Stettbacher Signal Processing
Margrit Rainer Strasse 12a
CH-8050 Zürich

Phone: +41 43 299 57 23
Fax: +41 43 299 57 25
E-Mail: cam@stettbacher.ch

O-3000 Camera Series

Image Frame Format

Version 1.20
2013-05-24

Abstract: This document describes the image data format as sent by the O-3000 camera series.

Contents

1 Purpose

Each image frame sent by the O-3000 camera, either in streaming or snapshot mode, is prefixed with an image header, describing the format and dimension of the video frame following the header.

This document describes the image data format, consisting of the header and the image data. For the header (version 1), the corresponding fields are detailed. Understanding the format and ability to parse this header is crucial to any user application.

2 The Image Header

The image header is defined in the C header file image_header.h freely available with the O-3000 camera series driver package (see [1]). The individual fields are described in table 1. All fields are of type uint32_t (unsigned integer, 4 bytes), which is also the base for the offset, in little endian byte order. The header size is 512 bytes; unused space is padded. Treat padding values as undefined for future compatibility.

Offset	Len	Field	Description
0	2	preamble	Preamble; search for the signature 0xaa55deadbeef55aa to identify the start of frame.
2	1	version	Header version, current version is 1.
3	1	payload_size	Size (in bytes) of the payload following the header; does not include header size.
4	1	image_start	Offset of actual image data within the payload (in bytes).
5	1	image_size	The size of the actual image (in bytes).
6	1	width	Image width (in pixels).
7	1	height	Image height (in pixels).
8	1	format	Image data format descriptor (see table [tbd]).
9	1	frame_count	Frame sequence number.
10	118	padding	Padding/stuffing bytes; treat as undefined data.

Table 1: Image header fields.

3 Payload Size vs Image Size

The O-3000 cameras may send statistics or histogram data embedded to the image. This data is contained in form of lines added before and/or after the actual image lines, and must be skipped therefore. The field *image_start* tells, how many bytes should be skipped after the header to get the actual start of the image. The field *image_size* tells how many bytes from the offset *image_start* are valid image data bytes. The relationship is depicted in figure 1. The payload arrives in little endian byte order too.

