OpenCV

Concrete Architecture

eLand

Eric Rodrigues, Laura Marin, Alp Baran Sirek, Negar Khalilazar, Danny Le

Topics that are going to be covered

1. Scope & goals
2. Architecture Derivation Process
3. Concrete architecture
4. Overview of G-API
5. Patterns and Trade-offs
6. Behavioral view
7. Limits of Our Findings
8. Conclusions
9. Questions

Architecture Derivation Process

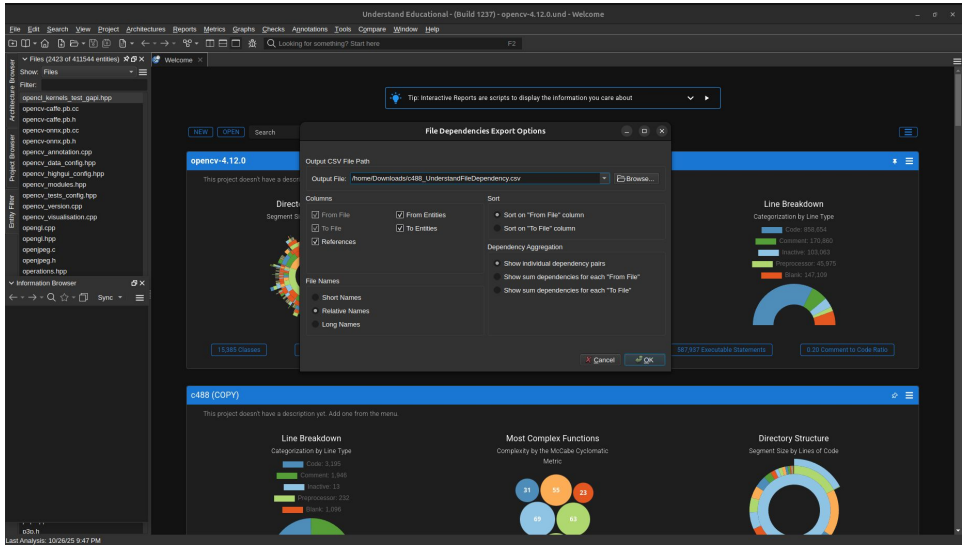
Architecture Derivation Process (1)

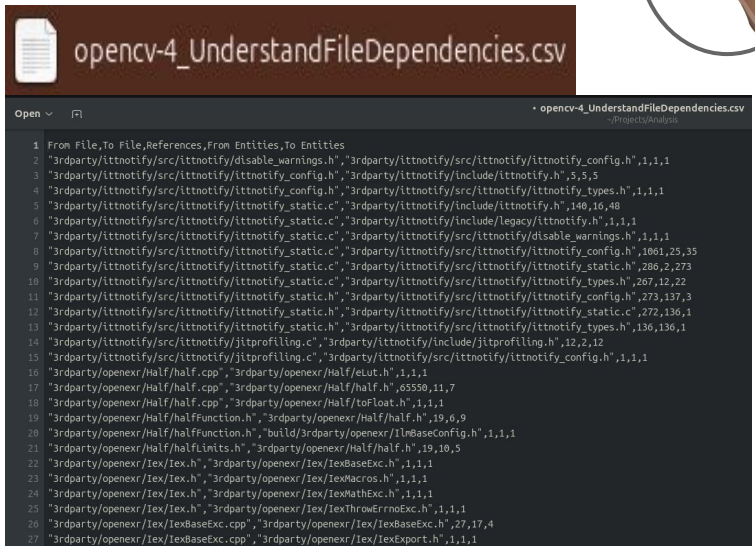
```
~/opencv-4.12.0/build
:~/opencv-4.12.0/build$ cmake ../build
-- ocv_init_download: OpenCV source tree is not fetched as git repository. 3rdparty resources will be downloaded from github.com by
 default.
-- Detected processor: x86_64
-- Looking for ccache - not found
-- Could NOT find AVIF (missing: AVIF_LIBRARY AVIF_INCLUDE_DIR)
-- Could NOT find OpenJPEG (minimal suitable version: 2.0, recommended version >= 2.3.1). OpenJPEG will be built from sources
-- OpenJPEG: VERSION = 2.5.3, BUILD = opencv-4.12.0-openjp2-2.5.3
-- OpenJPEG libraries will be built from sources: libopenjp2 (version "2.5.3")
-- libva: missing va.h header (VA_INCLUDE_DIR)
-- found Intel IPP (ICV version): 2022.1.0 [2022.1.0]
-- at: /home/metalgeardull/opencv-4.12.0/build/3rdparty/ippicv/ippicv_lnx/icv
-- found Intel IPP Integration Wrappers sources: 2022.1.0
-- at: /home/metalgeardull/opencv-4.12.0/build/3rdparty/ippicv/ippicv_lnx/iw
-- Could NOT find Atlas (missing: Atlas_CBLAS_INCLUDE_DIR Atlas_CLAPACK_INCLUDE_DIR Atlas_CBLAS_LIBRARY Atlas_BLAS_LIBRARY Atlas_LA
PACK_LIBRARY)
-- Could NOT find BLAS (missing: BLAS_LIBRARIES)
-- Could NOT find LAPACK (missing: LAPACK_LIBRARIES)
    Reason given by package: LAPACK could not be found because dependency BLAS could not be found.

-- Could NOT find JNI (missing: JAVA_INCLUDE_PATH JAVA_INCLUDE_PATH2 AWT)
-- VTK is not found. Please set -DVTK_DIR in CMake to VTK build directory, or to VTK install subdirectory with VTKConfig.cmake file
-- Checking for module 'gtk+-2.0'
--   Package 'gtk+-2.0', required by 'virtual:world', not found
-- Checking for modules 'libavcodec;libavformat;libavutil;libswscale'
--   Package 'libavcodec', required by 'virtual:world', not found
--   Package 'libavformat', required by 'virtual:world', not found
--   Package 'libavutil', required by 'virtual:world', not found
--   Package 'libswscale', required by 'virtual:world', not found
-- FFMPEG is disabled. Required libraries: libavcodec;libavformat;libavutil;libswscale. Missing libraries: libavcodec;libavformat;l
ibavutil;libswscale
-- Checking for module 'gstreamer-base-1.0'
```

