Robust 3D Computer Vision System for Robotic Applications

Prohászka Zoltán
Department of Control Engineering and Information Technology
Budapest University of Technology
Magyar Tudósok körútja u. 1-D, H-1117 Budapest, IX.
Hungary
prohaszka@seeger.iit.bme.hu

Abstract – **This paper decsribes the design and implementation concepts of a robust 3D vision system is under development. This system is based on a robust geometric algorithm which is able to determine the relative position of two cameras based on 2D images. The cameras need not be pin-hole type ones, they can have real optics. The pincushion effect of the optics is determined with a separate algorithm. The result of this can be used to correct the distorted input images, in order to obtain more precise output. The computational time is linear with the number of points, and enables real time processing on nowadays DSPs. The algorithm can be used with multiple cameras, but it is also able to operate with a single, moving and autonomously zooming camera.**

I. INTRODUCTION

The most important part of this system is the inner part, which determines the transformation matrix between the objects on image A and B. The features of this part (geometrical algorithm in the following) determine the possible features of the whole system. This part has already been implemented, and is under testing. Another level will be responsible to handle the complexity of the environment, in which the cameras operate. The basic methods are declared, and waiting for the implementation. A post-processing algorithm is also implemented which can fit curves and curved surfaces to unsorted and unlinked 3D points. Its testing was done with curves only The algorithms handling the effects of the real optics on the cameras are under implementation. This article will focus more on this part. The development of a demonstrative application is simultaneously going on. This would be able to track the fine movements of the user's head. These data will be used in a Virtual Reality application. Currently the head-mounted markers (HW) are under construction, which will be able to be used either in a helmet or in a window style VR installation.

II. THE GEOMETRICAL ALGORITHM

The algorithm needs paired two-dimensional points whose correspondence has to be determined during preprocessing. The exact spatial locations of these points and their precisions are also outputs of the method. The basic idea of this method (as for the simpler similar ones) was given by the linear correspondence of the two-dimensional projection of the points' locations and velocities. In the range of practical cases, the precision's degradation is linear with the amplitude of the noise distorting the input data.

The basic problem can be viewed in two different ways: Either we have one static object and two cameras (or one static object and one moving camera), or we have one static camera and a moving object. The mathematical deduction is based on the first aspect, but the case of multiple objects requires the second way of thinking. First we developed a simplified linearized model, which in the second phase was changed to an exact nonlinear model. A similar model can be found in [2, Chapter 8].

A. Specification

The input data consist of pair of two-dimensional points; these are the projected location of the real 3D points in the first and in the second image (image A and B in the following). This information can be provided by a preprocessing algorithm, or manually by a user of a 3D modeling program. The images can be made with real optics, thus the method is precise not only for the case of pin-hole cameras. [1, Chapter 3] In the case of enough input data (eight point-pairs) which belong to one rigid object, the method is able to determine the zoom of the optics in the two pictures, the 3D transformation between the two images (this can be interpreted as motion of either the camera or the object), the spatial location of the points and the precision of these. It is necessary to see, that the model of the object is undefined for a linear scaling, while inflated models can produce the same pictures on inflated cameras. Thus, an object with known size is required on the picture for concrete measurements. The post-processing determines the pincushion effect of the optics. A robust extension was developed which can handle noisy input data too. The error of the output is proportional to the amplitude of the noise added to the input.

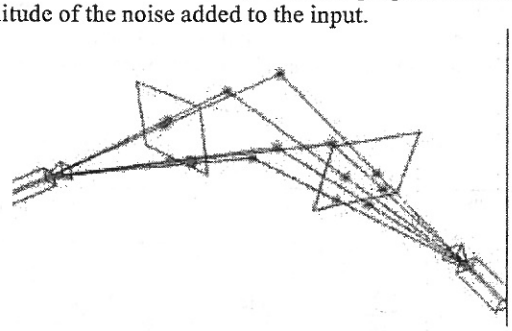

Fig. 1. The basic setup of the problem

In the beginning phase, simpler algorithms have also been developed, which do require less than eight correctly paired 2D points per rigid object, but they use linearized models of rotation and of perspective effects. These are currently not under our interest.

The geometrical algorithm consists of two parts, the first one determines the direction of the translation, and the second one calculates the zoom values. The parameters of rotation can be easily determined after this.

In the first part, a feature vector is calculated for each input point-pair. The spatial properties of these vectors can be used for various purposes, for example to calculate the direction of translation, to make the algorithm robust against noise and also to determine optical distortion.

