

## Installation guide for Robosuite on Windows 10 using Linux configured on Virtual Machine.

Guide can be used to install Robosuite, Mujoco on regular Linux as well

### Virtual Machine on Windows 10 (if you run ubuntu skip this part)

- 1) Install virtual machine by following this great tutorial. (Oracle VM VirtualBox)  
<https://www.youtube.com/watch?v=x5MhydiJWmc&t=1007s>
- 2) After creating the new Linux environment, we open Terminal and all of the below commands are inputted in the Terminal (in the VM, we do not use Windows anymore)
- 3) `sudo apt update`
- 4) `sudo apt upgrade`
- 5) `sudo apt install xrdp`
- 6) `sudo systemctl enable --now xrdp`
- 7) `sudo ufw allow from any to any port 3389 proto tcp`

### Mujoco

- 8) Download *mujoco200 linux* from the official website <https://www.roboti.us/index.html>

Create *.mujoco* folder in the following way

- 9) `mkdir ~/.mujoco`
- 10) `cd ~/.mujoco`

The unzipped *mujoco200\_linux* folder place in home directory and change name to *mujoco200*.

The created folder *.mujoco* can be seen by clicking **Ctrl+H** when in home directory. We place the downloaded folder *mujoco200* into the *.mujoco* folder.

- 11) Download the *mjkey.txt* license key and place it in the *home/.mujoco/mujoco200* and in the *home/.mujoco/mujoco200/bin* folder

To validate that mujoco try to run the following:

- 12) On the path */.mujoco/mujoco200/bin* enter in terminal `./simulate ../model/humanoid.xml`  
 If a window with 3D model appears it means we are all good to go and mujoco was installed successfully

### PyCharm

- 13) In the home directory: `sudo snap install pycharm-community --classic`

### Anaconda

We will create folder *Linux\_setup* where the anaconda will be installed (name of folder is arbitrary)

- 14) `cd`
- 15) `sudo apt-get update`
- 16) `cd`

- 17) `source ~/.bashrc`
- 18) `mkdir Linux_setup`
- 19) `cd Linux_setup`

Enter anaconda website <https://www.anaconda.com/products/individual#linux> and find the Linux installer which you then must download. Once download is completed place the installed file in the Linux\_setup folder we created before.

- 20) We check if installer is in the folder by typing in Terminal: `dir` and then execute
- 21) `bash Anaconda3-2021.05-Linux-x86_64.sh`

The anaconda should be installed now, we then create new environment called “mujoco-gym” (you can name it however you wish)

- 22) `conda config --set auto_activate_bash false`

For the above changes to take place **close and reopen** the terminal

- 23) `cd Linux_setup`

Create new environment using Anaconda in the Linux\_setup folder

- 24) `conda create -n mujoco-gym python3=3.8`

Check the list of existing environments and make sure our newly create environment is there and then activate it

- 25) `conda env list`
- 26) `conda activate mujoco-gym`

### Mujoco-py

Now onto the most difficult part which is installation of mujoco-py. What I recommend is to take 5 min brake and take few deep breathes before proceeding as what is about to come might ruin your day 😊

While still in Linux\_setup directory

- 27) `cd`
- 28) `sudo apt install git`
- 29) `cd`

Get back to home directory

- 30) `conda deactivate`
- 31) `sudo apt-get update`
- 32) `sudo apt-get install patchelf`
- 33) `sudo apt-get install python3 python3-dev python3-dev build-essential libssl-dev libffi-dev libxml2-dev libxslt1-dev zlib1g-dev libglew1.5 libglew-dev python3-pip`
- 34) `sudo apt-get install python3-pip`
- 35) `git clone https://github.com/openai/mujoco-py`
- 36) `cd mujoco-py`
- 37) `pip3 install -e . --no-cache`
- 38) `cd`

You will probably get a bunch of errors. Go to your home directory using by simply clicking on the needed folders and open `.bashrc` and add the following into the bottom of it

