## Data Buffer

### MP.1 Data Buffer Optimization

**Implement a vector for dataBuffer objects whose size does not exceed a limit (e.g. 2 elements). This can be achieved by pushing in new elements on one end and removing elements on the other end.**

The implementation uses the member function **erase** of the vector class to remove the front element.

## Keypoints

### MP.2 Keypoint Detection

**Implement detectors HARRIS, FAST, BRISK, ORB, AKAZE, and SIFT and make them selectable by setting a string accordingly.**

All the detectors are implemented using default input values.

### MP.3 Keypoint Removal

**Remove all keypoints outside of a pre-defined rectangle and only use the keypoints within the rectangle for further processing.**

The implementation uses the member function **contains** of the cv::Rect

## Descriptors

### MP.4 Keypoint Descriptors

**Implement descriptors BRIEF, ORB, FREAK, AKAZE and SIFT and make them selectable by setting a string accordingly.**

All the descriptors are implemented using default input values.

### MP.5 Descriptor Matching

**Implement FLANN matching as well as k-nearest neighbor selection. Both methods must be selectable using the respective strings in the main function.**

For descriptors other than SIFT, the features are binary features, so the indexParams is set as **cv::flann::LshIndexParams(20, 10, 2)**

### MP.6 Descriptor Distance Ratio

**Use the K-Nearest-Neighbor matching to implement the descriptor distance ratio test, which looks at the ratio of best vs. second-best match to decide whether to keep an associated pair of keypoints.**

The distance ratio is set to 0.8

## Performance

### MP.7 Performance Evaluation 1

**Count the number of keypoints on the preceding vehicle for all 10 images and take note of the distribution of their neighborhood size. Do this for all the detectors you have implemented.**

The result is written in the **detector.csv**. BRISK detects the largest number of keypoints, and HARRIS detects the smallest number of keypoints.

### MP.8 Performance Evaluation 2

**Count the number of matched keypoints for all 10 images using all possible combinations of detectors and descriptors. In the matching step, the BF approach is used with the descriptor distance ratio set to 0.8.**

The result is written in the **detector_descriptor_matches.csv**.

The number of matches is slightly proportional to the number of detected points.

### MP.9 Performance Evaluation 3

**Log the time it takes for keypoint detection and descriptor extraction. The results must be entered into a spreadsheet and based on this data, the TOP3 detector / descriptor combinations must be recommended as the best choice for our purpose of detecting keypoints on vehicles.**

The result is written in the **detector_descriptor_time.csv**.

My top 3 detector/descriptor combinations are **FAST**/**BRIEF**, **FAST**/**ORB**, **FAST**/**SIFT**

The choice is based on the number of matches and the time it takes to perform the detection and matching. The number of matches of these methods are ~130 or higher, and the time for the detection and matching takes are less than ~20 ms