

A Spherical Eye from Multiple Cameras (Makes Better Models of the World)

Patrick Baker, Cornelia Fermüller, and Yiannis Aloimonos
Computer Vision Laboratory
University of Maryland
College Park, MD 20742-3275, USA
{pbaker, fer, yiannis}@cfar.umd.edu

Robert Pless
Department of Computer Science
Washington University
St. Louis, MO, 63130
pless@cs.wustl.edu

Abstract

This paper describes an imaging system that has been designed specifically for the purpose of recovering egomotion and structure from video. The system consists of six cameras in a network arranged so that they sample different parts of the visual sphere. This geometric configuration has provable advantages compared to small field of view cameras for the estimation of the system's own motion and consequently the estimation of shape models from the individual cameras. The reason is that inherent ambiguities of confusion between translation and rotation disappear. We provide algorithms for the calibration of the system and the 3D motion estimation. The calibration is based on a new geometric constraint that relates the images of lines parallel in space to the rotation between the cameras. The 3D motion estimation uses a constraint relating structure directly to image gradients.

1. Introduction

Technological advances make it possible to mount video cameras in some configuration, connect them with a high-speed network and collect synchronized video. Such developments open new avenues in many areas, making it possible to address, for the first time, a variety of applications in surveillance and monitoring, graphics and visualization, robotics and augmented reality. But as the need for applications grows, there does not yet exist a clear idea on how to put together many cameras for solving specific problems. That is, the mathematics of multiple-view vision is not yet understood in a way that relates the configuration of the camera network to the task under consideration. Existing approaches treat almost all problems as multiple stereo problems, thus missing important information hidden in the multiple videos. The goal of this paper is to provide the first steps in filling the gap described above. We consider a multi-

camera network as a new eye. We studied and built one such eye, consisting of cameras which sample parts of the visual sphere, for the purpose of reconstructing models of space. The motivation for this eye stems from a theoretical study analyzing the influence of the field of view on the accuracy of motion estimation [5] and thus in turn shape reconstruction. The exposition continues by first describing the problems of developing models of shape using a common video camera and pointing out inherent difficulties.

In general, when a scene is viewed from two positions, there are two concepts of interest:

- (a) The 3D transformation relating the two viewpoints. This is a rigid motion transformation, consisting of a translation and a rotation (six degrees of freedom). When the viewpoints are close together, this transformation is modeled by the 3D translational and angular velocity of the eye (or camera).
- (b) The 2D transformation relating the pixels in the two images, i.e., a transformation that given a point in the first image maps it onto its corresponding one in the second image (that is, these two points are the projections of the same scene point). When the viewpoints are close together, this transformation amounts to a vector field denoting the velocity of each pixel, called an image motion field.

Perfect knowledge of both transformations described above leads to perfect knowledge of models of space, since knowing exactly how the two viewpoints and the images are related provides the exact position of each scene point in space. Thus, a key to the basic problem of building models of space is the recovery of these two transformations. Any difficulty in building such models can be traced to the difficulty of estimating these two transformations. This paper is concerned with removing the difficulties of task (a). What are the limitations in achieving this task?

2. Inherent limitations

Images, for a standard pinhole camera, are formed by central projection on a plane (Figure 1). The focal length is f and the coordinate system $OXYZ$ is attached to the camera, with Z being the optical axis, perpendicular to the image plane.

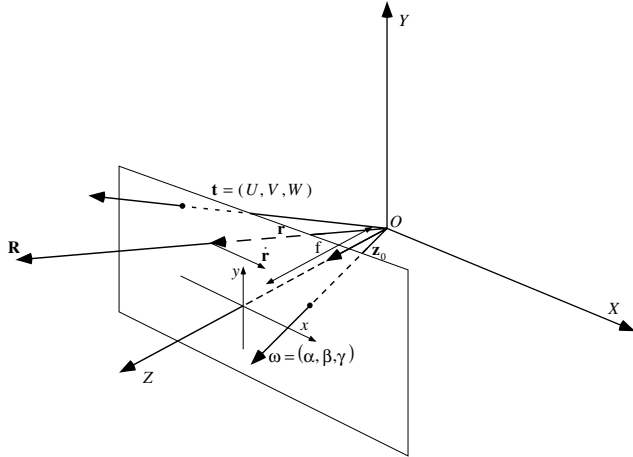


Figure 1. Image formation on the plane. The system moves with a rigid motion with translational velocity \mathbf{t} and rotational velocity $\boldsymbol{\omega}$. Scene points \mathbf{R} project onto image points \mathbf{r} and the 3D velocity $\dot{\mathbf{R}}$ of a scene point is observed in the image as image velocity $\dot{\mathbf{r}}$.