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/daniel/.mujoco/mujoco200/bin
```

Make sure to change *daniel* to your system name. In my example I added it in line 119 see how it should look like in the Appendix.

### **VERY IMPORTANT.**

After changing `bashrc` make sure to save it and restart your Ubuntu.

- 39) Open the terminal and type the following:
- 40) `sudo apt install libosmesa6-dev libgl1-mesa-glx libglfw3`
- 41) `sudo ln -s /usr/lib/x86_64-linux-gnu/libGL.so.1 /usr/lib/x86_64-linux-gnu/libGL.so`
- 42) `pip3 install -U 'mujoco-py<2.1,>=2.0'`

You should have no errors after executing the `pip3` command. If you managed to execute the command without errors – congratulation!

### **Robosuite**

- 43) Go to official robosuite GitHub repo and obtain their GitHub link
- 44) in the home directory execute the following:  
`git clone https://github.com/StanfordVL/robosuite.git`
- 45) `cd robosuite`
- 46) `pip3 install -r requirements.txt`
- 47) `pip3 install -r requirements-extra.txt`

For a test if robosuite works well type the following.

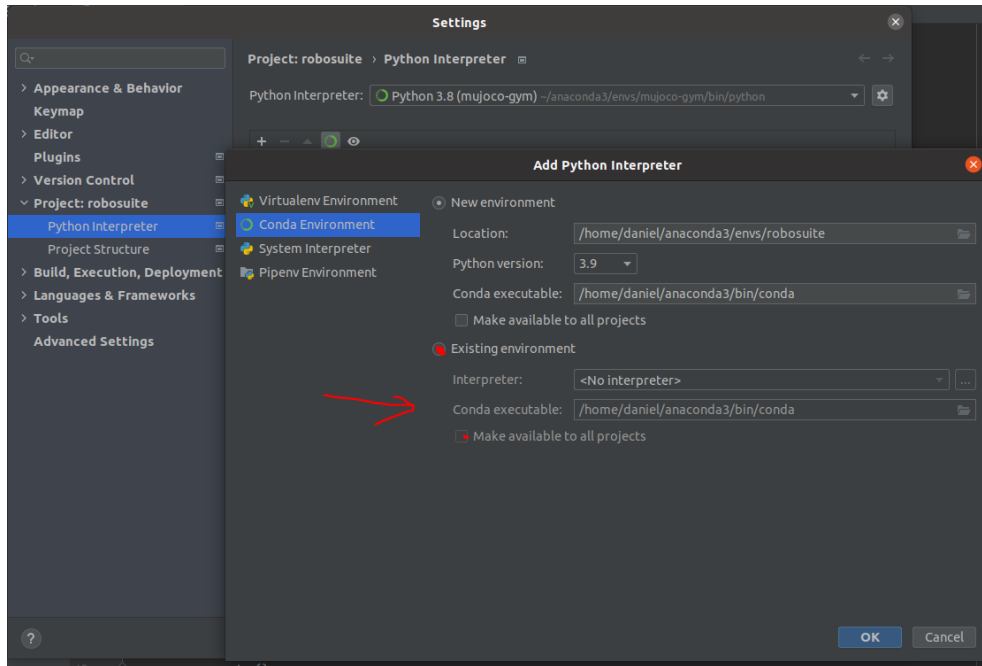
- 48) `python3 robosuite/demos/demo_random_action.py`

