HAL
open science

A Study towards Real Time Camera Calibration

Ragini Choudhury

To cite this version:

Ragini Choudhury. A Study towards Real Time Camera Calibration. [Technical Report] 2000, pp.41. inria-00590143

HAL Id: inria-00590143
https://inria.hal.science/inria-00590143v1

Submitted on 5 May 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Study towards Real Time Camera Calibration

Preliminary Report

Prepared for the **Project VISTEO**

by

Ragini Choudhury

Date : September 10, 2000

Project VISTEO supervised by : **Edmond Boyer**

Projet MOVI, INRIA Rhone Alpes

ZIRST, 655 avenue de l'Europe

38330, Montbonnot, St. Martin

FRANCE

1 Introduction

This report provides a detailed study of the problem of real time camera calibration. This analysis, based on the study of literature in the area, as well as the experiments carried out on real and synthetic data, is motivated by the requirements of the VISTEO project. VISTEO deals with a fusion of real images and synthetic environments, objects etc in TV video sequences. It thus deals with a challenging and fast growing area in virtual reality research - Augmented reality (AR). AR generates a composite view of the real scene viewed by the user and a virtual scene generated by the computer which augments the real scene with additional information.

The aim of VISTEO is to put a virtual environment around a real TV announcer and to make the announcer interact with this world : moving around virtual objects and handling them. The camera is moving while shooting the announcer in a studio (the background having a uniform known colour). The idea is to overlay a video of the real scene with virtual objects such that the announcer can interact with these objects. In order to merge the real world with the virtual, we will need to reproject the virtual objects into the real scene. Therefore it is essential to determine the parameters of the camera and the position of the announcer in the scene and with respect to the camera. Also the ideal merging between the real and virtual worlds in the augmented video is one in which there is no sway, jitter or drift. Therefore, the parameters of the camera have to be determined in real time as any delays in this would cause a misalignment between the virtual object and the real scene and give the impression that the augmented object lags behind in the scene. Thus the problem of merging the video of a real scene with the virtual objects has two aspects :

- determining how the image has been formed, the position and orientation of the camera. This requires computing the camera parameters, which is called *camera calibration*, in order to be able to reproject a virtual object into a sequence. The estimation of camera parameters has to be carried out in real time to merge the virtual objects into a video sequence.

- determining the position of the announcer with respect to the camera and his(her)

shape, which would involve reconstruction of parts and tracking the position of these parts across the sequence. Determining the position and shape of the announcer would aid in the interaction with the virtual environment and permit the virtual environment to be dynamic.

In this report we propose to do an extensive study of the first aspect of the problem - calibration of the camera in real time. Here we analyze the various aspects of calibration and the different approaches available for calibration and present a comparative study in order to determine the one best suited to the problem of reprojecting the virtual objects onto the real image sequence. Camera parameters can be intrinsic (focal length, distortion, aspect ratio) and extrinsic (pose, the position and orientation with respect to the world). The data available to us is a video sequence of the scene together with targets placed behind the announcer on a wall. We are are given the coordinates of the 3D target points (targets are placed on the wall behind the announcer and hence are coplanar) in a convenient frame and their images as taken by a camera. Using this information, the camera parameters must be determined in a robust manner as any fluctuations in its computation will manifest as a jitter in the visual image. Inaccuracy in calibration will cause the virtual objects to be incorrectly positioned and also lead to the perception that the virtual object is not stationary in the scene. This together with the fact that the computation of the camera parameters has to be carried out in real time, are the constraints with respect to which each of the methods are analyzed.

1.1 Organization of the Report

In this report, we study the problem of camera calibration in real time, analyze the various approaches for the computation of the intrinsic and extrinsic parameters. Although various methods of determining intrinsic parameters have been studied in this report, the emphasis is on real time estimation of extrinsic camera parameters, also called the *pose estimation problem*. The problem of robust pose estimation is a difficult problem due to noise and outliers. Real time requirement makes it even more difficult. In this report we explore various methods of pose computation, conduct experiments on real and synthetic data and finally present an approach where robustness is the key to performance, by

using RANSAC and a fast algebraic method of computing the initial pose. The report is organized into the following sections :

Camera Calibration In this section, we give brief introduction to the camera model used and a discussion of the camera calibration problem in general. This is to put the subsequent in depth theoretical and experimental analysis of the computation of camera parameters in perspective. The problem can be broadly divided into two parts (dealt with in individual sections)

Part 1: Intrinsic Camera Calibration Here we study two approaches for the computation of intrinsic camera parameters. Experiments are carried out in order to compare the values obtained with those supplied by GETRIS.

Part 2 : Extrinsic Camera Calibration : Pose Computation This presents an in-depth study of various approaches for computing the pose, while keeping in mind the issues of robustness and speed. Experiments have been conducted with the aim of determining the pose computation method best suited for our requirements of robustness and real-time.

Conclusions This summarizes the results and discusses their relevance to the VISTEO project. We also indicate the direction which we propose to take subsequently in the project.

2 Camera Calibration : A brief overview

In order to merge the virtual objects with the real scene, it is essential that the virtual and the real camera are perfectly aligned. This makes the robust determination of camera parameters imperative. As the aim of the report is the computation of camera parameters, the basic requirement is a camera model. This models the imaging process which obtains the 2D images of the 3D points and can therefore be used for determining the parameters.

2.1 Camera Model

We model the camera and hence, the imaging process, by a perspective (projective) projection. A perspective projection from the Euclidean 3D space to the image 2D space is represented as

$$m = PM$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

where the 3D and 2D points have been represented in homogeneous coordinates. P is a 3×4 camera projection matrix (11 parameters upto scale) with the decomposition

$$P = K[R|t] \tag{1}$$

where K is a 3×3 non-singular matrix which gives the intrinsic parameters of the camera which are required for any metric measurement and R and t are called the extrinsic parameters of the camera and determine the position and orientation of the camera with some world coordinate frame (Fig. 1).

2.2 Camera Calibration

Camera calibration is the process of determining the intrinsic (focal length, aspect ratio, principal point) and the extrinsic parameters (position and orientation of the camera frame with respect to some world coordinate frame). The information available are the coordinates of the 3D points and their images taken by the camera.

The algorithm for camera calibration consists of two parts

- Compute the matrix P from a set of points with known 3D positions and their measured image positions
- Decompose P into K, R and t

Determining K is called the calibration of the *intrinsic* parameters and determining R and t is known as the calibration of the *extrinsic* parameters or the problem of *Pose (position and orientation) estimation.*

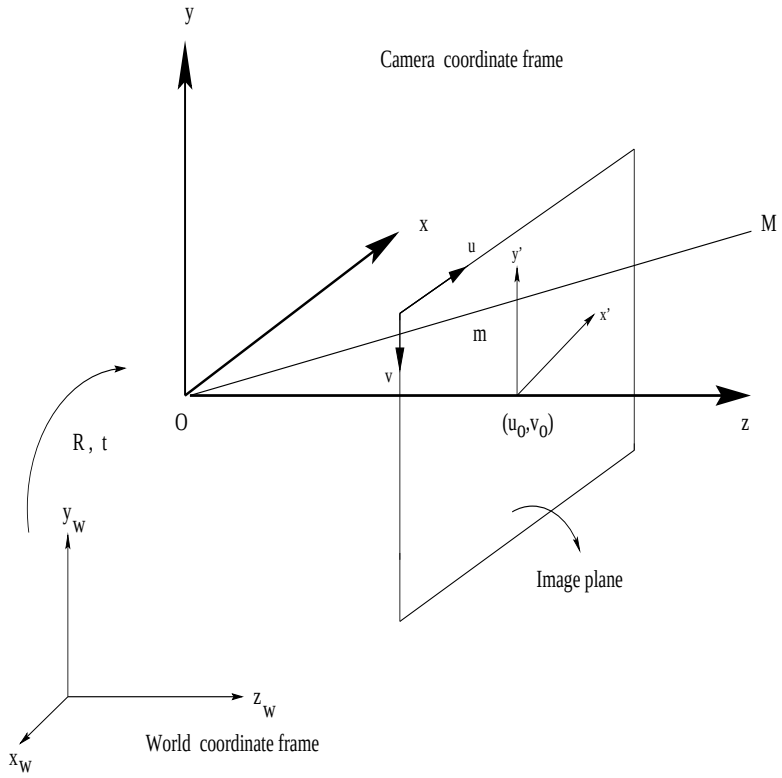


Figure 1: Camera and World Coordinate Frame

We use the correspondences between the 3D points M_i and their 2D images m_i to determine the matrix P. Each correspondence generates two linear equations on the matrix elements of P

$$x_i = \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}\ , \qquad y_i = \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}$$

which gives

$$\begin{aligned} x_i(p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) &= p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14} \\ y_i(p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) &= p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24} \end{aligned}$$

Given ($n \geq 6$) correspondences, a linear solution can be obtained for P from the set of 2 n linear simultaneous equations : $A\widehat{P} = 0$, where $\widehat{P}$ is the 12 vector representation of the matrix P and A is a 2 $n\times$ 12 matrix depending on the 3D and 2D coordinates of the reference points. This system of equations can be solved as a constrained minimization problem in two ways [3] :

Linear methods In this case we solve the constrained minimization problem for n

points

$$min_{\widehat{P}} \ ||A\widehat{P}||^2$$

subject to

$$||R_3|| = 1$$

where R_3 is the third row of R. This problem is a classical minimization problem and a closed form solution exists.

Non linear methods The above linear solution is then used as the starting point for a non-linear minimization of the reprojection error, that is, the difference between the measured image points (u_i, v_i) and the projected point (x_i, y_i):

$$min_{\widehat{P}} \sum_i ||(u_i, v_i) - (x_i, y_i)||^2$$

In this case there is no closed form solution and iterative (numerical) methods must be used.

Having computed the camera matrix, we need to decompose P into K, R and t. The first 3×3 submatrix of P is the product KR of an upper triangular and rotation matrix. This factorization can be obtained using QR decomposition. A closed form solution exists for the decomposition, based on the 5 parameters parameterization of K [3]. This determines K and R. Then

$$t = K^{-1}(p_{14}\ p_{24}\ p_{34})^T$$

Instead of estimating all the parameter of the camera matrix in one minimization problem, in some situations, we may have knowledge of the intrinsic parameters of the camera (as will be our case since the intrinsic parameters will be provided by GETRIS). In this scenario, we formulate a minimization problem just to solve for the extrinsic parameters, that is , the pose.

We now proceed to discuss calibration of intrinsic and extrinsic parameters in a more detailed manner but specific to the requirements of VISTEO.

