Universität Tübingen
Prof. Dr.-Ing. Hendrik P.A. Lensch
Lehrstuhl für Computergrafik
Lukas Ruppert (lukas.ruppert@uni-tuebingen.de)

February 19, 2024

Computer Graphics Group
Getting Started Guide

1 Connecting via SSH

The first thing you need to do is to connect to our machines via SSH. Simply run the following command in a terminal to test your connection.

```
ssh <your WSI user>@cgcontact.cs.uni-tuebingen.de
```

For signing in, use your *WSI* user name and password.

On your first sign-in you may be prompted to change your initial password. In this case, you need to type your initial password again, and then a two times a new password.

1.1 Computer Graphics Machines

Never work on `cgcontact` directly. This machine's entire purpose is to serve as a SSH gateway to our machines inside the WSI network. Within your SSH connection to `cgcontact`, use SSH again to connect to one of the following machines:

Pool Machines. Our pool machines are listed in our `pool-smi` tool (Ctrl+C to quit): `cgpool1801`..`cgpool1803`, `cgpool1900`..`cgpool1912`, `cgpoolsand1900`..`cgpoolsand1907`.

Compute Servers. If you're a staff member or if you have been explicitly granted permission, you can also use one of the staff PCs and the compute servers listed in the `cluster-smi` tool (Ctrl+C to quit): `cluster-gpu00`..`cluster-gpu04`, `glorifolia`, `heracleum`, `myristica`, `pulsatilla`.

1.2 SSH configuration

You can simplify the SSH connection process, especially to internal machines, by creating a SSH configuration file `.ssh/config` in your `home` folder on your local machine:

```
Host cgcontact
    HostName cgcontact.cs.uni-tuebingen.de
    User <your WSI username>
    ForwardAgent yes

Host cgpool???? cgpoolsand19?? cluster-gpu0? glorifolia heracleum myristica pulsatilla
    HostName %h.cs.uni-tuebingen.de
    User <your WSI username>
    ForwardAgent yes
    ProxyJump cgcontact
```

This will allow you to connect to any machine in the list by simply writing e.g.

```
ssh cgpool1905
```

rather than

```
ssh <your WSI user>@cgcontact.cs.uni-tuebingen.de
# then, from cgcontact
ssh cgpool1905
```

Visual Studio Code Remote: SSH (CG users only)

If you want to use VS Code's SSH extension, you have to spell out the host names explicitly, rather than using wildcards like `?` or `*`. I.e. type `cgpool1800 cgpool1801` ... instead of `cgpool????`.

Please note: Using VSCode's SSH feature only works for CG users (see section on AFS filesystem below).

1.3 Login via SSH Key (CG users only)

This section only applies if your account belongs to a computer graphics group (e.g., `cgstaff`, `cgstud`). To find out, run `groups` on one of our machines.

First, create a SSH key on your machine (if you don't have one already):

```
ssh-keygen -t rsa
```

and follow the instructions on screen (generally, just confirm the defaults by pressing *Enter*).

Then, copy your *public* key into the `~/.ssh/authorized_keys` file in your `home` on our machines:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub cgcontact
```

You will have to enter the password for your WSI account this time. For all following connections, your private SSH Key will be used to safely establish the connection without requiring a password.

You will *not* have access to the AFS filesystem when logging in with an SSH Key. To access the AFS, run `kinit`, type your password and then run `aklog`.

2 Shared Filesystems

In the WSI network, your home directory is shared across all machines and accessible to you as

```
 /home/<your WSI user>/
```

No matter which machine you use, you will see the same files. This means that you can easily switch between machines without having to reconfigure anything.

If you are a member of the computer graphics group or a student of the computer graphics group, your `/home/<your WSI user>/` is on our file servers. (Run `groups` to check.)

For non-CG users, that home is actually located on the *AFS* and also accessible at

```
 /afs/cs/home/<your WSI user>/
```

A few words on the AFS. To access the AFS, you need a "token", which is created during sign in and stays valid for up to 1 day or until you sign out. If you have processes running in the background, they will not be able to access your home after you sign out or once a day has passed. You can check the lifetime of your token using `tokens` (or `klist`). You can also renew your token by running `kinit`, which will ask for your password again. You may also have to run `aklog` afterwards.

Please note that tokens are per session, so they will only affect e.g. your current SSH connection. Also note that your quota on the AFS is *limited to 5 GiB*. To check your quota, run