→ compile_commands.json

- Configure OpenCV’s build system to create “compile_commands.json”.

Architecture Derivation Process (2)





- SciTools' "Understand" application analyses the file dependencies, uses the .json file for accuracy, and exports the file dependencies a .csv file

Architecture Derivation Process (3)

```
# EECS 4314 Lab Script
#       This script is used to transform the understand csv data into TA format
#
#/usr/bin/perl
#!****

use strict;
use warnings;

my ($header, $from, $to, $reference, $v1, $v2, $output_file);
my (%FILE_HASH, %CALL_HASH, $name, $value, $count, $hash, $line);

$header = 0;

open INPUT, "$ARGV[0]";

if ($ARGV[0] =~ /(.*)\.csv/) {
        $output_file = $1 . ".raw.ta";
}

open OUTPUT, ">$output_file";

print OUTPUT "FACT TUPLE :\n";


while (<INPUT>) {
        $line =~;
        chomp $line;

        if ($header == 0) {
                $header++;
                next;
        }

        ($from, $to, $reference, $v1, $v2) = split(/,/, $line);

        # comment out if we don't want to consider header files
#       if ($from =~ /\.h/ or $to =~ /\.h/) {
#               next;
#       }

        $FILE_HASH{$from}++;
        $FILE_HASH{$to}++;
        $CALL_HASH{$from}{$to}++;
}

# output the list of files
while (($name, $value) = (each %FILE_HASH)) {
        # $INSTANCE codeen.o[.text+0x0] cFunction
        print OUTPUT "$INSTANCE $name cFile\n";
}

# output the list of call relations
while (($name, $hash) = (each %CALL_HASH)) {
        while (($value, $count) = (each %$hash)) {
                # clinks codeen.o[.text+0x173] semantics.o[.bss+0x0]
                # print OUTPUT "contain $name $value\n";
                print OUTPUT "clinks $name $value\n";
        }
}

close INPUT;
close OUTPUT;
```

opencv-4_UnderstandFileDependencies.raw.ta

Open ~/Projects/Analysis opencv-4_UnderstandFileDependencies.raw.ta

```
FACT TUPLE :
$INSTANCE "3rdparty/include/opencl/1.2/CL/cl.h" cFile
$INSTANCE "3rdparty/ittnotify/include/ittnotify.h" cFile
$INSTANCE "3rdparty/ittnotify/include/jitprofiling.h" cFile
$INSTANCE "3rdparty/ittnotify/include/legacy/ittnotify.h" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/disable_warnings.h" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/ittnotify_config.h" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/ittnotify_static.c" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/ittnotify_static.h" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/ittnotify_types.h" cFile
$INSTANCE "3rdparty/ittnotify/src/ittnotify/jitprofiling.c" cFile
$INSTANCE "3rdparty/openexr/Half/eLut.h" cFile
$INSTANCE "3rdparty/openexr/Half/half.cpp" cFile
$INSTANCE "3rdparty/openexr/Half/half.h" cFile
$INSTANCE "3rdparty/openexr/Half/halfFunction.h" cFile
$INSTANCE "3rdparty/openexr/Half/halfLimits.h" cFile
```