3 Intrinsic Camera Calibration

Intrinsic camera calibration is the process of determining the parameters of the matrix K, that is, the intrinsic parameters. K is a 3×3 non-singular , upper triangular matrix which has the form

$$K = \begin{pmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

where $\alpha_u = k_u f$ and $\alpha_v = k_v f$. K provides the transformation between an image point and a ray in Euclidean 3-space. There are five parameters:

1. f is the *focal length*
2. (u_0, v_0) are the coordinates of the *principal point*, that is, the point where the optic axis intersects the image plane
3. s is the *skew parameter* which we take to be zero
4. the *scaling* in the x and y directions, α_u and α_v. The *aspect ratio* is $\alpha_u/\alpha_v = k_u/k_v$. Here k_u, k_v are the inverses of the dimensions of a pixel.

As can be seen α_u, α_v, u_0 and v_0 do not depend on the position and orientation of the camera in space and they are therefore called *intrinsic parameters*.

Along with the internal parameters, we need to consider the *distortion*, especially radial distortion. Distortion concerns the position of image points in the image plane. Therefore the expression for images of the points are

$$\begin{aligned} u_d &= u - \delta_u(u_d, v_d) \\ v_d &= v - \delta_v(u_d, v_d) \end{aligned}$$

where (u, v) are the ideal (non-observable distortion free) pixel image coordinates and (u_d, v_d) are the corresponding real observed image coordinates with distortion. The amount of error along each coordinate, depends on the position of the point. In order to correct distortion, various sources of distortion need to be analyzed and modeled. Distortion could be caused by imperfect lens shape and manifests itself by radial positional

error only, or it can be caused by improper lens and camera assembly and causes radial and tangential errors [22]. Both radial and tangential distortion are modeled using an infinite series.

$$\begin{aligned}\delta_u(u_d, v_d) &= u_d(k_1 r^2 + k_2 r^4 + \cdots) \\ \delta_v(u_d, v_d) &= v_d(k_1 r^2 + k_2 r^4 + \cdots)\end{aligned}$$

where $r = \sqrt{u_d^2 + v_d^2}$ and k_1 and k_2 are the coefficients of distortion. But a study of literature [22, 23] shows that the distortion function is usually dominated by the radial components and only one term of the sequence needs to be considered. the equations are therefore

$$\begin{aligned}\delta_u(u_d, v_d) &= u_d(k_1 r^2 + k_2 r^4) \\ \delta_v(u_d, v_d) &= v_d(k_1 r^2 + k_2 r^4)\end{aligned}$$

Therefore, in estimating the distortion, k_1, k_2 need to be determined.

For the purposes of VISTEO, the intrinsic parameters will be provided by the manufacturer and GETRIS. But in view of the importance of the accuracy and robustness of the values of these parameters to VISTEO, it becomes essential to verify these values. Also we need to determine the distortion in the images. We carry out a sequence of experiments which are aimed at identifying any discrepancy between the values provided and those computed from the data. The data available are non-coplanar 3D points and their images. Two approaches have been used to compute the intrinsic parameters.

3.1 Minimization approach

This is the non-linear approach to camera calibration discussed above in section 2.2. The initial solution has been computed using the Linear method of Faugeras and Toscani [4]. Then the optimization is done over all the camera parameters using bundle adjustment.

3.2 Tsai's approach

In this approach the camera parameters are determined in two stages. It involves a direct solution for most of the calibration parameters and some iterative solution is used for

the others [22, 21, 11]. A radial alignment constraint is used to derive a closed form solution for the external parameters, the effective focal length of the camera, the (X, Y) coordinates and scale. Then an iterative scheme is used to estimate three parameters : the depth component in the translation vector (external parameter), the effective focal length and the radial distortion coefficient. The image coordinates of the principal point are also iteratively determined.

Tsai's method also uses the linear method described above in order to get the initial solution which is refined using non-linear optimization. But other than the extrinsic and intrinsic parameters and the distortion, it also has 2 parameters for the distortion and 6 fixed intrinsic parameters. These are Ncx, the number of sensor elements in camera's x direction (in sels),Nfx, the number of pixels in frame grabber's x direction (in pixels),d_x, the X dimension of camera's sensor element (in mm/sel),d_y, the Y dimension of the camera's sensor element (in mm/sel),dpx, the effective X dimension of pixel in frame grabber (in mm/pixel) and dpy, the effective Y dimension of pixel in frame grabber (in mm/pixel). These are supplied by the manufacturer. The distortion function is taken upto the second order terms. Tsai's algorithm fails if the origin of the world coordinate system is near the origin of the camera's field of view or near the y-axis of the camera coordinate system. This ensures that the translation along the y-axis is not exactly zero which is an explicit requirement of the algorithm.

Thus in general the parameters used are $\alpha_u, \alpha_v, s, u_0, v_0$ with the notations having the same meaning as explained before. Tsai's algorithms uses the following parameters $\frac{s_x Nfx}{d_x Ncx}, \frac{Nfy}{d_y ncy}, 0, C_x, C_y$, where s_x is the uncertainty scale factor for x and (C_x, C_y) are the image coordinates for the origin in the image plane.

With this brief discussion about the algorithms, we are now in a position to describe the experiments which have been carried out with these algorithms.

3.3 Experiments related to Intrinsic parameters

In these experiments, the intrinsic parameters have been estimated using image information alone and parameters supplied by the manufacturer. The experiments have been carried out with the aim of studying which parameters can be estimated and which pa-

rameters obtained by using GETRIS's software can be used. Also we study the effect of change in the various parameters, especially the focal length and the image center.

The experiments carried out can be broadly categorized into

- Set A : Computation of $u_0, v_0, \alpha_u, \alpha_v, f$ and d, the distortion parameter, and comparing it to the ones obtained with GETRIS's calibration procedure. This would help in estimating the stability of the parameters with respect to the calibration method. This is repeated with several values of f.

- Set B : u_0, v_0 are set to the centre of the image, k_u, k_v are set to the values given by the manufacturer and f and the distortion are estimated. This will aid in the analysis of the influence of $u_0, v_0, \alpha_u, \alpha_v$. This is also repeated for several values of the focal length f.

3.3.1 Experimental setup

The experiments have been carried out using images taken by a camera provided by GETRIS. The range of the zoom was 8 mm - 128 mm. The calibration object was the one available in the MOVI team (Fig. 2). The images are in Fig. 5.

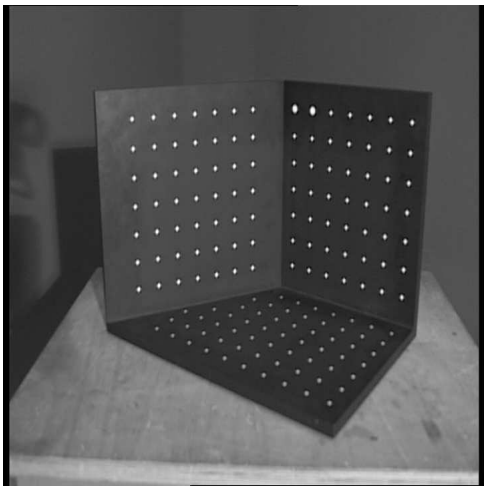


Figure 2: Image of the calibration object

- the calibration grid was placed on a movable trolley

- the camera was fixed at the expected focal lengths of 8mm, 28mm, 38mm, 50mm , 68 mm, 88 mm, 108mm ,120 mm (these were readings read off the camera).Using a software available at GETRIS, the Field of View (FOV) α was obtained at each position and the focal length was computed using the formula $f = \frac{4.4}{tan(\alpha/2)}$.

- for each focal length, three images of the grid were taken by translating and/or rotating the camera/grid but keeping the focal length fixed

- the hard constraints supplied by the manufacturer are

$$k_u = 720/8.8, \quad k_v = 576/6.6, \quad aspect\ ratio = k_u/k_v = .9375$$

  The size of the images are 720×576.

Note : The values for the parameters used are specific to these set of experiments and images provided by GETRIS. They by no means limit the scope of the experiments or the applicability of the results. These images form the input for the experiments on camera parameters.

3.3.2 Set A experiments

The aim of these experiments was the computation of $u_0,\ v_0,\ \alpha_u,\ \alpha_v,\ f$ and the distortion parameters and compare the values obtained with GETRIS's calibration procedure.

Stepwise Methodology

- Target points are extracted from the image of the calibration grid using a software available with the MOVI team . The software returns the 3D coordinates of the targets and their 2D images. Therefore the data available are the non-coplanar 3D points (targets on all 3 planes of the calibration grid) and their 2D images.

- This data is used as an input for the two routines handling the two approaches described above :

  - *Minimization approach* : Programs for this are available with the MOVI team. it uses data of non-coplanar 3D points and their 2D images. The data file is expected to be in the format

" no. of points

2D point coordinates 3D point coordinates "

The program returns the values of the intrinsic parameters and the extrinsic parameters. The rotation is parameterized in the form of Euler angles and values of these angles are returned by the program.

– *Tsai's approach* : the software package containing routines for calibrating Tsai's perspective projection camera model is available at

`http://www.cs.cmu.edu/afs/cs.cmu.edu/user/rgw/www/TsaiCode.html`

We run the program which handles non-coplanar calibration with full optimization. The values of the internal camera parameters provided by the manufacturer need to be provided to the program. Since we are using the calibration routines using non-coplanar points and full optimization, atleast 11 points are required. The data files in to be provided in the format " 3D point coordinates 2D point coordinates".

The programs compute the internal camera parameters, the pose and identifies the distortion. We use only the intrinsic camera parameters from these programs and that too for the purposes of comparison. The pose is to be computed using the information of target points placed on the wall behind the announcer.

Results and Discussion

We now discuss the results of the experiments with respect to the two approaches

Minimization approach A study of the results (Table 1) indicate that

1. There is a difference between the focal length computed using GETRIS's software (GETRIS's calibration setup) and that obtained by Minimization approach, but the difference is consistent throughout the result. The difference is initially (for small focal lengths) small but becomes more significant for the higher focal lengths.

2. The scale factor of GETRIS is .9375. Those obtained by the minimization approach are different but the difference is stable ($\approx .01$).

