown, whereas the lower row are the corresponding right images. The three columns represent image pairs from the 50th, 100th and 150th frame. The black points represent the extracted corner features, while the lines show valid matches between the left and right images.

capabilities of the SGI system were used for the implementation. Since the movements of the platform have to be known for the occupancy grid to make sense, the SGI implementation was done without occupancy grids.

VII. Real-time experiments

We performed a series of experiments, out of which one will be presented here in greater detail. During an interval of about 30 seconds the complete system was running, including the epipolar estimation process as well as rectification and disparity map calculation. In this case the position of the stereo head was kept static, but with the ability for the cameras to verge. It should be pointed out that the data was not processed off-line, but in real-time on the SGI Octane machine using the implementation described earlier.

A small number of images from the sequence are presented in Figure 19, the upper row showing images from the left camera and the bottom row come from the right camera. The three columns constitute image pairs from the 50th, 100th and 150th frame. Between the first and second columns the cameras have been slightly verged, from about 8° to around 11.5°. In order to test the ability of the system to reassume convergence after a failure, we let someone block the left camera with a hand for a few seconds. This can be seen in the images from the 150th frame. For the left images the locations of extracted corner features, as well as lines corresponding to valid matches, can be seen. The system was able to capture the relative tilt between the cameras, using the method presented in Section II-B and be able to adjust the coordinates correspondingly. An estimate of about 5.2 pixel rows was found in a few frames and kept during the whole sequence.

Figure 15 and 16 show the estimated vergence angles and their standard deviations. The values are processed through a moving average filter, with a time window of about 20 frames. The first frame in the figures is the very moment when the system starts running. Through the experiments performed in our lab, the actual bootstrap process never seems to fail, if the distribution of features have otherwise been good enough. However, if the lighting-conditions radically change or the cameras get partially occluded as in the 150th frame of the sequence shown here, one might get serious errors in the geometry estimates, primarily due to a dramatic increase in the number of outliers. In order to minimize the effect

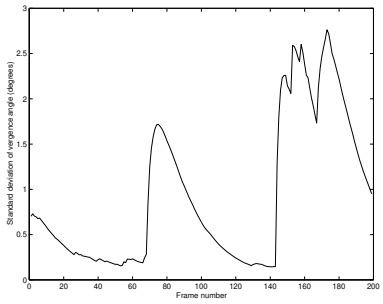

Fig. 15. Standard deviation of vergence angle

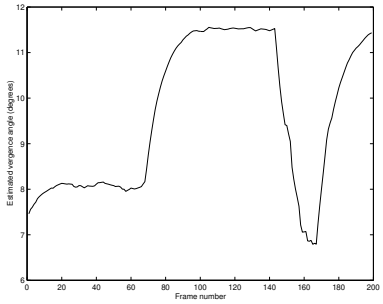

Fig. 16. Estimated vergence angle

of these problems, a gate was introduced just before the moving average filter. This gate simply discards inputs to the filter, if the estimated values are too far away from the range of values that are considered to be probable. This range is based on the physical limitations of the stereo-head and is not changed during execution.

In Figure 17 and 18 the corresponding graphs for the cyclopean gaze direction are shown. One may show that this direction is the same as the translation component in angles plus half the rotational one. The standard deviation stabilizes at about 1°, which is in the neighbourhood of the errors of the translational component shown in Figure 11. The same thing is true for the vergence angles, where the standard deviation reaches about 0.2° upon convergence.

