# Linear Algorithms for Multi-Frame Structure from Constrained Motion

**T. N. Tan, G. D. Sullivan and K. D. Baker**
**Department of Computer Science**
**The University of Reading, Berkshire RG6 2AY, UK**
Email: T.Tan@uk.ac.reading

**Abstract**

Linear algorithms are presented for computing the 3-D structure and motion of vehicles in traffic scenes under the ground-plane constraint. Point matches in multiple image frames are used. The algorithms first determine the inter-frame rotation angles, then the point depths, and finally the inter-frame translations. All computations are linear and require no parameter tuning. The algorithms are tested with both synthetic data and routine traffic images. Extensive experimental results are reported that demonstrate the validity of the algorithms.

## 1 Introduction

Much effort has been devoted to the computation of 3-D structure and motion (SFM) from point matches in multiple frames [1]. We call the SFM algorithms that use multiple (more than two) frames the Multiple Frame SFM, or MFSFM algorithms. A MFSFM algorithm can be either recursive or batch in nature, depending on whether it processes one frame at a time or all frames simultaneously. In general, batch approaches have been shown to be both more accurate and stable [2], but recursive techniques such as [3] are more convenient to use in practice.

Despite nearly two decades of active research in the area, a general robust SFM algorithm remains elusive [1]. The difficulties are primarily due to the scale of the task of solving the six degrees of freedom non-linear problem allowed by general SFM algorithms. Many practical tasks in vision need be concerned with fewer degrees of freedom, since object motion is often subject to physical constraints, such as the commonly occurring ground plane constraint (GPC) [4]. The GPC reduces the number of degrees of freedom (DOF) of a rigid object from 6 to 3. The 3 DOFs are most conveniently expressed as the translation $(X, Y)$ on the ground-plane (GP), and the rotation $(\theta)$ about the axis normal to the GP. In previous work [5], we have shown that the GPC allows simple and robust MFSFM algorithms which do not suffer from the usual pitfalls of the existing MFSFM algorithms. The algorithm described in [5] is based on the distance invariance property of the rigidity constraint [6]. The algorithm first computes the depths of object points by solving a set of quadratic equations, and then the inter-frame motion parameters are estimated by one of the absolute orientation algorithms (e.g., [7]).

In this paper we show that the GPC also permits entirely linear and robust MFSFM algorithms. The algorithm described here decomposes the computation into three stages. It first determines the inter-frame rotation angles, then computes the depths

BMVC 1994 doi:10.5244/C.8.58

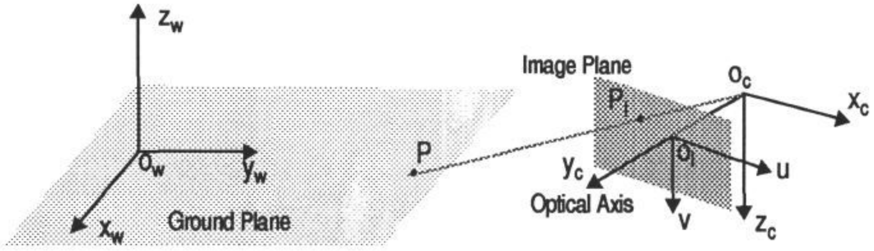

*Figure 1: Coordinate systems and the imaging model.*

of points, and finally estimates the inter-frame translations. The computation at each stage is linear. The new algorithm has several advantages over that given in [5]:

- The recovered structure and motion are guaranteed to be rigid since the full rigidity constraint (rather than the distance invariance property between points) is imposed.
- The algorithm is entirely linear.
- The algorithm is mathematically more elegant and involves no heuristics.

The work presented in this paper was conducted in the context of machine vision for traffic scene analysis, but is also applicable to a wide range of potential industrial applications such as mobile robot navigation [8].