Actual focal (mm)	no. of points	focal $\frac{\alpha_u}{k_u}$	focal $\frac{\alpha_v}{k_v}$	aspect ratio	distortion d	(u0 , v0)
9.09	157	10.453546	10.634181	0.92157539	-.127344	(350.080642,290.79368)
	157	10.436774	10.617129	.92157459	-.131908	(351.638637,291.836838)
	156	10.449862	10.628494	.92174357	-.138625	(350.905759,291.686256)
27.671	141	31.408837	31.946881	.92171081	.765077	(344.035227,297.277805)
	144	29.136797	29.538879	.92473879	.160415	(353.130310,399.079618)
	138	31.793775	32.339099	.92169124	.87628	(350.113158,290.505863)
37.0645	123	38.456047	39.027016	.9237843	.115193	(303.795986,430.458575)
	113	38.352247	38.927164	.92365404	.214197	(370.106813,399.348894)
	112	39.028129	39.58299	.92458022	.196375	(358.976684,428.960297)
51.1664	96	54.289905	55.074174	.9241498	.172617	(353.913595,496.615025)
	90	58.915434	59.917759	.92181718	1.72119	(343.537307,288.909719)
	93	53.364127	54.154398	.92382633	.139599	(327.767148,486.781792)
70.0335	81	75.979239	77.212325	0.92252807	0.166987	(224.952364,487.490953)
	82	74.379576	75.480375	.92382758	.0500917	(245.922779,641.651845)
	76	74.311086	75.451406	.92333128	.143038	(274.9996,546.119796)
86.85728	67	89.994773	91.368377	0.92340592	0.0324411	(369.328876,671.306195)
	74	92.197287	93.62223	0.92323112	0.0119122	(245.588516,661.963359)
	74	90.521081	91.840893	0.92402752	-0.0272115	(224.887960,838.580233)
106.3112	72	110.63205	112.21935	0.92423938	-0.0266908	(196.405669,1048.955206)
	76	110.43377	112.11838	0.92341381	-0.0377497	(236.869260,896.487544)
	69	112.39075	113.98563	0.92438261	-0.0139149	(236.057844,1044.046293)
139.2362	63	140.72633	142.98812	0.92263252	-0.0374617	(195.589302, 914.310365)
	60	140.86498	143.09771	0.92287239	-0.0133337	(81.401862,997.399929)
	49	136.84955	139.07171	0.92252013	-0.0262702	(121.487628,920.866803)

Table 1: Intrinsic parameters computed using the Minimization approach

3. The position of (u_0, v_0) alter a lot, even for a small focal length

Tsai's approach An analysis of the results (Table 2) shows that

1. The focal length varies a lot from the one provided by GETRIS. The difference is more pronounced than in the case of the results obtained by the minimization approach.
2. the scale factor is stable
3. (u_0, v_0) is very stable compared to the values obtained by the minimization approach

3.4 Set B Experiments

As a second stage of experiments, the principle point $(u_0,\ v_0)$ was fixed to the centre of the image for the images provided by GETRIS, that is, at (360,288) and the intrinsic parameters were computed using the minimization approach. This means that the prin-

Actual focal (mm)	no. of pts	focal	aspect ratio	distortion	(u0 , v0)
9.09	157	10.615935	.983063	1.217077e-3	351.526762,291.537962
	157	10.620707	.983019	1.214263e-3	352.025837,291.575888
	156	10.624311	.983230	1.317391e-3	349.691864,291.251038
27.671	141	32.060733	.983146	-7.250604e-4	346.024104,295.443012
	144	32.288181	.983197	-7.84317e-4	350.147675,289.783644
	138	32.342848	.983158	-7.83252e-4	350.033061,290.826635
37.0645	123	42.937880	.983282	-6.011083e-4	349.065844,290.865166
	113	42.985611	.983385	-5.967269e-4	348.113128,291.894832
	112	42.944172	.983381	-6.093481e-4	348.513538,291.106965
51.1664	96	59.406894	.983302	-4.433291e-4	346.348313,291.783740
	90	59.640693	.983250	-4.495042e-4	344.902035,291.130370
	93	59.555623	.983223	-4.356704e-4	343.902323,290.403266
70.0335	81	81.861506	0.983010	-3.664853e-04	342.853637, 286.666423
	82	82.219751	.983022	-3.796135e-4	341.706587,288.127381
	76	82.192942	.983116	-3.720118e-4	337.256934,283.699561
86.85728	67	102.031786	0.983106	-3.294327e-04	343.281280,291.654863
	74	101.773248	0.983204	-3.075101e-04	343.608198,291.400508
	74	101.790938	0.983146	-3.296992e-04	343.305832, 290.227323
106.3112	72	123.275997	0.983130	-2.737752e-04	341.779387, 298.155421
	76	124.043238	0.983138	-2.889017e-04	346.843929,293.717240
	69	123.365722	0.983128	-2.776888e-04	343.644272,297.018431
139.2362	63	159.176888	0.982973	-2.117128e-04	330.130200, 286.262602
	60	158.592867	0.983035	-2.100774e-04	345.915146,297.973628
	49	157.816615	0.982914	-1.803471e-04	329.160844,309.323857

Table 2: Intrinsic parameters using Tsai's calibration software

cipal point was fixed and minimization was carried out over the other parameters. These results are shown in Table 3 (with distortion) and Table 4 (without distortion).

3.5 Discussion

The results of the Set A experiments carried out using both the methods (Table 1 and Table 2), indicate that the focal length obtained by GETRIS's calibration software is different from the one obtained using the experiments with the calibration grid. The difference is not very high in the case of the lower focal lengths but it increases for the higher focal lengths. An alternative could be pre-calibrating the camera. But in order to calibrate using one of the approaches described above, we need the information of non-coplanar 3D points. This was possible using a calibration object, but in the case of the video sequence, we will be provided the coordinates of the target points placed on the wall behind the announcer. These are coplanar and will not suffice for calibration. Also calibration, for accuracy, should be done using information available from the scene

Actual focal (mm)	no. of pts	focal $\frac{\alpha_u}{k_u}$	$\frac{\alpha_v}{k_v}$	aspect ratio	distortion
9.09	157	10.495983	10.678435	0.92148183	-0.101761
	157	10.533612	10.71779	0.92138963	-0.102575
	156	10.46015	10.637724	0.92185041	-0.113038
27.671	141	31.309524	31.851189	0.92155678	0.803028
	144	31.576056	32.11727	0.91973324	0.8476
	138	29.02876	32.47855	0.9217132	0.915027
37.0645	123	42.324608	43.046324	0.92178185	1.23629
	113	41.871157	42.584598	0.92179361	1.09121
	112	41.867588	42.586176	0.92168087	1.14998
51.1664	96	58.197688	59.205345	0.92154404	1.60249
	90	58.471786	59.478519	0.92163189	1.59325
	93	58.305713	59.3087	0.92164567	1.58495
70.0335	81	80.631657	82.017481	0.92165935	2.63092
	82	79.94593	81.338625	0.92144795	2.64387
	76	80.656783	82.057746	0.92149416	2.74837
86.85728	67	99.00175	100.71654	0.9215382	3.13999
	74	99.489516	101.19711	0.92168072	3.17348
	74	99.346069	101.05608	0.92163615	3.3655
106.3112	*very*	*high*	distortion		
139.2362	*very*	*high*	distortion		

Table 3: Intrinsic parameters computed using the Minimization approach when the principal point has been fixed to the centre of the image. Distortion is computed

and not using a calibration object.

Also the results of the experiments carried out by fixing the principal point give good results for the other parameters except in the cases of higher focal lengths. In this case, Table 1 shows that the position of the principal point varies a lot for the higher focal lengths. The values which give a problem are close to the maximum value. But the cases of the maximum zoom may not occur in real life scenes. This is because in these cases very little of the wall behind the announcer will be seen. Therefore, few or no target points will be available and such a sequence will be a problem for computing the pose. (As will be explained in the next section, we require the coordinates of the targets placed on the wall behind the announcer and its images to be able to compute the pose).

Therefore, in view of the experimental results, we work with the intrinsic parameters provided by GETRIS and proceed to compute the pose. We will revert to precalibration only if the pose gives very bad results on reprojecting the virtual objects onto the real scene. We now proceed to study the more fundamental problem of pose computation.

Actual focal (mm)	no. of pts	focal $\frac{\alpha_u}{k_u}$	focal $\frac{\alpha_v}{k_v}$	aspect ratio
9.09	157	10.779153	10.981949	0.92018782
	157	10.756956	10.959774	0.92015095
	156	10.661797	10.853197	0.92096687
27.671	141	29.56964	30.045498	0.92265193
	144	29.935351	30.404828	0.92302418
	138	30.053801	30.120373	0.92285414
37.0645	123	40.007543	40.659779	0.92246127
	113	40.1068	40.742811	0.92286526
	112	40.187191	37.675492	0.92211061
51.1664	96	56.039654	52.537176	0.92169667
	90	55.344028	56.263384	0.92218105
	93	55.421757	56.351254	0.92203621
70.0335	81	78.101677	79.392962	0.92225205
	82	77.178087	78.448099	0.92232263
	76	76.663988	77.939798	0.92215391
86.85728	67	92.093794	93.670073	0.92172376
	58	93.914256	95.454121	0.92237626
	74	does	not	converge
106.3112	72	118.48623	120.49986	0.92183374
	76	116.51489	118.46275	0.92208485
	69	does	not	converge
139.2362	59	150.21497	152.74126	0.9219941
	60	148.3146	150.86262	0.92166594
	52	149.25372	151.80487	0.92174484

Table 4: Intrinsic parameters computed using the minimization approach, by fixing the principal point to the center of the image. Distortion is not estimated

4 Calibration of Extrinsic Parameters : Pose computation

Given a set of correspondences between the reference points in space and image points, pose estimation consists of determining the position and orientation of the calibrated camera (one whose intrinsic parameters have been computed) with respect to the known reference points. This problem is also known as the *Perspective n- Point problem, exterior (or extrinsic) camera calibration problem, or camera location or orientation problem.*This can be stated more formally as : Given a set of points that are described in an object centered frame, given the projections of these points onto an image, and given a projection model and the parameters of this model, determine the rigid transformation (rotation and translation) between the object centered frame and the camera centered frame. Therefore we need to determine the $[R\ t]$ matrix of the projection model (give in equation 1). Here

1. t : describes the translation which transforms the origin of the world coordinate system to the camera coordinate system. It has three unknown parameters.

2. R (Rotation matrix) : This describes the rotation that aligns the axes of the world coordinate system with those of the camera coordinate system. This also has three unknown parameters. (see Fig. 1)

The computation of pose, is in general, a difficult problem which has been highly researched in literature and various approaches have been suggested [3, 19, 18, 20, 16, 17, 7, 14]. There are three crucial choices associated with the problem. The choice of :

- the mathematical representation of the problem
- the error function to be minimized
- the optimization method to be used

The approaches in literature to solve the pose computation problem, fall into two categories :

Closed-form solution These are devised in the case when the number of point correspondences are limited. Such solutions exist for three points (which is not unique) (Fischler and Bolles [6], Haralick et al [8]), 4 coplanar points (Hung et al 1985), 4 points in general position (Horaud et al [10], Holt and Netravali [9]).

Numerical solutions Iterative numerical solutions have been used when the number of point correspondences is more than 4. (Haralick et al [8] , Lowe [12], Yuan [24]). These approaches are robust but they converge towards the correct solution only if a good initial estimate has been provided.

A combination of the above methods could also be applied. That is, the non-linear minimization in the iterative solutions are initialized with the solution obtained from a linear method. But this may take a long time to converge if the solution obtained by the linear method is far away from the actual pose.

