FAUCCAL: AN OPEN SOURCE TOOLBOX FOR FULLY AUTOMATIC CAMERA CALIBRATION

V. Douskos[1], L. Grammatikopoulos[2], I. Kalisperakis[1], G. Karras[1], E. Petsa[2]

[1] Department of Surveying, National Technical University of Athens (NTUA), GR-15780 Athens, Greece
[2] Department of Surveying, Technological Educational Institute of Athens (TEI-A), GR-12210 Athens, Greece
e-mail: mdouskos@googlemail.com, lazaros.pcvg@gmail.com, ilias_k@central.ntua.gr, gkarras@central.ntua.gr, petsa@teiath.gr

KEY WORDS: camera calibration, point operator, automation, bundle adjustment, radial distortion

ABSTRACT

Recently, the authors presented a fully automatic camera calibration algorithm. The open-source software *FAUCCAL* (Fully Automatic Camera Calibration) is now freely available on the Internet. Input is simply different images of standard chess-board patterns; the software then proceeds automatically to produce calibration results and statistical data for camera parameters selected by the user. With this software, feature points are first extracted with a Harris operator, among which valid pattern nodes are separated and subsequently ordered in columns and rows in correspondence with pattern nodes. Initial values for all unknown parameters are estimated automatically. Based on the established point correspondences and initial values, a final bundle adjustment allows fully recovering – without use of any external information – the camera geometry parameters selected by the user. Besides camera constant, principal point location and radial-symmetric lens distortion polynomial, these may include decentering lens distortion, aspect ratio and skewness. Graphical output is also provided. The *FAUCCAL* site provides users with the source code in Matlab, detailed documentation of the software (including tips), links to bibliographical references and an image test dataset with results. It is believed that our software will prove useful to everyone, and particularly non-photogrammetrists, involved in the field of cultural heritage documentation. The authors welcome any questions but also suggestions, comments and criticism which will help improve this toolbox.

1. INTRODUCTION

Image-based documentation of cultural heritage is a rapidly developing field of interest and research with numerous applications, notably presented at CIPA symposia. Today, such projects rely almost exclusively on the use of uncalibrated off-the-shelf digital cameras. These projects are carried out by different experts, many of who (architects, archaeologists, conservationists) are not familiar with photogrammetry. In view of this, the issue of camera calibration clearly becomes very important. In practical close-range photogrammetric tasks, furthermore, it is often preferable to precalibrate cameras (Remondino & Fraser, 2006).

For a typical user, cameras should ideally be calibrated automatically, exclusively from image sets taken rapidly with unknown exterior orientation. An answer to this demand is free availability of open-source user-friendly software for automatic camera calibration, based on simple 2D patterns of the chess-board type recorded in different perspective views. Such patterns are easy to construct and facilitate automation via feature extraction algorithms. Freely accessible tools of this kind have been mainly inspired by 'plane-based calibration' (Sturm & Maybank, 1999; Zhang, 1999), which rests on the projective transformations between a plane of known metric structure and its images, usually followed by a non-linear refinement step. The *Camera Calibration Toolbox for Matlab* of J.-Y. Bouguet (see cited site), implemented also in C++ and included in the Open Source Computer Vision library distributed by Intel, is the best known among functional tools of this type; other approaches (see Bouguet website) often represent 'add-ons' to this tool. Another interesting free calibration toolbox is *DLR CalDe – DLR CalLab* (see site).

Elaborating on the concept behind the semi-automatic calibration toolbox of Bouguet, the authors have presented a camera calibration algorithm which – unlike similar approaches – runs in a fully automatic mode, dispensing with special targets, non-symmetric patterns and assumptions regarding lens distortion or appearance of the full calibration pattern on images; exclusive assumption here is that different images of a typical chess-board pattern (succession of dark and light squares), acquired with the same camera, are available.