- Then a script reads the csv file to create a .ta file

Architecture Derivation Process (4)

```java
import java.io.*;
import java.util.*;

public class GenerateContainRelations {
    Run | Debug
    public static void main(String[] args) {
        String inputFile = "opencv-4_UnderstandFileDependencies.raw.ta";
        String outputFile = "opencv-4_UnderstandFileDependencies.contain";

        Set<String> seenRelations = new HashSet<>();

        try (BufferedReader br = new BufferedReader(new FileReader(inputFile));
             PrintWriter pw = new PrintWriter(new FileWriter(outputFile))) {

            String line;
            while ((line = br.readLine()) != null) {
                line = line.trim();

                if (!line.startsWith("$INSTANCE")) continue;

                int start = line.indexOf('"');
                int end = line.lastIndexOf('"');
                if (start == -1 || end == -1 || start >= end) continue;

                String path = line.substring(start + 1, end);
                String[] parts = path.split("/");

                for (int i = 0; i < parts.length - 1; i++) {
                    String parentPath = String.join("/", Arrays.copyOfRange(parts, 0, i + 1));
                    String childPath = String.join("/", Arrays.copyOfRange(parts, 0, i + 2));
                    String relation = "contain \"" + parentPath + "\" \"" + childPath + "\"";

                    if (seenRelations.add(relation)) {
                        pw.println(relation);
                    }
                }
            }

            System.out.println("Containment relationships written to " + outputFile);

        } catch (IOException e) {
            System.err.println("Error processing file: " + e.getMessage());
        }
    }
}
```

opencv-4_UnderstandFileDependencies.contain

Open — opencv-4_UnderstandFileDependencies.contain — ~/Projects/ExtractContains

```
contain "3rdparty" "3rdparty/include"
contain "3rdparty/include" "3rdparty/include/opencl"
contain "3rdparty/include/opencl" "3rdparty/include/opencl/1.2"
contain "3rdparty/include/opencl/1.2" "3rdparty/include/opencl/1.2/CL"
contain "3rdparty/include/opencl/1.2/CL" "3rdparty/include/opencl/1.2/CL/cl.h"
contain "3rdparty" "3rdparty/ittnotify"
contain "3rdparty/ittnotify" "3rdparty/ittnotify/include"
contain "3rdparty/ittnotify/include" "3rdparty/ittnotify/include/ittnotify.h"
contain "3rdparty/ittnotify/include" "3rdparty/ittnotify/include/jitprofiling.h"
contain "3rdparty/ittnotify/include" "3rdparty/ittnotify/include/legacy"
contain "3rdparty/ittnotify/include/legacy" "3rdparty/ittnotify/include/legacy/ittnotify.h"
contain "3rdparty/ittnotify" "3rdparty/ittnotify/src"
contain "3rdparty/ittnotify/src" "3rdparty/ittnotify/src/ittnotify"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/disable_warnings.h"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/ittnotify_config.h"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/ittnotify_static.c"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/ittnotify_static.h"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/ittnotify_types.h"
contain "3rdparty/ittnotify/src/ittnotify" "3rdparty/ittnotify/src/ittnotify/jitprofiling.c"
contain "3rdparty" "3rdparty/openexr"
contain "3rdparty/openexr" "3rdparty/openexr/Half"
contain "3rdparty/openexr/Half" "3rdparty/openexr/Half/eLut.h"
contain "3rdparty/openexr/Half" "3rdparty/openexr/Half/half.cpp"
contain "3rdparty/openexr/Half" "3rdparty/openexr/Half/half.h"
contain "3rdparty/openexr/Half" "3rdparty/openexr/Half/halfFunction.h"
contain "3rdparty/openexr/Half" "3rdparty/openexr/Half/halfLimits.h"
```

- Our Java program reads the .ta file to create our containment file.

Architecture Derivation Process (5)

createContainment.sh
~/Projects/Analysis

```
#!/bin/bash
java -Xms256M -Xmx1024M -classpath ql.jar ca.uwaterloo.cs.ql.Main
addcontain.ql opencv-4_UnderstandFileDependencies.contain
opencv-4_UnderstandFileDependencies.raw.ta
opencv-4_UnderstandFileDependencies.con.ta
cat schema.asv.ta opencv-4_UnderstandFileDependencies.con.ta >
opencv-4_UnderstandFileDependencies.ls.ta
```

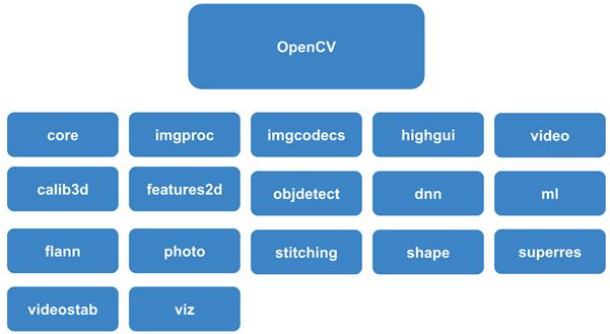
- A script reads our containment file to produce a landscape file of OpenCV’s architecture. This is then opened in LSEdit.

