Rolling Shutter Correction in 3DE4

SDV - public
compiled on 23.07.2010

1 *Introduction*

In this document we describe the parameter for 3DE4's rolling shutter functionality. Rolling shutter was introduced in release 1 beta 8.

2 *User Interface*

In the current implementation rolling shutter is modelled by one single parameter *Framewise Time Shift* which is located in the *Camera Panel*, section *Rolling Shutter Compensation* of 3DE4's Attribute Editor. 3DE4 allows optimization of this parameter by means of the *Parameter Adjustment Window*. Alternatively, the parameter can be entered by hand.

3 *Mathematical description*

In the following we consider a filmback given in length units (usually mm) with dimensions

$$w \times h \tag{1}$$

where positions are denoted by

$$(x, y), x \in [0, w], y \in [0, h]. \tag{2}$$

In pixel we denote the filmback by

$$w_{\text{px}} \times h_{\text{px}}. \tag{3}$$

We choose our coordinates as follows: Let (0,0) be the point in the filmback where the scanning begins and (w,h) the point where it ends. Assume we are recording a sequence of images indexed by j = 0, 1, 2,... at a frame rate r given in units sec^{-1}. Let us assume frame 0 is recorded at a time t_0. In an ideal non-rolling shutter scenario with shutter time tending towards zero, each frame j represents a point in time

$$t_j = t_0 + \frac{j}{r} \tag{4}$$

with frame rate r. In order to deal with tracking curves at non-equidistant moments in

time we will use the following notation: A tracking point is given by a triple [*x*,*y*,*t*], which means it has position (*x*,*y*) on the screen/filmback and was recorded at time *t*. We can simply perform calculations on these triples as if they were points in three-dimensional space. Given a feature moving through three-dimensional space at time-dependent position *p*(*t*) with respect to the camera, the projection onto the filmback and time read

$$[x(t), y(t), t] \tag{5}$$

Without rolling shutter the track is simply a sequence like

$$[x(t_j), y(t_j), t_j], \quad j = 0, 1, 2 \ldots \tag{6}$$

Now, with rolling shutter the entire filmback is not scanned in one single step but is scanned within a finite time interval from top to bottom (or vice versa). That means, the feature in three-dimensional space with projection (*x*,*y*) is recorded at a time

$$t = t_j + f(x, y) \tag{7}$$

That means, our tracking curve is a sequence

$$[x(t), y(t), t], \quad t = t_j + f(x(t), y(t)), \quad j = 0, 1, 2 \ldots . \tag{8}$$

The unpleasant thing from the point of view of a three-dimensional tracking system (as well as for compositing and rendering) is, that the tracking points for a given frame do not represent a single point in time, which leads to the well-known artefacts of rolling shutter shots. Therefore, in order to compensate for rolling shutter we have to model the delay function *f*(*x*,*y*) and correct the tracking points accordingly. What we need is

$$[x(t_j), y(t_j), t_j]. \tag{9}$$

In the following we shall find an expression for the delay function and formulate a correction scheme for tracking curves.

3.1 Scenario 1: *y*-dependency

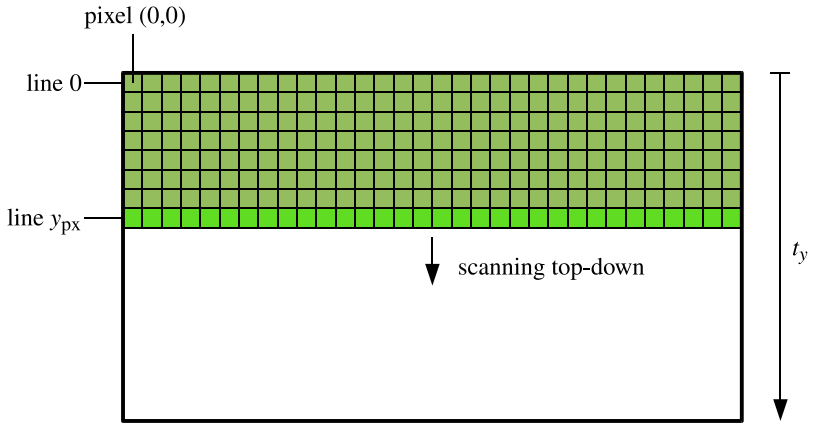
This scenario is fairly simple, and it can be formulated independently from image resolution and therefore we use it in 3DE4. In this simple model we assume, that lines on the filmback are scanned consecutively, but the pixels of one line are scanned simultaneously. Our delay function then reads

$$f(x, y) = t_y \frac{y}{h} = t_y \frac{y_{\mathrm{px}}}{h_{\mathrm{px}}} \tag{10}$$

where for line y_{px} we set

$$y = y_{\mathrm{px}} \frac{h}{h_{\mathrm{px}}} \tag{11}$$

The parameter t_y denotes the time required for scanning the entire filmback, as indicated in the figure below. It is referred to as *Framewise Time Shift* in section "Rolling Shutter Compensation" of 3DE4's Attribute Editor.



