extractStrip: User Manual

Xaliq Aghakerimov, Imran Ibrahimli

1 Usage

The program accomplishes the task of cutting a strip of given size out of middle of a BMP image.

1.1 How to compile the program

There are 3 source files:

`bmp.h`
`bmp.c`
`main.c`

and a makefile called `Makefile`. You can compile the program simply using the make utility. To do this you need to open a terminal, change directory to the project folder and type

```
$ make
```

Notice that the `$` here denotes the prompt and is not typed by the user. Also, you should have `gcc` and `make` installed (they are installed by default in most GNU/Linux OS'es so you should be good to go).

1.2 How to execute the program

To execute the program you need to call the executable with two parameters: `strip_width` and `filename`.

- `-strip_width` — The width of strip which you want to cut out from the file. This parameter should be a positive integer that does not exceed width of the image itself. Notice that you must put a preceding '-' in front of this number.
- `filename` — Name of the file from which the strip will be cut. Must be a correct path to an image in BMP format. The path can be absolute or relative.

If you do not know what the parameters are, or if you forget anything, you can always call the program with `-h` or `--help`. For example, if you currently are in the same directory as executable, and your image is in `./pot/` subdirectory, you can execute the program as follows to cut a 300 pixel stripe:

```
$ ./extractStrip -300 ./pot/L1019552.bmp
```

Notice that the program prints its output to `stdout`, this output contains basic information about the input file like its width, height, header size, bits per pixel etc.. This output can be suppressed in order to, for example, keep your terminal clutter-free during processing of a large number of images. This can be achieved using redirection of `stdout` to `/dev/null`:

```
$ ./extractStrip -300 ./pot/L1019552.bmp > /dev/null
```

1.3 Batch processing

You may also want to execute the program on every image in a folder, using the same `strip_width`. In the project directory there is a shell script called `execute.sh` that does that. It takes two arguments: `strip_width` which must be a positive integer (you can omit the preceding '`-`' here) and `directory` which must be a valid path to directory that contains the images. The program will be executed on **every** file with extension `.bmp` in given directory. An example that runs on the aforementioned `./pot` directory:

```
$ . execute.sh 100 ./pot
```

2 Error messages

If the program encounters some error during execution, it will print an error message and usage. The possible errors and their likely causes are the following:

- **Too few arguments.** — Check the arguments. You probably forgot to pass the strip width and/or the filename.
- **Too many arguments.** — Check the arguments. You have passed something extra.
- **Wrong extension: expected '.bmp'** — The file that you have passed does not end with ".bmp". This one is more like a warning and will not halt the execution by itself, but usually you should have your file extensions correct.
- **Couldn't open file** — The file does not exist, or there is a mistake in the path. Check the filename argument.

- **Wrong file signature** — The "magic number" (first two bytes of the file) does not match that of a BMP file (`42 4D`). Probably your image is not in the BMP format.
- **Width argument not a number** — The width argument cannot be resolved into a number. Check width argument for typos.
- **Strip width is too large.** — The width of strip is larger than the width of the file. Obviously, this cannot be done. Try with a smaller strip width.
- **Bad strip width:** — Strip width is negative. Should not happen often (at all).

All of these error messages can be captured using the shell, since the error outputs are sent to `stderr`. Output redirection like this can be used to save error messages to a file:

```
$ ./extractStrip -100 ./pot/L1019552.bmp 2> error.txt
```

3 Results

Here you can see the result of program's execution on a sample image of a pot.

L1019552.bmp

L1019552_S100.bmp

L1019552_S500.bmp

The second and the third images were obtained using commands

```
$ ./extractStrip -100 ./pot/L1019552.bmp
```

and

```
$ ./extractStrip -500 ./pot/L1019552.bmp
```

respectively.