The formulas of the deduction show symmetry between the two images. For noiseless and undistorted 2D data, we get the same result (both theoretically and practically) if we do an A-B and a B-A measurement based on image A and image B. In general cases (having noise or distortion), the two different, but similar solutions can be averaged in order to have better approximation of the results.

The second part consists of the solution of a nonlinear equation system. The result can be determined by successive expression and substitution of unknowns. The nonlinear system consists of four trilinear equations, which can be converted to a 12th degree polynomial. Fortunately, beautiful simplifications occur and the final equation becomes to a simple second degree one.

The combined algorithm is able to handle the unknown zoom of both of the optics. This was a primary expectation against the system.

There are situations, in which the axis of rotation and the vector of translation are in such a relation to the optical (z) axes, in which the equations result infinite solutions, and the required quantities can not be determined. The upper levels can easily handle these exceptional singular cases.

B. The robust extension and its performance

The algorithm was first tested with simulated input, which corresponds to perfect geometrical conditions. In this simulated case, the output contained the correct result. In the practice, there adds noise to the input in all cases, and the program should work with noisy data too. To solve this problem, three different versions has been implemented and tested. The best one leads to an algebraic eigenvalue problem, which gives three different solutions. We must choose from them with the verification of the output. This means, the resulting 3D locations are projected to produce 2D images similar to the input data.

TABLE I

THE PERFORMANCE OF THE ROBUST EXTENSIONS, SIMULATION RESULTS

Input noise, relative to image size [+1;-1]	Same noise level in S-VHS pixels	Precision of output, relative to image size [+1;-1]		
		Eigenvalue problem	2nd Version	3rd Version
±0.2	±40	0.33	*9.8*	*8.1*
±0.1	±20	0.168	*3.3*	*5.1*
±0.05	±10	0.086	*2.1*	*4.0*
±0.03	±6	0.053	*1.35*	*3.1*
±0.01	±2	0.0191	*0.49*	*2.3*
±0.005	±1	0.0097	0.23	*1.0*
±0.0025	±0.5	0.0052	0.11	*0.48*
±0.001	±0.2	0.0019	0.043	0.18
±0.0001	±0.02	0.00020	0.0042	0.018

Grey data: can not be used in practice due to the too big errors

The sum of squared differences between the resulting and original 2D locations shows the error of the actually verified solution. The other two versions lead only to single solutions. Five different solutions have to be evaluated. After this we can select the most accurate one. In all cases, the first version gave the most precise results. (See Table 1)

Because the algorithm does not require calibrated cameras, we can use archive image or video data for processing, and it is also possible to identify the type of the optics.

The obscurity (uncertainity) of the output 3D locations can easily be determined. Currently, each 2D input point contains an additional attribute which express the obscurity of its location (typically the pixel radius, or, on blunt images, the wavelength of the highest frequency). This data is used to determine the 3D location which minimizes the least square error after back-projection to the projective planes. When we construct two rays which are going through the two optical center of the cameras (A or B), and also through the 2D location of the same point's image on the proper projective plane (A or B, respectively), these rays would typically not intersect each other. The location which results the least error would be halfway between them, if the cameras are equally distant from this location, the aperture angles of the two cameras are identical, and the obscurity of each 2D point is the same. In general cases, these parameters (distance, zoom, 2D obscurity) have to be taken into account to determine the location with the highest probability. The current implementation works this way.

Currently, we do not care about the obscurity of the calculated transformations because their dependence on the input is very complex. In the case of successive measurements, the reciprocal of the 3D obscurity can be collected (by simple addition) to a 3*3 matrix. Thus, the more measurements we did, the less obscurity of 3D locations we have. In a succeeding step, we can update (and refine) the calculated camera positions based on the more precise 3D location of the points.

C. Test results performed on real input

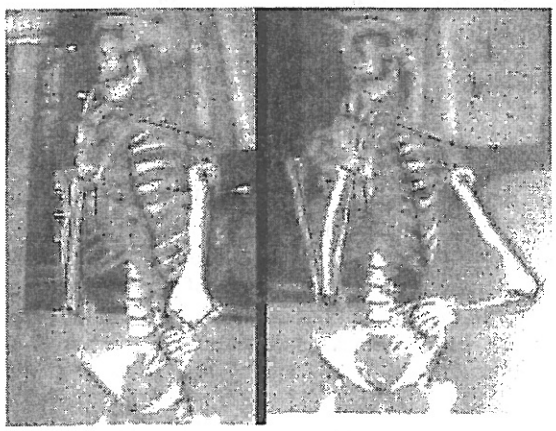

Fig. 2. Two images of the same skeleton (lines between points are shown only to help identifying the correspondence)