## 2 Geometry and Notations

The camera is regarded as a linear pinhole perspective camera without lens distortion. The geometry of the camera coordinate system (CCS) and the world coordinate system (WCS) is illustrated in Fig.1. The WCS is defined on the GP on which objects such as vehicles are confined to move. The CCS is related to the WCS by a known 3x3 rotation matrix $\boldsymbol{R}$ and a known 3x1 translation vector $\boldsymbol{T} = (C_x \ C_y \ C_z)^T$ ($\boldsymbol{R}$ and $\boldsymbol{T}$ are obtained by means of camera calibration [11]). Under the imaging geometry shown in Fig.1, the camera coordinates of an image point $(u, v)$ is given by $\boldsymbol{P}_I = (u \ \mathcal{F} \ v)^T$, and those of the corresponding 3-D point by $\boldsymbol{P}_c = \lambda \boldsymbol{P}_I$, where $\mathcal{F}$ is the focal length, and $\lambda$ the depth scale (the depth from $O_c$) of the 3-D point. The world coordinates $\boldsymbol{P}_w$ of the 3-D point is

$$\boldsymbol{P}_w = \lambda (U \ V \ W)^T + \boldsymbol{T} \tag{1}$$

where $(U \ V \ W)^T = \boldsymbol{R}\boldsymbol{P}_I$. Under the GPC, the world coordinates $\boldsymbol{P'}_w$ of a point at time $t'$ are related to its coordinates $\boldsymbol{P}_w$ at time $t$ by

$$\boldsymbol{P'}_w = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{P}_w + \begin{bmatrix} X \\ Y \\ 0 \end{bmatrix} \tag{2}$$

By combining (1) and (2), we obtain

$$\lambda' W' + C_z = \lambda W + C_z \tag{3a}$$

$$\lambda' U' + C_x = (\lambda U + C_x) \cos\theta - (\lambda V + C_y) \sin\theta + X \tag{3b}$$

$$\lambda' V' + C_y = (\lambda U + C_x)\sin\theta + (\lambda V + C_y)\cos\theta + Y \tag{3c}$$

From (3a) the depths $\lambda$ and $\lambda'$ of a point at time $t$ and $t'$ are related to each other by

$$\lambda' = \lambda W / W' \equiv Q\lambda \tag{4}$$

The following symbols are defined for use in the subsequent discussions:

$S_F = \{F_0, F_1, F_2, \ldots, F_{M-1}, F_M\}$ : the set of $M+1$ frames in which points have been detected and matched.
$S_P = \{P_1, P_2, \ldots, P_{N-1}, P_N\}$ : the set of points appearing in $S_F$.
$S_{Pm} = \{P_{m1}, P_{m2}, \ldots, P_{mN_m}\}$ : the set of points present in frame $F_m$, i.e., $S_{Pm} \in S_P$.

We do not require $S_{Pm} = S_{Pn}$, $m \neq n$, thus point occlusions are allowed. The motion of the image sequence is described by the inter-frame motion parameters between an arbitrarily chosen reference frame and each of the other frames. For convenience, we use $F_0$ as the reference frame and assume that all points are present in the reference frame, i.e., $S_{P0} = S_P$. Let the 3D structure of the object be defined by the depths $\lambda_1, \lambda_2, \ldots, \lambda_N$ of the $N$ points in the reference frame. The problem to be solved is: Given $S_F$ and $S_{Pm}, m \in \{0, 1, 2, \ldots, M\}$, determine $\lambda_1, \lambda_2, \ldots, \lambda_N$, and the motion $\{X_m, Y_m, \theta_m\}$ from the reference frame $F_0$ to each of the other $M$ frames.

# 3 Computation of Rotation Angle

We first discuss how to determine the single rotation angle from the reference frame $F_0$ to Frame $F_m$. By substituting (4) into (3b) and (3c), we obtain for each point $P_i$

$$\lambda_i Q_{m,i} U_{m,i} + C_x = \lambda_i U_{0,i}\cos\theta_m - \lambda_i V_{0,i}\sin\theta_m + C_x\cos\theta_m - C_y\sin\theta_m + X_m \tag{5}$$

$$\lambda_i Q_{m,i} V_{m,i} + C_y = \lambda_i V_{0,i}\cos\theta_m + \lambda_i U_{0,i}\sin\theta_m + C_x\sin\theta_m + C_y\cos\theta_m + Y_m \tag{6}$$