As explained earlier, the problem of pose computation assumes importance for VISTEO as we need the camera parameters in order to be able to reproject the virtual objects

into the real scene. In this report, we concentrate on approaches which are relevant with respect to VISTEO. As it is important in our context to have a low reprojection error and pose computed in real time, the approach that we employ should be *fast* and *robust* . In order to compute pose, the information required are coordinates of 3D points and their 2D images, taken by the camera whose pose is desired. In VISTEO, this information comes in the form of 3D coordinates of targets placed on the wall behind the announcer, and their image taken by the camera. This implies that the 3D points will be coplanar. Therefore, we will study two methods of pose computation which use coplanar points :

- Algebraic Perspective-3-Point Pose estimation [8, 6]
- Iterative pose computation methods [10]

Both these methods require the knowledge of 4 coplanar points and their images, to be able to compute the pose. So next, we will formulate a method of computing the pose using this as the initial solution and all the points whose 3D coordinates and 2D images are available. This would give the pose for the first frame. For every subsequent frame in the video sequence, the pose can be computed via a process of updation, based on the pose computed in the previous frame, by formulating this as a minimization problem. We now proceed to discuss each of these aspects in greater detail.

4.1 Pose computation using coplanar points

Here we briefly outline two approaches which use the 3D coordinates and its 2D image of 4 coplanar points.

4.1.1 Perspective-3-Point Pose Computation method

This method comes under the class of approaches used to solve the classical PnP (Perspective-n-Point) problem [16, 1]. This problem can be stated more formally as : Given a set of n 3D points whose coordinates are known, their images and the angle subtended by every pair of points at the center of perspectivity, to find the lengths of the line segments joining the center of perspectivity to the 3D points. This helps to determine the position of the camera. In our case, we are considering the P3P problem, (n = 3), which requires

that we determine the lengths of the three legs of the tetrahedron, given the base dimensions and the face angles (see Figure 3). The closed-form solutions for 3 points have been

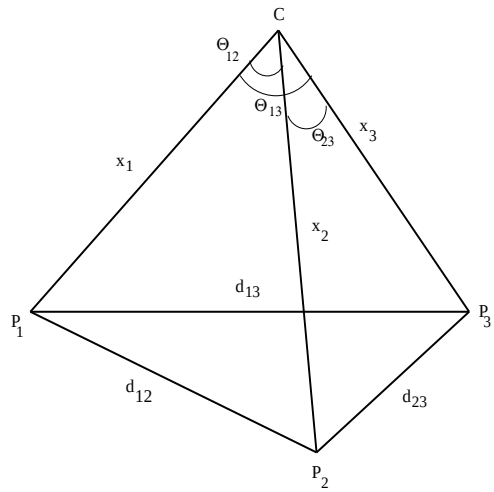


Figure 3: P3P problem formulation : The lengths $d_{ij}, i, j = 1, 2, 3$ and the angles $\theta_{ij}, i, j = 1, 2, 3$ are known, the problem is to determine the distance $x_i, i = 1, 2, 3$

known for a long time with several approaches having been developed in vision. One such approach can be found in Fischler and Bolles [6], and a detailed numerical analysis of the different algorithms was reported in Haralick et al [8]. We will describe an algebraic solution to the P3P problem in detail.

For n point correspondences $P_i \leftrightarrow u_i$ for $i = 1, 2 \cdots n$ between 3D reference points P_i in space and image points u_i, each pair of point correspondences $P_i \leftrightarrow u_i$ and $P_j \leftrightarrow u_j$ gives the following constraint on the unknown distances $x_i = ||P_i - C||$ and $x_j = ||P_j - C||$ of the points P_i and P_j to the perspective camera center C (see Fig. 3):

$$d_{ij}^2 = x_i^2 + x_j^2 - 2x_i x_j cos(\theta_{ij}) \tag{2}$$

where $d_{ij} = ||P_i - P_j||$ is the known distance between the i-th and j-th reference points and $\theta_{ij} = \angle u_i c u_j$ is the view angle subtended at the perspective center by the i-th and the j-th points, which can be measured from images for which the intrinsic parameters, that is K , is known. Given an image point $u = (x\ y\ 1)^t$, the direction vector n of the viewing line of the image point is given as $n = K^{-1}u$. The angle θ_{ij} between the viewing lines of a pair of image points u_i and u_j is given by

$$cos(\theta_{ij}) = \frac{n_i n_j}{||n_i||||n_j||} = \frac{u_i^t C u_i}{(u_i^t C u_i)^{1/2}(u_j^t C u_j)^{1/2}} \tag{3}$$

where $C = (KK^t)^{-1}$. Now the constraint 2 is quadratic in unknown depths and can be rewritten as

$$f_{ij}(x_i, x_j) \quad = \quad x_i^2 + x_j^2 - 2x_i x_j cos\theta_{ij} - d_{ij}^2 \tag{4}$$

For the case of $n = 3$ the following polynomial system is obtained

$$\begin{aligned} f_{12}(x_1, x_2) &= 0 \\ f_{13}(x_1, x_3) &= 0 \\ f_{23}(x_2, x_3) &= 0 \end{aligned}$$

for the three unknown distances x_1, x_2 and x_3. The system has a Bezout bound of 8 $= 2 \times 2 \times 2$ solutions. Using classical Sylvester resultants to eliminate x_3 between $f_{13}(x_1, x_3)$ and $f_{13}(x_1, x_3)$ to get a polynomial $h(x_1, x_2)$, then further eliminating x_2 between $f_{12}(x_1, x_2)$ and $h(x_1, x_2)$, gives an 8th degree polynomial in one variable x_1 with only even terms, i.e. a 4th degree polynomial in $x = x_1^2$:

$$g(x) \quad = \quad a_5 x^4 + a_4 x^3 + a_3 x^2 + a_2 x + a_1 = 0 \tag{5}$$

This has at most four solutions for x and can be solved in closed form. As x_i is positive, $x_1 = \sqrt{x}$. Then x_2 and x_3 are uniquely determined from x_1.

Due to multiplicity of solutions of this 3-point algorithm, in practice, we need a 4-th point for a unique solution. This is true only if the entire configuration together with the optical center does not sit on a critical surface (in this case a cylinder). One straightforward approach is to take subsets of 3 of the 4 points, solve the 4th degree polynomial equation for each subset and finally find the common solution.

The image coordinates and recovered depths give complete estimates $\widetilde{P_i}$ of the 3D coordinates of the reference points in camera-centered coordinates. There remains the determination of a similarity transformation between two sets of 3D points $\widetilde{P_i} \leftrightarrow P_i$. For this, we first remove the effect of translation by subtracting the mean from the data. Then the best rotation in the least squares sense can be found in closed-form using quaternions. The determination of the translation and scale immediately follow from the rotation [3].

4.1.2 Iterative Pose computation

These methods iteratively improve the pose computed with a weak perspective (Dementhon and Davis [2, 13]) or a para-perspective (Horaud et al [10]) model to converge, at the limit, to a pose computed with a perspective camera model. The perspective projection is modeled by a projective transformation mapping the 3D projective space to the 2D projective plane. Weak perspective is just an affine zero-order approximation of full perspective. Paraperspective is the first order approximation of the full perspective (Fig. 4). These methods start with computing the pose of an object using weak perspective/

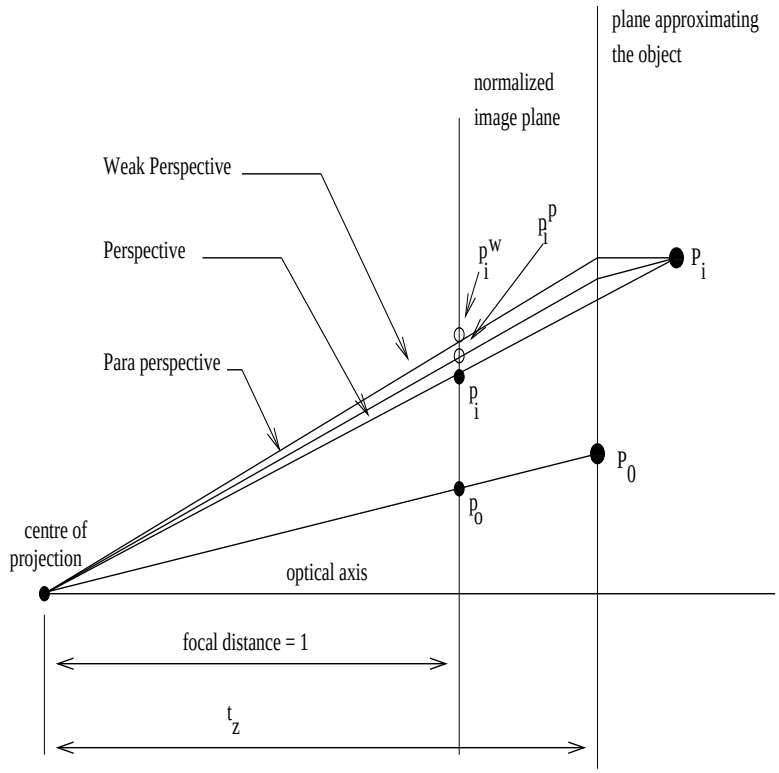


Figure 4: This figure shows p_0, p_i- the perspective projections of two object points, P_0 and P_i, and p_i^w, p_i^p - the weak and paraperspective projections of P_i. The quality of the paraperspective projection depends on the angle between the line of projection of P_i and the line of projection of P_0. The quality of the weak perspective projection depends on the angle between the line of projection of P_i and the optical axis

para-perspective and after a few iterations converges towards the pose estimated under perspective. But these methods have limited applicability to the situations when the

weak-perspective and paraperspective projections are valid. For the purpose of comparison, we use the iterative pose computation method of Horaud et al [10]. Although the paper, proposes methods for computing pose using non-coplanar points and coplanar points, we use the algorithm given for coplanar points alone.

4.2 Robust method of pose computation in each frame

We now have a initial pose estimate , computed with 4 points , using one of the methods described above. The goal is to develop a robust method of pose computation using coplanar points, which is fast and reliable. Thus the two criteria under scrutiny are :

- time required for computation : speed
- reprojection error : Suppose we are given N 3D points $M_i, i = 1, \cdots N$ and their images $m_i, i = 1 \cdots N$. Let K be the known matrix of intrinsic camera parameters. Let POSE denote the pose computed by one of the computation approaches. Then the camera matrix is $K * POSE$. Let $m'_i, i = 1 \cdots N$ be the image of the 3D points obtained using this camera, that is, the reprojected points . Then

  $$Reprojection\ \ Error = \Sigma_{i=1}^{N} ||m_i - m'_i||^2$$

  is the difference between the actual image and the reprojected image. The best pose is the one which gives the least reprojection error.

We have the images of a large number of target points. In order to compute the pose using all the points, we use a Least Squares formulation, for which the pose computed using the four points is the initial solution. Thus, we need to determine the best set of four points which would give a good initial estimate. Also the method would need to avoid degenerate cases like when the 3D points are collinear and the cases in which the pose computation is ambiguous. This occurs when the projection center is coplanar with the reference points or the reference points and the optical centre lie on a circular cylinder. Such points would be termed as outliers and ideally we should be able to eliminate them in the initial stages of pose computation. It is usually assumed that the standard least squares framework is sufficient to deal with outliers (these are data which

do not agree with the fitted model). But these classical techniques optimize (according to a specified objective function) the fit of a functional description (model) to all of the presented data. These techniques have no inherent mechanism for detecting and rejecting gross errors. But outliers can distort a fitting process to such an extent that the final result becomes arbitrary. In such circumstances, it is essential to use robust parameter estimation methods, to maintain the precision of the pose computed. We use a robust technique based on RANSAC to obtain a good estimation of pose, using all points after we have obtained an initial estimate using 4 coplanar points.