Scene points \mathbf{R} are projected onto image points \mathbf{r} . Let the camera move in a static environment with instantaneous translation \mathbf{t} and instantaneous rotation $\boldsymbol{\omega}$. The image motion field is described by the following equation:

$$\dot{\mathbf{r}} = -\frac{1}{(\mathbf{R} \cdot \hat{\mathbf{z}})} (\hat{\mathbf{z}} \times (\mathbf{t} \times \mathbf{r})) + \frac{1}{f} \hat{\mathbf{z}} \times (\mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r}))$$

where $\hat{\mathbf{z}}$ is a unit vector in the direction of the Z axis.

There exists a veritable cornucopia of techniques for finding 3D motion from a video sequence. Most techniques are based on minimizing the deviation from the epipolar constraint which in the continuous case takes the form $(\mathbf{t} \times \mathbf{r}) \cdot (\dot{\mathbf{r}} + \boldsymbol{\omega} \times \mathbf{r}) = 0$ [3] and in the discrete case is written as $x_2^T (\mathbf{t} \times R \mathbf{x}_1) = 0$, where \mathbf{x}_1 and \mathbf{x}_2 are corresponding image points and R is the rotation matrix. Other constraints that are used are the minimization of negative depth or the recently introduced minimization of the depth variability constraint [1]. Both these constraints relate image derivatives directly to the 3D motion, in contrast to the epipolar constraint which requires the intermediate computation of optical flow or the estimation of correspondences.

Whatever the constraint, mathematically one sets up a function which expresses deviation from the chosen constraint and which is expressed in terms of the 3D motion parameters, the image coordinates and their derivatives. Motion estimation amounts to finding the translation $\hat{\mathbf{t}}$ and rotation $\hat{\boldsymbol{\omega}}$ which minimize the chosen function.

Solving accurately for 3D motion parameters turned out to be a very difficult problem. The main reason for this has to do with the apparent confusion between translation and rotation in the image displacements. This is easy to understand at an intuitive level. If we look straight ahead at a shallow scene, whether we rotate around our vertical axis or translate parallel to the scene, the motion field or the correspondence at the center of the image are very similar in the two cases. Thus, for example, translation along the x axis is confused with rotation around the y axis. This really is a geometric problem and it exists in the cases of small and large baselines between the views, that is for the case of continuous motion as well as for the case of discrete displacements of the cameras. The basic understanding of this confusion has attracted few investigators over the years [2, 3, 6, 9].

Our work is motivated by some recent results analyzing this confusion. In previous work [5] we have conducted a geometrical analysis of the problem. We formulated the expected value of the different functions used in motion estimation as a five-dimensional function of the motion parameters (two dimensions for $\mathbf{t}/|\mathbf{t}|$ and three for $\boldsymbol{\omega}$). Independent of specific estimators the topographic structure of the surfaces defined by these functions explains the behavior of 3D-motion estimation. Intuitively speaking, it turns out that the minima of these functions lie in a valley. This is a cause for inherent instability because, in a real situation, any point on that valley or flat area could serve as the minimum, thus introducing errors in the computation (see Figure 2a).

In particular, the result obtained are as follows: Denote the five unknown motion parameters as (x_0, y_0) (direction of translation) and (α, β, γ) (rotation). Then, if the camera has a limited field of view, *no matter how 3D motion is estimated from the motion field*, the expected solution will contain errors $(x_{0\epsilon}, y_{0\epsilon})$, $(\alpha_\epsilon, \beta_\epsilon, \gamma_\epsilon)$ that satisfy two constraints:

- (a) The orthogonality constraint: $\frac{x_{0\epsilon}}{y_{0\epsilon}} = -\frac{\beta_\epsilon}{\alpha_\epsilon}$
- (b) The line constraint: $\frac{x_0}{y_0} = \frac{x_{0\epsilon}}{y_{0\epsilon}}$

In addition, $\gamma_\epsilon = 0$. The result states that the solution contains errors that are mingled and create a confusion between rotation and translation that cannot be cleared up, with the exception of the rotation around the optical axis (γ). The errors may be small or large, but their expected value will always satisfy the above conditions. Although the 3D-motion estimation approaches described above may provide

answers that could be sufficient for various navigation tasks, they cannot be used for deriving object models because the depth Z that is computed will be distorted.