where subscripts 0 and $m$ indicate the reference frame $F_0$ and Frame $F_m$ respectively. When confusion is unlikely to occur, we omit, for the sake of clarity, the subscript 0 in the following discussions. Therefore a pair of matched points $P_i$ and $P_j$ leads to the following four equations:

$$\lambda_i Q_{m,i} U_{m,i} + C_x = \lambda_i U_i\cos\theta_m - \lambda_i V_i\sin\theta_m + C_x\cos\theta_m - C_y\sin\theta_m + X_m \tag{7a}$$

$$\lambda_i Q_{m,i} V_{m,i} + C_y = \lambda_i V_i\cos\theta_m + \lambda_i U_i\sin\theta_m + C_x\sin\theta_m + C_y\cos\theta_m + Y_m \tag{7b}$$

$$\lambda_j Q_{m,j} U_{m,j} + C_x = \lambda_j U_j\cos\theta_m - \lambda_j V_j\sin\theta_m + C_x\cos\theta_m - C_y\sin\theta_m + X_m \tag{7c}$$

$$\lambda_j Q_{m,j} V_{m,j} + C_y = \lambda_j V_j\cos\theta_m + \lambda_j U_j\sin\theta_m + C_x\sin\theta_m + C_y\cos\theta_m + Y_m \tag{7d}$$

By subtracting both sides of (7a) from those of (7c), we get

$$\lambda_j Q_{m,j} U_{m,j} - \lambda_i Q_{m,i} U_{m,i} = (\lambda_j U_j - \lambda_i U_i)\cos\theta_m - (\lambda_j V_j - \lambda_i V_i)\sin\theta_m \tag{8}$$

Similar operations on (7b) and (7d) give rise to the following equation:

$$\lambda_j Q_{m,j} V_{m,j} - \lambda_i Q_{m,i} V_{m,i} = (\lambda_j U_j - \lambda_i U_i)\sin\theta_m + (\lambda_j V_j - \lambda_i V_i)\cos\theta_m \tag{9}$$

Eliminating the depth parameters from (8) and (9) yields

$$F\cos\theta_m + G\sin\theta_m = H \tag{10}$$

where the coefficients are given by

$$\begin{cases} F = V_i Q_{m,j} U_{m,i} + U_j Q_{m,i} V_{m,i} - U_i Q_{m,j} V_{m,j} - V_j Q_{m,i} U_{m,i} \\ G = V_i Q_{m,j} V_{m,j} - U_j Q_{m,i} U_{m,i} + U_i Q_{m,j} U_{m,j} - V_j Q_{m,i} V_{m,i} \\ H = Q_{m,i} V_{m,i} Q_{m,j} U_{m,j} - Q_{m,i} U_{m,i} Q_{m,j} V_{m,j} + V_i U_j - U_i V_j \end{cases} \tag{11}$$

Eqn. (10) states that each pair of points specifies one constraint on the rotation angle. If there are $N_m$ points in $F_m$, one can write a total of $N_m(N_m - 1)/2$ such constraints (by considering all possible point pairs):

$$F_n\cos\theta_m + G_n\sin\theta_m = H_n, \quad n = 1, 2, \ldots, N_m(N_m - 1)/2; \tag{12}$$

The equations in (12) only involve the rotation angle so the rotation parameter can be computed independently of the translation and depth parameters. In the following, we outline two linear techniques for solving the rotation angle $\theta_m$ from (12).

### 3.1 The linear least squares (LLS) technique

If $\cos\theta_m$ and $\sin\theta_m$ are regarded as independent variables, (12) becomes a set of linear equations in two unknowns. The standard LLS technique can then be used to compute $\cos\theta_m$ and $\sin\theta_m$. The rotation angle $\theta_m$ can easily be derived from $\cos\theta_m$ and $\sin\theta_m$.