This algorithm – whose exclusive purpose is camera calibration, i.e. camera exterior orientations are irrelevant – has been documented and tested in Douskos et al. (2007 and 2008). Now it is available as open-source free-access software *FAUCCAL* (Fully AUtomatic Camera CALibration), which can be downloaded at

http://www.survey.ntua.gr/main/labs/photo/staff/gkarras/fauccal.html.

The *FAUCCAL* site provides potential users with the source code in Matlab, a detailed documentation of the software (including tips), links to bibliographical references and a test image dataset with results. For the details of the algorithm readers are referred to the above publications. Purpose of this paper is to present the toolbox and describe its basic functions and tools. It is believed that our software will prove useful to those involved in the field of cultural heritage documentation. Any question, suggestion or critical comment will help improve the toolbox and, of course, is welcome by the authors; and so are modifications, extensions or possible add-ons.

2. OUTLINE OF THE ALGORITHM

Assuming that a user has acquired – with the same camera and focus – images of a black-and-white chess-board pattern with square tiles, the process flows automatically to produce results, statistical data and graphical outputs for the camera parameters selected by the user. A Harris point operator (estimated accuracy ~0.1 pixel) is first applied to extract the chess-board nodes. Actual nodes are then separated from noisy points; subsequently they are linked and ordered in columns and rows ('gaps' are tolerated, i.e. it is not necessary to identify all individual nodes or every row and column). Thanks to the square tiles, physical correspondence between image and pattern nodes is not needed.

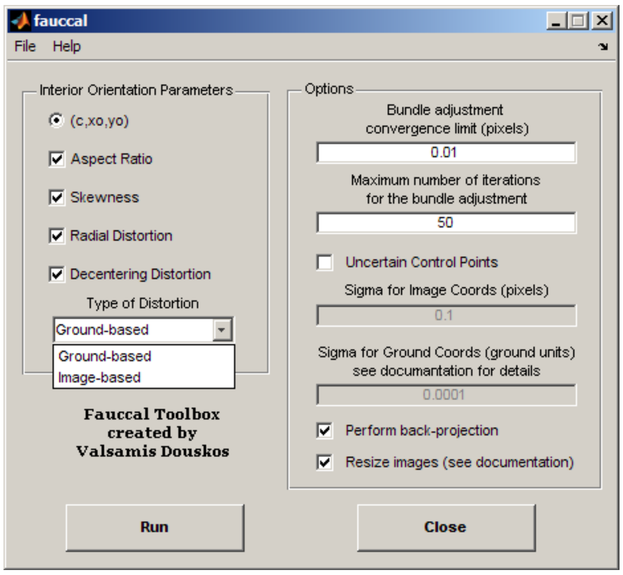
As a consequence, image exterior orientations given in the solution may differ by in-plane translation and/or rotation from the actual ones; this, however, does not affect interior orientation.

To exclude outliers, only points belonging to intersecting image lines are at first accepted as valid nodes. Approximations for the unknown parameter values are gained automatically through the vanishing points of the two pattern directions. After a first solution, remaining extracted nodes are checked by back-projection and – if their image residuals are tolerable – accepted as valid. With all established point correspondences, a final bundle adjustment allows recovering – without use of any external information – the camera geometry parameters selected by the user. For further details see Douskos et al. (2007 and 2008).

3. USING THE CALIBRATION TOOLBOX

3.1 Mathematical model, parameters, options

The basic menu of the *FAUCCAL* toolbox is the following:


fauccal
File Help

Interior Orientation Parameters
(●) (c,xo,yo)
[✓] Aspect Ratio
[✓] Skewness
[✓] Radial Distortion
[✓] Decentering Distortion
Type of Distortion
Ground-based
Ground-based
Image-based

Faucal Toolbox
created by
Valsamis Douskos

Options
Bundle adjustment convergence limit (pixels): 0.01
Maximum number of iterations for the bundle adjustment: 50
[ ] Uncertain Control Points
Sigma for Image Coords (pixels): 0.1
Sigma for Ground Coords (ground units) see documentantion for details: 0.0001
[✓] Perform back-projection
[✓] Resize images (see documentation)