The proofs described are of a statistical nature. Nevertheless, we found experimentally that there were valleys in the function minimized for any indoor or outdoor sequence we worked on. Often we found the valley to be rather wide, but in many cases it was close in position to the predicted one.

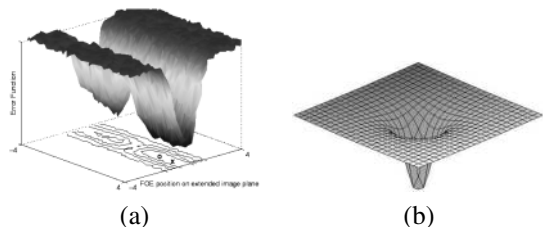


Figure 2. Schematic illustration of error function in the space of the direction of translation. (a) A valley for a planar surface with limited field of view. (b) Clearly defined minimum for a spherical field of view.

The error function, however, changes as the field of view changes. The remarkable discovery in our theoretical study is that when the field of view becomes 360° the ambiguity disappears. This means that there are no more valleys, but a well defined minimum, as shown in Figure 2b. This constitutes the basis of our approach.

Our interest is to develop techniques that, given video data, yield models of the shape of the imaged scene. Since conventional video cameras have an inherent problem, we should perhaps utilize different eyes. If, for example, we had a sensor with a 360° field of view, we should be able to accurately recover 3D motion and subsequently shape. Catadioptric sensors could provide the field of view but they have poor resolution, making it difficult to recover shape models. Thus we built the Argus eye, a construction consisting of six cameras pointing outwards (Figure 3). Clearly, only parts of the sphere are imaged. You can see the collective field of view in figure 2. When this structure is moved arbitrarily in space, then data from all six cameras can be used to very accurately recover 3D motion, which can then be used in the individual videos to recover shape. The next section shows how we calibrated the Argus eye and the subsequent section describes how 3D motion was estimated.

3. Calibration

In order to calibrate the Argus Eye, it is not possible to use ordinary stereo calibration methods, because the fields

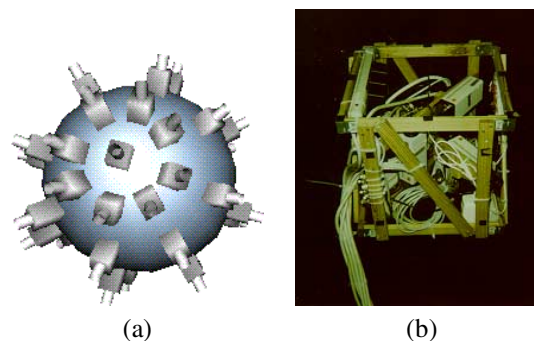
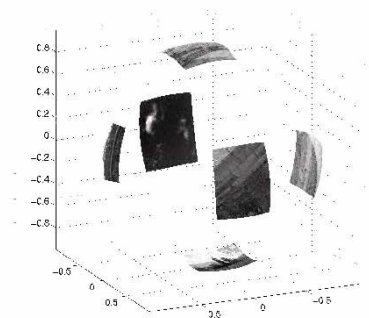


Figure 3. (a) A compound-like eye composed of conventional video cameras, and a schematic description of the Argus eye. (b) The actual Argus eye. The cameras are attached to diagonals of a polyhedron made out of wooden sticks.

of view do not overlap. Mechanical calibration is difficult and expensive, so we would like to use an image based method. There are a few possibilities for image based calibration, listed below.

Grid Calibration Construct a precisely measured calibration grid which surrounds the Argus eye, and then use standard calibration methods (such as [10] or [4]) from this. This method is difficult and expensive.

Self Calibration Use a self-calibration algorithm to obtain the calibration parameters. By matching the axes of rotation in the various cameras, this method should be able to obtain the rotation between the cameras. The accuracy of this method is not as high as we would like.

Many Camera Calibration If additional cameras were placed around the Argus eye pointing inwards in such a way that the cones of view of the cameras intersected with each other and with those of the Argus eye, then those cameras, properly calibrated, can be used to calibrate if we use an LED as a corresponding point. We use this method for translational calibration, but use a new method for rotational calibration.

Line Calibration If we can use lines as our calibration objects instead of points, the lack of intersecting fields of view is not a problem, because a line has the extent to cross multiple fields of view.