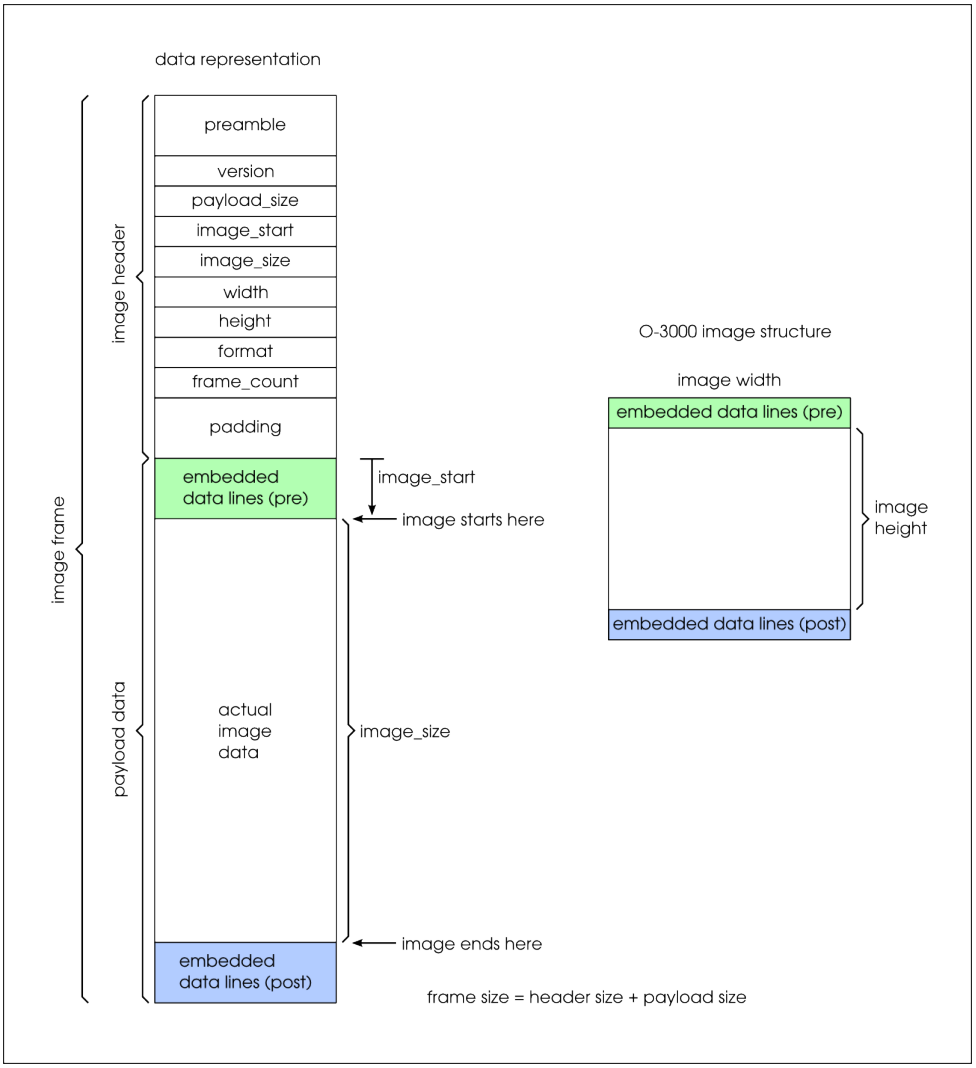

Figure 1: Frame structure diagram.

4 Data Format Descriptor

The data format descriptor (see table 2) indicates, in which format the images are sent. This information is required to identify how many bytes one pixel occupies and how to interpret the pixel values. For instance, data in RGB bayer format has to be transformed using a demosaicing algorithm.

Identifier	Properties	Values
DF_RAW_MONO_8	Color filter array: Quantisation: Bytes/pixel: Format:	monochrome 8 bits/pixel 1 raw/linear
DF_RAW_MONO_12	Color filter array: Quantisation: Bytes/pixel: Format:	monochrome 12 bits/pixel 2 (data right aligned) raw/linear
DF_RAW_BAYER_8	Color filter array: Quantisation: Bytes/pixel: Format:	RGB bayer 8 bits/pixel 1 raw/linear
DF_RAW_BAYER_12	Color filter array: Quantisation: Bytes/pixel: Format:	RGB bayer 12 bits/pixel 2 (data right aligned) raw/linear
DF_HDR_MONO_20_COMP	Color filter array: Quantisation: Bytes/pixel: Format:	monochrome 20 bits/pixel (compressed to 12 bits/pixel) 2 (data right aligned) HDR
DF_HDR_BAYER_20_COMP	Color filter array: Quantisation: Bytes/pixel: Format:	RGB bayer 20 bits/pixel (compressed to 12 bits/pixel) 2 (data right aligned) HDR

Table 2: Data format descriptors.

5 Image Data Format

The image data area following the header is organised in a way described by the data format descriptor. Three different formats are provided.

5.1 8 Bits per Pixel

In this format, every pixel on the camera is represented by 1 byte (8 bits). No padding takes place. Figure 2 illustrates that. Note that the notation $P_{1,2}$ means the pixel at row 1, column 2 of the image matrix. This format is used for DF_RAW_MONO_8 and DF_RAW_BAYER_8.

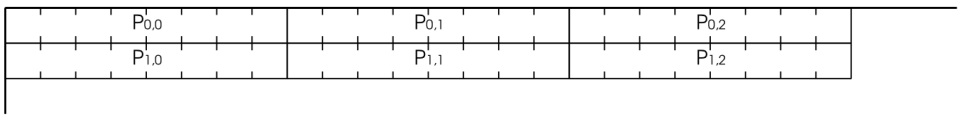

Figure 2: 8 Bits per pixel data format.

5.2 12 Bits per Pixel

Every pixel on the camera is represented by 2 bytes (16 bits), with only 12 significant bits (bits 0 to 11). The upper 4 bits are fixed to 0 as shown in figure 3. This format is used for DF_RAW_MONO_12 and DF_RAW_BAYER_12.

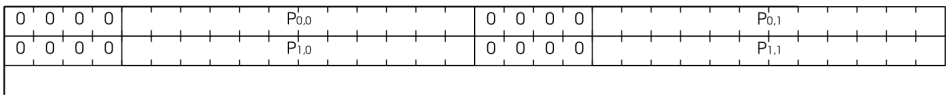

Figure 3: 12 Bits per pixel data format.

5.3 12 Bits per Pixel (HDR)

Similar to 5.2, every pixel is represented by 2 bytes with 12 significant bits. These 12 bits however, are compressed from a 20 bits HDR pixel value. The relation between uncompressed

and compressed intensity values are shown in figure 4 and equation 1. This format is used for DF_HDR_MONO_20_COMP and DF_HDR_BAYER_20_COMP.

