

The onboard optic flow algorithm uses the [temporal filtered Lucas Kanade method](#) . The presented implementation is focused on efficiency and uses integer arithmetic. The following pseudo-code illustrates the procedure:

P, n, pixel, GB, GBt=(0,0,0,0,0)

for x, y in P

gx = pixel[x+1,y] – pixel[x-1,y]

gy = pixel[x, y+1] – pixel[x, y-1]

gt = pixel[x, y] – pixel_old[x, y]

GBt += (gx * gx, gx * gy, gy * gy, gx * gt, gy * gt)

GB -= (GB(0) >> n, GB(1) >> n, GB(2) >> n, GB(3) >> n, GB(4) >> n)

GB += (GBt(0) >> n, GBt(1) >> n, GBt(2) >> n, GBt(3) >> n-1, GBt(4) >> n-1)

det = GB(0) * GB(2) – GB(1) * GB(1)

u = (GB(2)*GB(3)-GB(1)*GB(4))/det;

$$v = (GB(0)*GB(4)-GB(1)*GB(3))/det;$$

The sensor data for the current and previous time step is contained in *pixel* and *pixel_old*, respectively.

P
denotes the spatial integration patch and

GB
contains the integrated values for

G
and
B

.

Since

G
is symmetric it is sufficient to store 3 values for
G

instead of 4. The multiplications required for the temporal filtering are implemented as bitwise shift operations ($\gg n$) which limits the possible

α
values for the filter but improves performance.

Note that GB contains signed valued which requires an arithmetic shift with sign extension instead of a logic shift

. A shift value of 4 corresponds to a division by 16 (
 α

=
0.0625). In the computation of the spatial gradients a divisor of 2 is omitted. These factors cancel out in the final solving step up to a factor of 2 for the temporal gradient which is accounted for in the integration step on line 8.

[back to Controller code overview](#)

[back to Readout board optic flow software](#)