3D Digitization: Camera Autocalibration

Guillaume Lemaître
Heriot-Watt University, Universitat de Girona, Université de Bourgogne
g.lemaitre58@gmail.com

I. Introduction

This paper present an Matlab implementation of the camera self-calibration proposed by Mendonca and Cipolla [1]. Only a main points of [1] which allow the implementation will be presented as well as results obtained using this method.

II. Camera self-calibration via the essential matrix

The method proposed by Mendonca and Cipolla [1] is based on the properties demonstrated by Huang and Faugeras that two of the three singular values of the essential matrix have to be equal knowing that the remaining singular value have to be null [2]. Huang and Faugeras demonstrated also that the equalities of the singular values imposed two algebraic constraints depending of the essential matrix [2].

Hartley and Richard exploited these properties between the fundamental matrix and the intrinsic parameters and the essential matrix [3]:

$$\mathbf{E}_{ij} \approx \mathbf{A}_j^T \mathbf{F}_{ij} \mathbf{A}_i \tag{1}$$

where $\mathbf{F}_{ij}$ is the fundamental matrix associated the image i and j and $\mathbf{A}_i$ and $\mathbf{A}_j$ are the intrinsic parameters of the camera i and j as:

$$\mathbf{A} = \begin{pmatrix} \alpha_x & s & u_0 \\ 0 & \epsilon\alpha_x & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

where α_x is the product of focal length and magnification factor, ϵ is the aspect ratio, $[u_0 \; v_0]^T$ are the coordinates of the principal point and s is the skew.

Mendonca and Cipolla introduced two cost functions which have to be minimized [1]. These cost functions are defined as:

$$C(\mathbf{A}_i, i = 1, ..., n) \quad = \quad \sum_{ij}^{n} \frac{w_{ij}}{\sum_{kl}^{n} w_k l} \frac{\sigma_{ij}^{(1)} - \sigma_{ij}^{(2)}}{\sigma_{ij}^{(2)}} \tag{2}$$

or

$$C(\mathbf{A}_i, i = 1, ..., n) \quad = \quad \sum_{ij}^{n} \frac{w_{ij}}{\sum_{kl}^{n} w_{kl}} \frac{\sigma_{ij}^{(1)} - \sigma_{ij}^{(2)}}{\sigma_{ij}^{(1)} + \sigma_{ij}^{(2)}} \tag{3}$$

where $\sigma_{ij}^{(1)}$ and $\sigma_{ij}^{(2)}$ are the non-zero singular values of the essential matrix and w_{ij} is the degree of confidence in the estimation of the fundamental matrix $\mathbf{F}_{ij}$. These cost functions are implemented as in appendix B.

This problem is solved by nonlinear least-square optimization. In our implementation, the *Levenberg-Marquart* algorithm is used to solve this non linear system (appendix A).

III. Results

The initial and rough intrinsic parameters are:

$$\mathbf{A} = \begin{pmatrix} 870 & 0 & 279 \\ 0 & 812 & 261 \\ 0 & 0 & 1 \end{pmatrix}$$

After applying the algorithm presented in the section II, the accurate intrinsic parameters found are:

$$\mathbf{A} = \begin{pmatrix} 800 & 0 & 256 \\ 0 & 800 & 256 \\ 0 & 0 & 1 \end{pmatrix}$$

IV. Conclusion

The method presented by Mendonca and Cipolla [1] allow to find of an accurate way the intrinsics parameters of camera.

References

[1] P. Mendonca and R. Cipolla, "A simple technique for self-calibration," in *CVPR99*, 1999, pp. 500–505.

[2] T. S. Huang and O. Faugeras, "Some properties of the e matrix in two-view motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, pp. 1310–1312, December 1989. [Online]. Available: http://portal.acm.org/citation.cfm?id=75598.75604

[3] R. I. Hartley, "Estimation of relative camera positions for uncalibrated cameras," in *Proceedings of the Second European Conference on Computer Vision*, ser. ECCV '92. London, UK: Springer-Verlag, 1992, pp. 579–587. [Online]. Available: http://portal.acm.org/citation.cfm?id=645305.648678

Appendix A
Autocalibration

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Guillaume Lemaitre
%%% -----------------------------------------------------------------------
%%% Camera autocalibration
%%% -----------------------------------------------------------------------
%%% A simple technique for Self-Calibration
%%% Paulo S.R. Mendoca and Roberto Cipolla
%%% Computer Vision and Pattern Recognition, 1999. IEEE Computer Society
%%% Conference on., Vol. 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Initialisation

clear all;
close all;
clc;

%%% Data type
format long;

%%% Add repository
addpath('data');

%%% Read data
load('data.mat');

%%% Display the intrinsic parameters matrix
disp('Rough Intrinsic parameters: ');
disp(A);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Nonlinear least-squares optimization

options = optimset('Algorithm','levenberg-marquardt');

%%% Optimization
int_params_lin = lsqnonlin('cost_function',[A(1,1) A(1,3) A(2,2) A(2,3)],[],[],options);

%%% Reshape
int_params = [int_params_lin(1)  0 int_params_lin(2) ; 0 int_params_lin(3) int_params_lin(4) ; 0 0 1];

disp('Intrinsic parameters: ');
disp(int_params);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B
Cost function

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Guillaume Lemaitre
%%% -----------------------------------------------------------------------
%%% Cost function
%%% -----------------------------------------------------------------------
%%% A simple technique for Self-Calibration
%%% Paulo S.R. Mendoca and Roberto Cipolla
%%% Computer Vision and Pattern Recognition, 1999. IEEE Computer Society
%%% Conference on., Vol. 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function cost = cost_function(params)

global Fs;

Aj = [params(1)  0  params(2) ; 0 params(3) params(4) ; 0 0 1];
```

```
Ai = [params(1)  0  params(2) ; 0 params(3) params(4) ; 0 0 1];

cost = 0.0;

for i=1:size(Fs,3)
    for j=i+1:size(Fs,4)

        [U,S,V] = svd(Aj'* Fs(:,:,i,j) * Ai);

        cost = cost + (S(1,1) - S(2,2)) / (S(1,1) + S(2,2));
        %cost = cost + (S(1,1) - S(2,2)) / (S(2,2));

    end
end
```