### 3.2 The non-linear least squares (NLS) technique

Closed-form solutions to (12) that do not neglect the constraint $\cos^2\theta_m + \sin^2\theta_m = 1$ are also possible. The equations in (12) can be written in a matrix form as

$$\boldsymbol{A}q = \boldsymbol{H} \tag{13}$$

where $q = (\cos\theta \;\; \sin\theta)^T$, and $\boldsymbol{A}$ and $\boldsymbol{H}$ are the known coefficient matrices. The NLS solution of (13) is then given by

$$q = \arg\{\min_q \|\boldsymbol{A}q - \boldsymbol{H}\|^2\}; \text{ subject to } \|q\|^2 = 1 \tag{14}$$

The above constrained minimization problem can be mapped into a 4th-order polynomial equation in one unknown. The roots of the polynomial may be found in closed-form, and the rotation angle $\theta_m$ is computed accordingly [9].

In summary, we have shown that the single rotation angle $\theta_m$ from the reference frame $F_0$ to Frame $F_m$ can be computed independently of the depth and translational parameters. Two point matches specify one constraint on the rotation angle, thus allowing a maximum of two solutions. Three points define three constraints, and the solution for the rotation angle is in general unique. The rotation angles from the reference frame to other frames are computed similarly.

## 4 Computation of Point Depths

Once rotation is known, Equations (8) and (9) impose two constraints on $\lambda_i$ and $\lambda_j$:

$$J_{m,i}\lambda_i = J_{m,j}\lambda_j; \quad K_{m,i}\lambda_i = K_{m,j}\lambda_j \tag{15}$$

where

$$J_{m,i} = Q_{m,i}U_{m,i} - U_i\cos\theta_m + V_i\sin\theta_m; \quad K_{m,i} = Q_{m,i}V_{m,i} - U_i\sin\theta_m - V_i\cos\theta_m \tag{16}$$

Therefore, for all points in all frames, we get a set of constraints on the depths of the $N$ points in $F_0$:

$$\begin{cases} J_{m,i}\lambda_i = J_{m,j}\lambda_j \\ \\ K_{m,i}\lambda_i = K_{m,j}\lambda_j \end{cases}, \quad \forall m, F_m \in S_F;\ i = 1, \ldots, N_m - 1;\ j = i+1, \ldots, N_m \tag{17}$$

The equations in (17) can be written in a matrix form as

$$C\Lambda = 0 \tag{18}$$

where $\Lambda$ ($= (\lambda_1\ \lambda_2\ \ldots\ \lambda_{N-1}\ \lambda_N)^T$) is the unknown depth vector, and $C$ is the known coefficient matrix. Since the equations (18) are homogeneous, the depths can only be recovered up to a global scale - a well-known fact in SFM [1].

### 4.1 Biased depth normalisation

The global scale in depth is commonly resolved by fixing the depth of one of the points. Let $\lambda_1 = 1$. Then Equation (18) becomes

$$C_{-1}\Lambda_{-1} = -C_1 \tag{19}$$

where $\Lambda_{-1} = (\lambda_2\ \ldots\ \lambda_{N-1}\ \lambda_N)^T$, $C_1$ is the first column of $C$, and $C_{-1}$ is the matrix formed by the remaining $N-1$ columns of $C$. $\Lambda_{-1}$ can easily be solved from (19) using the standard LLS technique.

### 4.2 Unbiased depth normalisation

The normalisation method described above is biased in that the depth solutions obtained under different normalisation points are in general not related to each other by a scale factor as they should. An unbiased method is to compute the depths from (18) subject to $\|\Lambda\|^2 = 1$. In this case, the objective function to minimise becomes

$$\varepsilon(\Lambda) = \|C\Lambda\|^2 = (C\Lambda)^T(C\Lambda) \equiv \Lambda^T D\Lambda, \quad \text{subject to } \|\Lambda\|^2 = 1 \tag{20}$$

where $D = C^T C$ is a $NxN$ symmetric matrix. The solution of (20) is given by the unit eigenvector of $D$ corresponding to the smallest eigenvalue.

## 5 Computation of Translational Parameters

Once the rotation angle and the depth parameters are known, the translation parameters can be solved very easily by substituting the known parameters into (5) and (6). To combat noise, we take the average of the solutions given by all points.