Concrete Architecture Overview

OpenCV Architecture

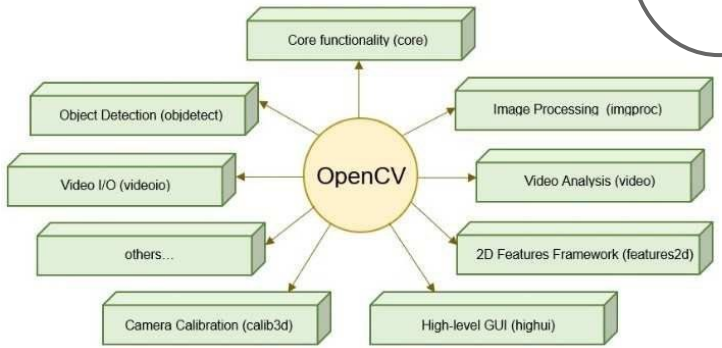
- Each module is implemented in C++ under the modules directory and compiles into its own library
- Lower-level modules handle fundamental functionality
- Higher-level modules build on top for specialized tasks like deep learning, video processing, and graph-based computation

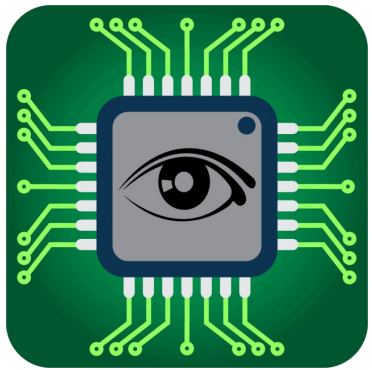
OpenCV



OpenCV Modules

- Core module defines key data structures like Mat and UMat for image and matrix representation
- Sitting above that is the Imgproc module provides core image processing functions
- Other modules handle more specific tasks





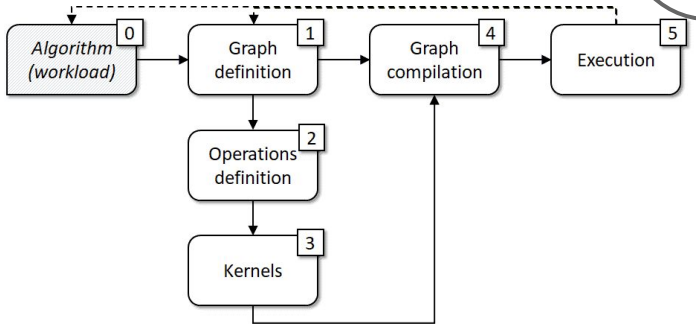
G-API Subsystem: Components, Interaction, & Analysis

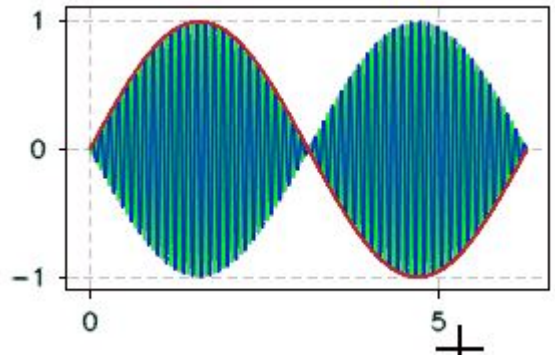
Overview of G-API

- Graph API subsystem provides OpenCV's graph-based computation engine
- Unlike traditional modules which execute sequentially, G-API represents computations as dataflow graphs
- The subsystem is made up of around 290 source and header files

Internal Structure

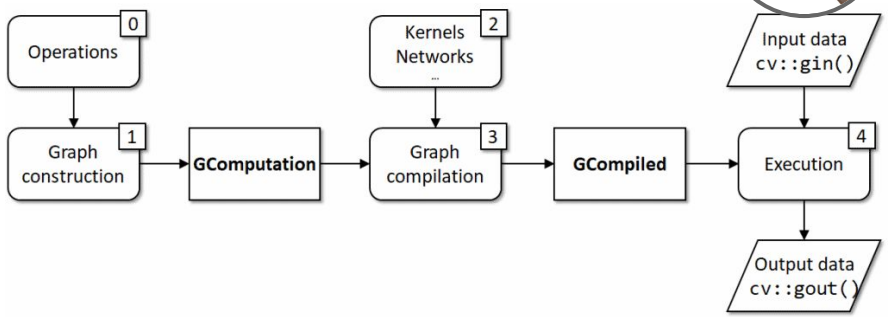
- The compiler directory includes the GCompiler that transform these abstract graphs into executable pipelines
- The executor folder manages runtime execution through classes such as GExecutor and GCompiled
- Backends holds backend-specific code that register its kernels through macros