3.1. Method overview

Lines have not been used extensively before in the calibration of cameras because the mathematical tools in computer vision used frequently are based on points. We would like to use lines because they extend over non-overlapping fields of view, which makes them simple objects that can be used for the Argus eye. An additional benefit of lines over other objects is that they are easier to locate with subpixel accuracy, thus allowing our calibration to be much more accurate than a method based on spheres or LEDs. There is an equation which constrains just the rotation of three cameras based on corresponding lines which is perfect for our purposes, since we have multiple cameras. One of the first appearances of these equations in the computer vision literature is in [7] in 1986, but they have not been used in vision systems. We present the mathematics in the next section, then describe in more detail the algorithm.

3.2. Mathematics

Given three normalized cameras and a world line L , we derive here the constraint on the rotation matrices. The projection matrices for our cameras are:

$$P_i = R_i^T [I_3 \quad -c_i] \quad (1)$$

where c_i is the position of the camera with respect to a coordinate system attached to the center of the Argus eye and R_i is the rotation of the camera with respect to that coordinate system.

Let the line L be imaged in each of the cameras and let $\hat{\pi}_i$ and $\hat{\sigma}_i$ be two unitized points on the image line in each camera i . If the six points Π_i and Σ_i (all on L), generate these camera points, then we know:

$$\hat{\pi}_i = \frac{1}{\lambda_i} R_i^T (\Pi_i - c_i) \quad \hat{\sigma}_i = \frac{1}{\delta_i} R_i^T (\Sigma_i - c_i) \quad (2)$$

$$\lambda_i R_i \hat{\pi}_i = \Pi_i - c_i \quad \delta_i R_i \hat{\sigma}_i = \Sigma_i - c_i \quad (3)$$

where λ_i and δ_i are scale factors chose to unitize $\hat{\pi}_i$ and $\hat{\sigma}_i$.

Since the R_i are orthonormal, we may write after some algebra

$$\lambda_i \delta_i R_i (\hat{\sigma}_i \times \hat{\pi}_i) = \Sigma_i \times \Pi_i + (\Pi_i - \Sigma_i) \times c_i \quad (4)$$

Since for each i , Π_i and Σ_i are both on the same line, we see that the RHS of the above equation must be perpendicular to the direction of the line L for all i . If we have three cameras, then we know that the $R_i (\hat{\sigma}_i \times \hat{\pi}_i)$ must lie in one plane (we may ignore the scale factors). So we get the following triple product constraint:

$$|R_1(\hat{\sigma}_1 \times \hat{\pi}_1) \quad R_2(\hat{\sigma}_2 \times \hat{\pi}_2) \quad R_3(\hat{\sigma}_3 \times \hat{\pi}_3)| = 0 \quad (5)$$

It is significant that the world lines imaged by each of the three cameras do not have to be identical, only be parallel, since the constraint only depends on the perpendicularity to the *direction* of the world lines.

Obviously we could use the trilinear constraint to accomplish the same task, but the above constraint has the parallel advantage. We do not need to correspond *identical* lines, only parallel lines. This opens up a wealth of possibilities for calibration objects. First, we can use the occlusion boundaries of cylinders, since all occlusion boundaries on a cylinder are parallel. Secondly, if we place two cylinders on opposite sides of a square (as in figure 4), then we can put the object *around* the Argus eye, so that we may rotationally calibrate all the cameras together, for greater accuracy. Thus, if our cameras are positioned in such a way that there are no world lines which three cameras see, we may still rotationally calibrate if we have parallel lines visible in three cameras.

In order to obtain translational calibration, we can use a similar method, using the trilinear constraint to constrain the positions of the camera centers with respect to camera 1. Let us call them the $T_{1,k}$. Similarly we can define the translation between any two cameras as $T_{j,k}$. Since we already have the internal calibration and rotations, then we may use a greatly simplified trilinear constraint on corresponding lines in mul-

triples cameras. The trilinear constraint in our notation becomes:

$$\ell_3^T \hat{\pi}_1' T_{1,2}^T \ell_2' = \ell_2^T \hat{\pi}_1' T_{1,3}^T \ell_3' \quad (6)$$

where $\ell_i = \hat{\pi}_i \times \hat{\sigma}_i$ for $i \in \{2, 3\}$, and the primes indicate that the image point or line has been derotated into the common coordinate system. That is:

$$\ell_i' = \widehat{R}_i \ell_i \quad (7)$$

$$\hat{\pi}_i' = \widehat{R}_i \hat{\pi}_i \quad (8)$$

We may use the $\hat{\pi}_i$ and the $\hat{\sigma}_i$ over many frames to obtain the translational calibration by means of a linear system which solves for the $T_{j,k}$. We have found this procedure to yield accurate translational calibration.