## 6 Experimental Results

Experiments have been carried out to investigate the validity and performance of the algorithms presented in the preceding sections. Both synthetic and real image data were used. With synthetic data, the performance of the algorithms was investigated by means of extensive Monte Carlo simulations. The set-up for the simulations was as follows. The parameters of the camera were those of a real outdoor calibrated camera having an effective focal length of 1475 pixels. The object was a cuboid of dimension 3x2x1.2 $m^3$ (=Length*Width*Height), and was located in front of the camera with a nominal depth of 23.6 metres. Object motion was generated by rotating the cuboid around the vertical axis, followed by a translation on the GP. The known object movement was chosen to represent typical vehicle movement in traffic scenes. A specified number of points was generated randomly within the cuboid. Data noise was simulated by adding zero mean, uniformly distributed random values to the ideal image coordinates of all points in all frames. The level of noise was indicated by the maximum perturbation $\Delta E$ (in pixels) on the image coordinates. The performance of the algorithms was measured by the relative errors (averaged over all frames) of the computed motion parameters, and the absolute standard scene error (SSE) which is the mean of the Euclidean distances between the original and reconstructed 3-D points in the reference frame. All images were of size 512x512 pixels. The average inter-frame rotation and translation were $5^o$ and 0.5m respectively.

## 6.1 Comparison of the LLS and NLS techniques

The performances of the LLS and the NLS technique are illustrated in Fig.2. The NLS technique is more accurate than the LLS technique in computing the rotation angle. The NLS technique exhibits a S-shaped noise sensitivity, whereas the LLS technique possesses a somewhat linear error curve.

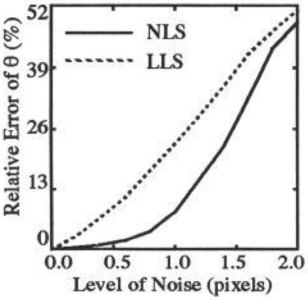

*Figure 2: Noise sensitivity of the LLS and the NLS technique in computing the rotation angle. Depth normalisation was unbiased. 10 points in 5 frames were used.*

## 6.2 Comparison of biased and unbiased depth normalisation

The two linear techniques for depth estimation described in Section 4 are compared by examining the accuracy of the recovered point structures. The SSE curves of the two techniques are shown in Fig.3. When the noise level is low (e.g., $\Delta E < 1.5$ pixels), there

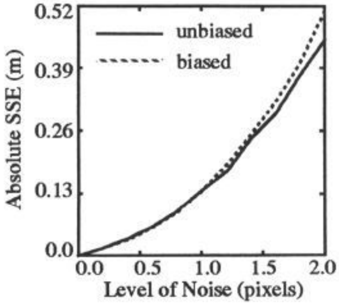

*Figure 3: Comparison of biased and unbiased depth normalisation. Rotation angles were computed by the LLS technique. 10 points in 5 frames were used.*

is little difference in the performance of the two techniques. However, when the noise level is higher, the advantage of the eigenvalue approach becomes clearer.

## 6.3 Noise reduction with more points

Monte Carlo simulations were conducted to determine the effectiveness of using more points in combating noise. During the simulations, the number of points in each frame was increased from the minimum of 3 to 19 in steps of 2. The results for the LLS-Biased algorithm (i.e., the combination of the LLS technique and the biased depth normalisation technique) are plotted in Fig.4. The error curves of all parameters are similarly shaped.

