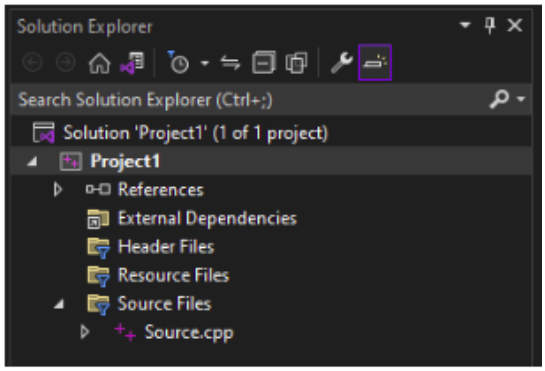
FClBrI

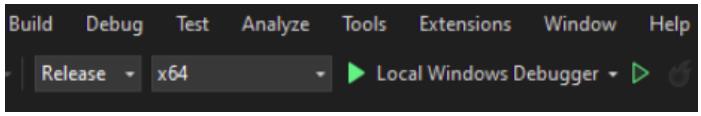
Halide-based CNN implementation and benchmarking.

Installing and Setting Up Halide in Windows

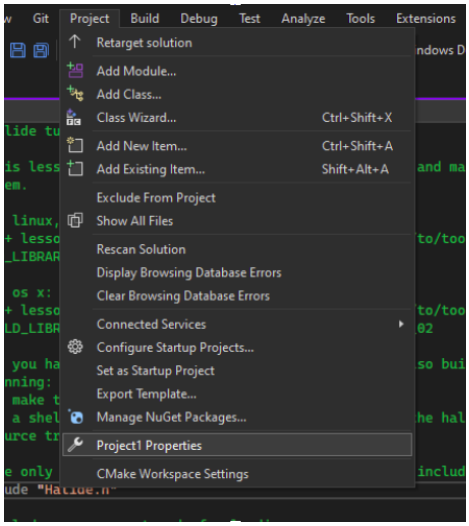
- Download Halide from here, Unzip the content and save it at the desired location.
- For our development purpose, we use Visual Studio. Download Visual Studio from here, and during installation make sure to install the C++ Development for Windows package.
- Make a new C++ empty project. `Make a source.cpp` file in Source Files as shown in the figure below. Write some Halide code in this file, you can get code snippets from - here



- Set the configuration as Release and x64.



- Go to Project → Project_Name Properties Options. Here we would add ‘include’ files, set the path of libraries and set the C++ language.



- Go to VC++ Directories → Include Directories. We will add the path of all the included libraries here. Set path as `Halide_location\include`
- Go to VC++ Directories → Library Directories. Set path as `Halide_location\lib\Release`
- Next Go to C/C++ → Language → C++ Language Standard. Select ISO C++17 Standard (/std:c++17)
- Lastly Go to Linker → Additional Dependencies. Add `Halide.lib` here.
- Copy `Halide.dll` file from Halide_location\bin\Release and paste this in the Project\x64\Release folder (where your Project.exe file is present). If this folder is not present, run your program using the Local Windows Debugger button as shown in Fig. 2. Add the following header file in the Source code -

```
#include “Halide.h”
```

This should be enough to run Halide code, You can verify by running the Example Code, it should give a Success message.

But with Halide, we use images. For this purpose we might include the opencv library for various operations on images such as opening image, saving image, etc. For installing opencv, follow these steps -

- Download the OpenCV pre-built library for Windows. Go to the OpenCV official download page here and download the latest pre-built library for Windows. Make sure to download the appropriate version (32-bit or 64-bit) depending on your system.
- Set up environment variables. You may want to add the OpenCV binaries to your system's PATH. This can be done by adding `opencv_location\build\x64\vc15\bin` (or the appropriate path for your system) to your system's PATH environment variable. Note that the 'x64' and 'vc15' parts may differ based on your OpenCV version and the Visual Studio version.
- Go to VC++ Directories → Include Directories. Add path `opencv_location\build\include`
- Go to VC++ Directories → Library Directories. Set path as `opencv_location\build\x64\vc16\lib`
- Lastly Go to Linker → Additional Dependencies. Add `opencv_worldXXXd.lib` for the debug configuration and `opencv_worldXXX.lib` for the release configuration, where XXX is the OpenCV version number. You can find this file at `opencv_location\build\x64\vc16\lib`.
- Copy `opencv_worldXXX.dll` file from `opencv_location\build\x64\vc16\bin` and paste this in the `Project\x64\Release` folder (beside the Project.exe file).
- Add the OpenCV header in your code. In your C++ source file, you can now include the OpenCV header with the following line:

```
#include <opencv2/opencv.hpp>
```