Lab4_final project

Report :

Introduction:

The main focus of my project was C++-based feature identification and image patch overlay on top of an original image.

---

1) Importing images:

My system's preset directory was used to load an original image, which served as the project's starting point. To avoid any mistakes in the following steps, I made sure the image had loaded successfully. I also imported three smaller "patch" photos, making sure that each one loaded properly.

2) Feature Detection

I applied the Scale-Invariant Feature Transform (SIFT) to identify and describe local features in the photos. I computed the SIFT features for the main image and each of the three patch images after constructing a SIFT object. This procedure laid the framework for key-point matching between the images.

3)Correspondence with Key Points:

After identifying the traits, I set out to compare critical points between the patch photos and the original image. I employed the help of the Brute Force Matcher (BFMatcher) to do this. After the matching process, I determined the match with the shortest distance, which prepared the way for the matches to be improved. You can look my terminal picture , for patch 0 I found 7784 matches and 207 refined matches . For patch1 7784 matches and 210 refined matches and finally for patch 2 7784 matches and 433 refiend matches

Additionally, I used a ratio test to exclude poor matches, choosing only those whose distance fell below a predetermined ratio of the minimum distance discovered earlier.

4) Homography Patch Overlay:

I calculated an affine transformation (homography) between the patch and the source image for each patch image with a sufficient number of matches—at least four. The RANSAC (RANdom SAmple Consensus) algorithm was used for this because it is reliable at estimating the transformation in the face of potential outliers.Using the warpPerspective function and the homography, I adjusted the patch's perspective to match the main focal points of the original image. The original image was overlaid at the appropriate location by the patch as a result of this phase.Finally, the successful completion of the processing pipeline was demonstrated visually by displaying the finished, patched image using OpenCV's imshow function.

Conclusion:

I've learned so much about OpenCV's broad capabilities in handling challenging image processing jobs thanks to this project. Key-point matching and homography estimates were used to overlay patches over an original image to show the effectiveness of these methods. For tasks like object recognition, tracking, or superimposing virtual objects onto real-world settings, I see potential applications in a variety of fields, including picture editing, computer vision, and Augmented Reality (AR).

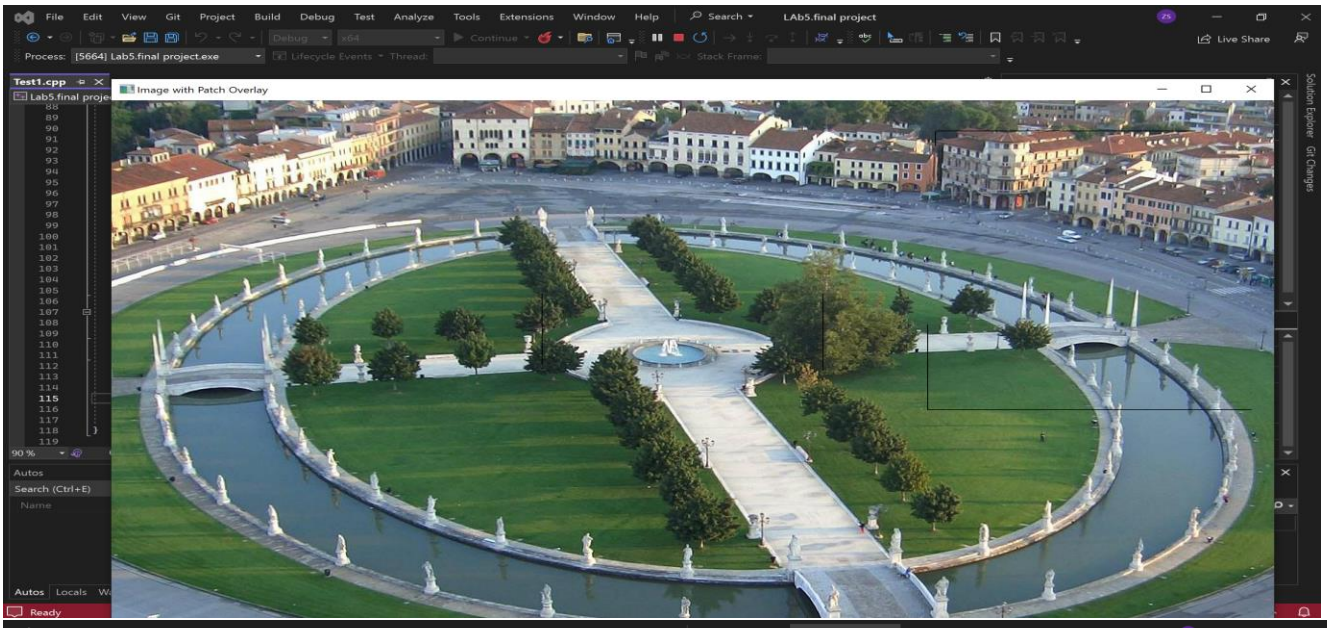
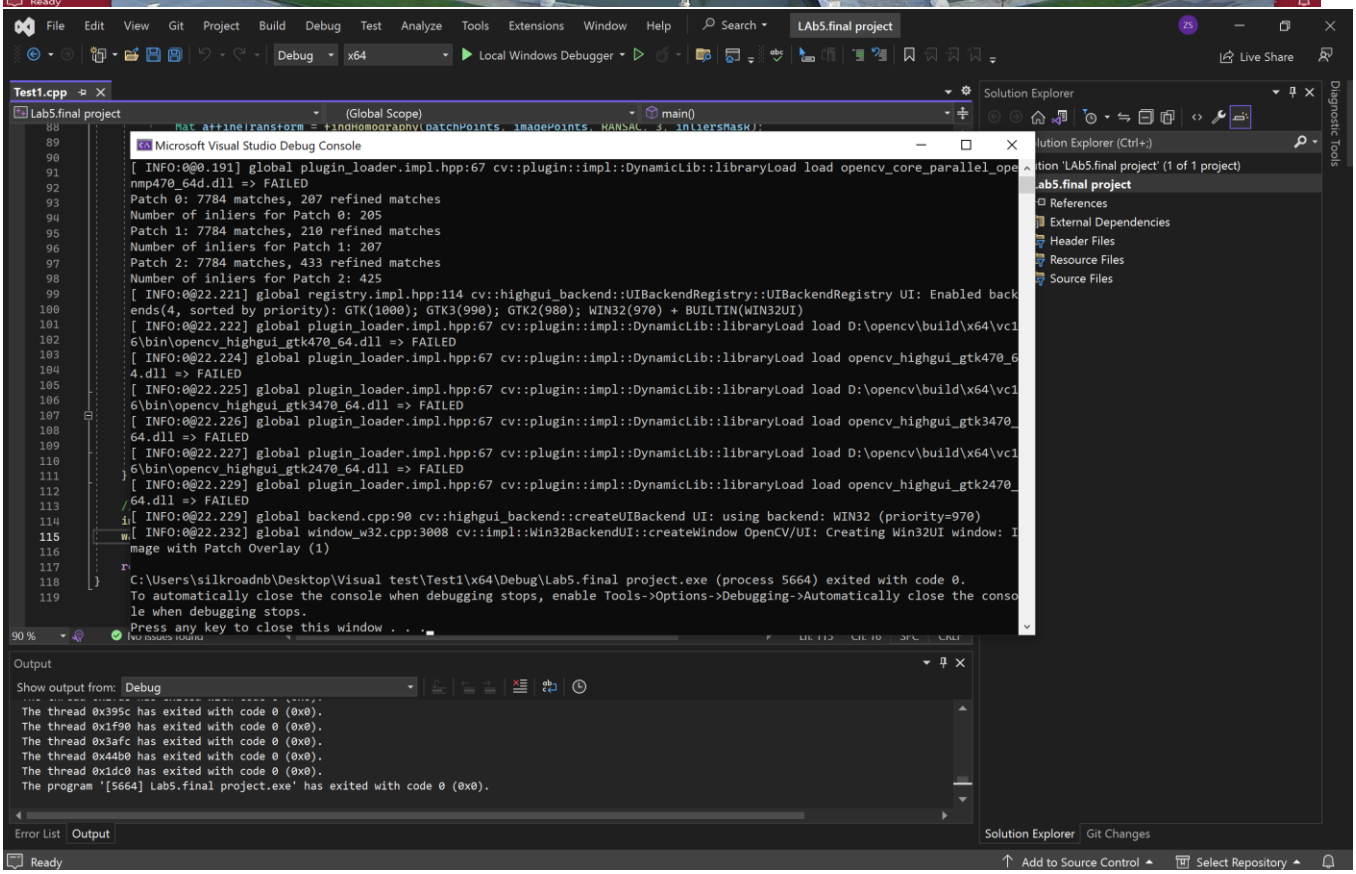