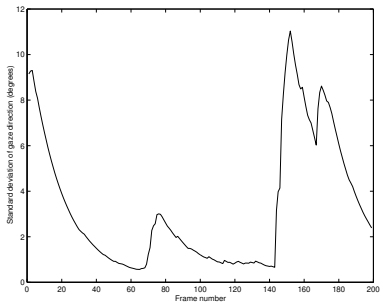

Fig. 17. Standard deviation of gaze direction

Finally, in Figure 19 the resulting disparity maps can be seen for a number of selected frames. The black parts represent areas, where no confident data could be calculated, either because they are too close to the border of the images or simply because there is hardly any structure in the images. The remaining shades are based on the range of available disparities for each individual disparity map. The lighter shades always mean areas located closer to the cameras, but what it means in metric values is different for each image. A small median filter has also been used on the data, which is able to eliminate some erroneous results, such as the typical streaking effect of dynamic programming

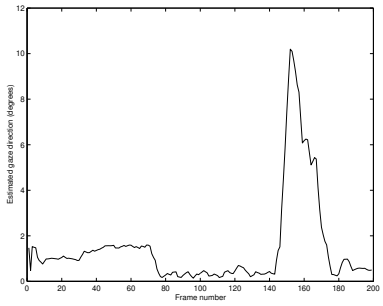

Fig. 18. Estimated gaze direction

based stereo routines [4].

As seen in the figure the disparity maps are relatively stable, when the cameras are not currently changing vergence angle and both cameras are free from disturbances, such as the hand blocking the view of the left camera in the 150th frame. When the cameras are verging, as in the 100th frame here, it usually takes about 15 frames or so for the data to converge and during that time the disparity maps can not be trusted. The convergence speed is usually not quick enough to keep up with the speed in which cameras are verged. This is primarily because data is low-passed in time to reduce the effect of temporary bad results, while the cameras move relatively easily.

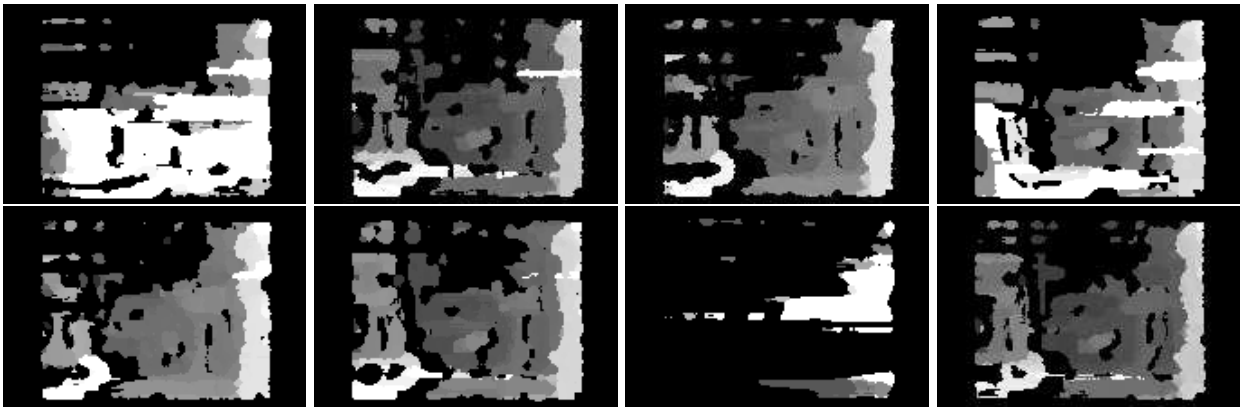

Fig. 19. These images show the resulting disparity maps of the 10th, 25th, 50th, 70th, 90th, 125th, 170th and 195th frames. The raw disparity data has been processed by a following median filtering operation in order to suppress sparse erroneous data.

A. Performance

The table in Figure 20 show the execution times of the methods presented above, performed using a 195 MHz MIPS R10K processor. Time required for feature extraction has not been included here. The data clearly shows that the methods that do not rely on any support calculations run considerably faster, than if a support measure has to be obtained for each and every sample set. It is worth noting that **LMedS** is not much slower than **support**, even if the number of degrees of freedom is much higher. This indicates that the actual support calculations, which are similar in both methods, constitutes most of the computational cost.