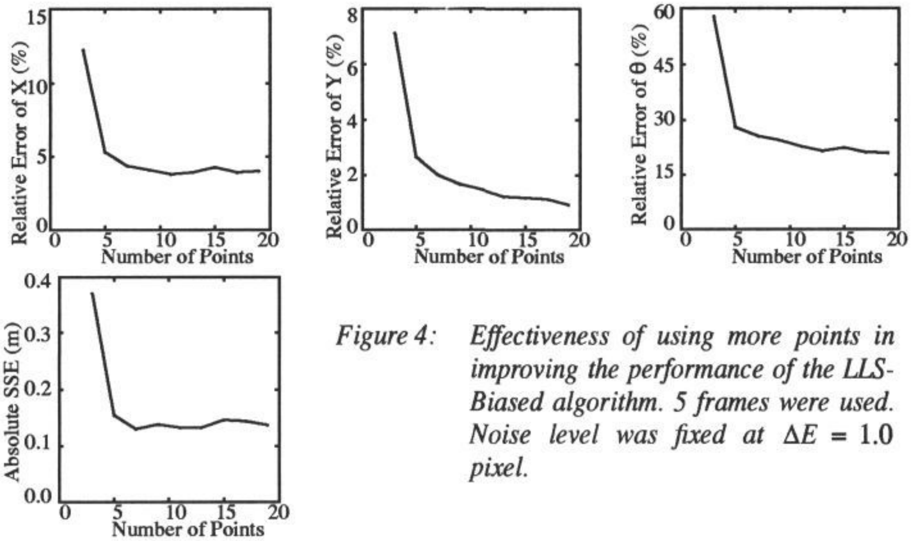

*Figure 4: Effectiveness of using more points in improving the performance of the LLS-Biased algorithm. 5 frames were used. Noise level was fixed at $\Delta E$ = 1.0 pixel.*

The improvement in performance is most significant when the number of points is increased from 3 to 5 - errors in all parameters are reduced by more than 50%. Further increase in the number of points beyond 10 results in little improvement. It should also be noticed that even with only 5 points, the relative error is no higher than 5.5% for X, 2.7% for Y, and 28% for the rotation angle (i.e., in absolute terms, ±28mm, ±14mm, and $\pm 1.4^o$), and the absolute SSE is no more than 0.16m (note the test cuboid is of size 3x2x1.2 $m^3$).

## 6.4 Noise reduction with more frames

Another way of combating noise is to use point matches from longer image sequences (i.e., more frames). Monte Carlo simulations were conducted to determine the effectiveness of using more frames in noise reduction. During the simulations, the number of frames was increased from the minimum of 2 to 16 in steps of 2. The results for the LLS-Biased algorithm are shown in Fig.5. Errors in all parameters are reduced when more frames are used. The most dramatic improvement occurs when the number of frames is increased from 2 to 4. This is true especially for the SSE. It also appears that the advantage of using more frames beyond 8 is very small. It is particularly encouraging to note that with 10 points in 16 frames, the relative errors are no higher than 1% for the translational parameters and less than 8% for the rotation angle. The SSE is as small as 0.03m.

It is worth of pointing out that naively, the relative error in rotation angle might

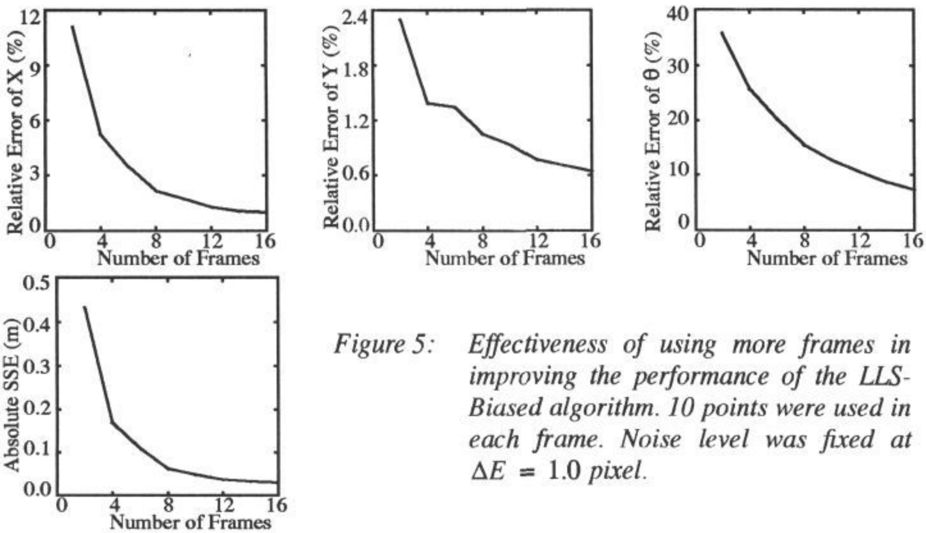

*Figure 5: Effectiveness of using more frames in improving the performance of the LLS-Biased algorithm. 10 points were used in each frame. Noise level was fixed at* $\Delta E = 1.0$ *pixel.*

