INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG2010/M17174
January 2010, Kyoto, Japan

Source **Poznan University of Technology,**
Chair of Multimedia Telecommunications and Microelectronics

Title **Preprocessing methods used for Poznan 3D/FTV test sequences.**

Authors Jakub Stankowski (jstankowski@multimedia.edu.pl),
Krzysztof Klimaszewski (kklima@et.put.poznan.pl),
Olgierd Stankiewicz (ostank@multimedia.edu.pl),
Krzysztof Wegner (kwegner@multimedia.edu.pl),
Marek Domanski (domanski@et.put.poznan.pl)

1 Introduction

This documents supplements our response [1] to call for test material – 3D/FTV test sequences and pre-processing methods for multiview sequences. All described methods are applied to Poznan test sequences (*Poznan_Hall1*, *Poznan_Hall2*, *Poznan_Street* and *Poznan_Carpark*) [1], but could be also applied for other sequences.

2 Geometric rectification

Rectified images are needed for proper depth map estimation. In our solution, we used chessboard calibration pattern and corners extraction algorithms. First step was to capture independent sequences for each camera and find chessboard corners coordinates. Coordinates obtained from every sequence was averaged to eliminate noise influence. Then intrinsic matrices (**I**) and distortion parameters (**D**) were calculated. These parameters were used in further processing.

For rectification, we used special calibration sequences, which contain chessboard pattern recorded simultaneously by all cameras. Translation (**T**) and rotation (**R**) matrices were calculated using known parameters (intrinsic, distortion) and corners coordinates from analyzed sequences.

Next operation was to calculate relative rotation ($\mathbf{R}_{rel}$) and translation ($\mathbf{T}_{rel}$) vectors between camera 0 and other cameras:

$$\mathbf{R}_{rel}\,(0 \rightarrow i) = \mathbf{R}(0)\ \cdot \mathbf{R}(i)^{-1},$$
$$\mathbf{T}_{rel}\,(0 \rightarrow i) = \mathbf{R}(0 \rightarrow i) \cdot \mathbf{T}(i) - \mathbf{T}(0),$$

where:
$\mathbf{R}_{rel}(0 \rightarrow i)$ – rotation matrix between camera 0 and camera i,
$\mathbf{R}(i)$ – camera i rotation matrix,
$\mathbf{T}_{rel}(0 \rightarrow i)$ – translation matrix between camera 0 and camera i,
$\mathbf{T}(i)$ – camera i translation matrix,
i – camera number.

Next, a new common coordinate system was found. This coordinate system has to be independent of camera 0 coordinate system. We did this by using linear regression, and tried to find a line nearest to optical centres of all cameras. This line was the x axis of the new coordinate system. Linear regression was calculated for y(x) and z(x) relation:

$$\mathrm{y(x)} = \mathrm{a}_y \cdot \mathrm{x} + \mathrm{b}_y,$$
$$\mathrm{z(x)} = \mathrm{a}_z \cdot \mathrm{x} + \mathrm{b}_z,$$

where:
a_y, a_z, b_y, b_z – respective linear regression coefficients.

Y- and z-intercepts (b_y, b_z) were used to create translation correction ($\mathbf{T}_c$) matrix:

$$\mathbf{T}_c = \begin{bmatrix} 0 \\ -\mathrm{b}_y \\ -\mathrm{b}_z \end{bmatrix}.$$

Slopes of the line (a_y, a_z) was used for calculation of rotation correction ($\mathbf{R}_c$) matrix. Angle of rotation around z axis (α_z) was calculated using a_y coefficient and rotation around y axis (α_y) was calculated using a_z coefficient:

$$\alpha_z = \tan^{-1} \mathrm{a}_y,$$
$$\alpha_y = \tan^{-1} \mathrm{a}_z.$$

Rotation angles were converted into rotation matrices, z-axis rotation matrix ($\mathbf{R}_z$) and y-axis rotation matrix ($\mathbf{R}_y$). Rotation correction ($\mathbf{R}_c$) was calculated using $\mathbf{R}_y$ and $\mathbf{R}_z$:

$$\mathbf{R}_c = \mathbf{R}_z \cdot \mathbf{R}_y$$

Relative translation matrices could be converted to the new coordinate system:

$$\mathbf{T}'_{rel}(0 \rightarrow i) = \mathbf{R}_c \cdot \mathbf{T}_{rel}(0 \rightarrow i) + \mathbf{T}_c,$$

where:
$\mathbf{T}'_{rel}(0 \rightarrow i)$ – translation matrix between camera 0 and camera *i* in new coordinate system
$\mathbf{T}_{rel}(0 \rightarrow i)$ – translation matrix between camera 0 and camera *i*
$\mathbf{R}_c$ – rotation correction matrix
$\mathbf{T}_c$ – translation correction matrix
i – camera number

Last correction parameter was average rotation matrix ($\mathbf{R}_a$), calculated by averaging relative rotation matrices for all cameras.