```
fs listquota
```

Since file access requires a valid token, using a SSH Key for signing in is sadly not an option, since the required `authorized_keys` file will not be accessible by the SSH server until you're already authenticated. (This will also mess with any attempts of using VS Code's SSH-Remote feature! – Only the initial connection works, connecting again will fail as you connect to a server running in a logged-out session!)

2.1 Computer Graphics File Systems

We have the following file systems:

/home (*only* for users in a `cg` group, e.g., `cgstaff` or `cgstud`) (snapshots + backup)

/graphics/opt for shared software (snapshots + backup)

/graphics/projects, /graphics/projects2 for project data (snapshots + backup)

/graphics/scratch, /graphics/scratch2 for temporary data and data that could easily be recreated (snapshots only!)

Temporary Data. Please put all your temporary data (e.g. training logs) into `/graphics/scratch2`. Create a folder in the `staff` or `students` folder for yourself if necessary.

Datasets. Download datasets directly into `/graphics/scratch2`, never into your home!

Permanent Storage. Once you created some data or results that needs to be kept long term without any changes, move it over to `/graphics/projects2`, so that it will be backed up.

Trash. Please empty your trash folder regularly. Trashed files in your home will still fill up our backups.

Snapshots. On all our file systems, snapshots are being created every few hours which are kept for several days. If you accidentally messed up or deleted a file or folder, ask a CG admin to get it back. :)

Backups. Except for `/graphics/scratch` and `/graphics/scratch2`, backups are made Monday to Friday early in the morning. All files visible when taking the snapshot for the backup will continue to stay around and occupy space until that snapshot is deleted (roughly two years later), so make sure to not store large temporary files on these file systems.

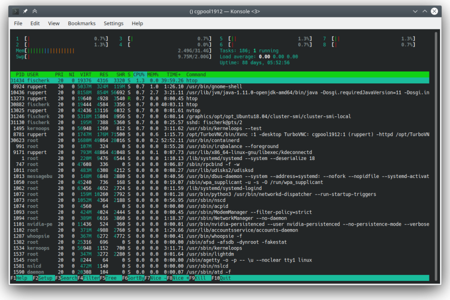
If you download large files which could easily be downloaded again, do not download them to your home, but directly to `/graphics/scratch2` or the machine's local `/tmp` folder.

Compute Servers. On our compute servers, there is an additional shared scratch space available at `/ceph`. Some machines have additional fast local SSD storage available at `/scratch`.

3 Rules on Computer Graphics Machines

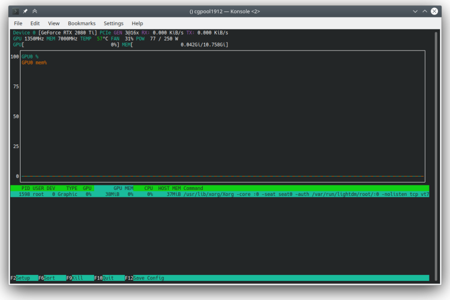
Make sure to not disturb existing jobs. Once you have chosen a machine, use `htop` and `nvtop` to check the CPU, GPU, and memory usage of the machine. On many of these machines, other staff members or students might already be running some jobs. Please do your best to not interfere with their work. If at all possible, use a machine without running processes from other users. If you use the compute servers, use `cluster-smi` to monitor the GPU usage. Similarly, use `pool-smi` to find a free pool machine to work on. Some machines are turned off to conserve energy and can easily be turned on.

Monitor your running processes. Just like when checking for existing jobs, you should also check on your own processes to make sure that everything is working correctly. Again, use `htop` and `nvtop` to monitor CPU, GPU, and memory usage.



```
  1  [|                         0.7%]   3  [|                         0.7%]   5  [||                        1.3%]   7  [||                        1.3%]
  2  [|                         1.3%]   4  [                          0.0%]   6  [|                         0.7%]   8  [|                         0.7%]
  Mem[|||||||||||||||||                                    2.49G/31.4G]   Tasks: 186; 1 running
  Swp[|                                                    9.75M/2.00G]   Load average: 0.00 0.00 0.00
                                                                          Uptime: 88 days, 05:52:56

