CSC4005 Development Environment DIY Guidance (many Linux version)

Thanks for Peilin Li for his vital contribution.

- If you want to DIY a CSC4005 enviornment on Linux, you can try this guidance.
- This guidance utilize `yum` (a Linux package management tool native to CentOS but NOT Ubuntu), if you don't have a yum, you need to install it before getting started.
- If your Linux virtual environment doesn't support `yum` natively (e.g. WSL Ubuntu version), you may need to install Linux distributions where `yum` is natively supported (e.g. CentOS).
- To install CentOS on WSL, check out CentOS 7.9 on GitHub
- To check whether you have installed `yum`, just type `yum` in your terminal, if your terminal tells you that `command not found...` , you need to install it.

1. Install g++

```
sudo yum install gcc-c++ -y
```

2. Install mpich-3.2

MPICH is a high performance and widely portable implementation of the Message Passing Interface (MPI) standard.

Step 1. Install `mpich-3.2`

```
sudo yum install mpich-3.2.x86_64 mpich-3.2-devel.x86_64 -y
```

Step 2. Add `mpich-3.2` to PATH

Note: You might need to install `vim` before running the following command if you doesn't have `vim` installed, but please feel free to use any other text editor your are comfortable with, especially considering beginners may feel uncomfortable using `vim`. This note also applies whenever we use `vim` in the following part of this guidance.

```
export PATH=$PATH:/usr/lib64/mpich-3.2/bin/
```

Step 3. Save the file and quit

If you are using `vim`, press `Esc` to quit editing mode, then type `:wq` to save file and quit `vim`.

Step 4. Reboot your system

Step 5. Test it out

```
mpic++
mpirun
```

If no `command not found...` appears, then you have succeeded.

3. Install OpenGL GUI library

Step 1. Install OpenGL GUI Library

```
sudo yum install mesa* -y
```

Step 2. Check it out

```
sudo yum list *glut* -y
```

If your output looks something like this:

```
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.neusoft.edu.cn
 * extras: ftp.sjtu.edu.cn
 * updates: mirrors.nju.edu.cn
Installed Packages
freeglut.x86_64                    3.0.0-8.el7                @base
Available Packages
freeglut.i686                      3.0.0-8.el7                base
freeglut-devel.i686                3.0.0-8.el7                base
freeglut-devel.x86_64              3.0.0-8.el7                base
```

, then you have succeeded.

4. Pthread and OpenMP may not need installing

5. Install Nvidia driver and CUDA Toolkit

Note: If your computer doesn't have Nvidia GPU (use the device manager on Windows to check it out), you don't need to do the following.

5.1. Install Nvidia driver

Go to https://www.nvidia.com/dow nload/index.aspx , choose your GPU model and platform. Then follow the instruction.

Choose `Game Ready Driver` at the `Download Type` option.

5.2 Install CUDA Toolkit

The NVIDIA® CUDA® Toolkit provides a development environment for creating high performance GPU-accelerated applications. With the CUDA Toolkit, you can develop, optimize, and deploy your applications on GPU-accelerated embedded systems, desktop workstations, enterprise data centers, cloud-based platforms and HPC supercomputers. The toolkit includes GPU-accelerated libraries, debugging and optimization tools, a C/C++ compiler, and a runtime library to deploy your application.

Using built-in capabilities for distributing computations across multi-GPU configurations, scientists and researchers can develop applications that scale from single GPU workstations to cloud installations with thousands of GPUs.

Note: You may need to upgrade your WSL 1 to WSL 2 before the following. Check out this tutorial on how to upgrade it.