3.3. Calibration Procedure

We have now the results to construct our final calibration procedure and algorithm. First, we internally calibrate the cameras using some well-known algorithm, such as that by Zhengyou Zhang in [11]. Second, we rotationally calibrate the cameras using the above algorithm and a square frame with two black poles measured to be parallel as in figure 4. We may do this step as a large nonlinear optimization over the homogeneous Rodrigues parameters of the rotation matrices. We have found this method to converge well.

Since we do not have sufficient field of view coverage with only six cameras to use a single line for the final calibration step, we instead calibrate with the Argus cameras and some additional cameras surrounding and pointing inwards. With the external cameras, we can guarantee that three cameras will see the lines, so that the trilinear constraint may be used. Note that this is only necessary because we have only used six cameras. If an Argus Eye was used which had more cameras, the additional cameras would not be necessary.

4. 3D motion from the Argus eye

4.1. The Problem

Consider a calibrated Argus eye moving in an unrestricted manner in space, collecting synchronized video from each of the video cameras. We would like to find the 3D motion of the whole system. Given that motion and the calibration, we can then determine the motion of each individual camera so that we can reconstruct shape. Before going into the details of the algorithm let us give a pictorial motivation for our camera system.

An important aspect of the results regarding the ambiguity in motion estimation for small fields of view is that they

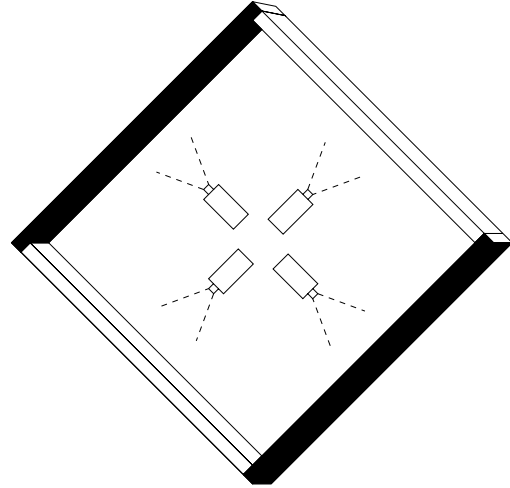


Figure 4. Calibration Frame to Rotationally Calibrate Cameras with Non-intersecting Fields of View

are algorithm independent. Simply put, whatever the objective function one minimizes, the minima will lie along valleys. The data is not sufficient to disambiguate further. Let us look at pictures of these ambiguities. We constructed a six camera Argus Eye and estimated the 3D motion independently in each of the six sequences. In particular, we used the Lucas-Kanade [8] algorithm to estimate optical flow. Then for every direction of translation we find the corresponding best rotation which minimizes deviation from the epipolar constraint. Figure 5 shows (on the sphere of possible translations) the residuals color coded. Noting that the red areas are all the points within a small percentage of the minimum, we can see the valley which clearly demonstrates the ambiguity theoretically shown in the proofs. Our translation could be anywhere in the red area. With such a strong ambiguity, it is clear that accurate shape models are difficult to construct.

These difficulties disappear for cameras with complete fields of view. Catadioptric cameras can obtain such fields of view, but their sampling properties are undesirable. Our Argus eye has better sampling properties, but it is not clear how to calculate the 6D motion of the system using current algorithms. A 6D search is undesirable, so we show in the sequel how to combine known algorithms to get an accurate starting point for a 6D gradient descent.

4.2. Combining the Estimates from Individual Cameras

Let us assume that for every camera i , since it is already calibrated, its projection matrix is given by (1). A rigid motion t, ω in the coordinate system of the Argus eye corre-

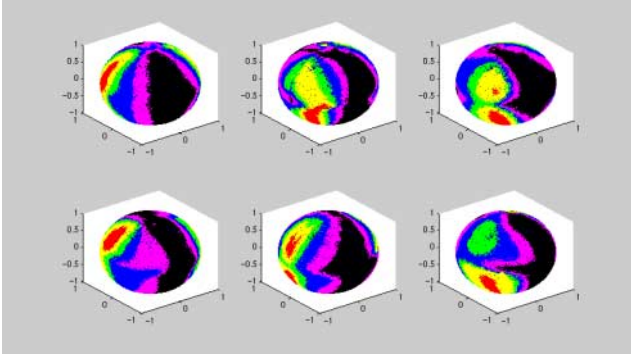


Figure 5. Deviation from the epipolar constraints for motion estimation from individual cameras