4.2.1 RANSAC

Random Sample Consensus (RANSAC) (Fischler and Bolles [6]) is a paradigm for robust parameter estimation. The idea is to find through random sampling of a minimal subset of data, the parameter set which is consistent with a subset of data as large as possible. The consistency check requires the user to supply a threshold on the errors, which reflects the priori knowledge of the precision of the expected estimation. RANSAC proceeds in a manner opposite to conventional fitting techniques. Instead of using as much data as possible to get the initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small an initial data set as feasible and enlarges this set with consistent data when possible. Herein lies the key to robustness.

The RANSAC paradigm, in the context of the pose estimation problem, is more formally stated as :

Inputs :

1. n : minimum number of data points required to instantiate the model , in our case, we need 4 coplanar points to compute the pose

2. S : set of all points which are to be used to compute the pose.
   no. of points in S = #(S) $\geq$ n

3. e : error tolerance used to determine whether or not a point is compatible with a model : this will be a threshold on the reprojection error

4. t : threshold (no. of compatible points used to imply that a correct pose has been found), percentage of outliers acceptable

Stepwise methodology :

1. Determine number of trials required (say n_{trials})

2. Randomly select a subset S_1 of n = 4 data points from S and instantiate the model, that is, compute the pose,$Pose_1$ (say) (The only check required at this stage are that the points are non-collinear in 3D)

3. Use $Pose_1$ and K (already obtained) to compute the reprojection error with other points in S and obtain a set $S_1^* \subseteq S$ such that these points are within some error tolerance e. S_1^* is called the *Consensus set* of S_1. The points lying in the consensus set are called *inliers* and those outside are called *outliers*.

4. If $\#(S_1^*) > t$ then
   use S_1^* and least squares to compute a better pose $Pose^*$
   else
   if $\#(S_1^*) < t$ then
   randomly select a new subset $S_2 \subseteq S$ and repeat step 1 onwards, till a consensus set of size t has been found. Then compute *pose* and exit.

5. if after n_{trials} a consensus set cannot be found with t or more members then compute pose using largest consensus set but indicate that it cannot be made better and exit

We are now in a position to consolidate the procedure which we use to compute pose. The general outline of the computational procedure is :

1. Compute the pose with 4 points using either of the two methods (we present results using both the methods in order to compare them in terms of time and reprojection error)

   (a) algebraic P3P approach

   (b) iterative approach

This provides the starting solution for the pose.

2. In order to determine the best starting solution, that is, the set of four points which gives the best pose (in terms of reprojection error), we use RANSAC [5]. In order to select the best set of 4 points, we use a threshold on the reprojection error as the criterion.

3. Compute the pose using the 4 points selected by RANSAC. Given the internal parameters of the camera, that is, the matrix K, we can compute the camera matrix, once the pose has been computed. We then categorize points as being an inlier depending on whether it is within a pre-specified error threshold (in our case this is taken as the pixel error permissible in the reprojection, that is , 1-2 pixels). Points which do not satisfy this criteria are called outliers.
   Note: The thresholds chosen in Step 2 and Step 3 are different

4. Compute the pose using all the inliers. For this we model the pose computation problem as a least squares problem. Here we are minimizing the reprojection error. The minimization is carried out using Levenberg-Marquardt method [15]. Also we parameterize the rotation matrix using quaternions [3]. The least squares solution gives a solution for the 4 parameters of the quaternion and 3 of the translation. The pose computed in this step is the starting point for the next RANSAC iteration.

5. The above step is repeated till RANSAC obtains the desired percentage of inliers. (It may be noted that RANSAC has already been provided with an accepted upper limit on the outliers)

This computational procedure has been applied to real and synthetic data. The experiments and their results will be discussed in detail subsequently. By the above procedure we have been able to determine the pose using all the target points in one frame of the video sequence. We now extend this procedure in order to compute pose for the subsequent frames.

Pixel Error in picking inliers	No. of inliers (tot. pt = 105)	Time (in sec)	reprojection error
1	37	2.16	0.165161
2	92	0.73	0.140561
3	105	.24	0.136533
4	105	0.16	0.136533
5	105	0.15	0.136532

Variation in time and reprojection error on varying the number of inliers for RANSAC

No. of points	Time (in sec)	Reprojection error
257	.27	0.0847847
195	.83	0.0991777
105	0.16	0.136533
86	0.14	0.158796
77	0.08	0.163187
41	0.08	.226224

Variation in time and reprojection error depending on the number of points available for computation (by varying the step size of the synthetic grid), pixel error = 4

Time taken to compute initial solution using four points : 0.04sec

Table 5: Pose computed using the algebraic P3P approach to obtain the initial estimate of pose

4.3 Robust approach to pose computation for an image sequence

In order to compute the pose for the subsequent frames of the sequence, the obvious method will be to repeatedly use the above procedure for each frame. But the pose will not differ too much between frames. As such we will be unnecessarily losing time by using the above procedure repeatedly. Thus we compute pose for all the subsequent frames by a process of updation, using the pose computed from the first frame as the initial estimate of pose. This is again carried out by modeling the pose computation as a *least squares problem* and *minimizing the reprojection error*. The next update is therefore the one with the least reprojection error using all the points in the subsequent frame.

4.4 Experiments

We will first describe the experiments carried out to validate the pose computation procedure for each frame. This will be discussed with the help of synthetic and real images. This will be followed by a discussion on the experiments carried out on the synthetic sequences.

Pixel Error in picking inliers	No. of inliers (tot. pt = 105)	Time (in sec)	reprojection error
1	does	not	converge
2	does	not	converge
3	105	1.73	0.136533
4	105	0.72	0.136533
5	105	0.26	0.136533

Variation in time and reprojection error on varying the number of inliers for RANSAC

No. of points	Time (in sec)	Reprojection error
257	1.86	0.0847847
195	.88	0.0991801
105	0.72	0.136533
88	0.4	0.156909
77	0.24	0.163185
53	5.73	does not converge
46	4.56	does not converge

Variation in time and reprojection error depending on the number of points available for computation (by varying the step size of the synthetic grid), pixel error = 4

Time taken to compute initial solution using four points : 0.2sec

Table 6: Pose computed using the Iterative approach to obtain the initial estimate of pose

4.4.1 Synthetic data

Here it is assumed that we have a 3m $\times$ 3m wall on which a grid has been placed. The size of the squares of the grid is varied throughout the experiments. The intersection points of the grid are the target points. We choose the camera as

$$= \begin{bmatrix} 600 & 0 & 256 \\ 0 & 600 & 256 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} .866025 & 0 & -.5 & 50 \\ 0 & 1 & 0 & 60 \\ .5 & 0 & .866025 & 700 \end{bmatrix}$$

(that is a camera which is at an angle of 30 degrees with the y-axis and at a translation of $(50\ 60\ 700)^t$ from the world coordinate frame. The images of the 3D targets are computed using this camera. A maximum of 100-120 targets are assumed. This is just a realistic assumption and by no means a limitation and can be increased or decreased.

One pixel Gaussian noise has been added to the image points. These points have been used to do the following

1. compute pose using RANSAC. The initial pose to RANSAC is provided by both the approaches - algebraic and iterative - pose computed using four points. The results for pose computation when the initial estimation is given by the algebraic

P3P approach are in Table 5 and when the initial estimation is given by the iterative approach is given in Table 6.

2. compute pose by changing the position of the wall on which the grid has been placed (Table 7). We study the pose computed using various positions of the grid with respect to the camera.

Type of motion	Pose	Reprojection error (pixels)	Time (sec)
Original position	$\begin{pmatrix} 0.864421 & 0.00179677 & -0.502776 & 50.039 \\ -1.39956e-05 & 0.999999 & 0.00354963 & 59.8756 \\ 0.50278 & -0.00306132 & 0.864416 & 700 \end{pmatrix}$	.150824	0.6
Translation (Tx Ty Tz) = (30 0 0)	$\begin{pmatrix} 0.864645 & 0.00192331 & -0.502379 & 50.1685 \\ 8.97485e-05 & 0.999992 & 0.00398284 & 59.954 \\ 0.502382 & -0.00348883 & 0.864638 & 700.545 \end{pmatrix}$	0.150582	0.26
Translation (Tx Ty Tz) = (60 0 0)	$\begin{pmatrix} 0.864491 & 0.00138483 & -0.502645 & 50.2107 \\ -0.000101437 & 0.999996 & 0.00258061 & 59.9287 \\ 0.502647 & -0.00217993 & 0.864489 & 699.951 \end{pmatrix}$	0.14682	0.27
Translation (Tx Ty Tz) = (90 0 0)	$\begin{pmatrix} 0.864483 & 0.00138046 & -0.502659 & 50.2566 \\ -0.000109089 & 0.999997 & 0.00255868 & 59.9251 \\ 0.502661 & -0.0021571 & 0.864481 & 699.84 \end{pmatrix}$	0.148732	0.14
Translation (Tx Ty Tz) = (120 0 0)	$\begin{pmatrix} 0.863204 & 0.00132922 & -0.504855 & 50.3557 \\ -0.000175788 & 0.999998 & 0.00233231 & 59.7801 \\ 0.504856 & -0.00192451 & 0.863202 & 698.574 \end{pmatrix}$	0.156924	0.14
Translation (Tx Ty Tz) = (150 0 0)	$\begin{pmatrix} 0.862095 & 0.000445956 & -0.506749 & 50.8397 \\ 0.000868084 & 0.999998 & 0.00235684 & 59.5676 \\ 0.506749 & -0.00247172 & 0.862092 & 698.158 \end{pmatrix}$	0.163672	0.43
Rotation $\Pi/20$, axis —— y-axis	$\begin{pmatrix} 0.864392 & 0.00179629 & -0.502827 & 50.0402 \\ -1.18084e-05 & 0.999999 & 0.00355208 & 59.8749 \\ 0.50283 & -0.00306443 & 0.864386 & 699.983 \end{pmatrix}$	0.150826	0.12
Rotation $\Pi/10$, axis —— y-axis	$\begin{pmatrix} 0.864321 & 0.00179748 & -0.502952 & 50.0454 \\ -1.50487e-05 & 1 & 0.003548 & 59.874 \\ 0.502955 & -0.00305902 & 0.864316 & 699.961 \end{pmatrix}$	0.150816	0.14
Rotation $\Pi/5$, axis —— y-axis	$\begin{pmatrix} 0.628703 & -0.0962833 & 0.771529 & 95.0231 \\ -0.308392 & 0.879936 & 0.361114 & 122.537 \\ -0.713739 & -0.465014 & 0.523579 & 959.723 \end{pmatrix}$	0.874085	0.91
Rotation $\Pi/4$, axis —— y-axis	$\begin{pmatrix} 0.633545 & -0.0882339 & 0.768548 & 92.1868 \\ -0.299327 & 0.888055 & 0.348701 & 117.192 \\ -0.713341 & -0.451003 & 0.536258 & 947.178 \end{pmatrix}$	.766339	0.79
Rotation $\Pi/3$, axis —— y-axis	$\begin{pmatrix} 0.628579 & -0.0814149 & 0.773417 & 88.7298 \\ -0.291241 & 0.897451 & 0.331172 & 108.664 \\ -0.721097 & -0.433437 & 0.540431 & 919.181 \end{pmatrix}$	0.530804	0.28