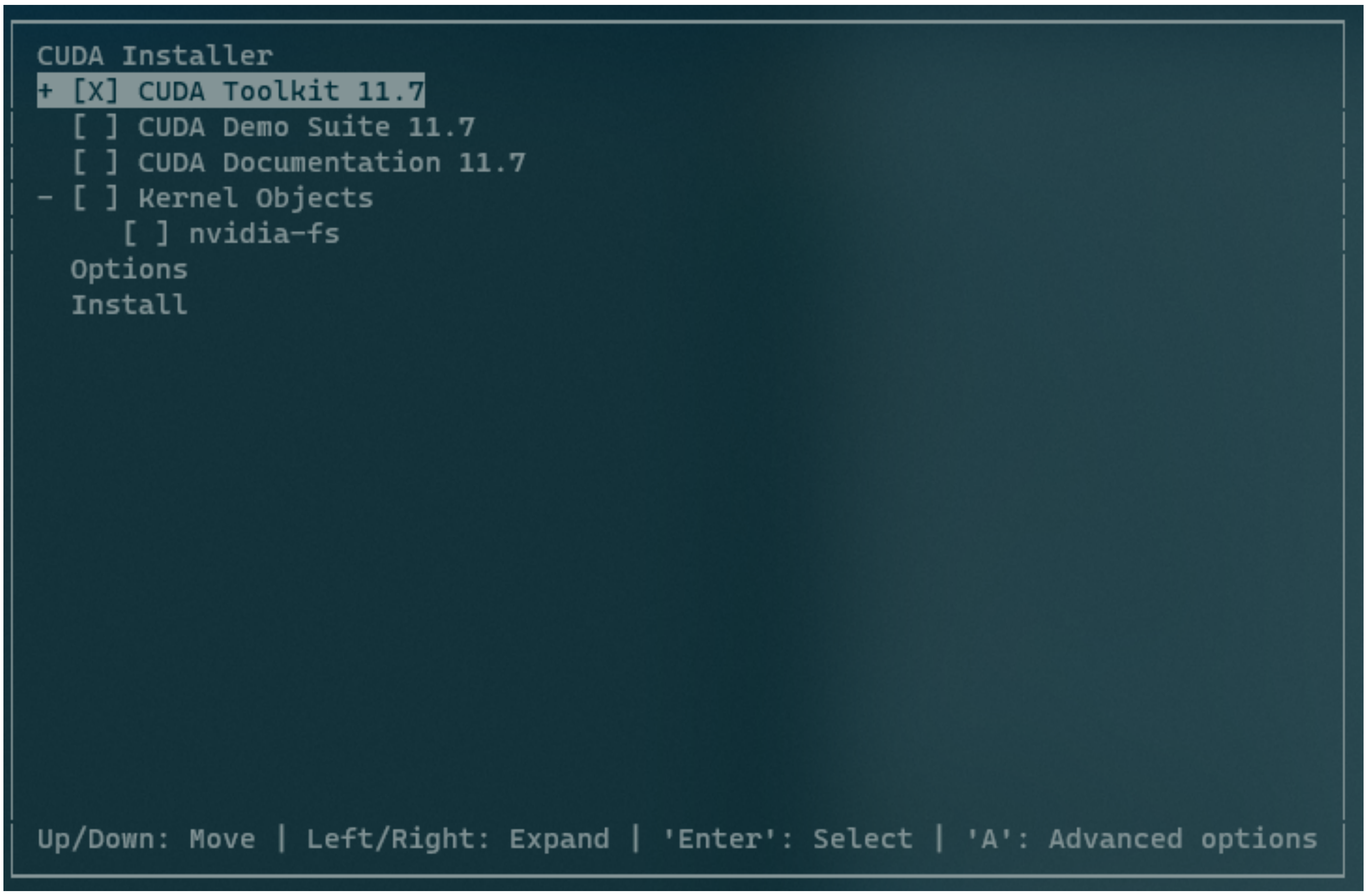
Step 1. Install CUDA Toolkit

Note: You may need to install `wget` before starting the following command. If you need to, just Google it.

Go to https://developer.nvidia.com/cuda-downloads , choose your Platform. Then follow the instruction. For `Installer Type`, we recommend `runfile (local)`. It is better to choose local installer (size is very large, but it is convenient).

Note: this process may take a while (e.g. 20 minutes), so make sure your Internet connection doesn't break during this time.

If you encounter a window like this,



```
CUDA Installer
+ [X] CUDA Toolkit 11.7
    [ ] CUDA Demo Suite 11.7
    [ ] CUDA Documentation 11.7
- [ ] Kernel Objects
       [ ] nvidia-fs
  Options
  Install

Up/Down: Move | Left/Right: Expand | 'Enter': Select | 'A': Advanced options
```

just choose the CUDA Toolkit 11.7 only.

Step 2. Add nvcc to PATH

`nvcc` is the CUDA compiler driver

```
sudo vim /etc/profile
```

Step 3. Add these to the end of file

```
export CUDA_HOME=/usr/local/cuda
export PATH=${CUDA_HOME}/bin:${PATH}
export LD_LIBRARY_PATH=${CUDA_HOME}/lib64:$LD_LIBRARY_PATH
```

Step 4.

```
source /etc/profile
```

Step 5. Reboot your system

Step 6. Test it out

```
nvcc
nvidia-smi
```

If no `command not found...` appears, then you have succeeded.

All done!

If you have any questions, please email bokaixu@link.cuhk.edu.cn