Assignment 6 Report

Akshit Vakati Venkata MM23B009

20 May 2024

1 Treasure Hunt II

In this task we were given an executable called ***script.sh.x*** . when this script is run in the home directory 3 branches are created and a **git** is initiated from the directory.
So by traversing through those committed branches we should find *Arceus*, the legendary Pokemon,which took refuge in Sinnoh mountains, after its creation.

1.1 Basic Working of GIT

Git is a distributed version control system that handles everything from small to extensive projects quickly and efficiently. It is free and open source, allowing multiple developers to work on a project simultaneously without overwriting each other's changes. The basic workflow of Git can be summarized in the following steps:

1. Make changes to files in your working directory.
2. Stage the changes you want to include in your next commit.
3. Commit the staged changes.

1.2 Git Commands

1.2.1 git add

The `git add` command adds changes in the working directory to the staging area. It tells Git that you want to include updates to a particular file or files in the next commit.

```
git add <file>
```

To add all changes in the current directory:

```
git add .
```

1.2.2 git commit

The `git commit` command captures a snapshot of the project's currently staged changes. Committed snapshots can be thought of as safe versions of a project.

```
git commit -m "Commit-message"
```

This command commits the staged changes with a message describing the changes.

1.2.3 git merge

The `git merge` command merges branches. It takes the contents of a source branch and integrates it with the target branch.

```
git merge <branch>
```

This command merges the specified branch into the current branch.

1.2.4 git checkout

The `git checkout` command is used to switch branches or restore working tree files. To switch to another branch:

```
git checkout <branch>
```

To create and switch to a new branch:

```
git checkout -b <new-branch>
```

1.2.5 git branch

The `git branch` command is used to list, create, or delete branches.

```
git branch
```

This lists all branches in the repository. To create a new branch:

```
git branch <new-branch>
```

To delete a branch:

```
git branch -d <branch>
```

1.2.6 git log

The `git log` command displays the commit history for the repository. It shows a list of commits along with details such as the commit hash, author, date, and commit message.

```
git log
```

To display a simplified version of the log with one line per commit:

```
git log --oneline
```

To see a specific number of most recent commits:

```
git log -n <number>
```

For example, to see the last 3 commits:

```
git log -n 3
```

1.3 Process of catching Arceus

- After knowing all those git commands : I went to branch_1 (**git checkout branch_1**) and searched for the log files (**git log**). We also know that Arceus is hiding in **question_2.cpp**.

```
Switched to branch 'branch_1'
mm23b009@ID2090:~/tmp/assignment_6$ git log
commit 5586fb1551e1c5a8a423f7bfdf8245ca67fff228 (HEAD -> branch_1)
Author: mm23b009 <mm23b009@id2090.iitm.ac.in>
Date:   Sun May 12 08:19:59 2024 +0000

    Commit #1
mm23b009@ID2090:~/tmp/assignment_6$
```

Figure 1: log file for branch_1

- after verifying that Arceus is not in this branch , then proceeded to branch_2 by (**git checkout branch_2**).
- after going through all the file version history by checkout to all the previous commits I still didn't find Arceus

```
mm23b009@ID2090:~/tmp/assignment_6$ git log
commit 2fe8b94e8a8c4f9016e2df27e072076c8011133d (HEAD -> branch_2)
Author: mm23b009 <mm23b009@id2090.iitm.ac.in>
Date:   Sun May 12 08:19:59 2024 +0000

    Commit #com+1

commit 46226a21f14d4d37d8a752e22b2f2d642a36125e
Author: mm23b009 <mm23b009@id2090.iitm.ac.in>
Date:   Sun May 12 08:19:59 2024 +0000

    Commit #com+1

commit 5586fb1551e1c5a8a423f7bfdf8245ca67fff228 (branch_1)
Author: mm23b009 <mm23b009@id2090.iitm.ac.in>
Date:   Sun May 12 08:19:59 2024 +0000

    Commit #1
mm23b009@ID2090:~/tmp/assignment_6$
```

Figure 2: log file for branch_2

- then finally proceeding to ***branch_3*** I found the file in commit id ***d4f5***.

Figure 3: Arceus Pokemon

- Then I created a new branch called *MASTER* by

  git checkout -b MASTER

  .