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
31434 fischerk   20   0 19376  4316  3320 S  1.3  0.0 39:59.26 htop
 8924 ruppert    20   0 5037M  324M  119M S  0.7  1.0  1:26.10 /usr/bin/gnome-shell
10436 ruppert    20   0 8158M  854M 56692 S  0.7  2.7  3:21.11 /usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java -Dosgi.requiredJavaVersion=11 -Dosgi.in
13273 ruppert    20   0 19640  4928  3540 R  0.7  0.0  0:00.45 htop
30882 fischerk   20   0 19444  4584  3356 S  0.7  0.0 40:03.11 htop
13025 ruppert    20   0 42436 13116  4032 S  0.7  0.0  0:01.61 nvtop
31246 fischerk   20   0 5318M 15804  8956 S  0.7  0.0  6:08.14 /graphics/opt/opt_Ubuntu18.04/cluster-smi/cluster-smi-local
31130 fischerk   20   0  195M  7388  5360 S  0.7  0.0  0:25.57 sshd: fischerk@pts/2
 1495 kernoops   20   0 56948  1260   812 S  0.7  0.0  3:11.62 /usr/sbin/kerneloops --test
 8781 ruppert    20   0 1747M  176M 71500 S  0.0  0.6  1:15.73 /opt/TurboVNC/bin/Xvnc :1 -desktop TurboVNC: cgpool1912:1 (ruppert) -httpd /opt/TurboVN
30623 root       20   0 1680M 49864 28016 S  0.0  0.2 52:52.11 /usr/bin/containerd
  991 root       20   0  107M   324     0 S  0.0  0.0  8:55.28 /usr/sbin/irqbalance --foreground
 9171 ruppert    20   0  793M 48864 41048 S  0.0  0.1  0:07.73 /usr/lib/x86_64-linux-gnu/libexec/kdeconnectd
    1 root       20   0  220M  9476  6544 S  0.0  0.0  1:10.13 /lib/systemd/systemd --system --deserialize 18
  747 root       20   0 47608   336     0 S  0.0  0.0  0:06.87 /sbin/rpcbind -f -w
 1011 root       20   0  483M  6308  4212 S  0.0  0.0  0:08.27 /usr/lib/udisks2/udisksd
 1013 messagebu  20   0  148M  6848  2880 S  0.0  0.0  0:40.56 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activat
 1050 root       20   0 45240   736   168 S  0.0  0.0  0:19.07 /sbin/wpa_supplicant -u -s -O /run/wpa_supplicant
 1062 root       20   0 63456  4652  2724 S  0.0  0.0  0:11.59 /lib/systemd/systemd-logind
 1072 root       20   0  159M 10260  2792 S  0.0  0.0  0:01.28 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
 1073 root       20   0 1052M  4364  2188 S  0.0  0.0  0:56.95 /usr/sbin/nscd
 1074 root       20   0  4560    64     0 S  0.0  0.0  0:00.00 /usr/sbin/acpid
 1093 root       20   0  424M  4024  2444 S  0.0  0.0  0:00.45 /usr/sbin/ModemManager --filter-policy=strict
 1094 root       20   0  389M  6616  3860 S  0.0  0.0  1:18.37 /usr/sbin/NetworkManager --no-daemon
 1101 nvidia-pe  20   0 11436   524   360 S  0.0  0.0  0:00.01 /usr/bin/nvidia-persistenced --user nvidia-persistenced --no-persistence-mode --verbose
 1102 root       20   0  371M  4988  2760 S  0.0  0.0  1:29.66 /usr/lib/accountsservice/accounts-daemon
 1287 whoopsie   20   0  367M  6272  4772 S  0.0  0.0  0:00.41 /usr/bin/whoopsie -f
 1382 root       20   0 25316   696     0 S  0.0  0.0  0:00.00 /sbin/afsd -afsdb -dynroot -fakestat
 1534 kernoops   20   0 56948  1152   700 S  0.0  0.0  3:11.71 /usr/sbin/kerneloops
 1537 root       20   0  347M  3272  2280 S  0.0  0.0  0:01.64 /usr/sbin/lightdm
 1545 root       20   0  8244    64     0 S  0.0  0.0  0:00.00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 1581 nslcd      20   0  472M  1140     0 S  0.0  0.0  0:00.00 /usr/sbin/nslcd
 1590 daemon     20   0 20308   104     0 S  0.0  0.0  0:00.07 /usr/sbin/atd -f
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```



```
Device 0 [GeForce RTX 2080 Ti] PCIe GEN 3@16x RX: 0.000 KiB/s TX: 0.000 KiB/s
GPU 1350MHz MEM 7000MHz TEMP  57°C FAN  31% POW  77 / 250 W
GPU[                                 0%] MEM[                  0.042Gi/10.758Gi]

  PID USER DEV    TYPE  GPU        GPU MEM    CPU  HOST MEM Command
 1598 root   0 Graphic   0%    38MiB   0%     0%     37MiB /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7

F2Setup  F6Sort   F9Kill   F10Quit   F12Save Config
```

Figure 1: Use `htop` to monitor CPU and memory usage, use `nvtop` to monitor GPU usage. This machine is idle and can be used freely.

