



# 系统开发工具基础实验报告

题目：  
命令行环境；Python 入门基础；Python 视觉应用；

学生姓名 周洋迅 学号 24020007175  
学部、学院 (中心) 信息科学与工程学部  
专业 计算机科学与技术  
日期 2025 年 9 月 12 日  
github 链接 <https://github.com/zyzgusg/SysDevelopmentTools>

中国海洋大学

## 目录

<b>1</b>	<b>练习内容</b>	<b>2</b>
1.1	命令行环境	2
1.1.1	任务控制	2
1.1.2	别名	2
1.1.3	配置文件	2
1.1.4	远端设备	2
1.2	python 图像处理	3
<b>2</b>	<b>练习结果</b>	<b>3</b>
2.1	结果截图	3
<b>3</b>	<b>心得体会</b>	<b>11</b>

## 1 练习内容

### 1.1 命令行环境

#### 1.1.1 任务控制

1. 使用 `pgrep` 来查找 `pid` 并使用 `pkill` 结束进程而不需要手动输入 `pid`。(提示: 使用 `-af` 标记)。
2. 请编写一个 `bash` 函数 `pidwait`, 它接受一个 `pid` 作为输入参数, 然后一直等待直到该进程结束。您需要使用 `sleep` 来避免浪费 CPU 性能。

#### 1.1.2 别名

3. 创建一个 `dc` 别名, 它的功能是当我们错误的将 `cd` 输入为 `dc` 时也能正确执行。
4. 执行 `history | awk ' $1="" ; print substr($0,2)' | sort | uniq -c | sort -n | tail -n 10` 来获取您最常用的十条命令, 尝试为它们创建别名

#### 1.1.3 配置文件

5. 为您的配置文件新建一个文件夹, 并设置好版本控制
6. 在其中添加至少一个配置文件, 比如说您的 `shell`, 在其中包含一些自定义设置 (可以从设置 `$PS1` 开始)。
7. 建立一种在新设备进行快速安装配置的方法 (无需手动操作)。最简单的方法是写一个 `shell` 脚本对每个文件使用 `ln -s`, 也可以使用专用工具
8. 在新的虚拟机上测试该安装脚本。
9. 将您现有的所有配置文件移动到项目仓库里。
10. 将项目发布到 GitHub。

#### 1.1.4 远端设备

11. 前往 `./ssh/` 并查看是否已经存在 SSH 密钥对。如果不存在, 请使用 `ssh-keygen -o -a 100 -t ed25519` 来创建一个。
12. 在 `ssh/config` 加入以下内容:

```
1 Host vm
2     User username_goes_here
3     HostName ip_goes_here
4     IdentityFile ~/.ssh/id_ed25519
5     LocalForward 9999 localhost:8888
6
```

13. 使用 `ssh-copy-id vm` 将您的 `ssh` 密钥拷贝到服务器。
14. 使用 `python -m http.server 8888` 在您的虚拟机中启动一个 Web 服务器并通过本机的 `http://localhost:9999` 访问虚拟机上的 Web 服务器

15. 使用 `sudo vim /etc/ssh/sshd_config` 编辑 SSH 服务器配置，通过修改 `PasswordAuthentication` 的值来禁用密码验证。通过修改 `PermitRootLogin` 的值来禁用 root 登录。然后使用 `sudo service sshd restart` 重启 ssh 服务器，然后重新尝试。

## 1.2 python 图像处理

16. 使用 Pillow 库增强图像对比度
17. 使用 Numpy 库分离图片 rgb 通道
18. 使用 Scipy 库进行图片高斯模糊
19. 使用 OpenCV 库进行图像裁剪
20. 使用 Skimage 库进行图像缩放

## 2 练习结果

### 2.1 结果截图

```
zysgusg@aaa:~$ pgrep -a sleep
476 sleep 10000
zysgusg@aaa:~$ pkill -f sleep
[1]+  Terminated                  sleep 10000
zysgusg@aaa:~$ pgrep -a sleep
zysgusg@aaa:~$
```

图 1: 使用 pgrep 来查找 pid 并使用 pkill 结束进程

```
zysgusg@aaa:~$ source pidwait.sh
zysgusg@aaa:~$ sleep 60 &
[1] 980
zysgusg@aaa:~$ pid=$!
zysgusg@aaa:~$ pidwait "$pid"
ls
[1]+  Done                          sleep 60
zysgusg@aaa:~$ ls
```

图 2: pidwait 使用

```
zysgusg@aaa:~$ dc
Command 'dc' not found, but can be installed with:
sudo apt install dc
zysgusg@aaa:~$ alias dc="cd"
zysgusg@aaa:~$ dc /
zysgusg@aaa:/$ |
```

图 3: 创建 dc 别名

```

zyzgusg@aaa:/$ history | awk '{ $1="" ; print substr($0,2) }' | sort | uniq -c | sort -n | tail -n 10
  7 vim marco.sh
  7 vim test.py
  8 cd
  9 gdb
 11 vim start.sh
 13 cd ~
 14 fish
 14 pwd
 16 clear
 57 ls
zyzgusg@aaa:/$ alias cl="clear"
zyzgusg@aaa:/$ alias fi="fish"
zyzgusg@aaa:/$