- Then I merged this branch with branch_3 by

  git merge branch_3

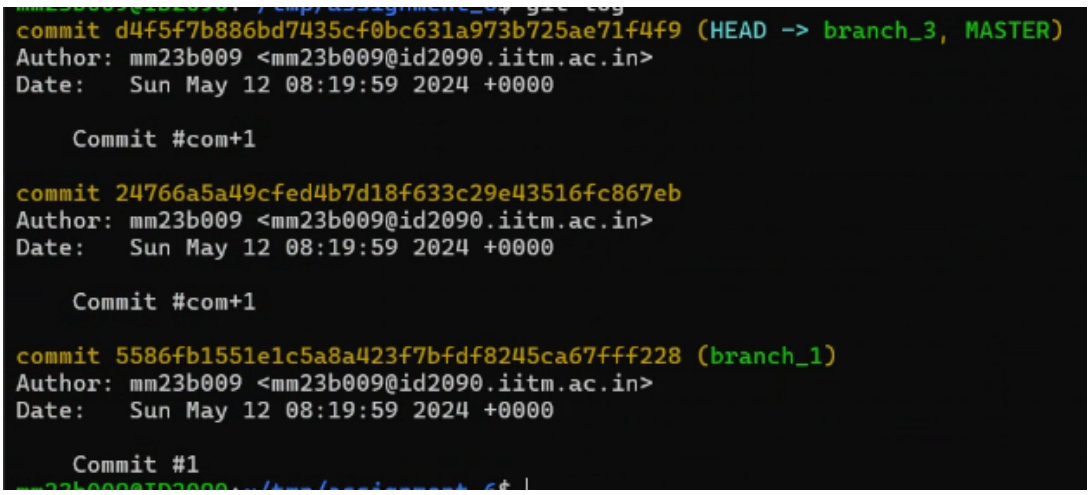


Figure 4: log file for branch_3

2 Image Processing

In image processing, a kernel is a small matrix used for transforming an image. This is achieved by convolving the image with the kernel matrix. Consider a 3×3 kernel which is a matrix A_{ij} that will operate on an image as follows: For a pixel located at (m, n) in the image, the new value of the pixel $P^{\text{new}}_{m,n}$ is given by the following formula:

$$P^{\text{new}}_{m,n} = \sum_{i=0}^{2} \sum_{j=0}^{2} A_{ij} P^{\text{old}}_{m-1+i,n-1+j} \tag{1}$$

2.1 Kernel Operations

2.1.1 Edge Detection

One common kernel operation is edge detection. An example of a 3x3 kernel for edge detection (Sobel operator) is:

$$A = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

When this kernel is convolved with an image, it highlights the edges by calculating the gradient of the image intensity.

2.1.2 Sharpening

Sharpening is another important image operation that enhances the edges and fine details in an image. An example of a 3x3 kernel for sharpening is:

$$A = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

This kernel works by amplifying the center pixel's value while subtracting the values of the surrounding pixels, which enhances the edges and details.

2.1.3 Blur (Smoothing)

Another common kernel operation is blurring, which can help reduce noise. A simple 3x3 averaging kernel is:

$$A = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Convolving an image with this kernel will smooth the image by averaging the pixel values in the neighborhood.

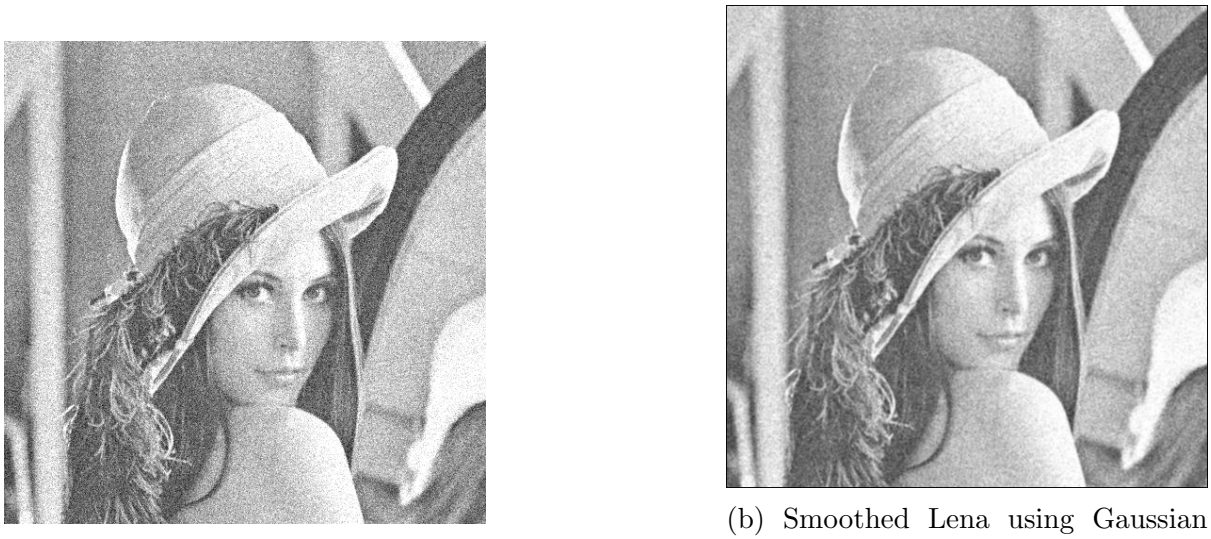
(a) Original Lena (b) Edge Detection (c) Sharpened Lena