Table 7: Pose when the grid is transformed and the camera is kept fixed (synthetic data)

The results in Table 5 and Table 6 indicate that the Perspective - 3 - point approach is faster and provides a better pose, in terms of reprojection error. These results indicate that this approach also provides a better initial solution and hence the robust approach (using RANSAC and Levenberg-Marquardt method) takes lesser time and gives a better reprojection error. Therefore we will be using this approach for our purposes in the

VISTEO project. The results of Table 7 indicate that the pose computation procedure can handle changes in the angle between the camera axis and the target plane to a very large extent. Also we have carried out all the synthetic experiments using 100 target points. This number can be reduced or increased, and pose is well computed as long as the points are well spaced out. This can be seen in the second table of Table 5 and Table 6. Here the experiments were carried out by varying the size of the grid. This causes the number of points to decrease or increase. It can be seen from a comparative study of the second table of Table 5 and Table 6 that the algebraic approach can handle the increase/decrease in points better. The reprojection error is maintained low even in the cases of 53 and 46 points, when the iterative approach fails to converge. Also the first table of Table 5 and 6 indicate the threshold which needs to be chosen for the RANSAC. We study various pixel errors which can be adopted. On the basis of this study we use 4 pixel errors to obtain the results of the second table and for subsequent experiments.

4.4.2 Real Images

These experiments were carried out using the images taken at GETRIS (procedure explained before). We use one of the planes (other than the base plane) in order to compute the pose using both the approaches. The results are in Table 8 and Table 9 respectively. The intrinsic parameters here are taken to be the ones supplied by GETRIS.

focal length (mm)	No. of points	Reprojection error (pixels)	Time (sec)
9.09	46	0.166559	0.15
27.671	39	0.0546567	0.12
37.0645	35	0.0555662	0.13
51.1664	24	0.0416007	0.1
70.0335	20	0.0367705	0.07
86.85728	16	0.0325903	0.06
106.3112	16	0.0355522	0.04
139.2362	16	0.0299043	0.04

Table 8: Pose computed using points on one plane of GETRIS images (one image of each focal length used). The K is the one supplied by GETRIS and (u_0, v_0) is the actual centre of the image. The Algebraic P3P approach is used for obtaining the initial estimate.

As in the case of the synthetic images, the reprojection error and the time indicates

focal length (mm)	No. of points	Reprojection error (pixels)	Time (sec)
9.09	46	0.769757	0.32
27.671	39	0.0547382	18.98
37.0645	35	0.055573	20.05
51.1664	24	0.0415743	0.16
70.0335	20	does not	converge
86.85728	16	0.0325672	0.97
106.3112	16	0.035144	0.95
139.2362	16	0.0300639	.97

Table 9: Pose computed using points on one plane of GETRIS images (one image of each focal length used). The K is the one supplied by GETRIS. The Iterative approach is used for obtaining the initial estimate.

that the algebraic approach is better than the iterative approach.

4.4.3 Synthetic sequences

We generate several synthetic sequences in order to develop a procedure for computing pose. this would then be used on a real video sequence. The first frame of the sequence is used to compute the pose with the algebraic method, using the target points. Then the pose for the subsequent frames are computed using the minimization approach, the target points being given. By first carrying out the experiments on the synthetic data we can determine whether the pose computed has acceptable reprojection error and also whether the approach is fast enough. This can then be applied to a real video sequence.

1. $Sequence1_{image1}$ - $Sequence1_{image5}$:

   Initial pose : rotation + translation

   Change in camera position : translation (no rotation)

   The first image is used to compute the initial pose (using the algebraic approach) and all the other images (frames) are used to compute the updates. this is true for all the updations.

   Result : The updations indicate that the camera motion is a translation.

2. $Sequence2_{image1}$ - $Sequence2_{image5}$:

   Initial pose : translation

Change in camera position : translation (no rotation)

Result : The updations indicate that the camera motion is a translation.

3. $Sequence3_{image1}$ - $Sequence3_{image5}$:

   Initial pose : rotation + translation

   Change in camera position : rotation (no translation)

   Result : The updations indicate that the camera motion is a rotation.

Total number of points = 99

Time to compute initial estimate using Algebraic approach : 0.04 sec

RANSAC : computing inliers at each stage

Stage	No. of inliers	Time (sec)
1	6	0.01
2	76	0.01
3	99	0.01

>>> 3 RANSAC iterations

>>>> Compute pose using all inliers : Levenberg-Marquardt method + Least Squares

Stage	No. of inliers	Time (sec)	Reprojection error (pixels)
1	6	0.07	0.3332
2	76	0.21	0.21
3	99	0.24	0.24

Table 10: Time profiling for pose computed using Algebraic P3P approach

Although the time profiling (Table 10) shows that the minimization step (with RANSAC and LM) is most expensive time wise. But results in Table 11 indicate that the method of computing the pose for subsequent frames of the sequence by updating the initial solution through a minimization approach (with Levenberg-Marquardt method) gives the update very fast and a very low reprojection error. This is the approach to be followed in the project.

4.4.4 Experiments : Effect of change in intrinsic parameters on the Pose

In all the experiments carried out on the real images, we use the parameters of GETRIS. But as was found on using the minimization and Tsai's approach to compute the intrinsic parameters,

there is a difference between the parameters provided by GETRIS and those estimated. Also we had found in our Set B experiments that high focal lengths give a problem in estimation when the principal point is fixed to the centre of the image. Therefore,

it becomes important to investigate the effect the change of these parameters has on the pose computed as that is our main problem. In order to do this, we carried out a final set of experiments on semi-synthetic data. We took the parameters provided by GETRIS to be the internal parameters. Points were taken on a 300 $\times$ 300 grid. A suitable pose was chosen so that images of atleast 60 -75 points could be taken. (One pixel noise has been introduced in the images). With this data (focal length between 8 - 140 mm), the pose was estimated at each stage. The results are shown in Table 12, 13 and 14.

Then we randomly changed the intrinsic parameters (focal length by 10 - 30 %, skew by 10 % and the principal point was placed at various position inside and outside the image). These changes were modeled on real data. The intrinsic parameters supplied by GETRIS and those computed by us differ by these amounts. Therefore, these experiments will help us confirm the validity of choosing GETRIS parameters over precalibration. For each such K, the pose was computed. Along with the average reprojection error, we also computed the reprojection error along the x and y direction and the standard deviation of the overall reprojection error and the standard deviation of the reprojection error in the x and y coordinates. These results are also shown in Table 12,13 and 14. As can be seen, that the reprojection errors remain low even with a large change in the focal length and skew and a very large difference in the principal point (the principal point is moved around till it lies outside the image). In certain cases, problems do arise when the when the principal point lies outside the image. But in most cases, the results are stable. On the basis of this, we can conclude that GETRIS parameters can be used as intrinsic parameters and proceed to compute the pose. The problems in very high and very low focal lengths can be ignored as these will rarely occur in real life scenarios.

5 Conclusion and Future Directions

The study carried out in the report as well as the experimental results have led us to conclude the following :

1. The parameters provided by GETRIS are used as the intrinsic parameters.

2. In order to compute the pose, we will use the algebraic Perspective -3-Point ap-

proach for obtaining an initial estimate. This is for the first frame of the sequence. The pose for the subsequent frames will be computed using the minimization approach.

Also for the purpose of VISTEO it is essential that the pose is computed in real time. As the time profiling shows, we will not be able to compute the pose 25 times a second. The pose change between frames is very minimal. As such computing it 25 frames per second, may lead to an unnecessary waste of time. An alternative could be that we compute pose for 3-4 frames per second and use a filter to obtain the pose in between the frames. Therefore as the next step, we will investigate various filtering approaches, especially the Kalman Filter. Experiments will be carried out to determine which techniques can be used so that the pose computation can be speeded up so that it is done in real time.

References

[1] D. Dementhon and L.S. Davis. Exact and approximate solutions of the perspective-three-point problem. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1100–1105, November 1992.

[2] D. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2):123–141, 1995.

[3] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. Artificial intelligence. The MIT Press, Cambridge, MA, USA, Cambridge, MA, 1993.

[4] O.D. Faugeras and G. Toscani. Camera calibration for 3D computer vision. In *Proceedings of International Workshop on Machine Vision and Machine Intelligence, Tokyo, Japan*, 1987.

[5] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis. Technical report, SRI International, Menlo Park, CA, 1980.

[6] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381 – 395, June 1981.

[7] R.M. Haralick, H. Joo, C. Lee, X. Zhuang, V.G. Vaidya, and M.B. Kim. Pose estimation from corresponding point data. IEEE *Transactions on Systems, Man and Cybernetics*, 6(19):1426–1446, November/December 1989.

[8] R.M. Haralick, C. Lee, K. Ottenberg, and M. Nölle. Analysis and solutions of the three point perspective pose estimation problem. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, USA*, pages 592–598, 1991.

[9] R.J. Holt and A.N. Netravali. Camera calibration problem: Some new results. *Computer Vision, Graphics and Image Processing: Image Understanding*, 54(3):368–383, November 1991.

[10] R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy. Object pose: The link between weak perspective, paraperspective and full perspective. *International Journal of Computer Vision*, 22(2):173–189, March 1997.

[11] R.K. Lenz and R.Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3D machine vision metrology. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 10(5):713–720, September 1988.

[12] D. Lowe. Fitting parameterized three-dimensional models to images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, May 1991.

[13] D. Oberkampf, D.F. DeMenthon, and L.S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, May 1996.

[14] T.Q. Phong, R. Horaud, A. Yassine, and P.D. Tao. Object pose from 2D to 3D point and line correspondences. Technical Report 95 IMAG 16 LIFIA, LIFIA–IMAG, Grenoble, France, February 1993.

[15] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.

[16] L. Quan and Z.D. Lan. Linear $n \geq 4$-point pose determination. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, January 1998.

[17] L. Quan and Z.D. Lan. Linear n-point camera pose determination. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, August 1999.

[18] P. Sturm. Algorithms for plane-based pose estimation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Hilton Head Island, South Carolina, USA*, pages 1010–1017, June 2000.

[19] P. Sturm and S. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, pages 432–437, June 1999.

[20] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 278–284, Kerkyra, Greece, September 1999.