```

图 4: 获取最常用的十条命令, 尝试为它们创建别名

```

zyzgusg@aaa ~> mkdir dotfiles
zyzgusg@aaa ~> cd dotfiles/
zyzgusg@aaa ~/dotfiles> git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/zyzgusg/dotfiles/.git/

```

图 5: 为配置文件新建一个文件夹, 并设置好版本控制

```

zyzgusg@aaa ~/dotfiles (master)> mv ~/.vimrc .vimrc
zyzgusg@aaa ~/dotfiles (master)> mv ~/.bashrc .bashrc
zyzgusg@aaa ~/dotfiles (master)> git add *
fish: No matches for wildcard '*'. See 'help expand'.
git add *
^
zyzgusg@aaa ~/dotfiles (master) [124]> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .bashrc
    .vimrc

nothing added to commit but untracked files present (use "git add" to track)
zyzgusg@aaa ~/dotfiles (master)> git add .*

```

图 6: 在其中添加至少一个配置文件

```

zyzgusg@aaa:~/dotfiles$ bash start.sh
Linked /home/zyzgusg/dotfiles/.bashrc -> /home/zyzgusg/.bashrc
Linked /home/zyzgusg/dotfiles/.vimrc -> /home/zyzgusg/.vimrc
Linked /home/zyzgusg/dotfiles/.gitconfig -> /home/zyzgusg/.gitconfig

```

图 7: 快速安装配置

```
(zyzgusg@kali)-[~]
└─$ git clone https://github.com/zyzgusg/dotfiles.git
正克隆到 'dotfiles' ...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 1), reused 7 (delta 1), pack-reused 0 (from 0)
接收对象中: 100% (7/7), 4.46 KiB | 4.46 MiB/s, 完成.
处理 delta 中: 100% (1/1), 完成.

(zyzgusg@kali)-[~]
└─$ bash dotfiles/start.sh
Backed up /home/zyzgusg/.bashrc to /home/zyzgusg/.bashrc.bak
Linked /home/zyzgusg/dotfiles/.bashrc → /home/zyzgusg/.bashrc
Linked /home/zyzgusg/dotfiles/.vimrc → /home/zyzgusg/.vimrc
Linked /home/zyzgusg/dotfiles/.gitconfig → /home/zyzgusg/.gitconfig

(zyzgusg@kali)-[~]
└─$
```

图 8: 在新的虚拟机上测试安装脚本

```
zyzgusg@aaa:~/dotfiles$ ls -a
.  ..  .bashrc  .git  .gitconfig  .vimrc  start.sh
```

图 9: 将现有的所有配置文件移动到项目仓库里

```
zyzgusg@aaa:~/dotfiles$ git remote add origin git@github.com:zyzgusg/DotFiles.git
error: remote origin already exists.
zyzgusg@aaa:~/dotfiles$ git remote add origin_ssh git@github.com:zyzgusg/DotFiles.git
zyzgusg@aaa:~/dotfiles$ git push -u origin_ssh main
The authenticity of host 'github.com (20.200.245.247)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 32 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 4.46 KiB | 4.46 MiB/s, done.
Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To github.com:zyzgusg/DotFiles.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin_ssh'.
```

图 10: 将项目发布到 GitHub。

```
zyzgusg@aaa:~$ cd ~/.ssh
zyzgusg@aaa:~/.ssh$ ls
id_rsa  id_rsa.pub  known_hosts  known_hosts.old
```

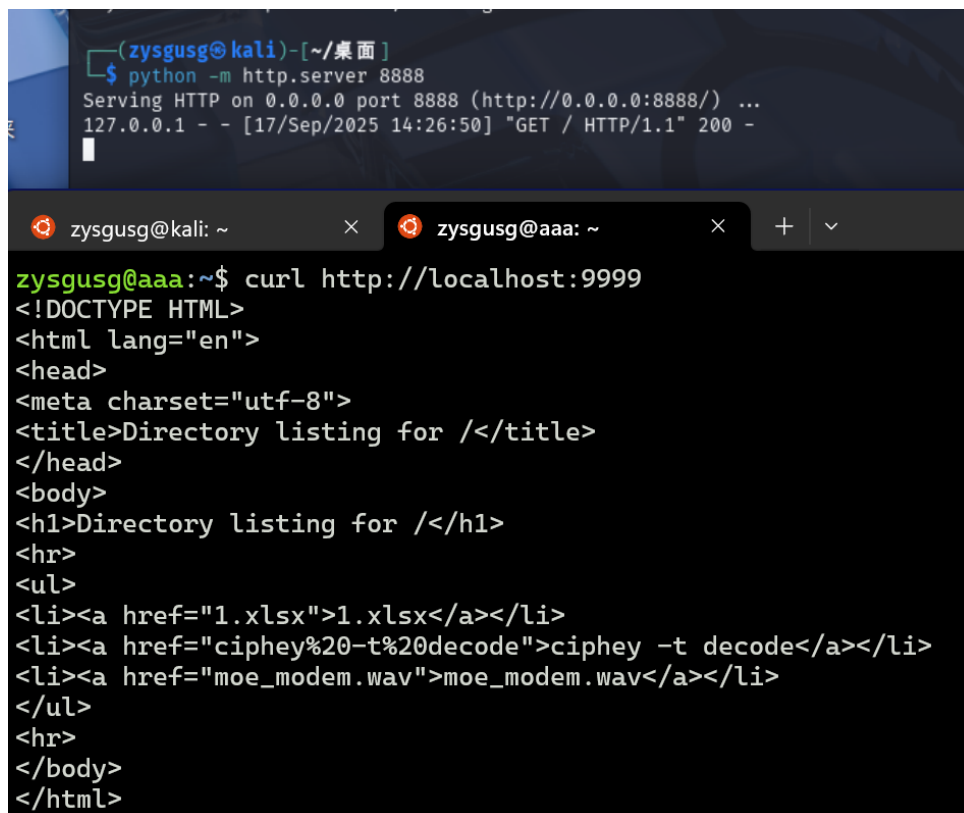
图 11: 查看是否已经存在 SSH 密钥对

```
4 Host vm
3     User zyzgusg
2     HostName 192.168.83.128
1     IdentityFile ~/.ssh/id_rsa
    LocalForward 9999 localhost:8888
```