To get the decompressed pixel intensity P_{out} for a given compressed pixel intensity P_{in}, the following formula can be used:

$$P_{out} = \begin{cases} P_{in} & \text{if} \quad 0 \leq P_{in} \leq 2048 \\ (P_{in} - 2016) \cdot \frac{63488}{992} & \text{if} \quad 2048 < P_{in} \leq 3040 \\ (P_{in} - 2976) \cdot \frac{983040}{960} & \text{if} \quad 3040 < P_{in} \end{cases} \tag{1}$$

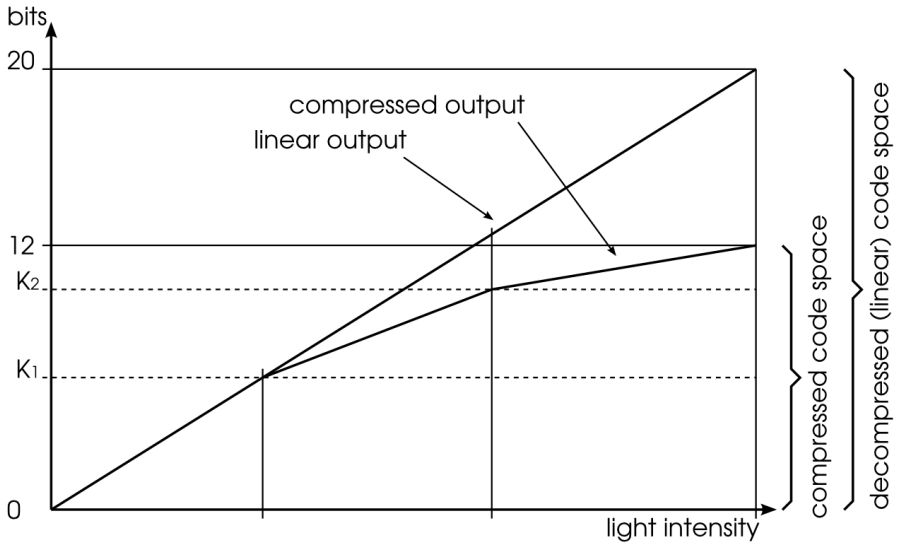

Figure 4: HDR compression.

6 Filter pattern

Color versions of the O-3000 Camera feature a RGB color filter array (see figure 5), which requires to run a demosaicing algorithm over the image retrieved from the camera. Please refer to [2] for a detailled explanation of the principle. For reconstruction of an RGB pixel, the adjacent pixels have to be taken into account. The intensity value of the current and the (weighted) intensity values of the adjacent pixels are summed and distributed to the three color channels of the current pixel. Depending on the requirements, either only the pixels directly adjacent to the current pixel are regarded (3x3 interpolation), or even a larger area is taken into account (e.g. 5x5 interpolation), where weighting decreases with increasing distance from the center (current) pixel. It is up to the user to implement the algorithm, or to use a third-party library like OpenCV (see [3]), which offers out-of-the-box functions.

For monochrome camera versions, no demosaicing has to be performed.

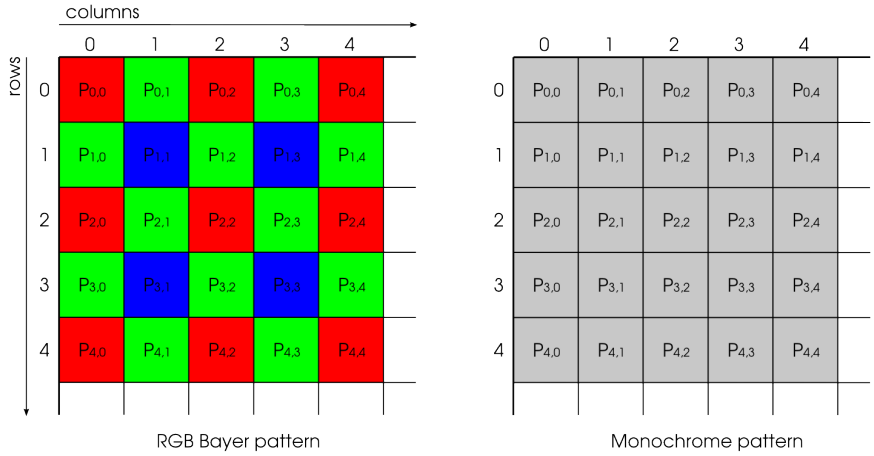

Figure 5: Color filter array versus monochrome pattern.

7 Literature and References

[1] Stettbacher Signal Processing Website, http://www.stettbacher.ch/cameras

[2] Wikipedia entry on Bayer filters, http://en.wikipedia.org/wiki/Bayer_filter

[3] OpenCV Project, http://opencv.org