3.1.1 *Implementation in 3DE4*

3DE4's calculation core is based on the principle that each frame represents one point in time, as if there was no rolling shutter. Therefore the task in 3DE4 is to compensate the effect of rolling shutter. Given a tracking point at position (*x*,*y*) in frame *j*, by using the model presented above we know that this position belongs to a time

$$t = t_j + f(x,y) = t_j + t_y \frac{y}{h} \tag{12}$$

We are given a tracking curve from a track of rolling shutter images. Let us denote the space/time positions for this track by

$$[x_j^{\mathrm{rs}}, y_j^{\mathrm{rs}}, t_j^{\mathrm{rs}}], \quad j = 0, 1, 2 \ldots \tag{13}$$

where

$$t_j^{\mathrm{rs}} = t_j + f(x_j^{\mathrm{rs}}, y_j^{\mathrm{rs}}) \tag{14}$$

Our task is to estimate the position

$$[x(t_j), y(t_j), t_j] \tag{15}$$

of this tracking point for time t_j. We do this by interpolating between the tracking positions of three consecutive frames

$$[x_{j-1}^{\mathrm{rs}}, y_{j-1}^{\mathrm{rs}}, t_{j-1}^{\mathrm{rs}}] \tag{16}$$

and

$$[x_j^{\mathrm{rs}}, y_j^{\mathrm{rs}}, t_j^{\mathrm{rs}}] \tag{17}$$

and

$$[x_{j+1}^{\mathrm{rs}}, y_{j+1}^{\mathrm{rs}}, t_{j+1}^{\mathrm{rs}}]. \tag{18}$$

We interpret these triples as three points in a 2+1-dimensional space-time and calculate a parameter curve, more precise a 2+1-tuple-valued parabola

$$p : [-1, +1] \subset \mathrm{R} \to \mathbf{R}^{2+1} : \lambda \mapsto [p_x(\lambda), p_y(\lambda), p_t(\lambda)] \tag{19}$$

having the following properties

$$\begin{aligned} p(-1) &= [x_{j-1}^{\mathrm{rs}}, y_{j-1}^{\mathrm{rs}}, t_{j-1}^{\mathrm{rs}}] \\ p(0) &= [x_j^{\mathrm{rs}}, y_j^{\mathrm{rs}}, t_j^{\mathrm{rs}}] \\ p(+1) &= [x_{j+1}^{\mathrm{rs}}, y_{j+1}^{\mathrm{rs}}, t_{j+1}^{\mathrm{rs}}]. \end{aligned} \tag{20}$$

Now, let λ be the point in [−1,+1] where

$$\mathrm{p}_t(\lambda) = t_j. \tag{21}$$

Then we interpolate the position as

$$[x(t_j), y(t_j), t_j] = p(\lambda) \tag{22}$$

and that's it. Note, that this implementation also gives an interpretation to negative values for t_y. A negative value corresponds to the opposite scanning direction.

3.2 Scenario 2: *x-y*-dependency

For reasons of completeness we mention here a more general (and yet reasonable) *ansatz* by using an *x-y*-dependent delay function *f*(*x*,*y*). We can imagine some "ray" running from left to right on the recording chip, and passing from one scan line to the next. The model we describe here can only be formulated by incorporating image resolution. We need two parameters for this model:

1. Horizontal scanning time t_x, time required for scanning w_{px} pixels
2. Vertical scanning time t_y, time required for scanning h_{px} rows of pixels

Given pixel position (x_{px}, y_{px}) with x_{px} in 0...w_{px}-1 and y_{px}-1 in 0...h_{px}-1. Pixel (*x*,*y*) is scanned later than pixel (0,*y*). If the entire line is scanned in a time interval t_x, pixel (*x*,*y*) is scanned at

$$f(x_{\mathrm{px}}, y_{\mathrm{px}}) = f(0, y_{\mathrm{px}}) + t_x \frac{x_{\mathrm{px}}}{w_{\mathrm{px}}} \tag{23}$$

For the vertical delay we define t_y as the time required for scanning the entire chip.

Then for any horizontal position x for our delay function we have

$$f(x_{\mathrm{px}}, y_{\mathrm{px}}) = f(x_{\mathrm{px}}, 0) + t_y \frac{y_{\mathrm{px}}}{h_{\mathrm{px}}} \tag{24}$$

In total, our delay function reads:

$$f(x_{\mathrm{px}}, y_{\mathrm{px}}) = f(0, 0) + t_x \frac{x_{\mathrm{px}}}{w_{\mathrm{px}}} + t_y \frac{y_{\mathrm{px}}}{h_{\mathrm{px}}} \tag{25}$$

with $f(0,0) = 0$ by definition. Note, that it is very important to formulate the delay in terms of pixel. In this model (i.e. if horizontal delays are considered to be relevant) using down-scaled proxy images falsifies the delay function: If an image is downscaled by, say, a factor 2, consecutive lines are blended. The content of two vertically adjacent pixels belongs to two different points in time. By doing so we override the horizontal rolling shutter correction (the t_X term) by some orders of magnitude. In this sense, scenario 2 and proxy images are incompatible and therefore we implemented the y-dependent delay as described in scenario 1.