As seen in the table, the **linear** and **nonlinear** are far faster than the methods based on support. However, it is hard tell which one of the two methods to prefer. In applications where the bias of the **linear** method is not critical, one

Method	Cost on R10K
Linear	58 ms (26 ms)
Nonlinear	81 ms (36 ms)
Support	480 ms
LMedS	850 ms
Iterative	27 ms (12 ms)

Fig. 20. Computational costs of the different methods performed on a 195MHz MIPS R10K processor.

might benefit from a slightly lower computational cost. The speed of the two methods may also be somewhat improved, by setting a limit on the number of feature pairs one wishes to accept. Instead of generating as many as 10000 feature pairs, one might be satisfied when a certain number of accepted pairs have been found (we use 200) and then end the random sampling procedure. The data in parentheses show the execution times when such a limit has been set. It should be noted that the accuracy of such an approach is not notably worse. The disadvantage is the fact that the computational cost will be less deterministic, but our simulations show that as long as the vergence angles exceeds about 3°, the execution times will not vary very much.

The **iterative** approach turned out to be considerably faster, than all the other methods. This is basically because of the fact that we already have a clue about the distance to the objects we are observing and use this knowledge to improve the estimation process. In cases where the external calibration is performed continuously and the camera has not moved too much since the last update, one might benefit from using the previous camera parameters, when performing a new estimation. Instead of using as many as 25 iterations, the method has often converged after a much lower number of iterations. Using the difference between two consecutive iterations, we end the iterative procedure if convergence has been met earlier. At the cost of a less deterministic computational cost, we get the average execution time as shown in parentheses.

VIII. Discussion

For autonomous robots stereo is an important cue for a variety of visually guided behaviours, such as navigation, obstacle avoidance and manipulation. However, different behaviours have different requirements on working ranges and because of the limitations in field of view and image resolution, dynamic vergence often becomes a necessity. Dynamic fixation may also be used to simplify problems like ego-motion estimation and figure-ground segmentation.

In this paper a real-time framework for epipolar geometry estimation has been presented. Instead of using counters on the motors of the stereo-head, the external calibration is performed continuously at a low computational cost. Two models have been explored, with the number of unknown parameters minimized without constraining the system. The first model based on the essential matrix showed to be somewhat sensitive to near parallel camera systems, but for vergence angles larger than about 5° the errors stay within about 1° in both translational and rotational component.

An alternative approach, based on a differential formulation of optical flow, was also presented and tested towards methods based on the essential matrix. Using an initial estimate of the median distance to objects in 3D space, the geometry parameters and point depths were sought in an iterative fashion. Since the model only works for limited rotations and translations, it could be expected not to be sufficient for modeling the disparities of a stereo system. However, this study showed that it could in fact be used for stereo-head under normal working conditions. The method

was faster and also more robust, especially for small vergence angles, than methods based on the essential matrix.

The binocular stereo system in this paper will be implemented on the various mobile platforms of our lab, and used for behaviours, such as obstacle avoidance, navigation and visually based manipulation. A number of stereo based processes has already been added to the current system. Ego-motion estimation has been greatly simplified using the available stereo data. The system also includes a process for identification of independently moving objects. Through forward prediction of an image from one instance in time to another, using estimated ego-motion and disparities, regions of independent motion can be identified as areas where the predicted image does not match the true new image.

References

[1] P.K. Allen, B. Yoshimi and A. Timcenko, Real-Time Visual Servoing, In *Proc. of the IEEE Int'l Conf. on Robotics and Automation*, pp. 851-856, 1991.

[2] D.H. Ballard, Animate Vision, *Artificial Intelligence*, 48, pp. 57-86, 1991.

[3] M.J. Brooks, L. de Agapito, D.Q. Huynh and L. Baumela, Direct Methods for Self-Calibration of a Moving Stereo Head, In *Proc. of the 4th European Conf. on Computer Vision*, pp. 415-426, 1996.

