MP.0 Mid-Term Report

I added some files and modified the return types of the provided functions so that the added can be gleaned for post processing. Further, I used the original file `MTPCS.cpp` for all the work.

The original file is `MidTermProject_Camera_Student.cpp`.

In addition, I created a folder named `reports` where the results for problems 7, 8, and 9 are output.

The results are generated every time the program mtpcs is run. The results are stored in the `results` folder.

The graphs are in the `.xlsx` files in the reports folder.

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.

Added the following lines of code so that we erase the previous contents and keep only fixed number (`dataBufferSize`) entries at any point.

```
                if (dataBuffer.size() == dataBufferSize) {
                    dataBuffer.erase(dataBuffer.begin());
                 }
                dataBuffer.push_back(frame);
```

MP.2 Keypoint Detection

IMPLEMENT DETECTORS HARRIS, FAST, BRISK, ORB, AKAZE, AND SIFT AND MAKE THEM SELECTABLE BY SETTING A STRING ACCORDINGLY.

Made the required modifications in the files provided.

MP.3 Keypoint Removal

REMOVE ALL KEYPOINTS OUTSIDE OF A PRE-DEFINED RECTANGLE AND ONLY USE THE KEYPOINTS WITHIN THE RECTANGLE FOR FURTHER PROCESSING.

```
        if (bFocusOnVehicle)
        {
            std::vector<cv::KeyPoint> filtered;
            for (const auto & keyPoint : keypoints) {
                if (vehicleRect.contains(keyPoint.pt)) filtered.push_back(keyPoint);
            }
            keypoints = filtered;
            metrics.carKeyPoints.push_back(keypoints.size());
            if (LOG) std::cout << "keypoint count after vehicle filtering: " <<
keypoints.size() << std::endl;
        }
```

MP.4 Keypoint Descriptors

IMPLEMENT DESCRIPTORS BRIEF, ORB, FREAK, AKAZE AND SIFT AND MAKE THEM SELECTABLE BY SETTING A STRING ACCORDINGLY.

Made the required modifications in the files provided.

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.

Made the required modifications in the files provided.

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.

Made the required modifications in the files provided.

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.

I added up the number of keypoints per each image and noted the number of keypoints in the preceding vehicle and those outside. Then I averaged them for each run of detector. Then I noted the ratios of the means for each detector. The results of this are shown in `mp7.xslx` in the reports folder.

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 results are tabulated in `mp8.xslx` under the reports folder.

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.

As shown in `mp9.xslx` in the reports folder, the top three (detector, descriptor) pairs are `(FAST, BRISK)`, `(FAST, BRIEF)`, and `(FAST, ORB)`.

The detection is where most of the time is spent. The descriptor business is relatively fast. Since the detection time is least for FAST, it is no surprise that FAST shows up in the top three detector/descriptors.