Run Close

Figure 1. Starting window

On the left section of this GUI the combination of the camera interior orientation elements which will be estimated is selected. These include the camera constant *c*, the image coordinates (x_o,y_o) of the principle point, the aspect ratio α, the skewness *sk*, and the coefficients of radial-symmetric (k_1, k_2) and decentering (p_1, p_2) lens distortions. The camera model is the following:

$$x = x_o - c \cdot \frac{r_{11}(X-X_o)+r_{12}(Y-Y_o)+r_{13}(Z-Z_o)}{r_{31}(X-X_o)+r_{32}(Y-Y_o)+r_{33}(Z-Z_o)} -$$

$$-c \cdot \alpha \cdot sk \cdot \frac{r_{21}(X-X_o)+r_{22}(Y-Y_o)+r_{23}(Z-Z_o)}{r_{31}(X-X_o)+r_{32}(Y-Y_o)+r_{33}(Z-Z_o)} + \Delta x^r + \Delta x^d$$

$$y = y_o - c \cdot \alpha \cdot \frac{r_{21}(X-X_o)+r_{22}(Y-Y_o)+r_{23}(Z-Z_o)}{r_{31}(X-X_o)+r_{32}(Y-Y_o)+r_{33}(Z-Z_o)} + \Delta y^r + \Delta y^d$$

with

$$\begin{aligned}
\Delta x^r &= (\tilde{x}-x_o)\cdot(k_1\cdot r^2 + k_2\cdot r^4)\\
\Delta y^r &= (\tilde{y}-y_o)\cdot(k_1\cdot r^2 + k_2\cdot r^4)\\
\Delta x^d &= p_1\cdot(r^2+2\cdot(\tilde{x}-x_o)^2)+2\cdot p_2\cdot(\tilde{x}-x_o)\cdot(\tilde{y}-y_o)\\
\Delta y^d &= p_2\cdot(r^2+2\cdot(\tilde{y}-y_o)^2)+2\cdot p_1\cdot(\tilde{x}-x_o)\cdot(\tilde{y}-y_o)\\
r^2 &= (\tilde{x}-x_o)^2+(\tilde{y}-y_o)^2
\end{aligned}$$

(X_o, Y_o, Z_o denote the object space coordinates of the projection centre, while r_{ij} are the elements of the image rotation matrix).

Depending on the application, it may be necessary to calculate distortion parameters in either of the following ways, an option offered by the program:

- by referring distortion to the centrally projected ground point coordinates on the image plane (*Ground-based*), i.e. by adopting the rigorous perspective model (this expression of distortion is needed e.g. if later in the project the images will have to digitally rectified);
- by referring distortion to the measured image point coordinates (*Image-based*). This is the case when only image coordinates may be used (e.g. if relative orientation will be performed later with the coplanarity condition).

In the first case, quantities $\tilde{x}$, $\tilde{y}$ in the model are expressed as:

$$\tilde{x} = x_o - c \cdot \frac{r_{11}(X-X_o)+r_{12}(Y-Y_o)+r_{13}(Z-Z_o)}{r_{31}(X-X_o)+r_{32}(Y-Y_o)+r_{33}(Z-Z_o)}$$

$$\tilde{y} = y_o - c \cdot \frac{r_{21}(X-X_o)+r_{22}(Y-Y_o)+r_{23}(Z-Z_o)}{r_{31}(X-X_o)+r_{32}(Y-Y_o)+r_{33}(Z-Z_o)}$$

In case the *Image-based* option is chosen, $\tilde{x}$ and $\tilde{y}$ are simply the measured image coordinates of the point.

On the right of Fig. 1, certain parameters concerning the bundle adjustment may be set. These include the convergence limit for the corrections of the c, x_o, y_o parameter values (for corrections below this limit the toolbox exits the bundle adjustment loop) and the maximum number of allowed iterations (above this limit the adjustment is considered as failed). Further, it is possible to regard the pattern nodes not as 'fixed' control points, i.e. error-free, but rather as 'observed', i.e. subject to uncertainty (*Uncertain control points*). In this case, σ-values (standard deviations) must be inserted for the image and the ground coordinates of the pattern nodes to allow calculation of a weight matrix. As scale is here irrelevant, the squares of the pattern could be given any arbitrary size (as to how this issue is handled here the reader is referred to the available documentation text).