sponds in the coordinate systems of the individual cameras to a translation

$$\mathbf{t}_i = R_i(\mathbf{t} + \boldsymbol{\omega} \times \mathbf{c}_i) \quad (9)$$

and a rotation

$$\boldsymbol{\omega}_i = R_i \boldsymbol{\omega} \quad (10)$$

The dependence of the translational components in the different cameras on the rotation complicates the translational motion estimation. However, we may equate the rotation $\boldsymbol{\omega} = R_i^T \boldsymbol{\omega}_i$ for every camera. It is easy to estimate the rotation if the translation is known, even for cameras with limited fields of view. This fact is exploited in the algorithm, which works as follows. We perform the rigid motion estimation for every camera individually, and we consider the valleys with the best candidate translations. To each translation corresponds a best rotation and the intersection of these rotations corresponding to all valleys provides the rotation of the system. Then subtracting the rotational component we find in each camera candidates for the translation. Their intersection provides the translation of the system.

In more detail, we use the depth variability constraint [1] which relates the spatiotemporal derivatives to the 3D motion and structure. The algorithm is designed as a search in the space of translational directions. We assume the scene in view is patchwise smooth. The image is partitioned into regions and for every region \mathcal{R} we define a measure

$$\Theta_0(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R}) = \sum_i W_i \left[\mathbf{r}_i \cdot \mathbf{n}_i - \frac{1}{f} \hat{\mathbf{z}} \times (\mathbf{r}_i \times (\hat{\boldsymbol{\omega}} \times \mathbf{r})) \cdot \mathbf{n}_i - \frac{1}{Z} (\hat{\mathbf{z}} \times (\hat{\mathbf{t}} \times \mathbf{r}_i)) \cdot \mathbf{n}_i \right]^2 \quad (11)$$

where \mathbf{r}_i and \mathbf{n}_i are the flow and gradient direction at point i and W_i is a weight.

We minimize Θ_0 with respect to $1/\hat{Z}$ (the best scene structure) to obtain

$$\frac{1}{\hat{Z}} = \frac{\sum_i W_i \left[\mathbf{r}_i \cdot \mathbf{n}_i - \frac{1}{f} \hat{\mathbf{z}} \times (\mathbf{r}_i \times (\boldsymbol{\omega} \times \mathbf{r})) \cdot \mathbf{n}_i \right]}{\sum_i W_i \left[(\hat{\mathbf{z}} \times (\hat{\mathbf{t}} \times \mathbf{r}_i)) \cdot \mathbf{n}_i \right]^2} \quad (12)$$

and substitute (12) into (11) which yields $\Theta_1(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R})$. Then we sum all local Θ_1 to obtain a global function

$$\Theta_2(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}) = \sum_{\mathcal{R}} \Theta_1(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}}, \mathcal{R}) \quad (13)$$

whose minimization yields a closed form solution to the best rotation $\hat{\boldsymbol{\omega}}$ for every translation candidate $\hat{\mathbf{t}}$ as well as a measure of the depth variation corresponding to $\hat{\mathbf{t}}$. The smallest values $\Theta_2(\hat{\mathbf{t}}, \hat{\boldsymbol{\omega}})$ correspond to the best motion candidates.

For each camera i , let us consider the set of translations $\{\mathbf{t}_{i,j}\}$ with error close to the minimum. Given a translational estimate $\mathbf{t}_{i,j}$, we estimated the rotation $\boldsymbol{\omega}_{i,j}$. Call $f: \mathbf{T} \rightarrow \Omega$, a function from the set of translations to the set of rotations. This function f is a diffeomorphism, so that given the 2D manifold of candidate translations (the ones with low error), we have a 2D manifold of candidate rotations, which we can derotate by R_i^T , to obtain a 2D manifold of rotational estimates in the fiducial coordinate system. Significantly, the *rotations exist in 3D space*, so that from six cameras, we have six 2D manifolds of candidate rotations in the 3D space of possible rotations. We can then find their intersection, which in general should be a single point. The attached video shows these manifolds growing and intersecting as rotational candidates in the valley are added, in order of increasing error.

This video confirms two basic tenets of this paper. First, it shows that the motion estimates of lowest error in individual cameras are *not* the correct motions, since if they were, the lowest error points would be coincident in rotation space. Thus even though we are using state-of-the-art algorithms, it is not possible to extract the correct motion from a single camera with limited field of view, as is shown in the proof. Second, the video shows that if we look at *all* the motion candidates of low error, the correct motion is in that set, shown by the intersection of the six manifolds at a single point.