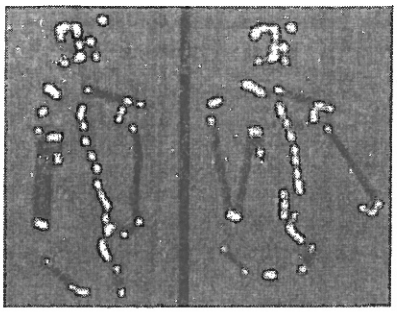

Fig. 3. The location of coupled 2D point pairs (lines between points are shown only to help identifying the correspondence).

The first real-life test was done on three images of the same human skeleton. Let us call these pictures A, B and C respectively. In the following, we focus on the measurement A-B. The pictures (Fig. 2) were taken with a mobile phone, whose camera has high aperture angle, and therefore a relatively high radial distortion (Fig. 5). Several corresponding points on these images were coupled manually, in the actual phase (Fig. 3).

Then, the 2D coordinates of these points were feed to the MATLAB implementation of the algorithm, and the output 3D points were exported to a 3D editor (Fig. 4). When the 3D locations were projected back to 2D the mean square of the differences to the original inputs turned out to be around the size of 1 pixel.

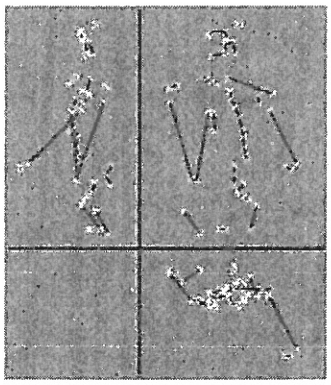

Fig. 4. Axonometric views of the resulted 3D locations

III. OPTICAL DISTORSION

A. Handling the pincushion effect of the optics

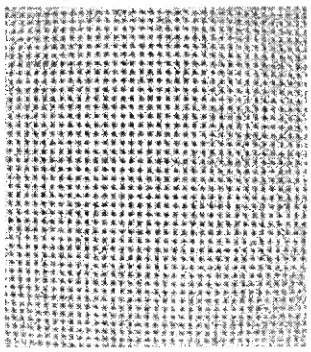

Fig. 5. Photograph of an equally spaced grid with real camera. This image can be used to verify the determined distortion function.

In the initial phase two methods were selected to handle this problem:

Method A divides the 2D point's on image A into equally sized clusters based on their distance from the image's center. After this, the normal geometric algorithm determines the zoom for the different clusters. In the case of five clusters, a cubic distortion function can be computed by LS method.

Method B assumes that the distortion of the optics has the same effect to the geometric algorithm as the effect of random noise. Thus, the preliminary calculation would give almost correct transformation matrices. After knowing the location of the two cameras, one can determine, which distortion gives the least error of the back-projected 3D points.

Method B uses a model which assumes that the real and the projected positions of a point satisfy that if both 2D positions are expressed in polar coordinates (f ,r) then (f) is not affected by the lens in the optics.

The radius is distorted in the following manner:

$$r_{projected} = f(r_{original}), \tag{1}$$

where f needs to be strictly monotonic, and thus invertable, which is always true in the practice. The model

$$r_{original} = g(r_{projected}) = f^{-1}(r_{projected}) = \\ = polynomial\ (r_{projected}) \tag{2}$$

is the basis of this post processing algorithm.

In the following we deduct the determination of the distortion parameters. This method works properly if the transformation between the camera A and B is known.
Let $\bar{p}_i$ be the 3D location of the actually examined point. For simplicity, we will neglect the i index for a moment. Denote $\bar{p}_{A\_}$ and $\bar{p}_{B\_}$ the projected locations of $\bar{p}$ on the projective planes of the cameras A and B, respectively. $\bar{c}_A$ and $\bar{c}_B$ are the center of the perspective planes.

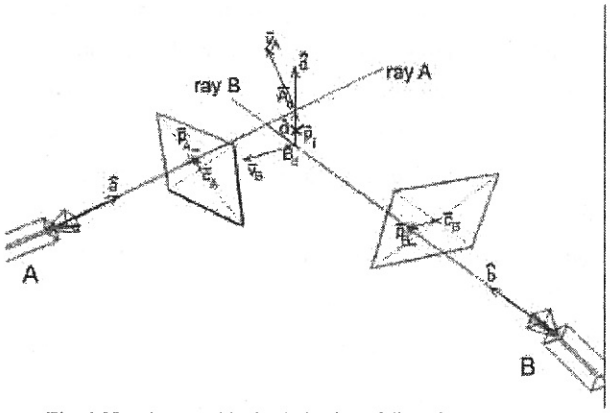

Fig. 6. Notations used in the derivation of distortion parameters.