With selection of the '*Perform back-projection*' option, the algorithm – after a first solution has been reached – will automatically back-project all missing pattern nodes, plus 3 additional outer rows and columns on all sides of the chess-board. This is to detect any nodes extracted but missed or discarded in the initial bundle adjustment and introduce them in the final solution.

The *Resize images* option will increase speed. Provided that the Matlab 'Image Processing Toolbox' has been installed, images larger than 920 pixels in width or 720 in height will then be automatically resized to 640 pixels width (the original aspect ratio is retained). The Harris operator is applied initially to the down-sized images; the extracted points are transferred to the original images, on which the Harris operator is applied only in a small area around these positions. Thus, the point extraction algorithm runs approximately 10 times faster on a 7 Mpixel image. By default, this option is set to '*on*'. Yet, if in some cases (mainly due to unsuitable imagery) the point operator will not manage to extract enough points in the down-sized image, it must be applied directly to the original image. Computation time will increase, but as many points as possible will be extracted from the image.

3.2 Running the Program

The '*Run'* button initializes the camera calibration process, and in the dialog box the images to be used for calibration must be selected (the 'Tips' in the documentation text give advice about image number and configurations). Throughout the initialization phase (details are found in Douskos et al., 2008), a wait bar will show which image is being processed at the moment (Fig. 2).

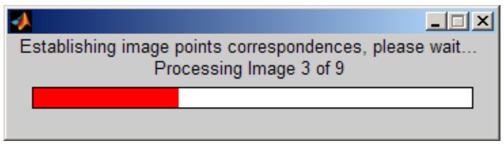

Establishing image points correspondences, please wait...
Processing Image 3 of 9

Figure 2. Wait bar indicating the image being processed.

After the final bundle adjustment (following the step of back-projection) has converged, two windows will appear. The first (Fig. 3) displays all image points with residuals >3 times larger than the standard error σ_o of the adjustment. In this window one can select and exclude all, or some, of these points from the solution; in this case the adjustment is resumed.

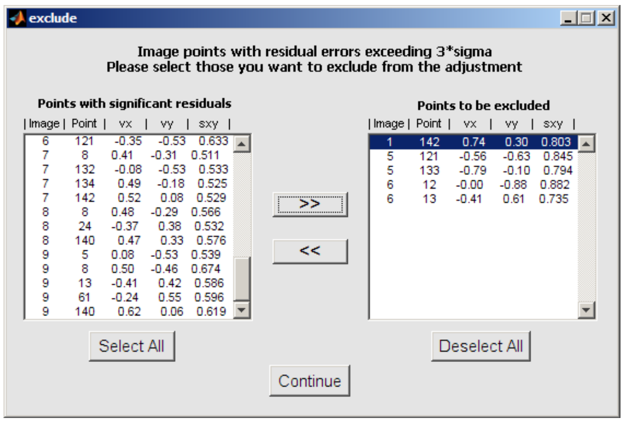

exclude

Image points with residual errors exceeding 3*sigma
Please select those you want to exclude from the adjustment

