

Supplementary Material for “Descriptor-In-Pixel : Point-Feature Tracking for Pixel Processor Arrays”

Laurie Bose^{1,2} Jianing Chen^{1,2} Piotr Dudek^{1,2}

¹The University of Manchester, Manchester, United Kingdom

²Visionchip Limited, Manchester, United Kingdom

<https://lauriebose.github.io/DIP>

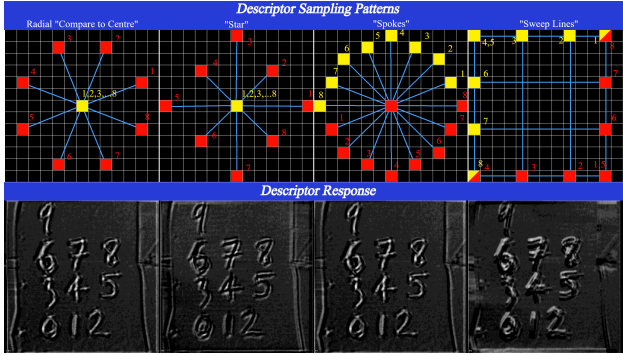


Figure 1. Some examples of different sampling patterns for 8-bit binary descriptors which could be used in our approach. The left radial pattern is the default presented in the paper. A descriptor response map is shown for each pattern.

1. Descriptor Sampling Patterns

In the main paper we made use of a simple radial sampling pattern for our in-pixel 8-bit descriptors, comparing a ring of pixels to a central pixel. However, there are many different sampling patterns could be used instead, with some examples shown in Figure 1. This figure also shows response maps for each example pattern. These are all computed using the same descriptor, however, because these are using different sampling patterns they highlight different visual structures.

2. Rotational Invariance

One of the most significant limitations of our current implementation upon SCAMP-7 is the lack of rotational invariance. This means features must maintain a visual orientation close to that at which they were initialized, otherwise they will lose tracking. As such, features cannot be tracked throughout certain motions, such as camera roll.

To solve this problem we need to incorporate some mechanism for tracking features through orientation



Figure 2. Examples of descriptors using a rotationally symmetric sampling pattern. The sampling pattern shown on the left is for 16-bit descriptors, and involves comparing 16 pairs of pixels, each comparison is between the central pixel and a pixel taken from a ring around this centre. Three descriptors are shown, each one is the right bit-shift of the previous. Note how this effectively rotates the visual structure the descriptor represents, as illustrated by the red (0s) and green (1s) on the pixel pairs of the sampling pattern.

changes. One approach is to have each feature’s orientation be tracked “in-pixel”, along with slightly longer descriptors, which can be used to enable feature tracking with rotational invariance. We will describe this idea next, however this is very challenging to implement upon SCAMP-7 due to the very limited memory available to each pixel-processor. We anticipate future PPAs will have significantly more memory per PE, enabling this idea.

Bit-shifting Rotationally Symmetric Patterns: Our method for implementation of rotational invariances requires the sampling patterns used for the “in-pixel” binary descriptors to be rotationally symmetric patterns. Specifically patterns where descriptor’s length (number of bits) is a multiple of the order of rotational symmetry. For example, Figure 2 shows a simple sampling pattern for 16-bit descriptors, which has 16 orders of rotational symmetry. The reason we want this property for our sampling pattern, is because bit-shifting any descriptor using such a pattern simply rotates the visual structure that descriptor represents by one order of pattern’s rotational symmetry. For example, as shown in Figure 2, bit-shifting a descriptor using this pattern essentially rotates its by $\frac{2\pi}{16}$ radians. Such descriptor bit shifting is simple to implement with “in-pixel” compute.

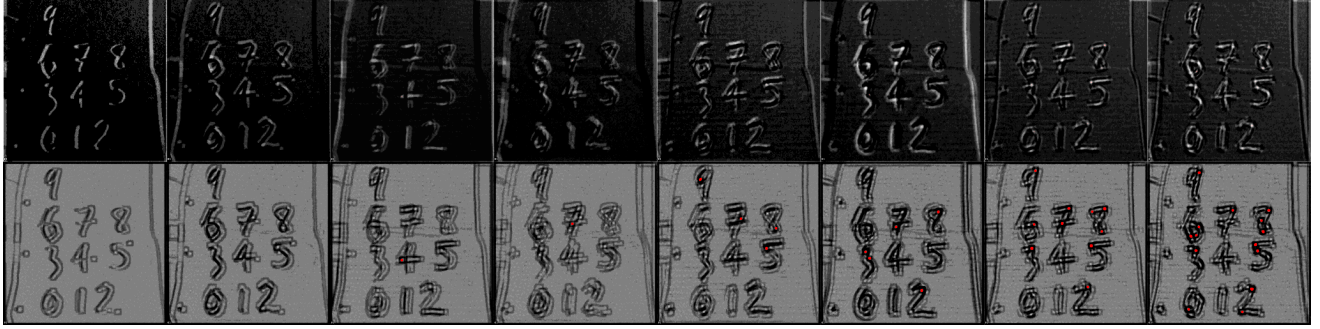


Figure 3. Examples of descriptor response maps (top) & corresponding blob detection result (bottom) showing potential point-features that can be tracked by that descriptor in red. From right to left, descriptors of size 1-bit, 2-bit, 3-bit... 8-bit. In general, the number of point feature candidates increases (though diminishingly) with descriptor size.