Let $\hat{a}$ and $\hat{b}$ (with unit length) be the direction of the rays coming from the optical centers of the camera A and B, and going through $\bar{p}_{A\_}$ and $\bar{p}_{B\_}$ respectively. In the general case, these rays will be skew lines. Let $\bar{d}$ be the shortest section between these rays (the normal transversal), and $\hat{d}$ be its direction (with unit length). Let

$\bar{A}_d$ be the closest point of ray A to ray B, and $\bar{B}_d$ be the closest point of ray B to ray A. Thus $\bar{d} = \bar{A}_d - \bar{B}_d$. Let us note with

$$\bar{r}_A = \bar{p}_{A\_} - \bar{c}_A \text{ and } \bar{r}_B = \bar{p}_{B\_} - \bar{c}_B \tag{3}$$

the positions of the projected points relative to the image's center. These vectors are the radii of the images of point $\bar{p}$. For the unit length directions of these radii we get:

$$\hat{r}_A = \bar{r}_A / \|\bar{r}_A\| \text{ and } \hat{r}_B = \bar{r}_B / \|\bar{r}_B\|. \tag{4}$$

If the location of $\bar{p}_{A\_}$ is artificially changed in the direction of $\hat{r}_A$ with the amount of Δr_A, then the displacement of $\bar{A}_d$ will be the following:

$$\bar{v}_A = \Delta r_A \cdot \hat{r}_A \cdot z_A. \tag{5}$$

Similarly, for the displacement of $\bar{B}_d$ we get:

$$\bar{v}_B = \Delta r_B \cdot \hat{r}_B \cdot z_B, \tag{6}$$

where z_A and z_B notes the z coordinate of $\bar{p}$ in the coordinate frame of the camera A and B, respectively.

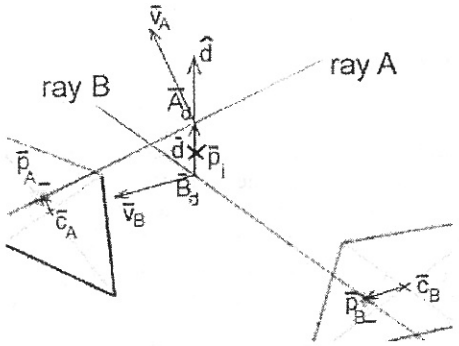

Fig. 7. Details used in the derivation

Displacing $\bar{A}_d$ with $\bar{v}_A$ and $\bar{B}_d$ with $\bar{v}_B$, these points will not be the closest ones. Let $\bar{A}'_d$ and $\bar{B}'_d$ denote the closest points in all instance. For small Δr_A and Δr_B values, the direction of $\bar{d}$ ($\bar{d} = \bar{A}'_d - \bar{B}'_d$) will not change, since the directions $\hat{a}$ and $\hat{b}$ are quasi constant. The components of $\bar{v}_A$ and $\bar{v}_B$ in the $\hat{d}$ direction (let these be called $\bar{v}_{A\_d}$ and $\bar{v}_{B\_d}$) will be the displacements of $\bar{A}'_d$ and $\bar{B}'_d$ in the same direction:

$$\bar{v}_{A\_d} = <\bar{v}_{A\_d}, \hat{d}> \hat{d} \text{ and } \bar{v}_{B\_d} = <\bar{v}_{B\_d}, \hat{d}> \hat{d} \tag{7}$$

$$\bar{d} = \bar{d}_0 + \bar{v}_{A\_d} - \bar{v}_{B\_d}. \tag{8}$$

In the equation above, every vector's direction is $\hat{\mathbf{d}}$, so this equation is true for the signed length of these vectors:

$$d = d_0 + v_{A\_d} - v_{B\_d} \tag{9}$$

Where

$$d = <\bar{d}, \hat{d}>, \quad d_0 = <\bar{d}_0, \hat{d}> \quad \text{and}$$

$$v_{A\_d} = <\bar{v}_{A\_d}, \hat{d}>, \quad v_{B\_d} = <\bar{v}_{B\_d}, \hat{d}>. \tag{10}$$

The deduction is continued in the following for the case when we want to minimalize the spatial distance between the rays A and B. This gives simpler formulas than the minimalization of the 2D error.