Points with significant residuals

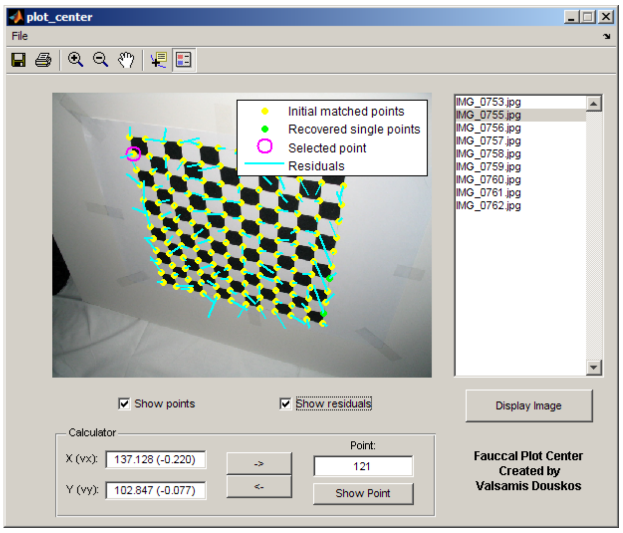
Image	Point	vx	vy	sxy
6	121	-0.35	-0.53	0.633
7	8	0.41	-0.31	0.511
7	132	-0.08	-0.53	0.533
7	134	0.49	-0.18	0.525
7	142	0.52	0.08	0.529
8	8	0.48	-0.29	0.566
8	24	-0.37	0.38	0.532
8	140	0.47	0.33	0.576
9	5	0.08	-0.53	0.539
9	8	0.50	-0.46	0.674
9	13	-0.41	0.42	0.586
9	61	-0.24	0.55	0.596
9	140	0.62	0.06	0.619

Points to be excluded

Image	Point	vx	vy	sxy
1	142	0.74	0.30	0.803
5	121	-0.56	-0.63	0.845
5	133	-0.79	-0.10	0.794
6	12	-0.00	-0.88	0.882
6	13	-0.41	0.61	0.735

Figure 3. Details for image point residuals $>3\times\sigma_o$.

In the second window (Fig. 4) the user may review all images of the dataset, and also visualize on each image all nodes involved in the bundle adjustment as well as the corresponding residuals (vectors of the residuals are enlarged by a factor of 100).

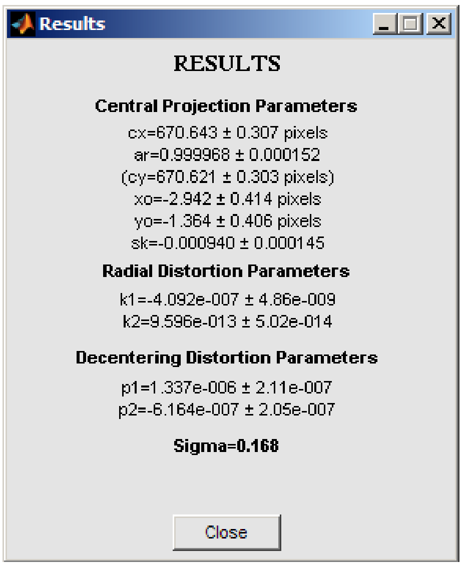


X (vx): 137.128 (-0.220)
Y (vy): 102.847 (-0.077)
Point: 121

Figure 4. Image points and image residuals.

Nodes which have taken part in the initial solution will appear in yellow, nodes recovered by back-projection appear in green, points which belong to the additional outer rows and columns appear in red (no such points are present in Fig. 4). A calculator at the bottom of the window allows finding the point closest to coordinates inserted by users in the corresponding textboxes, or shows the coordinates and residuals of a point given by its code number. Points can also be seen with the *'Show point'* button.

If it is chosen not to exclude any points, the toolbox displays a window which presents the final estimated values for the calibration parameters with their standard errors and the precision σ_o of the adjustment (Fig. 5).


Results

RESULTS

Central Projection Parameters

cx=670.643 ± 0.307 pixels
ar=0.999968 ± 0.000152
(cy=670.621 ± 0.303 pixels)
xo=-2.942 ± 0.414 pixels
yo=-1.364 ± 0.406 pixels
sk=-0.000940 ± 0.000145

Radial Distortion Parameters

k1=-4.092e-007 ± 4.86e-009
k2=9.596e-013 ± 5.02e-014

Decentering Distortion Parameters

p1=1.337e-006 ± 2.11e-007
p2=-6.164e-007 ± 2.05e-007

Sigma=0.168

Figure 5. Window with calibration results.