4 VNC Connections

You can start a VNC server using the command `vncserver`:

```
$ vncserver

Desktop 'TurboVNC: <cgpool>:2 (<your WSI user>)' started on display <cgpool>:2

Starting applications specified in /opt/TurboVNC/bin/xstartup.turbovnc
Log file is /home/<your WSI user>/.vnc/<cgpool>:2.log
```

On the first startup, it will ask you to set a password for incoming VNC connections. You can choose whatever you like, but you are limited to 8 characters. It will also ask for a view-only password, which you can skip. To reset your VNC password, simply remove the `.vnc/passwd` file in your home and you will be asked to set a new password on the next start of the VNC server.

Always check for running VNC servers using `vncserver -list`, before starting a new one:

```
$ vncserver -list

TurboVNC sessions:

X DISPLAY #     PROCESS ID
:2              1125884
```

Make sure to sign out of the VNC session when you're done using it. If necessary, kill the VNC session:

```
$ vncserver -kill :2
Killing Xvnc process ID 1125884
```

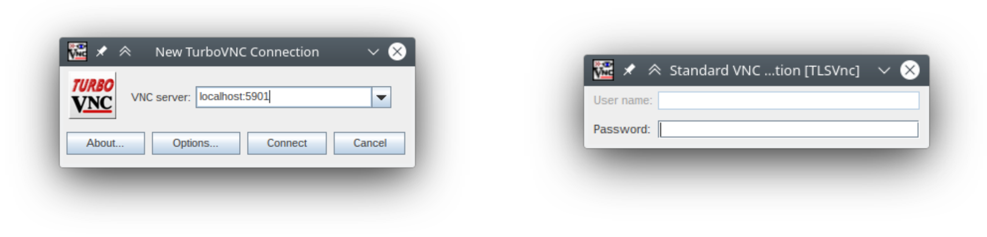
Each VNC server will be accessible from port 5900 + its display id, i.e., port 5902 for display `:2` and so on. Only the ports from 5900 to 5910 are accessible from other machines (like `cgcontact`). Higher ports are blocked by the firewall.

Port forwarding. To be able to access the VNC server from your own machine, use port forwarding:

```
ssh -L<local port>:<cgpool>:<5900+display id> cgcontact
# e.g. ssh -L5902:cgpool1905:5902 cgcontact
```

This will connect you to `cgcontact` via SSH and will connect the `<local port>` on your own machine (which can be any number between 1025 and 65535 – simply use the same number) to the port on the pool machine. When you close the SSH connection to `cgcontact`, the tunnel to the VNC server's port will also close. Consider running a program like `tmux` or `htop` to keep that SSH session alive.

Connecting with your VNC client. While the SSH tunnel is active, you can – on your machine – connect to the VNC server via `localhost:5902` (or the port number you chose). We recommend using the TurboVNC viewer, which allows to adjust the screen size, but any VNC client should work.



VNC server: localhost:5901

User name:

Password:

Figure 2: Connect to the VNC server forwarded to your local machine and type your password.

Your VNC client will ask you for the password you set during the first start of the VNC server.
You should now have access to the graphical desktop environment.

GNOME screen lock. Never have more than one GNOME session running at a time or you will not get past the unlock screen. Always log out when you're done using a GNOME session.
Alternatively, it may work to disable the automatic screen locking.

KDE sessions. You can start a KDE session in the VNC server with the following command:

```
 vncserver -wm startplasma-x11
```

OpenGL support in VNC sessions. By default, OpenGL in the VNC session is emulated in software. You can selectively run graphical programs on the GPU using

```
vglrun <your program here>
```

To check your version of OpenGL, run

```
glxinfo | grep "OpenGL version"
# or via VirtualGL
vglrun glxinfo | grep "OpenGL version"
```

5 Transferring Files

You can transfer files via SSH using SFTP: `sftp://cgcontact`
(Simply click the link or type it into the address bar of your file manager.)
If necessary, navigate to `/home/<your WSI user>/`, e.g. `sftp://cgcontact/home/<your WSI user>`.
On Windows, you can use WinSCP.