The model of distortion will be used in the sequel. The model stays that the required correction of any point in the radial direction is given by a polynomial of the original radius:

$$\Delta r_A = <\bar{C}_A, [1 \; r_A \; r_A^2 \; r_A^3]> \text{ and}$$

$$\Delta r_B = <\bar{C}_B, [1 \; r_B \; r_B^2 \; r_B^3]> \tag{11}$$

where $\bar{C}_A$ and $\bar{C}_B$ are the cofactors of the correction polynomials, which were chosen the grade 3 for the following. We get:

$$d = d_0 + <\bar{C}_A, [1 \; r_A \; r_A^2 \; r_A^3]> \frac{v_{A\_d}}{\Delta r_A} - <\bar{C}_B, [1 \; r_B \; r_B^2 \; r_B^3]> \frac{v_{B\_d}}{\Delta r_B} \tag{12}$$

where

$$v_{A\_d} / \Delta r_A = <\hat{r}_A \cdot z_A, \hat{d}> \text{ and}$$

$$v_{B\_d} / \Delta r_B = <\hat{r}_B \cdot z_B, \hat{d}>. \tag{13}$$

Grouping all unknown parameterss to the row vector $\bar{x}$:

$$\bar{x} = [\bar{C}_A \; \bar{C}_B] \tag{14}$$

and noting

$$\bar{m}_i = \left[[1 \; r_A \; r_A^2 \; r_A^3] \frac{v_{A\_d}}{\Delta r_A} \quad -[1 \; r_B \; r_B^2 \; r_B^3] \frac{v_{B\_d}}{\Delta r_B} \right] \tag{15}$$

the equation simplifyes to:

$$d = d_0 + \bar{m}_i \cdot \bar{x}^T \tag{16}$$

Now we take care again of the index i:

$$d_i = d_{0\_i} + \bar{m}_i \cdot \bar{x}^T \tag{17}$$

is to be minimalized for all i.

Collecting d_i-s to $\bar{D}$, the d_{0_i}-s to $\bar{D}_0$, and the $\bar{m}_i$-rows to the matrix $\bar{\bar{M}}$, we derive at a simple Least-Square (LS) problem:

$$\left\| \bar{D}_0 + \bar{\bar{M}} \cdot \bar{x}^T \right\|^2 \text{ is to be minimalized.} \tag{18}$$

If we would like to minimize the 2D error during the verification instead of the 3D error (skew), this will result in the addition of weighting factors to d_i, d_{0_i} and $\bar{m}_i$. This does not effect the behaviour of (18). The more complex problem which handle the effect of the zoom values and the 2D obscurity results also in an LS problem. The resulting matrix equation is poorly conditioned, but gives proper results.

B. Implementation results

Method B has been implemented. It is working perfectly on synthetic distortion, if the preliminary determined transformations are the simulated ones (Fig. 8). Unfortunately, the effect of distortion is not a white noise process, and the geometrical algorithm finds transformation and zoom values which result approximately one fifth of the error after verification, what we would get by using the original transformations. This means that the effect of radial distortion is not nearly 'perpendicular' to every effect of a combined change in zoom and camera location. The running results show, that in some cases, the algorithms find distortion parameters, which are further from the simulated ones than the undistorted (linear) pin-hole model. (Fig. 9) Thus Method B can be used only with cameras having calibrated locations, for example in stereo vision systems.

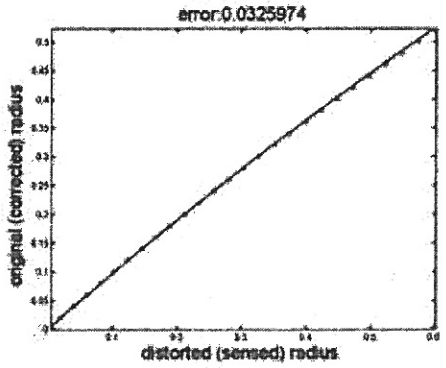

Fig. 8. The simulated (dots) and determined (solid curve) distortions if the transformations are known

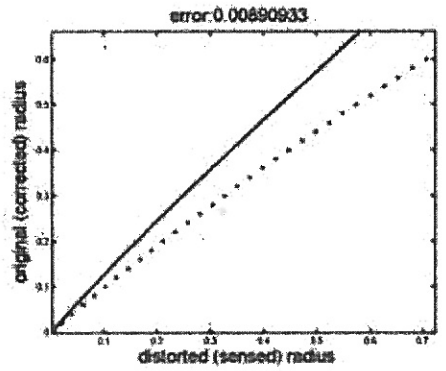

Fig. 9. The simulated (dots) and determined (solid curve) distortions if the transformations are calculated from 2D data

C. Possible solutions to the emerging problems

Currently several methods has been sketched to help handling this problem.