Then one can give a name to the file in which the results will be saved; else, results will be stored automatically in a *'results.txt'* file. The results file includes σ_o, values of all camera parameters with standard errors and correlations, the values of the 'exterior' orientation parameters and all image residuals. Included are also further useful details such as date, number and names of images, type of control, namely 'fixed' or 'observed' (in the second case the a priori precision estimates used for image and pattern coordinates are recorded), number of observations and of unknowns, degree of freedom, number of iterations and time elapsed. And, finally, the toolbox will plot the radial distortion curve (calibrated after Karras et al., 1998), as seen in Fig. 6.

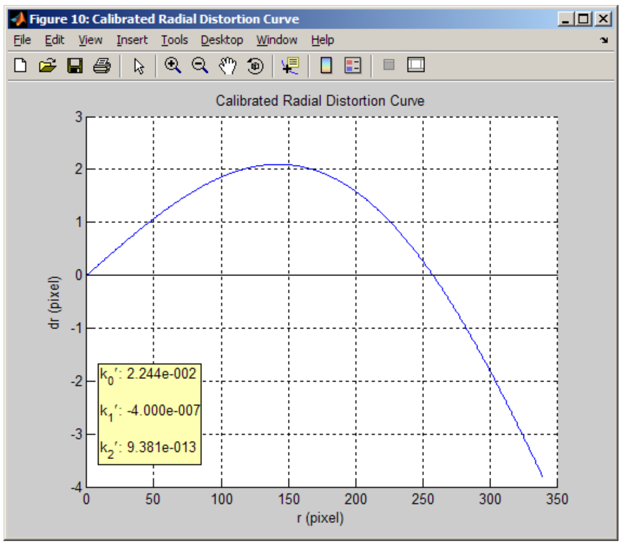


Figure 6. Calibrated curve of the radial lens distortion.

The above results and all examples in previous windows refer to a camera calibration process based on the image dataset (seen in Fig. 7) which is available to download, along with a results file, in the *FAUCCAL* website. All options had been set to default, except for the back-projection option which was set to '*on*'.

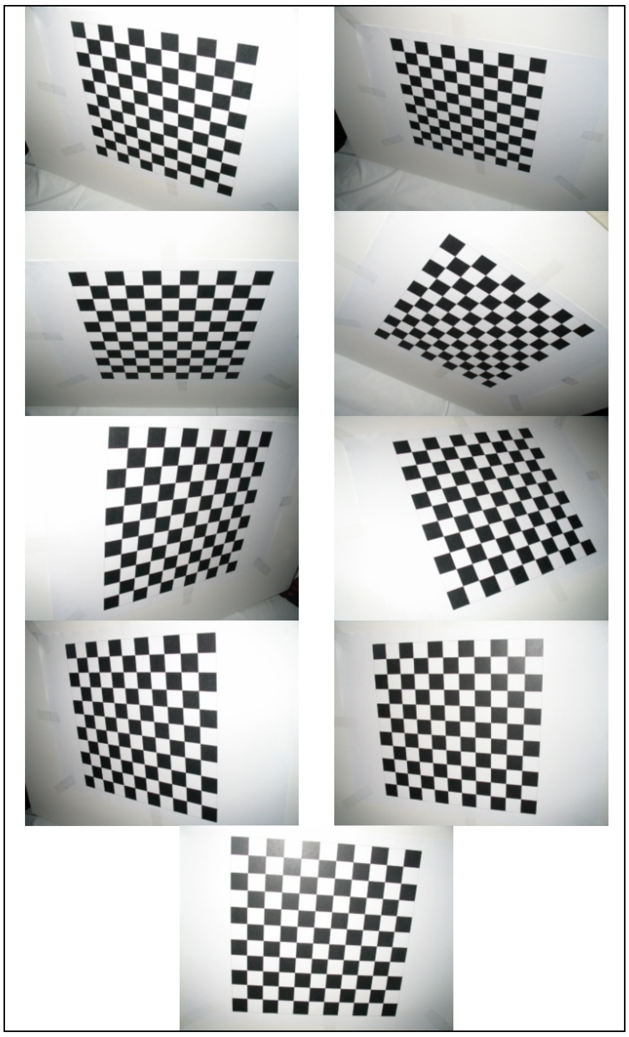


Figure 7. Image dataset available in the *FAUCCAL* website.

