TurboPixels

TurboPixels is a geometric-flow based algorithm for computing the superpixels of an image. It produces segments that on one hand respect local image boundaries, while on the other hand limit under-segmentation through a compactness constraint. This algorithm can yield less under-segmentation than algorithms that lack a compactness constraint, while offering a significant speed-up over N-cuts, which dose enforce compactness.

Inspired by active contours, after placing superpixel centers on a regular grid, the superpixels are grown based on an evolving contour. The contour is implemented as level set of the function:

The evolution is formally defined by: . Where denotes the gradient of  and  is the temporal derivative. Here, the speed describes the future evolution of the contour. In practice,  will be the signed euclidean distance and evolution is carried out using a first-order discretization. The contour in iteration (T+1) is given by

.

The speed  is split up into two components:  which depends on the image content and  which ensures that superpixels do not overlap. Iteratively, the superpixels are grown by computing  and  and then applying the equation above, see the paper for details. The procedure is summarized in algorithm below.

function turbopixels (

I, // Color Image

K, // Number of superpixels)

Place initial superpixel centers on a regular grid

Initialize

Repeat {

Compute

Evolve the contour by computing

Update assigned pixels

T = T + 1

} until all the pixels are assigned

Derive superpixels segmentation S from

Return S

This algorithm is guided by five basic principles:

1. Uniform size and coverage
2. Connectivity
3. Compactness
4. Smooth, edge-preserving flow
5. No superpixel overlap.