That the manifolds intersect so closely shows we can find the rotation well. Next, let us look at what the translations are in each camera. Given this accurate rotation, the translational ambiguity in each camera is confined to a very thin valley, shown in Figure 6. That the translations are not completely specified in some cameras is a result of the fact that those cameras look at scenes very far away and have sparse data (e.g. the sky). Next we have to intersect the transla-

tions represented by these valleys, to find the complete 3D translation.

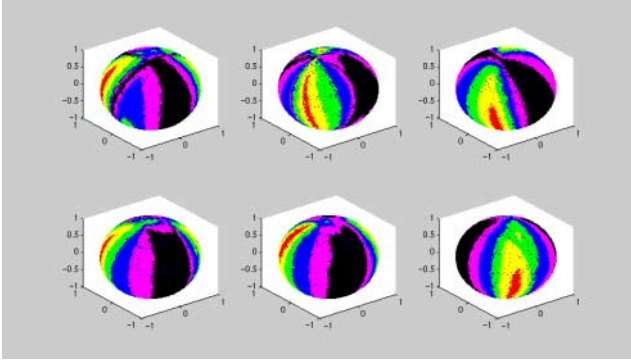


Figure 6. Error measure in the individual cameras after estimation of the system's rotation

The translation t_i in individual cameras is related to the overall translation t by (9). Since motion only allows us to estimate the direction of translation, we have

$$\lambda t_{io} = R_i t + R_i \omega \times c_i \quad (14)$$

with λ a constant and t_{io} a unit vector, from which we obtain a constraint on the common translation of the form

$$t = \lambda R_i^T t_{io} - R_i \omega \times c_i \quad (15)$$

This constraint defines for every candidate translation t_{io} a line. The intersection of all the constraints for the candidate translations in the six cameras provides the 3D translation of the system.

4.3. Improving on the Starting Point

At this point, the motion is still not optimum, since cameras which view very little texture will have contributed an equal amount to the determination of the motion as cameras with a lot of texture. But since we have a good starting point, we use a gradient descent algorithm over the 6D rotation and translation space, with the error function being the depth variability error (13). We put the rotation and translation into the appropriate coordinate frame using the calibration, and then add together the error obtained from each individual camera. This gradient descent reaches a well-defined minimum in the rotation and the direction of translation. The size of the translation vector is not accurate because our cameras are too close together, compared to the depth of the scene. However, we do get an estimate of the translational scale, and the metric depth calculated is within a meter at distances of 10m.

In Figure 8 we see the location of the low error translations in a spherical slice of 3D translation space. Notice the

Camera	Frame number	
	200	215
1	4.6038×10^{-4}	4.1906×10^{-4}
2	9.7312×10^{-5}	3.9176×10^{-4}
3	1.0381×10^{-5}	4.2621×10^{-4}
4	4.9273×10^{-5}	2.7382×10^{-3}
5	6.0546×10^{-4}	1.5422×10^{-3}
6	2.9612×10^{-5}	7.9458×10^{-5}
all	1.1782×10^{-2}	6.9629×10^{-2}

Figure 7. Ratio of the second smallest to the largest singular value in the Hessian of the error function

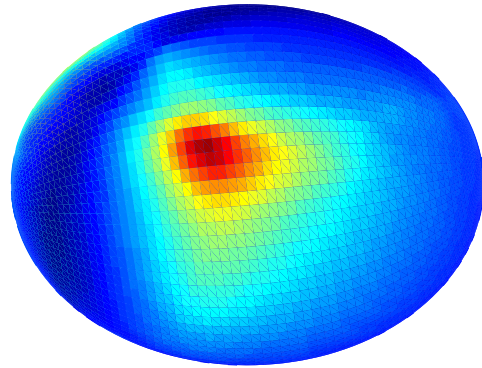


Figure 8. Translation estimation for the whole system

well-defined minimum (in red), indicating that the direction of the translation obtained is not ambiguous. We have done empirical tests on the Hessian of the error function at its minimum. This demonstrates that our Argus Eye gives a much better defined minimum than any single camera, showing that we have removed the ambiguity that makes egomotion recovery difficult. In Table 7, we show the ratio of the second smallest singular value (the smallest is zero due to the scale ambiguity) for four frames from our sequence for each individual camera and for the camera system as a whole. The ratio is at least $\frac{1}{100}$ for all of the frames where all the cameras were used and is never more than $\frac{1}{600}$ where only one camera was used (usually much worse). So we can see that the valley for one camera is rather flat, and for all the cameras together, there is a much better defined minimum.