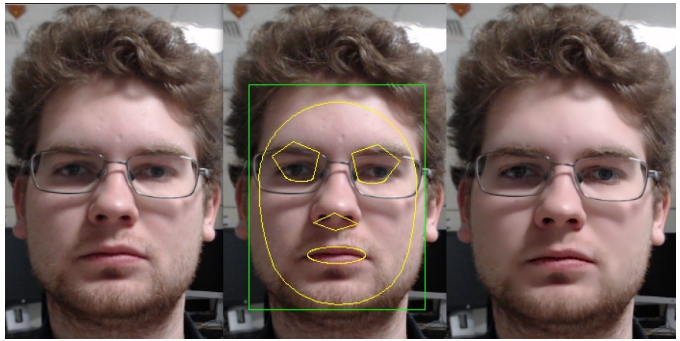




Deeper Dive

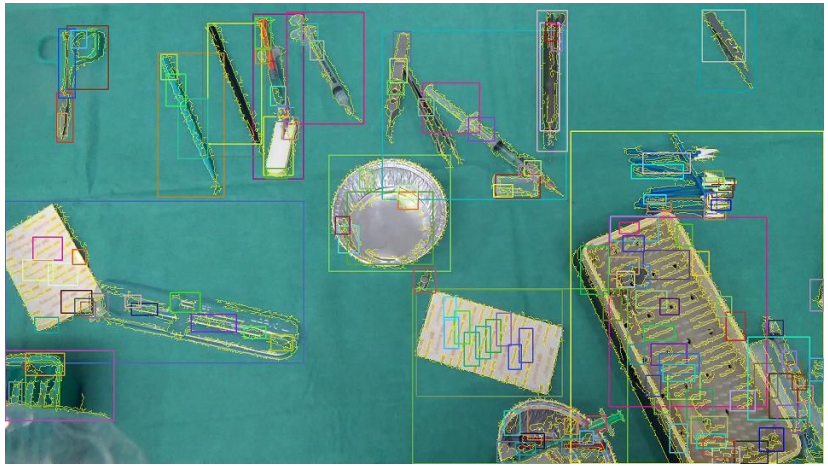
- The process starts with graph construction where the user defines a computation graph
- In the compilation phase GComputation triggers which analyzes dependencies and maps operations to available backends
- During execution GExecutor runs the compiled graph passing real data between nodes





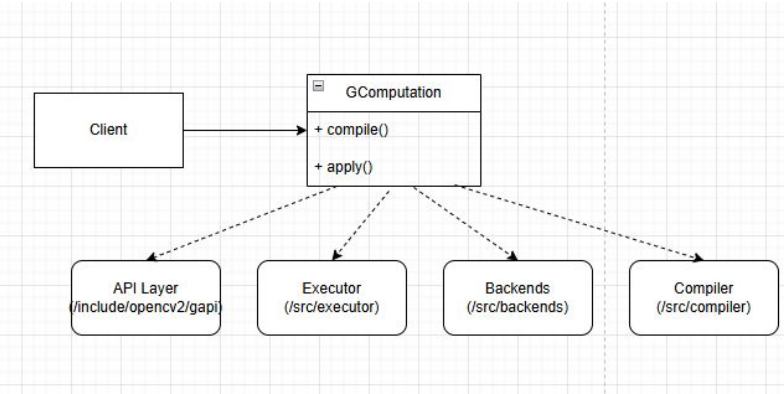
Component Interactions

- G-API’s internal components work together to balance flexibility and performance
- Performance is enhanced through optimizations like fusion, lazy evaluation, and data reuse
- Remains tightly integrated with the Core module but loosely coupled with others maintaining clean architectural boundaries