图 12: .ssh/config

```
zyzgusg@aaa:~/.ssh$ ssh-copy-id vm
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zyzgusg/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
zyzgusg@192.168.83.128's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'vm'"
and check to make sure that only the key(s) you wanted were added.
```

图 13: 使用 ssh-copy-id vm 将 ssh 密钥拷贝到服务器



The screenshot shows a Kali Linux terminal window at the top with the command `python -m http.server 8888` running. Below the terminal is a web browser window with two tabs: `zyzgusg@kali: ~` and `zyzgusg@aaa: ~`. The active tab shows the output of `curl http://localhost:9999`, which is an HTML directory listing for the root directory. The listing includes files `1.xlsx`, `ciphey -t decode`, and `moe_modem.wav`.

```
(zyzgusg@kali)-[~/桌面]
$ python -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
127.0.0.1 - - [17/Sep/2025 14:26:50] "GET / HTTP/1.1" 200 -

zyzgusg@kali: ~
zyzgusg@aaa: ~
zyzgusg@aaa:~$ curl http://localhost:9999
<!DOCTYPE HTML>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Directory listing for /</title>
</head>
<body>
<h1>Directory listing for /</h1>
<hr>
<ul>
<li><a href="1.xlsx">1.xlsx</a></li>
<li><a href="ciphey%20-t%20decode">ciphey -t decode</a></li>
<li><a href="moe_modem.wav">moe_modem.wav</a></li>
</ul>
<hr>
</body>
</html>
```

图 14: 虚拟机中启动一个 Web 服务器

```
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys
#AuthorizedKeysFile .ssh/

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser none

# For this to work you will a
#HostbasedAuthentication no
# Change to yes if you don't
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts
#IgnoreRhosts yes

# To disable tunneled clear t
PasswordAuthentication no
```

图 15: 编辑 SSH 服务器配置



```
(zyzgusg@kali)~$  
$ Connection to 192.168.83.128 closed.  
zyzgusg@aaa:~$ ssh vm  
Linux kali 6.12.13-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.13-1kali1 (2025-02-11)  
x86_64  
  
The programs included with the Kali GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Wed Sep 17 14:24:32 2025 from 192.168.83.1
```

图 16: 重新连接 ssh 服务器

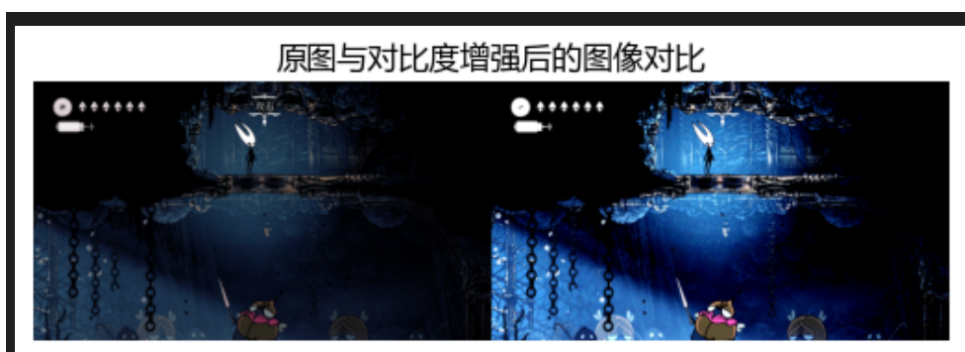


图 17: 增强图像对比度



图 18: 分离图片 rgb 通道置



图 19: 高斯模糊



图 20: 图像裁剪



图 21: 图像缩放

### 3 心得体会

- 通过本次实验，我学到了命令行环境和 Python 视觉应用相关的知识，加深了对 ssh 连接远端的理解。