[21] R.Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Miami Beach, Florida, USA*, pages 364–374, 1986.

[22] R.Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE *Journal of Robotics and Automation*, 3(4):323–344, August 1987.

[23] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accurate evaluation. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, October 1992.

[24] J.S.C. Yuan. A general phogrammetric solution for the determining object position and orientation. IEEE *Transactions on Robotics and Automation*, 5(2):129–142, April 1989.

Position	Pose	Reprojection Error	Time
Initial	$\begin{pmatrix} 0.864421 & 0.00179677 & -0.502776 & 50.039 \\ -1.39956e-05 & 0.999999 & 0.00354963 & 59.87 \\ 0.50278 & -0.00306132 & 0.864416 & 700 \end{pmatrix}$	.150824	0.3
Translation (30 0 0)	$\begin{pmatrix} 0.864453 & 0.00172394 & -0.502711 & 80.0315 \\ -1.53052e-05 & 0.999994 & 0.00340295 & 59.8677 \\ 0.502714 & -0.002934 & 0.864448 & 699.95 \end{pmatrix}$	0.150807	0.04
Translation (60 0 0)	$\begin{pmatrix} 0.864497 & 0.00165861 & -0.502636 & 110.021 \\ -1.98148e-05 & 0.999995 & 0.00326572 & 59.8613 \\ 0.502638 & -0.00281325 & 0.864492 & 699.912 \end{pmatrix}$	0.150799	0.06
Translation (90 0 0)	$\begin{pmatrix} 0.861718 & 0.00154681 & -0.507386 & 140.208 \\ 0.00102556 & 0.999988 & 0.00479031 & 59.6928 \\ 0.507387 & -0.00464825 & 0.861706 & 699.474 \end{pmatrix}$	0.168264	0.05

Sequence 1: Initial pose = Rotation + translation, subsequent frames are generated by translating the camera

Position	Pose	Reprojection Error	Time
Initial	$\begin{pmatrix} 0.999999 & 0.000430215 & -0.00174737 & 50.1604 \\ -0.000424283 & 0.999995 & 0.00339371 & 60.1263 \\ 0.00174882 & -0.00339296 & 0.999993 & 801.047 \end{pmatrix}$	0.145443	0.12
Translation (30 0 0)	$\begin{pmatrix} 0.999977 & 1.14174e-05 & -0.00681463 & 80.2679 \\ 2.66336e-05 & 0.999984 & 0.00558361 & 60.0403 \\ 0.00681458 & -0.00558366 & 0.999961 & 800.801 \end{pmatrix}$	0.155052	0.05
Translation (60 0 0)	$\begin{pmatrix} 0.999925 & -0.00137254 & -0.0121563 & 110.543 \\ 0.00145179 & 0.999978 & 0.00651259 & 59.9361 \\ 0.0121471 & -0.00652976 & 0.999905 & 800.536 \end{pmatrix}$	0.161422	0.06
Translation (90 0 0)	$\begin{pmatrix} 0.999901 & -0.00188634 & -0.0139387 & 140.47 \\ 0.00198316 & 0.999974 & 0.00693565 & 59.7367 \\ 0.0139253 & -0.00696261 & 0.999879 & 799.88 \end{pmatrix}$	0.176442	0.05

Sequence2 : Initial pose = Translation, subsequent frames are generated by translating the camera

Position	Pose	Reprojection Error	Time
Rotation $\Pi/20$ (about y-axis)	$\begin{pmatrix} 0.7751087 & 0.00221409 & -0.631824 & -60.063 \\ 0.000174841 & 0.999993 & 0.00371875 & 59.9109 \\ 0.631828 & -0.00299291 & 0.775102 & 699.564 \end{pmatrix}$	.149897	0.12
Rotation $\Pi/10$ (about y-axis)	$\begin{pmatrix} 0.666246 & 0.00238381 & -0.745728 & -168.61 \\ 5.22342e-05 & 0.999994 & 0.00324327 & 59.8042 \\ 0.745731 & -0.00219977 & 0.666242 & 680.813 \end{pmatrix}$	.150016	0.07
Rotation $\Pi/10$ (about y-axis)	$\begin{pmatrix} 0.41302 & -0.00061822 & -0.910723 & -372.614 \\ -0.00256993 & 0.999996 & -0.0018443 & 60.5463 \\ 0.910719 & 0.00310222 & 0.413016 & 596.178 \end{pmatrix}$	.195232	0.05

Sequence 3 : Initial pose = Rotation + translation,subsequent frames are generated by rotating the camera

Table 11: Pose computed using a synthetic image sequence. The first frame of the sequence is used to estimate the pose using the approach (algebraic + minimization). The pose for the subsequent frames are computed using minimization with the pose estimated from the first frame as the initial estimate

Focal length (mm)	Status	Rep. err (mean)	Rep err (sd)	Rep err (mean : x)	Rep err (sd : x)	Rep err (mean : y)	Rep err (sd : y)
9.09	actual	0.0909689	1.05093	-0.00268687	0.999217	-0.00255062	0.924786
10.09	changed	0.12071	1.20441	0.0179825	1.13379	0.0088641	1.40641
11.09	changed	0.173687	1.43657	0.0321897	1.44129	0.0171498	2.16305
12.09	changed	0.225487	1.6344	0.0412652	1.77357	0.0255242	2.87075
13.09	changed	0.272152	1.79438	0.0462401	2.08663	0.0319588	3.49768
14.09	changed	0.313545	1.92576	0.0504106	2.37066	0.0379854	4.04939
15.09	changed	0.35022	2.03559	0.053275	2.62566	0.0432464	4.53605
17.09	changed	0.411924	2.20861	0.0568066	3.0596	0.0521112	5.35177
18.09	changed	0.438039	2.27804	0.059427	3.24455	0.0542331	5.69623
10.09	changed	0.104515	1.12352	0.0138409	1.08399	0.00262525	1.12765
11.09	changed	0.133473	1.26956	0.0216127	1.29616	0.0138144	1.51982
12.09	changed	0.164667	1.40758	0.0301712	1.53995	0.0244537	1.92375
13.09	changed	0.194073	1.5277	0.0294608	1.77796	0.0232571	2.29656
14.09	changed	0.220804	1.62938	0.0303138	1.99846	0.0210322	2.63169
17.09	changed	0.285808	1.85484	0.0378293	2.54498	0.0365962	3.43774
18.09	changed	0.303282	1.91127	0.0360975	2.69334	0.0409018	3.65328
27.671	actual	0.123757	1.01424	0.00447582	0.938519	0.00637077	0.921849
28.671	changed	0.131998	1.05985	-0.00171303	0.97695	-0.000959353	1.00718
30.671	changed	0.172377	1.21136	-0.0120815	1.24509	-0.0149804	1.34425
31.671	changed	0.197125	1.29268	-0.0169085	1.41988	-0.0234589	1.5408
33.671	changed	0.247418	1.44514	-0.0199357	1.78257	-0.0268979	1.93356
34.671	changed	0.271843	1.51638	-0.0635693	1.95731	-0.0359593	2.12464
36.671	changed	0.317932	1.6391	-0.0328291	2.29902	-0.0463655	2.47657
37.065	actual	0.132015	0.994945	0.0230888	0.920694	-0.00829414	0.868964
38.065	changed	0.139335	1.01956	0.0188395	0.977335	-0.0113332	0.911188
39.065	changed	0.156042	1.07475	0.0146749	1.10712	-0.0136791	1.00684
40.065	changed	0.178276	1.14704	0.0113557	1.27844	-0.0165231	1.13523
42.065	changed	0.229466	1.30418	0.00610329	1.66873	-0.0251829	1.43469
44.065	changed	0.282074	1.44828	0.00450167	2.06644	-0.0331046	1.74587
45.065	changed	0.307724	1.51335	0.00222541	2.25961	-0.0288972	1.89855
46.065	changed	0.332718	1.57447	0.000231363	2.4475	7-0.0366856	2.0474
51.1664	actual	0.137671	0.970635	0.0203766	0.89709	-0.00722779	0.831911
52.1664	changed	0.142903	0.984184	0.0178928	0.9289	-0.00774388	0.866053
53.1664	changed	0.151384	1.00805	0.0155094	0.984543	-0.00820724	0.91696
54.1664	changed	0.162246	1.04298	0.0132207	1.05769	-0.00862304	0.980114
55.1664	changed	0.17475	1.08265	0.0110216	1.14286	-0.00899588	1.05173
58.1664	changed	0.217121	1.20995	0.00491542	1.43369	-0.00989581	1.29174
60.1664	changed	0.246554	1.2927	0.0012122	1.63614	-0.0103462	1.45783
70.0335	actual	0.12897	1.0036	0.0219342	0.93969	-0.00820669	0.863996
72.0335	changed	0.132517	1.01473	0.0197515	0.969511	-0.00922929	0.883478
74.0335	changed	0.138973	1.03494	0.0179916	1.02198	-0.010538	0.920783
75.0335	changed	0.143025	1.04745	0.0170728	1.05453	-0.01107	0.944588
77.0335	changed	0.152316	1.07727	0.0153514	1.12857	-0.0118307	0.999782
86.8573	actual	0.125955	1.0149	0.0222404	0.968215	-0.00619035	0.871513
87.8573	changed	0.12654	1.01695	0.0216258	0.970921	-0.00645252	0.877562
88.8573	changed	0.127416	1.0201	0.0210259	0.976026	-0.00670577	0.885448
90.8573	changed	0.129941	1.02892	0.019868	0.992641	-0.00718909	
91.8573	changed	0.131539	1.03425	0.0193091	1.00376	-0.00741928	0.918377
94.8573	changed	0.1374	1.05304	0.0177075	1.04614	-0.00806846	0.961892
106.311	actual	0.135705	0.965321	0.0197491	0.898352	-0.00696154	0.827164
107.311	changed	0.136463	0.967262	0.0192099	0.902756	-0.00702663	0.832469
108.311	changed	0.137452	0.970357	0.0186818	0.908803	-0.00708874	0.839053
110.311	changed	0.14005	0.978705	0.017658	0.925328	-0.00720366	0.855648
111.311	changed	0.141624	0.983695	0.0171619	0.93556	-0.00725602	0.865459
112.311	changed	0.143359	0.989125	0.0166449	0.946939	-0.00840078	0.876159
113.311	changed	0.145239	0.994992	0.0161565	0.95937	-0.00846544	0.887648
115.311	changed	0.149375	1.00829	0.0152076	0.986943	-0.00858764	0.91268
139.236	actual	0.162389	1.02814	0.00685356	1.02047	-0.0101306	0.911465
142.236	changed	0.162118	1.02249	0.00392857	1.01125	-0.0145691	0.918241
143.236	changed	0.162309	1.02209	0.00140399	1.00969	-0.0186871	1.02209
144.236	changed	0.162606	1.02291	0.00251994	1.00884	-0.0146724	0.926982
145.236	changed	.163036	1.02443	0.00170072	1.00868	-0.0155613	0.932494
146.236	changed	0.16358	1.02619	-0.00812594	1.00918	-0.0137696	0.938695
147.236	changed	0.164233	1.0286	-0.00885565	1.0103	-0.0145414	0.945543