Next step was to find target cameras parameters – intrinsic, rotation and translation. Intrinsic matrix was calculated by averaging all camera's intrinsic matrices. Target translation ($\mathbf{T}_t$) and rotation ($\mathbf{R}_t$) matrices for first camera was calculated using known matrices and correction parameters:

$$\mathbf{T}_t(0) = \mathbf{R}_c \cdot (\mathbf{T}(0) + \mathbf{T}_c),$$
$$\mathbf{R}_t(0) = \mathbf{R}_a^{-1} \cdot \big(\mathbf{R}_c \cdot \mathbf{R}(0)\big),$$

where:
$\mathbf{T}_t(0)$ – target translation matrix for first camera,
$\mathbf{T}(0)$ – first camera's translation matrix,
$\mathbf{R}_t(0)$ – target rotation matrix for first camera,
$\mathbf{R}(0)$ – first camera's rotation matrix,
$\mathbf{T}_c$ – translation correction matrix,
$\mathbf{R}_c$ – rotation correction matrix,
$\mathbf{R}_a$ – average rotation matrix.

Target intrinsic and rotation matrices for all cameras were the same. Translation matrices for other cameras were calculated by moving theirs optical centres along x axis by defined step – average distance between neighbouring cameras.

With use of the target cameras parameters, we have calculated projection of points that correspond to the corners of calibration pattern. We also have selected four points from corners of calibration pattern, and corrected theirs coordinates to remove distortion influences. Having two groups of four points (real and target), we were able to find perspective transformation for these points.

Example of two groups of points, real and target, are shown below (Figure 1,2):

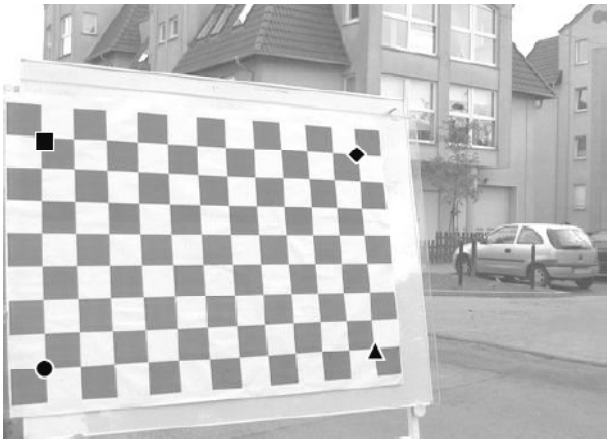

Figure 1. Image of four real corners of calibration pattern

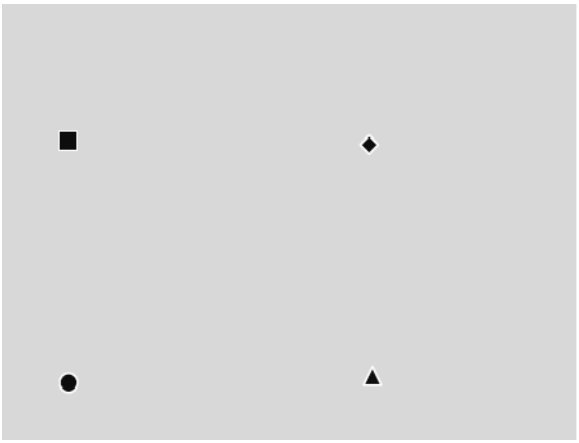

Figure 2. Image of four target corners corresponds to real ones

Final part of the described solution was image transformation. The first step consisted of distortion removal with use of known intrinsic and distortion matrices. Then, a perspective transformation was applied to images.

Correctness of rectification was verified by manual check and by generating depth maps for rectified sequences using Depth Estimation Reference Software 3.5 (DERS 3.5). Manual checking was done by comparing views from different cameras. Both verification methods yielded the conclusion, that geometrical correction was accurate enough.