Let us also look at the complete eigenvalue structure for the Hessian of the error function at frame 215, for each camera individually and all cameras together (Figure 9). We see that each camera individually does not provide a good constraint on the motion, but when put together using the Argus

Cam.	Sorted Eigenvalues (Smallest 3)		
1	8.8×10^{-4}	1.4×10^{-2}	3.0×10^{-2}
2	4.1×10^{-4}	4.9×10^{-3}	3.0×10^{-2}
3	9.4×10^{-3}	1.3×10^{-2}	4.9×10^{-2}
4	2.7×10^{-2}	4.6×10^{-2}	6.6×10^{-2}
5	1.3×10^{-2}	2.1×10^{-2}	1.4×10^{-1}
6	1.2×10^{-5}	4.2×10^{-5}	8.7×10^{-3}
all	9.8×10^{-2}	3.5×10^0	1.3×10^1

Cam.	Sorted Eigenvalues (Largest 3)		
	200	215	
1	3.0×10^{-1}	4.0×10^{-1}	3.3×10^1
2	1.4×10^{-1}	2.8×10^{-1}	1.2×10^1
3	4.4×10^{-1}	3.6×10^0	3.1×10^1
4	2.5×10^{-1}	3.4×10^0	1.7×10^1
5	2.8×10^{-1}	1.1×10^0	1.4×10^1
6	2.1×10^{-2}	2.2×10^{-2}	5.4×10^{-1}
all	1.6×10^1	3.5×10^1	5.0×10^1

Figure 9. Eigenvalues of the Hessian for individual cameras and all cameras together

Eye, we get a very well defined minimum, shown by the fact that all (but the sixth) eigenvalues of the Hessian are large. Note that while *none* of the individual cameras has a fifth smallest eigenvalue over 5×10^{-2} , the fifth largest eigenvalue of all cameras together is 3.5×10^0 . Clearly we get a minimum which is much better behaved. The ambiguities are in different directions for different cameras, and they support each other for a definite minimum in the full space.

The fact that the six cameras don't have the same nodal point complicates the 3D motion estimation, but it also brings advantages: It allows us to estimate all three parameters of the translation. Since from the calibration we have metric information, in the case where the translation is not much larger than the rotation and the distance between the cameras is significant, it is possible to calculate the absolute translation. Thus camera construction techniques which force the centers of projection to be coincident may have simpler algorithms, but the data is not as rich. Here we can obtain metric depth without using stereo techniques.

5. Conclusions

Our work is based on recent theoretical results in the literature that established the robustness of 3D motion estimation as a function of the field of view. We built a new imaging system, called the Argus eye, consisting of six high-resolution cameras sampling a part of the plenoptic function. We calibrated the system and developed an algorithm for recovering the system's 3D motion by processing all synchronized videos. Our solution provides remarkably accurate re-

sults that can be used for building models from video, for use in a variety of applications.

References

- [1] T. Brodský, C. Fermüller, and Y. Aloimonos. Structure from motion: Beyond the epipolar constraint. *International Journal of Computer Vision*, 37:231–258, 2000.
- [2] K. Daniilidis. *On the Error Sensitivity in the Recovery of Object Descriptions*. PhD thesis, Department of Informatics, University of Karlsruhe, Germany, 1992. In German.
- [3] K. Daniilidis and M. E. Spetsakis. Understanding noise sensitivity in structure from motion. In Y. Aloimonos, editor, *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, Advances in Computer Vision, chapter 4. Lawrence Erlbaum Associates, Mahwah, NJ, 1997.
- [4] O. Faugeras and G. Toscani. The calibration problem for stereo. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.
- [5] C. Fermüller and Y. Aloimonos. Observability of 3d motion. *International Journal of Computer Vision*, 37(1):43–62, June 2000.
- [6] A. D. Jepson and D. J. Heeger. Subspace methods for recovering rigid motion II: Theory. Technical Report RBCV-TR-90-36, University of Toronto, 1990.
- [7] Y. Liu and T. S. Huang. Estimation of rigid body motion using straight line correspondences: Further results. In *Proc. International Conference on Pattern Recognition*, 1986.
- [8] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.
- [9] S. J. Maybank. Algorithm for analysing optical flow based on the least-squares method. *Image and Vision Computing*, 4:38–42, 1986.
- [10] R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.
- [11] Z. Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, 1998.