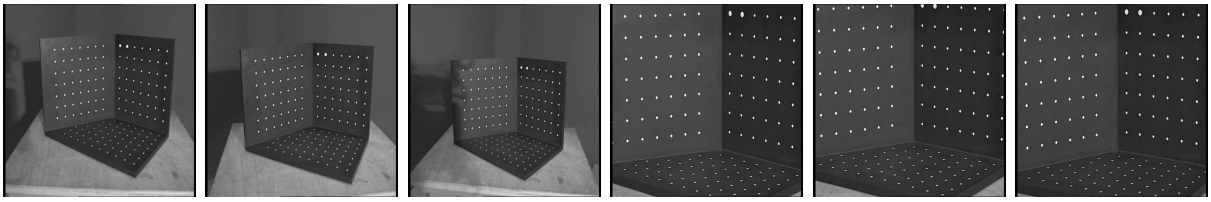
Table 12: Effect of the change in focal length on the pose computed. The skew is 0 and the principal point is the center of the image. The K used are those used for GETRIS images. The initial pose is so chosen such that atleast a minimum number of points are

Focal length (mm)	(u0,v0)	Status (mean)	Rep. err (sd)	Rep err (mean : x)	Rep err (sd : x)	Rep err (mean : y)	Rep err (sd : y)	Rep err
9.09	(360,288)	actual	0.0909689	1.05093	-0.00268687	0.999217	-0.00255062	0.924786
	(760,688)	changed	0.506456	2.42593	-0.135995	3.94987	-0.197925	6.46498
	(120,828)	changed	0.722099	2.84077	-0.886745	5.98161	2.84077	8.84201
	(940,868)	changed	0.615428	2.65855	-0.325225	4.70884	-0.635597	7.88386
	(780,368)	changed	0.388613	2.14431	0.0831665	2.94011	0.0700258	5.0172
	(840,308)	changed	0.372627	2.11441	0.0747304	3.31162	0.0377461	4.48651
	(780,168)	changed	0.364676	2.08239	0.0548976	3.2561	0.012835	4.37995
	(880,808)	changed	0.463791	2.36717	0.0584148	4.12486	2.36717	5.58184
27.671	(360,288)	actual	0.123757	1.01424	0.00447582	0.938519	0.00637077	0.921849
	(940,108)	changed	0.308643	1.60611	0.026716	2.35758	0.066828	2.28058
	(880,808)	changed	0.333661	1.67726	0.0799983	2.5723	1.67726	2.439
37.065	(360,288)	actual	0.132015	0.994945	0.0230888	0.920694	0.994945	0.868964
	(180,768)	changed	0.160624	1.08289	0.0525184	1.14902	0.00839404	1.02493
	(880,808)	changed	0.260585	1.37997	0.0559436	1.88209	0.0370218	1.6433
	(880,168)	changed	0.217835	1.25007	0.0314918	1.6123	0.00592654	1.32854
51.1664	(360,288)	actual	0.137671	0.970635	0.0203766	0.89709	-0.00722779	0.831911
	(920,128)	changed	0.176653	1.08152	0.0247396	1.19353	0.00357633	1.01987
	(860,688)	changed	0.170378	1.06317	0.0495426	1.16151	0.0114635	0.970328
	(220,728)	changed	0.138111	0.975973	0.0291034	0.920509	-0.00308261	0.811594
70.0335	(360,288)	actual	0.12897	1.0036	0.0219342	0.93969	-0.00820669	0.863996
	(240,908)	changed	0.130477	1.00463	0.0289118	0.957623	0.00156162	0.86631
	(960,888)	changed	0.143357	1.03464	0.0319521	1.08236	0.00461473	0.917319
	(260,188)	changed	0.129948	1.00881	0.0194199	0.939016	-0.00996616	0.878996
86.8573	(360,288)	actual	0.125955	1.0149	0.0222404	0.968215	-0.00619035	0.871513
	(240,168)	actual	0.125715	1.01564	0.020524	0.95714	-0.00871202	0.880015
	(900,828)	changed	0.134444	1.03741	0.0282411	1.06579	1.03741	0.89294
	(600,528)	changed	0.128288	1.01947	0.0269369	1.00246	-0.00261685	0.869092
	(1020,128)	changed	0.133268	1.03763	0.0233889	1.04129	-0.00314634	0.903059
	(220,928)	changed	1.16212	2.94503	-1.70366	9.06132	1.18592	7.61903
106.311	(360,288)	actual	0.135705	0.965321	0.0197491	0.898352	-0.00696154	0.827164
	(980,168)	changed	0.138001	0.969321	0.0220976	0.922752	-0.00439052	0.831026
	(1040,968)	changed	0.13629	0.963979	-0.0092288	0.921693	0.00223053	0.809293
	(120,928)	changed	0.134356	0.964224	0.0173164	0.894817	0.00908812	0.81308
	(700,628)	changed	0.134504	0.959239	0.0189839	0.900358	-0.00915162	0.808895
	(880,508)	changed	0.135737	0.962152	0.0230464	0.910052	-0.00399577	0.814636
139.236	(360,288)	actual	0.162389	1.02814	0.00685356	1.02047	-0.0101306	0.911465
	(940,568)	changed	0.163891	1.03306	-0.00669109	1.02758	1.03306	0.922542

Table 13: Effect of the change in principal point on the pose computed. The skew is 0 and the focal length is maintained unchanged.The K used are those used for GETRIS images. The initial pose is so chosen such that atleast a minimum number of points are visible on the grid. (sd = standard deviation

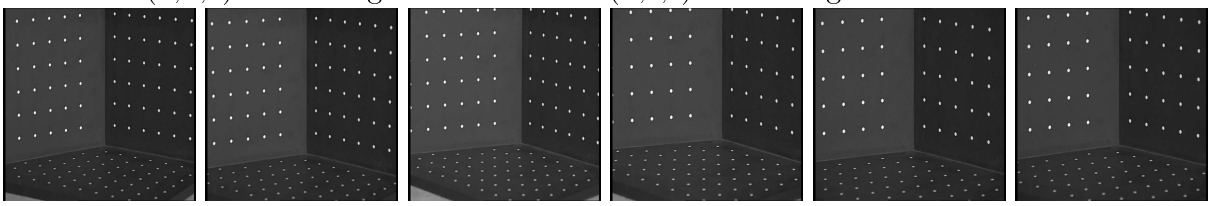
Focal length (mm)	Skew	Status (mean)	Rep. err (sd)	Rep err (mean : x)	Rep err (sd : x)	Rep err (mean : y)	Rep err (sd : y)	Rep err
9.09	0	actual	0.0909689	1.05093	-0.00268687	0.999217	-0.00255062	0.924786
	1.15782	changed	0.0910942	1.05157	-0.00288207	0.999356	-0.00339633	0.927392
27.671	0	actual	0.123757	1.01424	0.00447582	0.938519	0.00637077	0.921849
	1.55741	changed	0.123747	1.01437	0.00458975	0.938493	0.00646987	0.921724
37.065	0	actual	0.132015	0.994945	0.0230888	0.920694	-0.00829414	0.868964
	-2.18504	changed	0.132002	0.995029	0.0231441	0.92053	-0.00826402	0.868943
51.1664	0	actual	0.137671	0.970635	0.0203766	0.89709	-0.00722779	0.831911
		changed						
70.0335	0	actual	0.12897	1.0036	0.0219342	0.93969	-0.00820669	0.863996
	-0.291006	changed	0.128968	1.00359	0.021936	0.939669	-0.00820608	0.86399
86.8573	0	actual	0.125955	1.0149	0.0222404	0.968215	-0.00619035	0.871513
	1.55741	changed	0.125961	1.01492	0.02223	0.968257	1.01492	0.871569
106.311	0	actual	0.135705	0.965321	0.0197491	0.898352	-0.00696154	0.827164
	-0.291006	changed	0.135704	0.965319	0.0197508	0.898344	-0.00696186	0.827156
139.236	0	actual	0.162389	1.02814	0.00685356	1.02047	-0.0101306	0.911465
	-2.18504	changed	0.162393	1.02817	0.00686551	1.02053	-0.0101167	0.911454

Table 14: Effect of the change in skew on the pose computed. The focal is unchanged and the principal point is the center of the image (360,288) The K used are those used for GETRIS images. The initial pose is so chosen such that atleast a minimum number of points are visible on the grid. (sd = standard deviation)



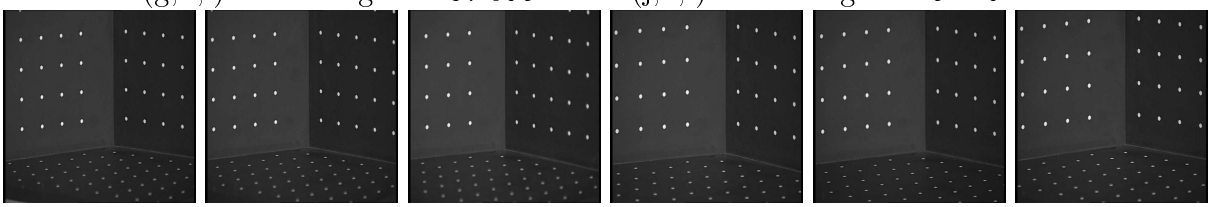
(a) (b) (c) (d) (e) (f)

(a,b,c) Focal length = 9.09 mm : (d,e,f) Focal length = 27.671 mm



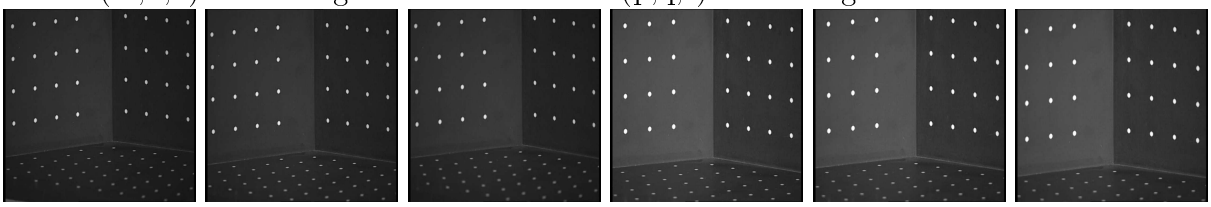
(g) (h) (i) (j) (k) (l)

(g,h,i) Focal length = 37.065 mm : (j,k,l) Focal length = 51.164mm



(m) (n) (o) (p) (q) (r)

(m,n,o) Focal length = 70.0335 mm : (p,q,r) Focal length = 86.85728 mm



(s) (t) (u) (v) (w) (x)

(s,t,u) Focal length = 106.3112 mm : (v,w,x) Focal length = 139.2362 mm

Figure 5: Images of the Calibration grid of MOVI team taken by the camera provided by GETRIS