Examples of rectified images are shown below (Figure 3,4):

Figure 3. Rectified image from camera 0

Figure 4. Rectified image from camera 8

3 Color correction

Depth estimation is very sensitive for color mismatch. Although, our test sequences have been recorded with high-class calibrated cameras and produce subjectively almost indistinguishable color profiles, color correction was required.

In our algorithm, views' color profiles are adjusted to center reference view. Color calibration is histogram-based and is performed on RGB color components independently.

In order, to reduce influence of differences in viewing areas between cameras, only neighboring views' histograms are compared for calibration. The following steps are performed in order to produce color corrected sequence:

1. Histograms $H_{v,c}(i)$ of each input view $v \in [0..8]$ are generated, for all three RGB components $c \in [R,G,B]$.

2. Histograms of input views neighboring with each other are compared and calibration parameters are estimated.

   Color adjustment parameter $p_{adj,c}$ of view being adjusted *adj* directing to the neighboring reference view *ref,* for component *c* is calculated as:

$$p_{adj,c} = \frac{\sum_i H_{adj,c}(i) \cdot i}{\sum_i H_{ref,c}(i) \cdot i}$$

   Neighboring reference view *ref* always resides into the direction of center reference view.
   Parameters $p_{adj,c}$ describe function which corrects colors of view beging adjusted *adj* to the colors of neighboring reference view *ref*.

$$I'_{adj,c}(x,y) = I_{adj,c}(x,y) \cdot p_{adj,c}$$

   Where $I_{v,c}(x,y)$ was an image component *c* of view *v*.

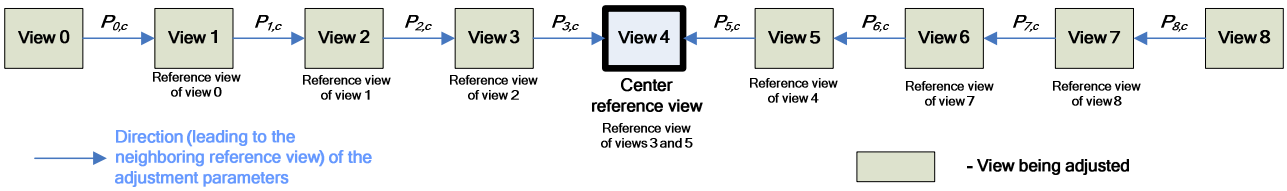

Figure 5. Color adjustment between neighboringviews.

3. Basing on the adjustment parameters between neighboring views, final adjustment parameters $f_{adj,c}$ (leading directly to the center reference view, Figure 6) are calculated.

$$f_{adj,c} = \prod_{x \in (adj \to ref)} p_{adj,c}$$

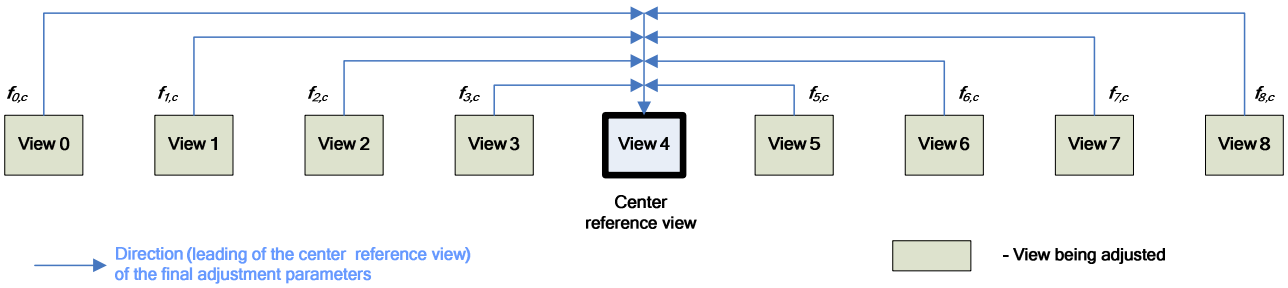

Figure 6. Final color adjustment parameters.

Final adjustment parameters $f_{adj,c}$ are used to transform sequences being calibrated into a color profile of the center reference view.

4 Rectification and correction of the test sequences

The following steps were performed on each of Poznan sequences:

1. Acquisition of calibration pattern.
2. Acquisition of sequences (resolution 1920x1080).
3. Estimation of camera parameters for each RGB component separately, to reduce chromatic aberration.
4. Rectification of sequences (each RGB component separately) and reprojection into resolution 1920x1088.
5. Estimation of color-space profiles.
6. Color correction of sequences.