be expected to be independent of the number of frames used (so the error curve for rotation in Fig.5 should be flat) since the computation of the rotation angle is done on a frame basis (see Section 3). Nevertheless, because the relative error is an average taken across all frames, and because the further a frame is from the reference frame, the more accurate the rotation angle estimate is (due to the larger magnitude of motion from the reference frame to the frame under consideration), the error curve shown in Fig.5 is thus expected.

## 6.5 Performance with real image data

The algorithms were also applied to compute the 3-D structure and motion of vehicles in routine traffic images. Segments of two frames from an outdoor traffic image sequence are shown in Fig.6. We initially tried to use automatic corner detectors (e.g., [10]) to locate and provide the point matches required by the algorithms. It turned out that the automatically detected corners were so unstable that they were hardly usable, since the surface of the vehicle is smooth, the corners of the vehicle are not well defined, and their positions in the image were affected by self-occlusion and specularities. Therefore, the corners were located and matched by eye. Eight corner points in 5 frames (sampled at 5 Hz) were identified as shown in the left column of Fig.6 (only two frames are shown due to space limitation). Although the detection and matching of points was done manually, there is still significant inaccuracy in the matches because of the low contrast of the images and the poor definition of the corners mentioned above.

With Frame 200 being used as the reference frame, the 8 point matches in the five frames are used by the LLS-Biased algorithm to compute the 3-D coordinates of the 8 points and the inter-frame motion parameters between Frame 200 and each of the other four frames. The global scale is resolved by assuming a known height (=1.0m) of Point $P_5$. Because the exact ground-truth is not available, the accuracy of the recovered structure and motion parameters cannot be measured quantitatively. We hence adopt the following qualitative assessment.

The accuracy of the motion parameters is measured by the goodness-of-fit

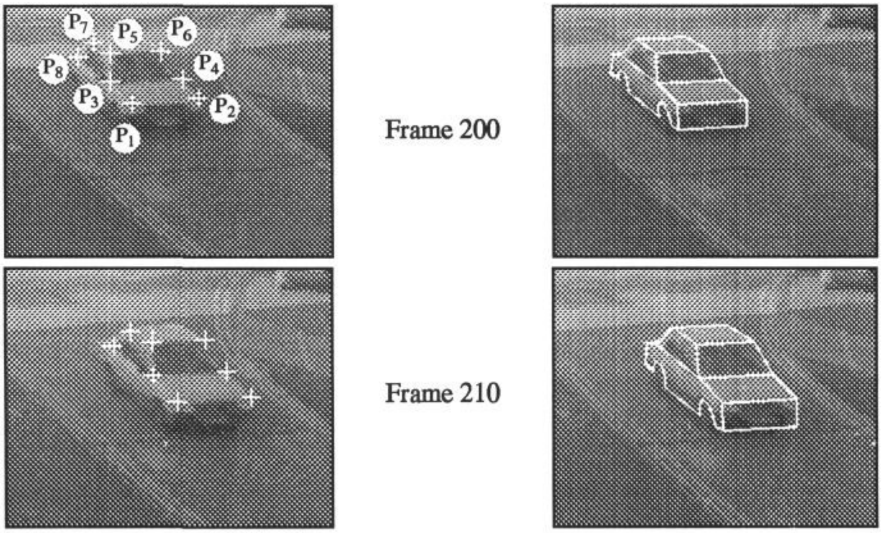

*Figure 6: Computation of 3-D structure and motion of vehicles in traffic images. Left column: 8 points in 2 frames; right column: model projection using recovered motion parameters.*

between the image and the projections of a vehicle model. The projection pose for the reference frame is determined by eye, and the poses for the other four frames are then derived from the corresponding motion parameters delivered by the LLS-Biased algorithm. Two model projections are shown in the right column of Fig.6. The fit between the image and the model is very close in all frames, indicating good accuracy of the computed motion parameters.