The presence of radial distortion raises the total 2D error, and the difference between the 3D points resulted by the measurements AB and BA. (The term 'measurement XY' means that we run the algorithm with the first picture to be image X and the second to be image Y.) One could numerically differentiate these errors respect to a distortion function either on image A or on image B. It will be very interesting to test or deduct the effect of orthogonality of the basic distortion functions.

Instead of differentiating the total 2D error, we can focus on the requirements against the point-pairs' feature vectors (these vectors should lie in the same hyperplane). We can deduct any basic distortion function's effect (or its cofactor's) to these requirements. This results to a very complex matrix eigenvalue problem, but we hope in the practical cases this can be solved.

Method A should be implemented and tested also, but it has stronger limitations than the other methods, if we have few points on the image, or to much objects on the image.

All methods handling this problem can be iterated: first we should determine the distortions, apply the inverse of this to the input, and repeat these steps.

IV. FUTURE DEVELOPMENT

A. Problems waiting for implementation

For several occurring problems, we have already worked out algorithms, which will be implemented in the near future. These problems are:

1) The handling preliminary knowledge: For example, information, that the zoom (and thus the radial distortion) is constant, or if we have two or more cameras connected stiffly to each other, then we know that the transformation between them is not changing. This method is based on the gradient of the verification error respect to the parameters of the cameras (including location).

2) The identification of points belonging to differently moving objects.

3) The identification of point-pairs which come from false pairing.

4) The combination of successive measurements (This can be handled also as preliminary information.).

B. Possible continuation of the development

The case of flexible bodies: If two spatial transformations (object movements) differ only in an axial rotation or prismatic translation, and this relation holds in the previous time instances also, than the model of the two rigid bodies can be linked, showing this relation. The geometry of industrial robots [3, Chapter 1] can be determined without prior information. Portions of flexible structures (for example bending branches of trees) can be linked in this manner.

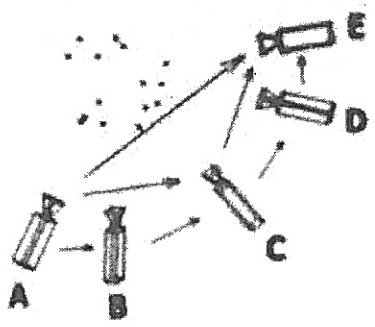

Fig. 10. Sequential processing of images.

The possibility of iterative refinement: If several images are processed (as an image sequence) it is possible to refine the results. Given images A, B, C, if the correspondence between the points of A, B and that of between B, C could be easily determined due to small movements, then the transformation can also be determined. If the spatial locations of the points are also known, then the correspondence between images A and C can be easily determined. The algorithm can be run with input images A and C, which measurement has twice as great basis distance as the previous two. The precision in the direction z can be reduced to the half.

This process can be applied to the sequence A, B, C, D, and E. After running measurement C-E, one can make a run for A and E. There is also an other effect, due to the fact that improving the precision of the locations improves the element of the matrices describing the spatial transformations. This refines the location of the points. This can be repeated until infinity, but the convergence to the exact values has to be analyzed mathematically.

V. CONCLUSIONS

We presented the main design and implementation steps of a three-dimensional perspective vision system, which differs from the commonly used ones by the ability to handle varying zoom, real optics and by the way of handling noisy inputs. We demonstrated the properties and test results of the most important parts, the algorithms handling the 2D-3D transformation and the optical distortion. The properties of these parts fundamentally determine the properties of the whole system.

The future research will be done on the further workout on proper pre- and post processing algorithms. Further aim is to develop such a collection of image processing routines, which can automatically construct the partial or full 3D model of the environment.

VI. ACKNOWLEDGMENT

This research was supported by the Hungarian National Research Fund under grant No. OTKA T 042634.

V. REFERENCES

[1] Faugeras, O.: *Tree-dimensional computer vision. A geometric viewpoint*, MIT Press, 1993.

[2] Trucco, E.-Verri, A.: *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, 1998.

[3] Lantos, B.: *Robot Control* (in Hungarian), Akadémiai Kiadó, Budapest 2002.