Figure 5: Comparison of original and processed images

2.2 Noise Reduction with Gaussian Blur

To reduce noise in the second image (the distorted version of Lena), we can apply the Gaussian blur kernel described above. The Gaussian blur works by weighing the pixel values in the neighborhood according to the Gaussian function, which gives more importance to the central pixels and less to the peripheral ones.

- **Why it works:** Gaussian blur effectively reduces high-frequency noise while preserving the edges better than a simple averaging filter. The Gaussian function's weighting helps to maintain more of the image's structure while smoothing out the noise.
- **Why it might not work:** While Gaussian blur reduces noise, it can still cause some loss of detail and may not be sufficient for very high levels of noise. It is a compromise between noise reduction and detail preservation.



(a) Noisy Lena (b) Smoothed Lena using Gaussian Blur

Figure 6: Comparison of Noisy and Smoothed Lena images

The figures above illustrate the original Lena, the distorted version, and the noise-reduced version after applying the Gaussian blur kernel.

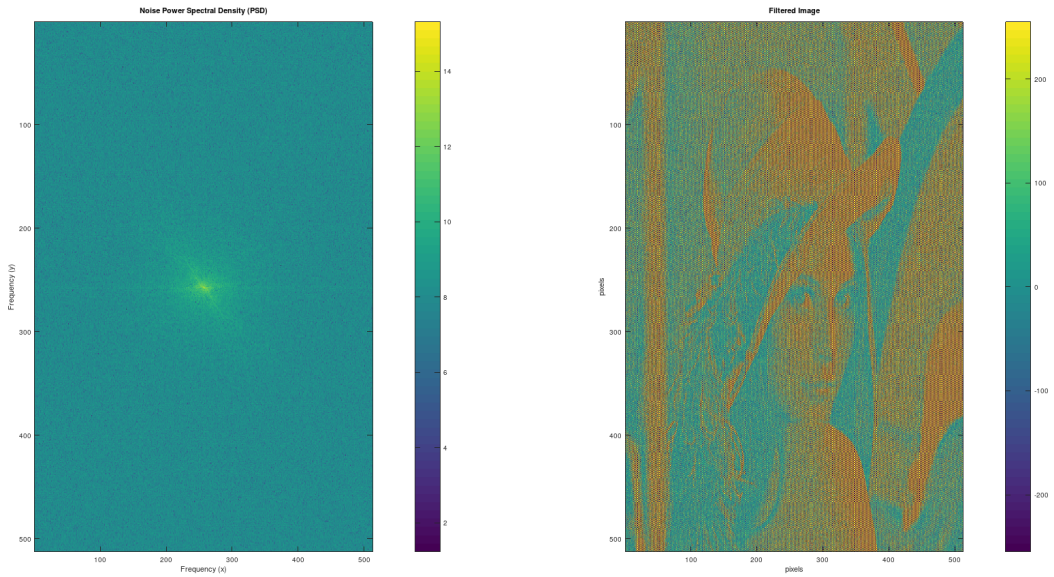


Figure 7: Noise analysis for Noisy_Lena

2.3 USAGE:

- First untar the MM23B009.tar
- For running the scripts , go to ***assignment_6/*** directory and run

  ./question_2.sh

- Then files named *sharp.jpg* , *edgedetection.jpg*, *gaussianblur_image.png* will be created.
- I have also included these pictures into my report.