VG101 Lab Manual

Lab 4

Instructor: Dr. Yifei ZHU
TA: Hangrui CAO
TA: Qinhang WU
TA: Muchen XU

Table of Contents

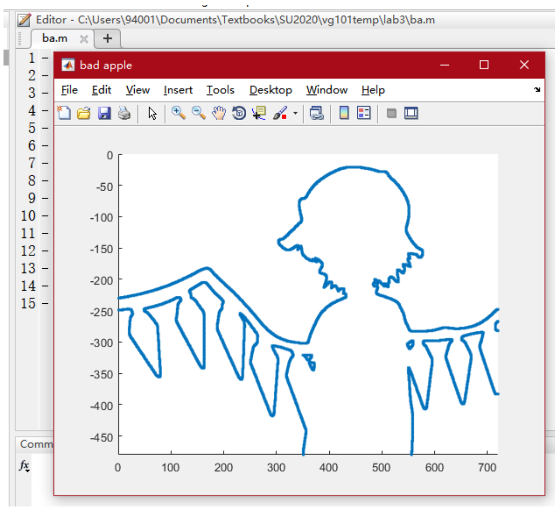
Workflow

Content	Approx. Time
Lab Design: Bad Apple Player	20 mins
Syntax Exercise: Recursion	30 mins
Break	10 mins
C Environment Configuration	60 mins
Break	10 mins
Practical Exercise	50 mins

Lab Design

In this section, we will explore how interesting MATLAB can be. We will try to visualize *bad apple* in a MATLAB style.

Bad Apple Player


Editor - C:\Users\94001\Documents\Textbooks\SU2020\vg101temp\lab3\ba.m
ba.m
bad apple
File Edit View Insert Tools Desktop Window Help
0
-50
-100
-150
-200
-250
-300
-350
-400
-450
0
100
200
300
400
500
600
700
Comm

Tags: `#plot` `#loop` `#image-processing`

The general steps are

1. Read frames from the video
2. Process image until it's only contain 0 or 1 (Binarization)
3. Find the edge in the binary image.
4. Plot the edge.

Read Video

The video should be in the working folder.

```
obj = VideoReader('./out.mp4');
while hasFrame(obj)
    ...
end
```

Process Image

Read frames from the video and binarize the image.

```
frame = readFrame(obj);
frame = rgb2gray(frame);
frame = imbinarize(frame);
```

Find Edge

MATLAB build-in function `edge` will return the function describing the edge of binary image.

```
roberts = edge(frame,'roberts');
[x,y] = find(roberts == 1);
```

Show Image

```
scatter(y, -x,'.');
axis([0 720 -480 0]);
```

Frame Rate Control

Left to the reader.

Hint: recall similar mechanism in *Threebody Problem Animation* in Lab 3.

From MATLAB to C: Environment Setup

In this section, we will set up the environment for C, that is, installing compiler *gcc*, as well as the recommended **IDE** (Integrated Development Environment) *clion*.

GCC Installation

Please refer to "Lab4_reference.pdf" (section 3, *GNU Complier Collection(GCC)*).

Clion Installation

Please follow the instruction in "Lab4_reference.pdf" (section 4, *from CLI to GUI*).

Appendix

Hello World

A sample C program *Hello World* is provided as follow:

```c
#include<stdio.h>
int main()
{
    printf("Hello World.");
    return 0;
}
```

Compile in Command Line with gcc

We list some useful gcc arguments here for your reference:

- `gcc`: GNU project C and C++ compiler
    - `infile`: pass in the file to be compiled.
        - `gcc test.cpp` will generate an executable called "a.out".
    - `-std=standard`: specify the standard you are using.
        - `-std=c11` means the program is compiled under standard `c11`.
    - `-o outfile`: specify the name of the output file.
        - `-o test` will generate an executable called "test.out".
    - `-Dmacro`: specify the macro that you apply before compiling.
        - `-DScarlet` is equivalent to `#define Scarlet` for all input files.
    - `-g`: turn on debugging options.
    - `-Wwarn`: turn on compile warning options.
        - `-Wall`: turn on **all** compile warning. Always add `-Wall` will help you a lot.
        - `-Werror`: stop compiling when any warning occurs.
    - `-Olevel`: turn on the optimization.
        - `-O3` will optimize more compared with `-O2`.
        - `-O0` preforms no optimization (default option).

Example:

```
gcc helloworld.c -o helloworld -Wall
```

Visual Studio Code

Visual Studio Code is a lightweight but powerful editor. It is available for Windows, macOS and Linux.

Concepts:

- Workspace
  - Current working folder.
- Integrate Terminal
  - ``Ctrl + Shift + ` `` to call out the terminal.
  - Open shell (command line interface).
  - Default folder is current workspace.
- Formatter
  - `Shift + Alt + F` to automatically format the code.
- Extension Market
  - Provide infinite possibility for customization.

Useful Extensions

- Readability Enhancement
  - *Bracket Pair Colorizer*
    - Rainbow matching bracket.
  - *Polacode*
    - Generate exquisite screenshot of code.
  - *Markdown All in One*
    - Markdown preview, syntax highlight.
- Development Facilitation
  - *C/C++*
    - C/C++ IntelliSense, debugging, and code browsing.
  - *Code Runner*
    - Run code snippet or code file for multiple languages.
  - *Latex Workshop*
    - Boost LaTeX typesetting efficiency with preview, compile, autocomplete, colorize, and more.
  - *Live Share*
    - Real-time collaborative development from the comfort of your favorite tools.
  - *Remote - SSH*
    - Open any folder on a remote machine using SSH and take advantage of VS Code's full feature set.
  - *Remote - WSL*
    - Open any folder in the Windows Subsystem for Linux (WSL) and take advantage of Visual Studio Code's full feature set.

Grading Rubric

Criteria	Weight	Available Time	Due Time	Entry
Attendance	30%	**4:00pm**, June.5	**11:59pm**, June.5	Canvas Assignment
In-lab quiz	70%	**9:00pm**, June.5	**11:59pm**, June.7	Canvas Quiz
Challenge bonus	10%	**9:00pm**, June.5	**11:59pm**, June.7	Canvas Quiz

- For the attendance score, you need to submit your code for **exercises** in the **worksheet** on Canvas. We won't judge the correctness of your code. You'll earn full credits as long as you've tried these exercises.
- **The challenge bonus will be counted towards your final lab score.**

Reference

[1] UM-SJTU VG101-Introduction to Computers and Programming-lab3.pdf 2019.

[2] "Visual Studio Code," Visualstudio.com, 14-Apr-2016. [Online]. Available: https://code.visualstudio.com/. [Accessed: 04-Jun-2020].