6 Customize your Environment

Our machines run Ubuntu 22.04 with lots of software already installed. If you need something that is not yet installed, ask for it to be installed. Software that is available on the Ubuntu repositories will usually be installed, no questions asked. Software that needs to be manually (compiled and) installed, can be installed to `/graphics/opt/opt_Ubuntu22.04/`.

Do *not* pollute your `home` directory with software that could be shared with the entire group. In general, there is *no need* for local package managers like conda!

If you have any questions regarding our setup, ask cg-admin@cs.uni-tuebingen.de.

Login Shell / .bash_profile. Whenever you login to a machine, `bash` is running in the terminal, awaiting your commands. It can be customized by creating two hidden files in your home: `.bashrc` and `.bash_profile`. Your `.bash_profile` file is sourced (executed as if typed in the current shell) by `bash` whenever you login. It should only contain the following content (you need to create it!):

```
if [ -f ~/.bashrc ]; then
    . ~/.bashrc # does the same as "source ~/.bashrc"
fi
```

This will check for your `.bashrc` file and source it (execute it as if typed in the current shell). By having this file, your login shell (the shell that runs when you login via SSH) will behave the same as when you run `bash` manually from the command line or when you open a terminal from a graphical session. In these cases, only the `.bashrc` file is sourced and the `.bash_profile` is ignored.

Your environment / .bashrc. You can override our default settings in your `.bashrc` file. E.g., to use a different version of the **CUDA Toolkit**, you can simply copy the following into your `.bashrc` file and adjust the version number:

```
# CUDA
export CUDA_TOOLKIT_VER=11.8.0
export CUDA_PATH="${GRAPHICS_OPT}/cuda/toolkit_${CUDA_TOOLKIT_VER}/cuda"
export PATH="${CUDA_PATH}/bin:${CUDA_PATH}/nvvm/bin:${PATH}"
export LD_LIBRARY_PATH="${CUDA_PATH}/lib64:${LD_LIBRARY_PATH}"
```

You can also add further configurations depending on your needs.

The `PATH` variable is a list of directories in which to check for executable programs. Similarly, the `LD_LIBRARY_PATH` variable is a list of directories in which to check for libraries (`.so`). Individual paths are separated by a colon `:`. Make sure to always append from the front, and prefer using quotes (you need them if there is a space somewhere in the path).

In addition to exporting environment variables, you can also set aliases to act like programs or to set default arguments. E.g., the following alias is set by default on most modern Linux systems:

```
alias ls='ls --color=auto'
```

You might also want to enable auto-completion on the command line:

```
source /usr/share/bash-completion/bash_completion
```

Whenever you press `Tab` on the command line, `bash` will try to auto-complete the command you're writing. By default, it can complete partially written names of files and directories. If you press `Tab` twice, it will also print all the available options for you. By sourcing this file, the arguments of many programs like `git` can also be completed by pressing `Tab`.

Editing text files on the command line. If you're connecting via SSH, you can use the command line editors `nano` or `vi` (VIM) to create and edit text files on the command line.

If you're not familiar with either one, `nano` is probably easier to use. To save files in `nano`, press `Ctrl+O` and confirm with `Y`. To exit `nano`, press `Ctrl+X`.

In `vi` (VIM), press `i` or `Insert` to enter edit mode and make your changes. Press `Esc` to switch back to command mode. In command mode, write `:w` and press `Enter` to save, `:q` to quit and `:q!` to force-quit without saving. You can find more documentation online.

More command line tools. More command line tools you might want to look into are:

`htop` (system monitor), `nvtop` (Nvidia GPU usage), `less` (pager - view files more easily, quit with `q`), `cat` (concatenate - easiest way to print a text file), `tmux` (terminal multiplexer - for having multiple terminals in one session and keeping your session alive when you sign out), `find` (find files), `grep` (filter by regular expressions), `sed` (stream editor), `man` (manual), `ls` (list), `cd` (change directory), `pwd` (print working directory), `touch` (create file or update its last modified date), `mkdir` (make directory), `rm` (remove), `rmdir` (remove empty directory), `df` (disk free), `ncdu` (ncurses disk usage), `git` (version control system).

Some also prefer using `zsh` over `bash`.

6.1 Virtual (Python) Environments

If you want (need) to use virtual environments, consider still using the packages from the system-wide environment and not re-installing everything. E.g., create an environment with

```
python3 -m venv --system-site-packages /path/to/your/new/environment
```

or

```
virtualenv --system-site-packages /path/to/your/new/environment
```

and activate it with

```
. /path/to/your/new/environment/bin/activate
```

Then,

```
python3 -m pip install <whatever else you may need here>
```