This idea of descriptor rotation via bit-shifting can be used to implement rotationally invariant feature tracking. This would involve computing two descriptor response patches for each feature per frame. The first patch would be computed using each features current descriptor, and the second patch computed using a bit-shifted copy of their descriptor (ie. a rotated version of their descriptor). If the patch from the rotated descriptor contained a stronger response, then the visual feature is better represented by the rotated descriptor, which then overwrites the current descriptor. By also alternating the direction in which descriptors are rotated every frame (ie. alternating between bit-shifting left & right), the descriptors of point-features will rotate to match any orientation changes in the visual features they are tracking, ie. rotationally invariant tracking. If sufficient memory is available, the orientation angle of each tracked feature can also be represented in PE memory, and updated accordingly each time a rotated copy of the feature's descriptor has a stronger response than its current descriptor. If required, this feature orientation could be readout in addition to the feature's location.

Unfortunately this idea is difficult to implement on SCAMP-7, due to the very limited memory per PE. Our current implementation uses descriptors only 8-bits in length, only allowing us to use patterns with a maximum of 8 degrees of rotational symmetry. Bit-shifting such descriptors would rotate $\frac{2\pi}{4}$ radians (ie. 45 degrees), which is a huge jump per bit-shift, and not fit for tracking a feature's orientation. We thus leave this idea for future PPA sensors.

3. Viable Descriptor Size

There is an interesting question in what size of descriptor is actually needed to reliably detect and track features (ignoring rotational & scale invariance for now). Figure 3 illustrates the descriptor response for some example descriptors of size 1,2,3-8-bits, along with blob detection and thresholding computed upon these responses to identify point-

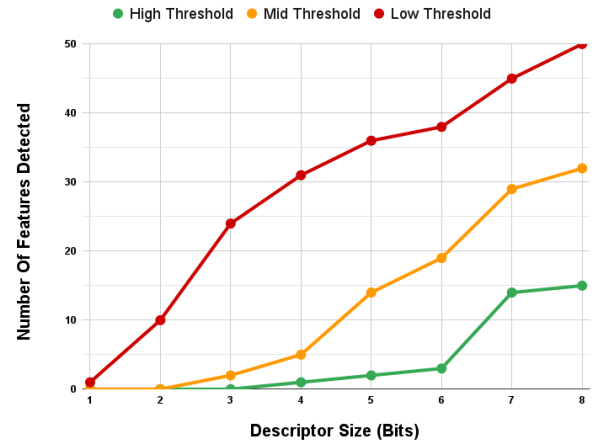


Figure 4. Number of features detected and tracked when using descriptors of various sizes and analogue thresholds for feature initialization.



Figure 5. Features detected when using descriptors of various sizes

features which they could be used to track. It is clear that the response from the 1-bit descriptor is useless for point feature detection/tracking, which is to be expected as it essentially performs edge detection. Similarly the descriptors of size 2,3,4 fail to identify a single feature here. The num-

ber of identifiable features sharply increases and levels off as descriptor size increases from 5-bits to 8-bits.

To evaluate this further we tested our system viewing the same scene of MNIST digits shown in Figure 5, multiple times. Each time using descriptors of different sizes (1-8 bits), each using a similar "compare to centre" style sampling pattern. We then recorded the number of features that were detected and tracked, giving us an idea of how the size of descriptors used effects the system's performance. Additionally we also investigated adjusting the analogue threshold used to initialize features for each descriptor size, where a higher threshold in general will only initialize features which can be reliably tracked. Three thresholds were tested, Low:24, Mid:48 (which is our approaches default), and High:72. The number of features detected and tracked for each threshold indicates what descriptor size is required to provide reliable features.

The results gathered from this are shown in Figure 4. It is clear that the number of features in general increases with descriptor size, and lowering the feature initialisation threshold also increases the number of features as would be expected. When using the Low threshold, many of the features are unreliable and lose tracking, (often because the underlying visual structure of these features is not very salient or discriminative). Higher thresholds (ie. Mid/High) are required to guarantee reliable features. Notably very few features are detected by descriptors of size 1-4 bits when using the Mid initialisation threshold, and similarly for descriptors of size 1-6 when using the High threshold. This indicates that in general larger descriptors allow for the detection and tracking of more reliable features. Unfortunately the maximum feature size we can currently implement is 8-bit, but features tracked by these descriptors (using the Mid/High thresholds) are already reliable and do not drift / lose tracking. Figure 5 shows some examples of features detected for each of the descriptor sizes investigated, using the Mid threshold. Again it is clear how the number of features increases with descriptor size.