Image with Patch Overlay


Microsoft Visual Studio Debug Console
[INFO:0@0.191] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load opencv_core_parallel_ope
nmp470_64d.dll => FAILED
Patch 0: 7784 matches, 207 refined matches
Number of inliers for Patch 0: 205
Patch 1: 7784 matches, 210 refined matches
Number of inliers for Patch 1: 207
Patch 2: 7784 matches, 433 refined matches
Number of inliers for Patch 2: 425
[INFO:0@22.221] global registry.impl.hpp:114 cv::highgui_backend::UIBackendRegistry::UIBackendRegistry UI: Enabled back
ends(4, sorted by priority): GTK(1000); GTK3(990); GTK2(980); WIN32(970) + BUILTIN(WIN32UI)
[INFO:0@22.222] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load D:\opencv\build\x64\vc1
6\bin\opencv_highgui_gtk470_64.dll => FAILED
[INFO:0@22.224] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load opencv_highgui_gtk470_6
4.dll => FAILED
[INFO:0@22.225] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load D:\opencv\build\x64\vc1
6\bin\opencv_highgui_gtk3470_64.dll => FAILED
[INFO:0@22.226] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load opencv_highgui_gtk3470_
64.dll => FAILED
[INFO:0@22.227] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load D:\opencv\build\x64\vc1
6\bin\opencv_highgui_gtk2470_64.dll => FAILED
[INFO:0@22.229] global plugin_loader.impl.hpp:67 cv::plugin::impl::DynamicLib::libraryLoad load opencv_highgui_gtk2470_
64.dll => FAILED
[INFO:0@22.229] global backend.cpp:90 cv::highgui_backend::createUIBackend UI: using backend: WIN32 (priority=970)
[INFO:0@22.232] global window_w32.cpp:3008 cv::impl::Win32BackendUI::createWindow OpenCV/UI: Creating Win32UI window: I
mage with Patch Overlay (1)
C:\Users\silkroadnb\Desktop\Visual test\Test1\x64\Debug\Lab5.final project.exe (process 5664) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
The thread 0x395c has exited with code 0 (0x0).
The thread 0x1f90 has exited with code 0 (0x0).
The thread 0x3afc has exited with code 0 (0x0).
The thread 0x44b0 has exited with code 0 (0x0).
The thread 0x1dc0 has exited with code 0 (0x0).
The program '[5664] Lab5.final project.exe' has exited with code 0 (0x0).