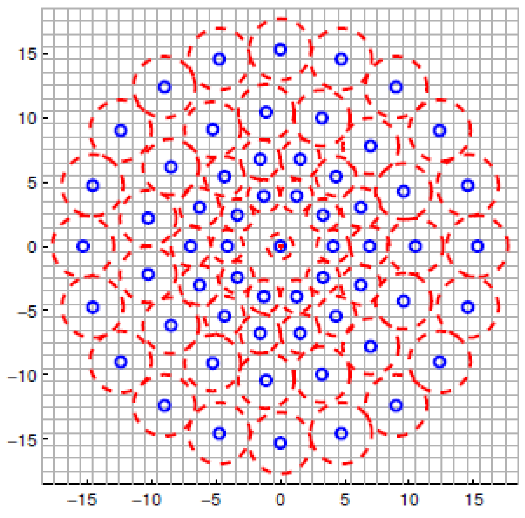
BRISK: Binary robust invariant scalable keypoints

Recall that to build the binary string representing a region around a keypoint we need to go over all the pairs and for each pair (p1, p2) – if the intensity at point p1 is greater than the intensity at point p2, we write 1 in the binary string and 0 otherwise.

The BRISK descriptor is different from the descriptors we talked about earlier, BRIEF and ORB, by having a hand-crafted sampling pattern. BRISK sampling pattern is composed out of concentric rings:



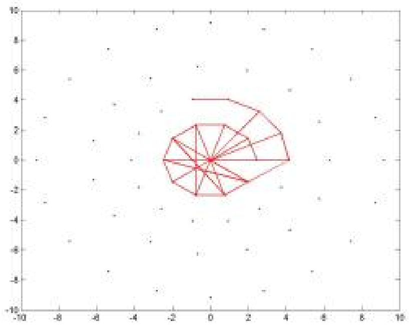
When considering each sampling point, we take a small patch around it and apply Gaussian smoothing. The red circle in the figure above illustrations the size of the standard deviation of the Gaussian filter applied to each sampling point.

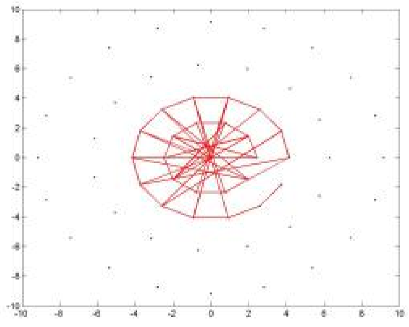
Short and long distance pairs

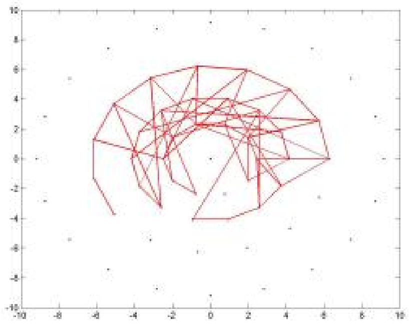
When using this sampling pattern, we distinguish between short pairs and long pairs. Short pair are pairs of sampling points that their distance is below a certain threshold d_max and long pairs are pairs of sampling points that their distance is above a certain different threshold d_min, where d_min>d_max, so there are no short pairs that are also long pairs.

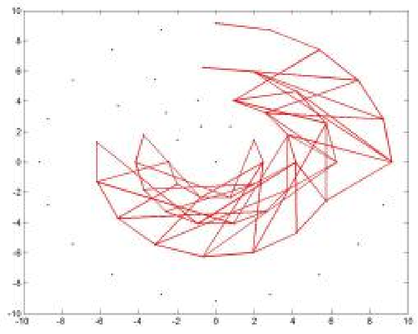
Long pairs are used in BRISK to determine orientation and short pairs are used for the intensity comparisons that build the descriptor, as in BRIEF and ORB. The

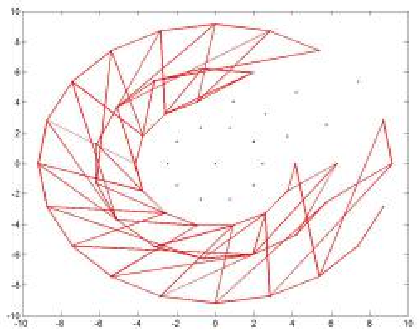
To illustrate this and help make things clear, here are figures of BRISK's short pairs – each red line represent one pair. Each figure shows 100 pairs:











Computing orientation

BRISK is equipped with a mechanism for orientation compensation; by trying to estimate the orientation of the keypoint and rotation the sampling pattern by that orientation, BRISK becomes somewhat invariant to rotation.

For computing the orientation of the keypoint, BRISK uses local gradients between the sampling pairs which are defined by

$$\mathbf{g}(\mathbf{p}_i, \mathbf{p}_j) = (\mathbf{p}_j - \mathbf{p}_i) \cdot \frac{I(\mathbf{p}_j, \sigma_j) - I(\mathbf{p}_i, \sigma_i)}{\|\mathbf{p}_j - \mathbf{p}_i\|^2}$$

Where g(pi,pj) is the local gradient between the sampling pair (pi,pj), I is the smoothed intensity (by a Gaussian) in the corresponding sampling point by the appropriate standard deviation (see the figure above of BRISK sampling pattern).

To compute orientation, we sum up all the local gradients between all the long pairs and take arctan(gy/gx) – the arctangent of the the y component of the gradient divided by the x component of the gradient. This gives up the angle of the keypoint. Now, we only need to rotate the short pairs by that angle to help the descriptor become more invariant to rotation. Note that BRISK only use long pairs for computing orientation based on the assumption that local gradients cancel each other thus not necessary in the global gradient determination.

Building the descriptor and descriptor distance

As with all binary descriptors, building the descriptor is done by performing intensity comparisons. BRISK takes the set of short pairs, rotate the pairs by the orientation computed earlier and makes comparisons of the form:

$$b = \begin{cases} 1, & I(\mathbf{p}_j^\alpha, \sigma_j) > I(\mathbf{p}_i^\alpha, \sigma_i) \\ 0, & \text{otherwise} \end{cases}$$

Meaning that for each short pair it takes the smoothed intensity of the sampling points and checked whether the smoothed intensity of the first point in the pair is larger than that of the second point. If it does, then it writes 1 in the corresponding bit of the descriptor and otherwise 0. Remember that BRISK uses only the short pairs for building the descriptor.

As usual, the distance between two descriptors is defined as the number of different bits of the two descriptors, and can be easily computed as the sum of the XOR operator between them.

You probably ask what about performance. Well we'll have a detailed post that will talk all about performance of the different binary descriptors, but for now I will say a few words comparing BRISK to the previous descriptors we talked about – BRIEF and ORB:

BRIEF outperforms BRISK (and ORB) in photometric changes – blur, illumination changes and JPEG compression.

BRISK slightly outperforms BRIEF in viewpoint changes, but performs about the same as ORB in overall.

Stay tuned for the next post in the series that will talk about the FREAK descriptor, the last binary descriptor we will focus on before giving a detailed performance evaluation.

OpenCV implementation

C++:

BRISK(int thresh=30, int octaves=3, float patternScale=1.0f)

Parameters:

thresh – FAST/AGAST detection threshold score.

octaves – detection octaves. Use 0 to do single scale.

patternScale – apply this scale to the pattern used for sampling the neighbourhood of a keypoint.