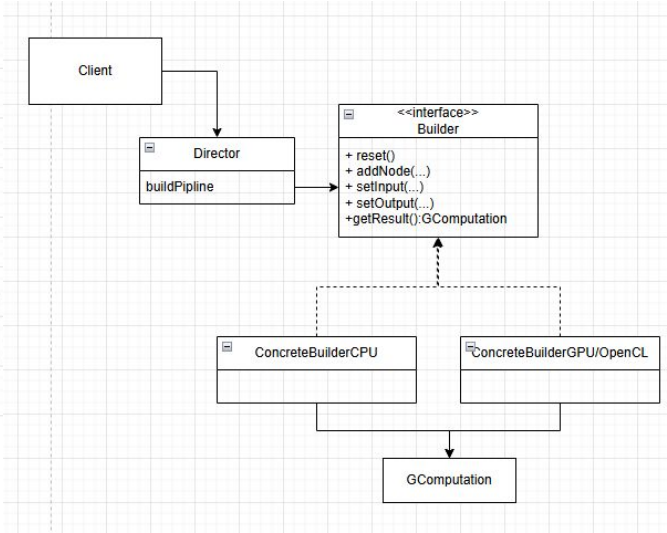


Design Patterns

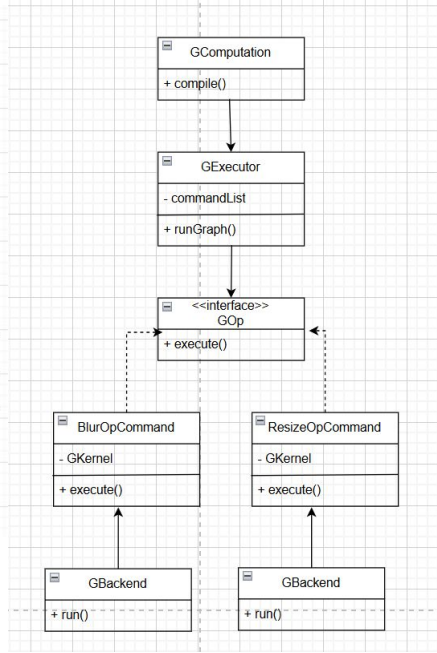
Facade Pattern



Builder Pattern

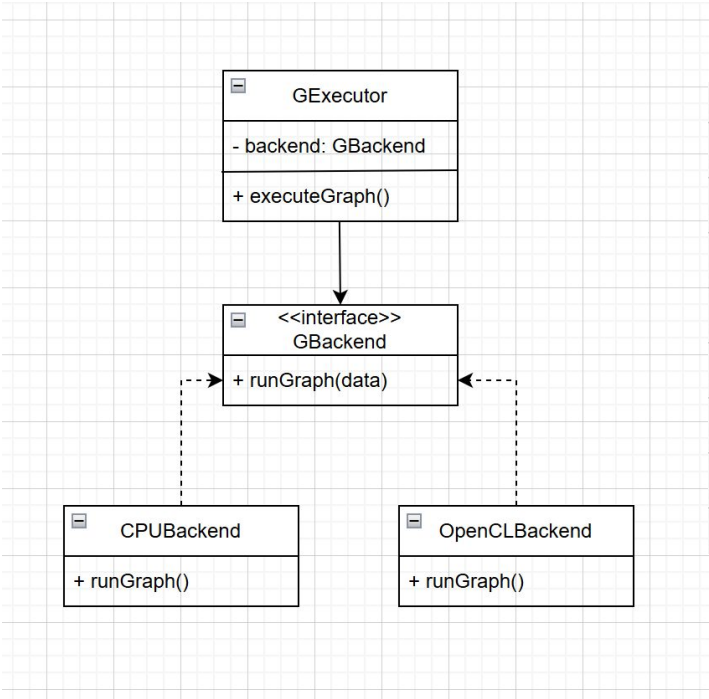


Command Pattern

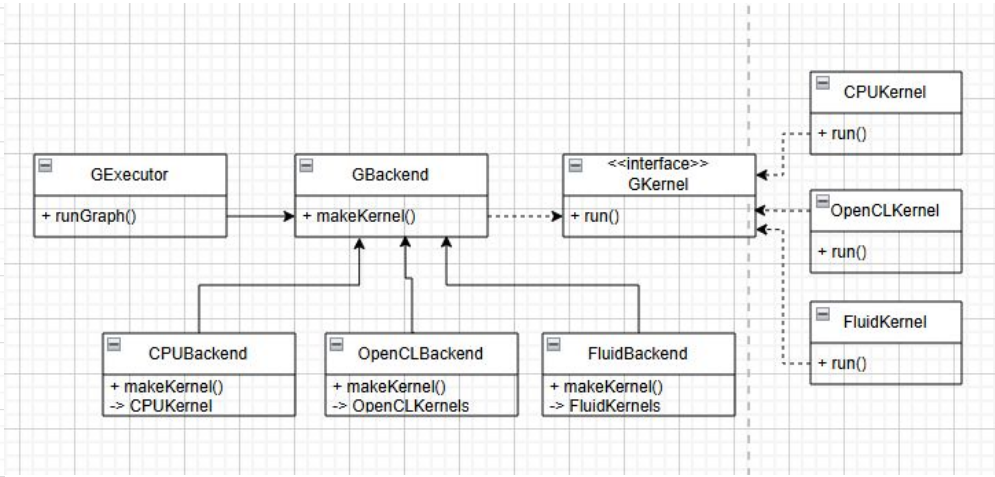