Please note that the original *Poznan_Hall* sequence has been split into two parts (*Poznan_Hall1* and *Poznan_Hall2*) in order to avoid problems with the pilar depth impressions.
The following sequences are available:

1. Poznan_Hall sequence

   a) original 1920x1080, 9 cameras, 600 frames
   b) rectified 1920x1088, 9 cameras, 600 frames

   In order to avoid problems with the pillar depth impressions, this has been split into two parts, which we recommend to use during the EE.

   c) Poznan_Hall1, rectified, 1920x1088, 200 frames, 5 cameras (frames 0..199 and cameras 0..4 extracted from original Poznan_Hall sequence).
   (we recommend to use views 1,2 (2-view case) and 1,2,3 (3-view case) in the EE)

   d) Poznan_Hall2, rectified, 1920x1088, 200 frames, 5 cameras (frames 350..549 and cameras 4..8 extracted from original Poznan_Hall sequence).
   (we recommend to use views 6,7 (2-view case) and 5,6,7 (3-view case) in the EE)

2. Poznan_Street sequence

   a) original 1920x1080, 9 cameras, 450 frames
   b) rectified 1920x1088, 9 cameras, 450 frames

   For the EE, we recommend to use frame range [150..349], views 3,4 (2-view case) and 3,4,5 (3-view case)

3. Poznan_Carpark sequence

   a) original 1920x1080, 9 cameras, 600 frames
   b) rectified 1920x1088, 9 cameras, 600 frames
   For the EE, we recommend to use frame range [200..399], views 3,4 (2-view case) and 3,4,5 (3-view case).

Split of *Poznan_Hall* sequence (and view selection) implied changes in configuration for 2-view and 3-view cases of Exploration Experiment described in [2]. Corrected versions of Table 1 and Table 2 (from [2]) for Poznan sequences are shown below (modified fields are grayed).

Table 1: Input and output views for MVD representation format in 2-view configuration

Data set	Original Pair OL-OR	Synthesized Pair SL-OR (OL-SR)	Frame Range for EE1
Poznan_Hall1	1-2	1.5-2	0~199 (0~199 from original Poznan_Hall)
Poznan_Hall2	6-7	6.5-7	0~199 (350..549 from original Poznan_Hall)
Poznan_Street	3-4	3.5-4	150 ~349
Poznan_Carpark	3-4	3.5-4	200~399

Table 2: Input and output views for MVD representation format in 3-view configuration

Data set	Original Views OL-OC-OR	Views to Synthesize for stereo viewing[1]	Views to Synthesize for 9-view display[2]
Poznan_Hall1	1-2-3	1.875, 2.125	1.5, 1.625, 1.75, 1.875, 2.125, 2.25, 2.375, 2.5
Poznan_Hall2	5-6-7	5.875, 6.125	5.5, 5.625, 5.75, 5.875, 6.125, 6.25, 6.375, 6.5
Poznan_Street	3-4-5	3.875, 4.125	3.5, 3.625, 3.75, 3.875, 4.125, 4.25, 4.375, 4.5
Poznan_Carpark	3-4-5	3.875, 4.125	3.5, 3.625, 3.75, 3.875, 4.125, 4.25, 4.375, 4.5

5 Summary

- Poznan test sequences have been rectified and color corrected,
- Resolution of corrected sequences is 1920x1088,
- *Poznan_Hall* sequence has been split into *Poznan_Hall1* and *Poznan_Hall2*,
- Adaptation of EE configuration [2] for 2-view and 3-view case is needed for *Poznan_Hall* sequences,
- The sequences are available on FTP site.

References

[1] M. Domanski, T. Grajek, K. Klimaszewski, M. Kurc, O. Stankiewicz, J. Stankowski, K. Wegner, "Poznan Multiview Video Test Sequences and Camera Parameters", ISO/IEC JTC1/SC29/WG11 m17050, Xian, China, October 2009.

[2] "Description of Exploration Experiments in 3D Video Coding" ISO/IEC JTC1/SC29/WG11 N10926, Xian, China, October 2009.

[1] The selected baseline distances correspond to emulation of a 22-view display on a stereo display.

[2] The views to synthesize for a multiview display are subject to change according to specific display requirements that would be available at a given meeting.