2 Simple image processing

Complete a simple unfinished UI application which allows the application of image filters and displays the results.

2.1 Open the project 02_BitmapPlayground

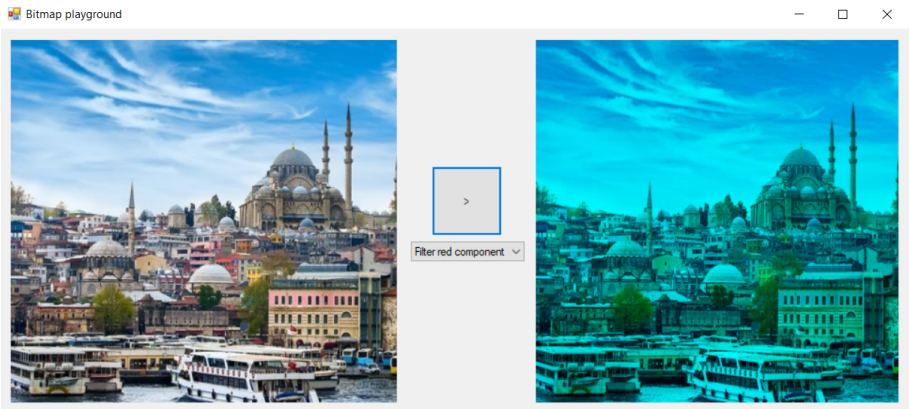

Figure 1: The application with the already implemented red filter.

Familiarize yourself with the project. There is already one filter function available that removes the red component from the image. Try it out. Identify the parts marked with “TODO” in the source code.

2.2 Implement a greyscale filter

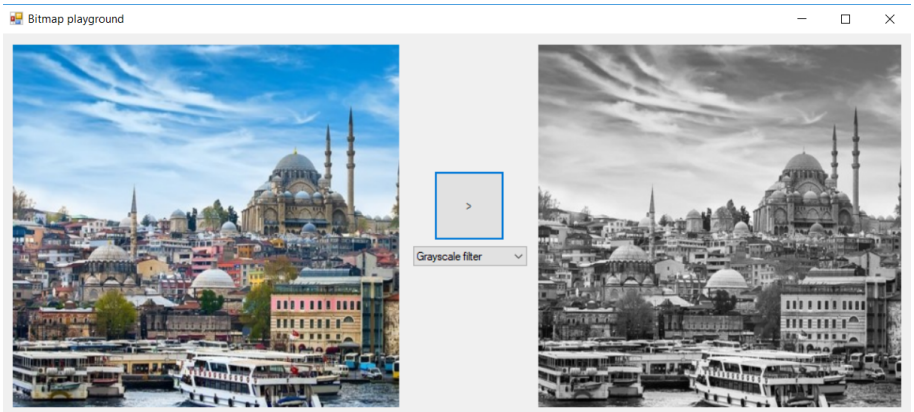

Figure 2: The desired grayscale filter effect.

Implement a greyscale filter. There are several possibles formulas for computing greyscale images, you are free to choose any.

2.3 Implement a moving average filter

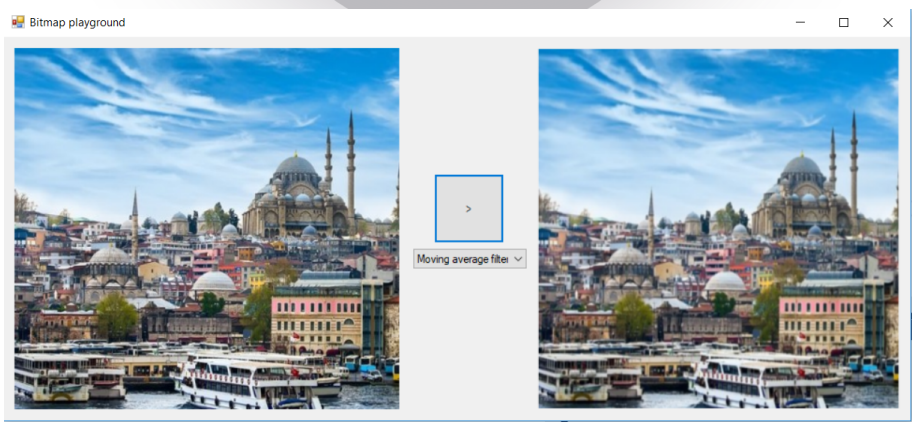

Figure 3: The desired moving average filter effect.

Implement a moving average filter, as follows: The value of each output pixel at the position $output(x, y)$ is the average value of the input pixel $input(x, y)$ and the 4 pixels left, right, above and below it. If the input pixel lies at the border, its value may remain the same.

2.4 Parallelize the filter

Optimize the filters by using thread pooling.

2.5 Extract the filters to an external project

Extract `IFilter` and all of its implementations to an a separate dll project and reference it from the main project.

2.6 Automatically find all filters in the assembly using reflection

Implement `PopulateListBox()` such that all implementations of `IFilter` in the assembly containing `IFilter` are found and added to the dropdown. The text in the Listbox should correspond to their "Name" property.