Design Patterns

Strategy Pattern

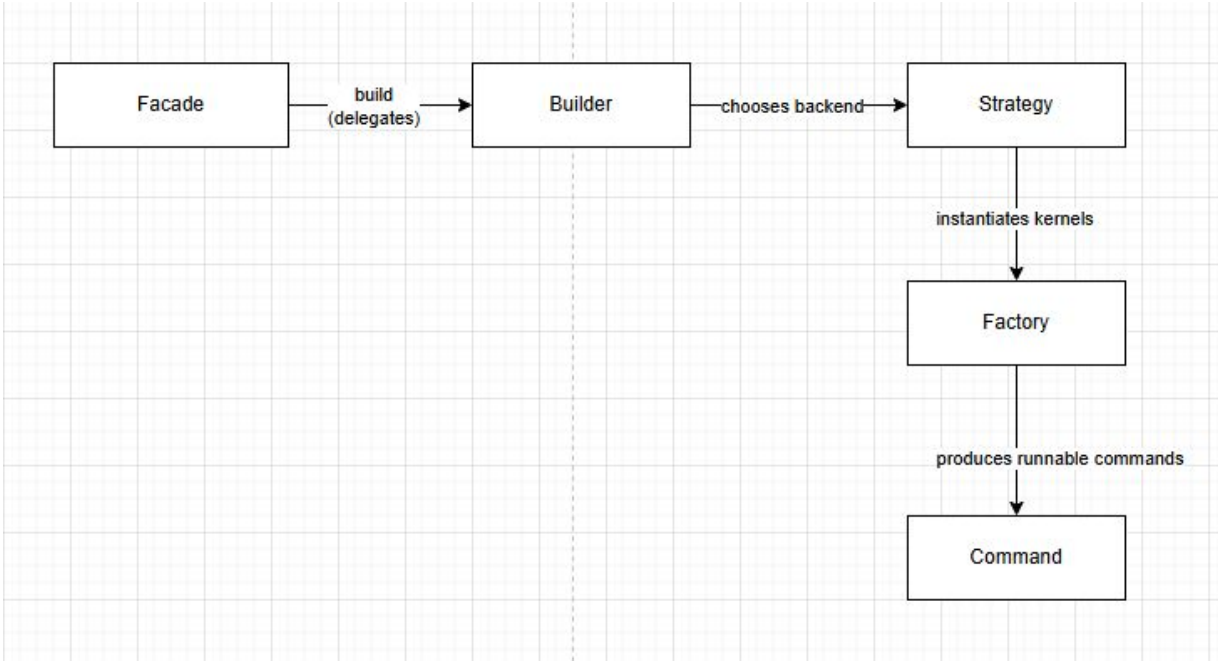


Factory Pattern



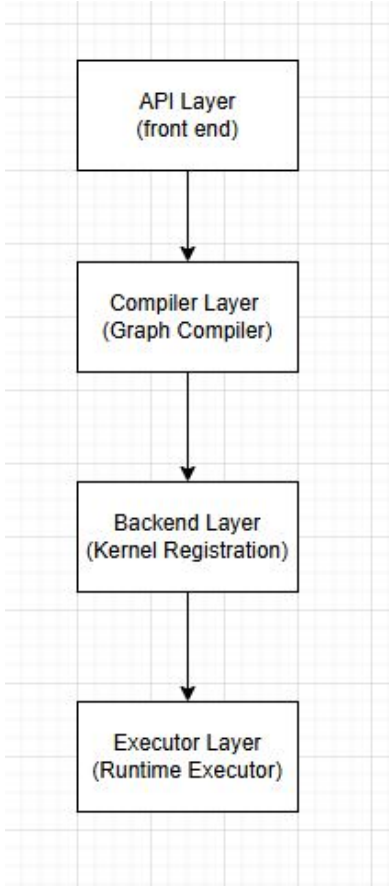
Design Patterns

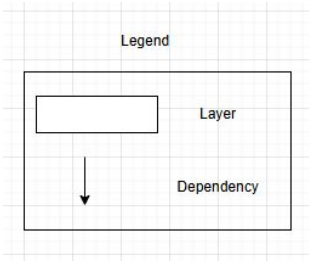
- Facade
- Builder
- Strategy
- Command
- Factory



