

### Algorithms Overview

#### Theory

The optic flow computation from Lucas and Kanade is based on the image brightness constancy assumption which states that for a motion  $(u,v)$  of a point in an image  $I$  the brightness of the point does not change:

$$I(x, y, t) = I(x+u, y+v, t+1)$$

Using first order Taylor expansion leads to the gradient constraint equation

$$I_x u + I_y v + I_t = 0$$

This underdetermined system is solved by a least squares estimate over an image patch given by

$$(u \ v)^T = [\sum (I_x \ I_y)^T (I_x \ I_y)]^{-1} \sum I_t (I_x \ I_y)^T$$

#### References

Lucas, B. D., & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. International joint conference on artificial intelligence (Vol. 3, pp. 674–679).

Baker, S., & Matthews, I. (2004). Lucas-Kanade 20 years on: A unifying framework: Part 1: The quantity approximated, the warp update rule, and the gradient descent approximation. *International Journal of Computer Vision*, 56(3), 221–255.

Bouguet, J. (1999). Pyramidal Implementation of the Lucas-Kanade Feature Tracker Description of the algorithm. Intel Corporation, Microprocessor Research Labs., 1(2), 1-9.

Bruhn, A., Weickert, J., & Schnörr, C. (2005). Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods. *International Journal of Computer Vision*, 61(3), 1-21.

Shi, J., & Tomasi, C. (1994). Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*.

Tomasi, C., & Kanade, T. (1991). Detection and tracking of point features. *Image* (Rochester, N.Y.).

### Algorithm

An implementation of the described algorithm using integer or floating point arithmetics is provided in:

["visual\\_processing/optic\\_flow/LucasKanade.hpp"](#) " ["visual\\_processing/optic\\_flow/LucasKanade.cpp"](#)

The sample code computes a single optic flow vector over an image patch. Different kernels can be used for the spatial gradient computation. The method returns the estimated optic flow vector as a homogeneous vector  $(x, y, d)$ .

An example for using the sample code is given in

" [visual\\_processing/optic\\_flow/LucasKanade\\_test.cpp](#) "