In the documentation at the *FAUCCAL* website, tips are given regarding the smooth functioning of the toolbox. Here it is merely pointed out that a pattern should consist of a sufficiently large number columns and rows (e.g. ≥ 8), however not too large (e.g. ≤ 20) since this might result in imaged tiles too small and, thus, confused with their neighbours as in Fig.8. This image also represents an example to be avoided. The pattern includes an excessively large number of nodes (33×27), i.e. the squares may appear too small on the images. Combined with strong perspective, this causes difficulties in distinguishing nodes from each other, mainly at the far end (particularly if the *'Resize images'* option is used). Besides, the pattern is not planar (see arrows).

4. CONCLUSION

A toolbox implemented in Matlab has been presented which, on the sole assumption that a squarely-tiled chess-board has been captured from different views, allows a camera to be calibrated in a fully automatic way. The authors have tested it with various data, including datasets available on the Internet, and it appears that if functions satisfactorily with any 'resonable' dataset. The source code can be downloaded, along with its documentation. The authors hope that this tool will be of use to those involved in cultural herirage documentation, and of course welcome any questions, critical comments and futher extensions of the code (Prokos et al., 2009, have modified it for the automatic calibration and relative orientation of a stereocamera).

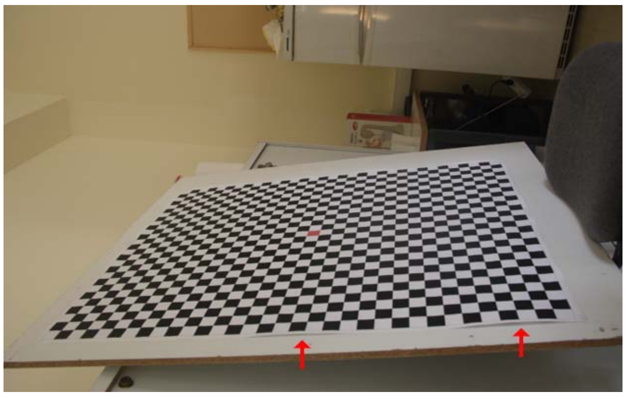


Figure 8. Image type not recommended.

REFERENCES

Bouguet J.-Y., Camera Calibration Toolbox for Matlab *http://www.vision.caltech.edu/bouguetj/calib_doc/*

DLR CalLab – CalDe Software: *http://www.dlr.de/rm/desktopdefault.aspx/tabid-4853/*

Douskos V., Kalisperakis I., Karras G., 2007. Automatic calibration of digital cameras using planar chess-board patterns. 8th Conf. Opt. 3-D Meas. Techn., Wichmann, vol. I, pp. 132-140.

Douskos V., Kalisperakis I., Karras G., Petsa E., 2008. Fully automatic camera calibration using regular planar patterns. Int. Arch. Phot. Rem. Sens., 37(B5), pp. 21-26.

Karras G., Mountrakis G., Patias P., Petsa E., 1998. Modeling distortion of super-wide-angle lenses for architectural and archaeological applications. Int. Arch. Phot. Rem. Sens., 32(B5), pp. 570-573.

Prokos A., Karras G., Grammatikopoulos L., 2009. Design and evaluation of a photogrammetric 3D surface scanner. In these Proceedings.

Remondino F., Fraser C., 2006. Digital camera calibration methods: considerations and comparisons. Int. Arch. Phot. Rem. Sens. & Spatial Sciences, 36(5), pp. 266-272.

Sturm P.F., Maybank S.J., 1999. On plane-based camera calibration: a general algorithm, singularities, applications. IEEE Conf. CVPR'99, Fort Collins, USA, pp. 432-37.

Zhang Z., 1999. Flexible camera calibration by viewing a plane from unknown orientations. IEEE. Conf. ICCV '99, Corfu, pp. 666-673.