Reflexion Analysis

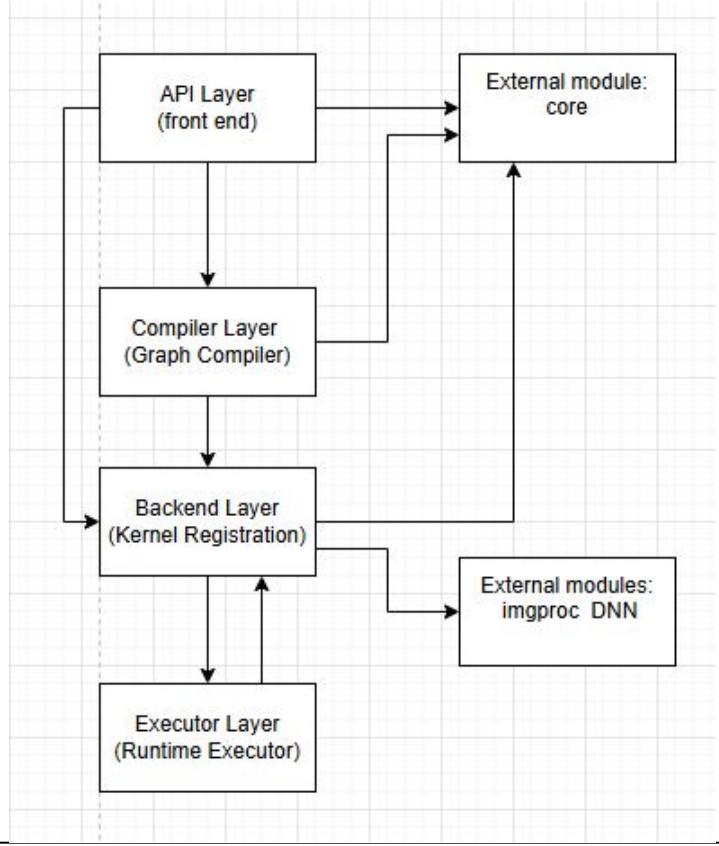
- Conceptual Architecture
- Expected

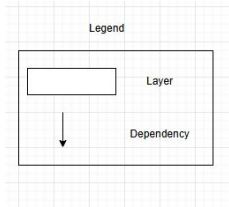




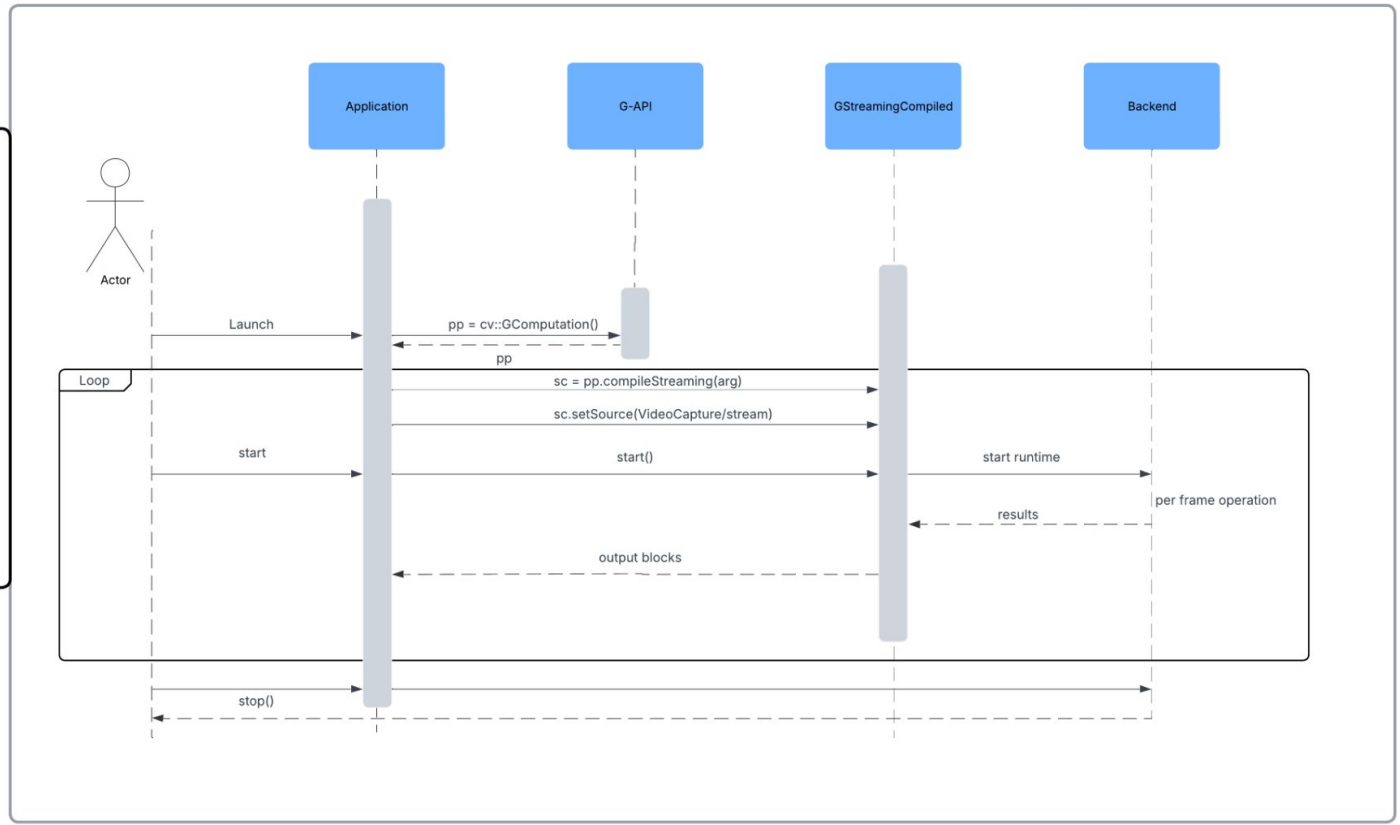
Reflexion Analysis

- Implementation and Mapping
- Actual





G-API Webcam Object Detection with Overlay (Streaming)



Team Synchronization

- Same steps and tools for everyone
- Match our earlier “layers” to real folders
- Show only what’s readable in the big graph

Limits of Our Findings

Conclusion
&
Lessons Learned

Questions?