[4] I.J. Cox, S.L. Hingorani, S.B. Rao and B.M. Maggs, A Maximum Likelihood Stereo Algorithm, *Computer Vision and Image Understanding*, 63(3), pp. 542-567, 1996.

[5] K. Daniilidis, Fixation Simplifies 3D Motion Estimation, *Computer Vision and Image Understanding*, 68(2), pp. 158-169, November 1997.

[6] C. Fermüller and Y. Aloimonos, The Role of Fixation in Visual Motion Analysis, *Int'l J. of Computer Vision*, 11(2), pp. 165-186, 1993.

[7] M.A. Fischler and R.C. Bolles, Random Sampling Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, *DARPA Image Understanding Workshop*, pp. 71-88, 1980.

[8] C. Harris and M. Stephens, A Combined Corner and Edge Detector, In *Proc. of the 4th Alvey Conf.*, pp. 189-192, 1988.

[9] R.I. Hartley, Estimation of Relative Camera Positions for Uncalibrated Cameras, In *Proc. of the 2nd European Conf. on Computer Vision*, pp. 579-587, 1992.

[10] J. Heel, Dynamic Motion Vision, *DARPA Image Understanding Workshop*, pp. 702-713, 1989.

[11] K. Konolige, Small Vision Systems: Hardware and Implementation, *8th Int'l Symposium on Robotics Research*, Hayama, Japan, 1997.

[12] T. Kanade, M. Okutomi and N. Nakahara, A Multiple-Baseline Stereo Method, *DARPA Image Understanding Workshop*, pp. 409-426, 1992.

[13] H.C. Longuet-Higgins and K. Prazdny, The Interpretation of a Moving Retinal Image, *Proc. of Royal Society of London*, B-208, pp. 385-397, 1980.

[14] H. Longuet-Higgins, A Computer Algorithm for Reconstructing a Scene From Two Projections, *Nature*, 293, pp. 133-135, 1981.

[15] Y. Ma, J. Kosecka and S. Sastry, Optimization Criteria and Geometric Algorithms for Motion and Structure Estimation, UC Berkeley Memorandom No. UCB/ERL M99/33, June 1999.

[16] L. Matthies and A. Elfes, Integration of Sonar and Stereo Range Data Using a Grid-Based Representation, *IEEE Int'l Conference on Robotics and Automation*, pp. 727-733, 1988.

[17] H. Moravec, The Stanford Cart and the CMU Rover, In *Proc. of the IEEE*, 71(7), pp. 872-884, 1983.

[18] H. Moravec and A. Elfes, High Resolution Maps from Wide Angle Sonar, *IEEE Int'l Conference on Robotics and Automation*, pp. 116-121, 1985.

[19] Y. Ohta and T. Kanade, Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(2), pp. 139-154, 1985.

[20] J. Oliensis and V. Govindu, An Experimental Study of Projective Structure From Motion, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(7), pp. 665-671, 1999.

[21] S.M. Smith and J.M. Brady, SUSAN - a New Approach to Low Level Image Processing, *Int'l J. of Computer Vision*, 23(1), pp. 45-78, May 1997.

[22] P.H.S. Torr and D.W. Murray, The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix, *Int'l J. of Computer Vision*, 24(3), pp. 271-300, 1997.

[23] V. Tucakov, M. Sahota, D. Murray, A. Mackworth, J. Little, S. Kingdon, C. Jennings and R. Barman, Spinoza: A Stereoscopic Visually Guided Mobile Robot, In *Proc. the 13th Annual Hawaii Int'l Conf. on System Sciences*, pp. 188-197, January 1997.

[24] W.Y. Yau and H Wang, Fast Relative Depth Computation for an Active Stereo Vision System, *Real-Time Imaging*, 5, pp. 189-202, 1999.

[25] Z. Zhang, R. Deriche, O. Faugeras and Q.T. Luong, A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry, *Artificial Intelligence*, 78, pp. 87-119, October 1995.