The accuracy of the recovered structure may also be assessed qualitatively by projecting the point coordinates onto the X-Y, X-Z, and Y-Z planes of a vehicle-centred coordinate system. The projections are shown in Fig.7, where the grey lines are added to help the delineation of the vehicle shape. It is not difficult to relate the recovered structure to the vehicle. The only obvious oddity appears to be $P_6$ which is in fact one of the hardest points to locate consistently due to the self-occlusion of the near-side of the vehicle (see Fig.6).

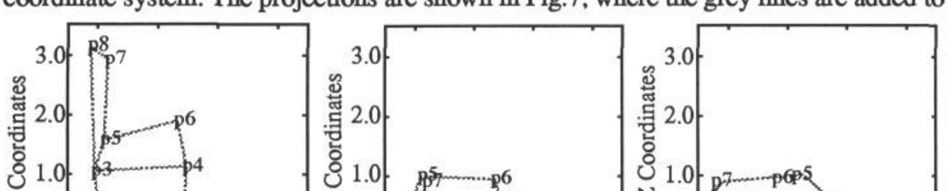

*Figure 7: Projection of the recovered 3-D coordinates of the 8 vehicle points onto the three planes of a vehicle-centred coordinate system.*

## 7 Conclusions

Novel algorithms have been presented in this paper for the computation of 3-D structure and motion parameters of vehicles in traffic images which, under normal conditions, are constrained to move on the ground-plane. The algorithms use point correspondences in multiple frames, and directly exploit the ground-plane constraint. The algorithms first compute the inter-frame rotation angles, then the depths of the points, and finally the inter-frame translations. Computation at each stage is linear.

The algorithms have been tested with both synthetic and routine traffic images. Extensive experimental results have been reported. The results have demonstrated the validity of the algorithms, and have confirmed their satisfactory performance with both synthetic and real image data.

## References

[1] T. S. Huang and A. N. Netravali, Motion and Structure from Feature Correspondences: A Review, *Proc. of IEEE*, vol.82, no.2, 1994, pp.252-268.

[2] T. J. Broida and R. Chellappa, Estimating the Kinematics and Structure of a Rigid Object from a Sequence of Monocular Images, *IEEE Trans. PAMI*, vol.13, 1991, pp.497-513.

[3] C. G. Harris and J. M. Pike, 3D Positional Integration from Image Sequences, *Proc. of Alvey Vision Conf. 1987*, University of Cambridge, 1987, pp.233-236.

[4] T. N. Tan, G. D. Sullivan and K. D. Baker, Structure from Motion Using the Ground Plane Constraint, *Proc. of 2nd European Conf. Computer Vision*, LNCS-588, Springer-Verlag, 1992, pp.277-281.

[5] T. N. Tan, G. D. Sullivan and K. D. Baker, 3D Structure and Motion Estimation from 2D Image Sequences, *Image and Vision Comput.*, vol.11, 1993, pp.203-210.

[6] S. Ullman, *The Interpretation of Visual Motion*, Cambridge, MA: MIT Press, 1979.

[7] B. K. P. Horn, Closed-form Solution of Absolute Orientation Using Quaternions, *J. Opt. Soc. Amer.*, vol.4, 1987, pp.629-642.

[8] X. Lin and Z. Zhu, Detecting Height from Constrained Motion, *Proc. of 3rd ICCV*, Osaka, Japan, 1990, pp.503-506.

[9] T. N. Tan, G. D. Sullivan and K. D. Baker, Pose Determination and Recognition of Vehicles in Traffic Scenes, in *Computer Vision - ECCV'94*, J.-O. Eklundh (Ed.), LNCS-800, Springer-Verlag, May 1994, Sweden, pp.501-506.

[10] J. A. Noble, Finding Corners, *Proc. of 3rd Alvey Vision Conf.*, University of Cambridge, England, 15-17 September 1987, pp.267-274.

[11] T. N. Tan, G. D. Sullivan and K. D. Baker, On Computing The Perspective Transformation Matrix and Camera Parameters, *Proc. of British Machine Vision Conference 1993*, BMVA Press, 1993, pp.125-134.