### **Setting up Pycharm for usage with anaconda, Mujoco and robosuite**

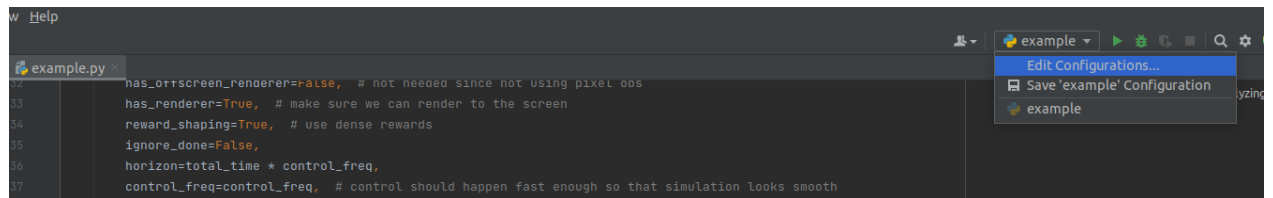
- 49) Open Pycharm and click “*open*”
- 50) Navigate to the robosuite folder and confirm your folder choice where new project will be created

We now set up the interpreter to be the anaconda environment we created. While in Pycharm and in robosuite folder:

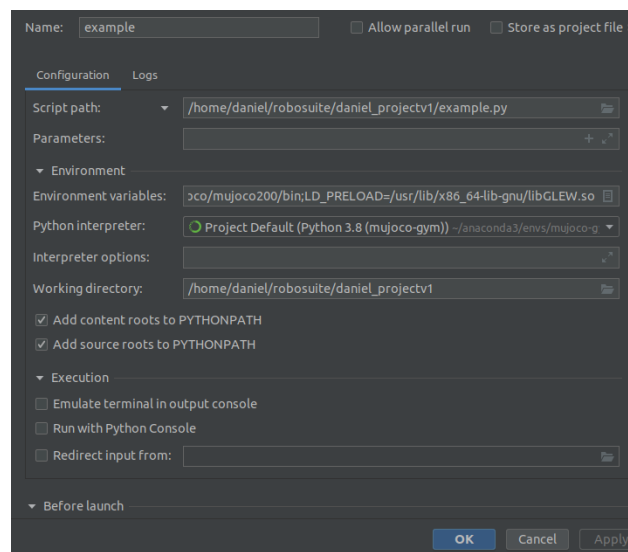
- 51) Go to *file->settings->project robosuite->python interpreter* then click on the setting icon (gear icon) of the new window and *click add*. Select *conda environment* and existing environment. Give it a second and click *ok*. Then the environment should switch to *mujoco-gym* i.e. our custom made environment



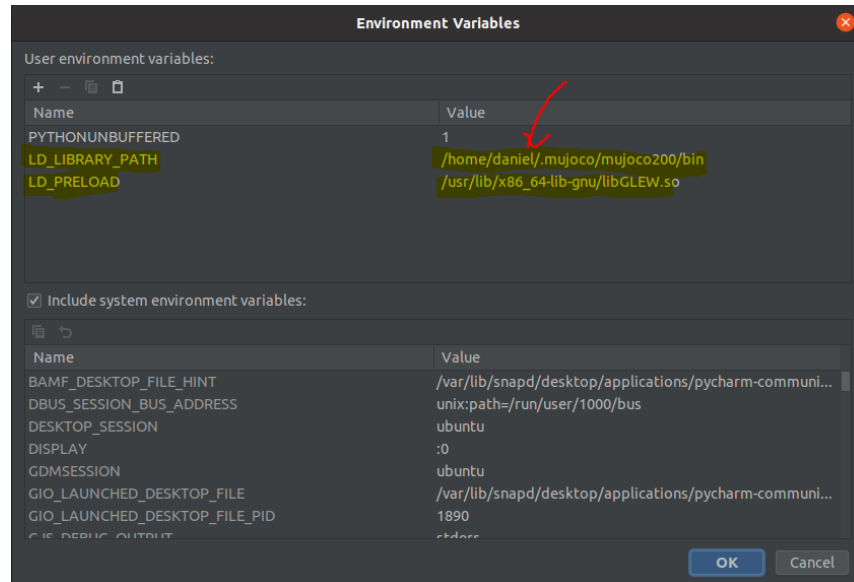
52) Last small changes. Initially you will only have *Edit configuration* option here so click on it



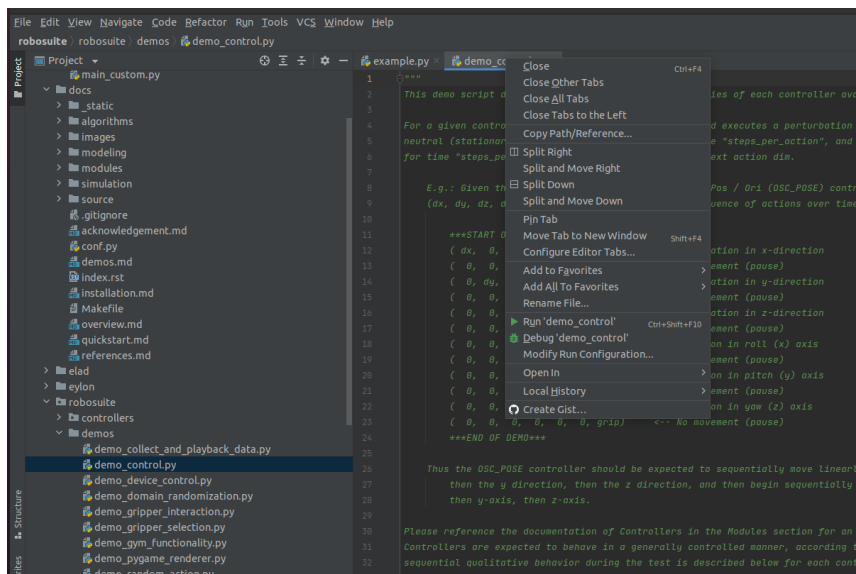
53) In the lower left corner, you will see option to make new template click on it, select python and then in the *Environment variables* we need to add 2 lines (see two pictures below)



54) The lines we add are (make sure to change *daniel* to your system name)



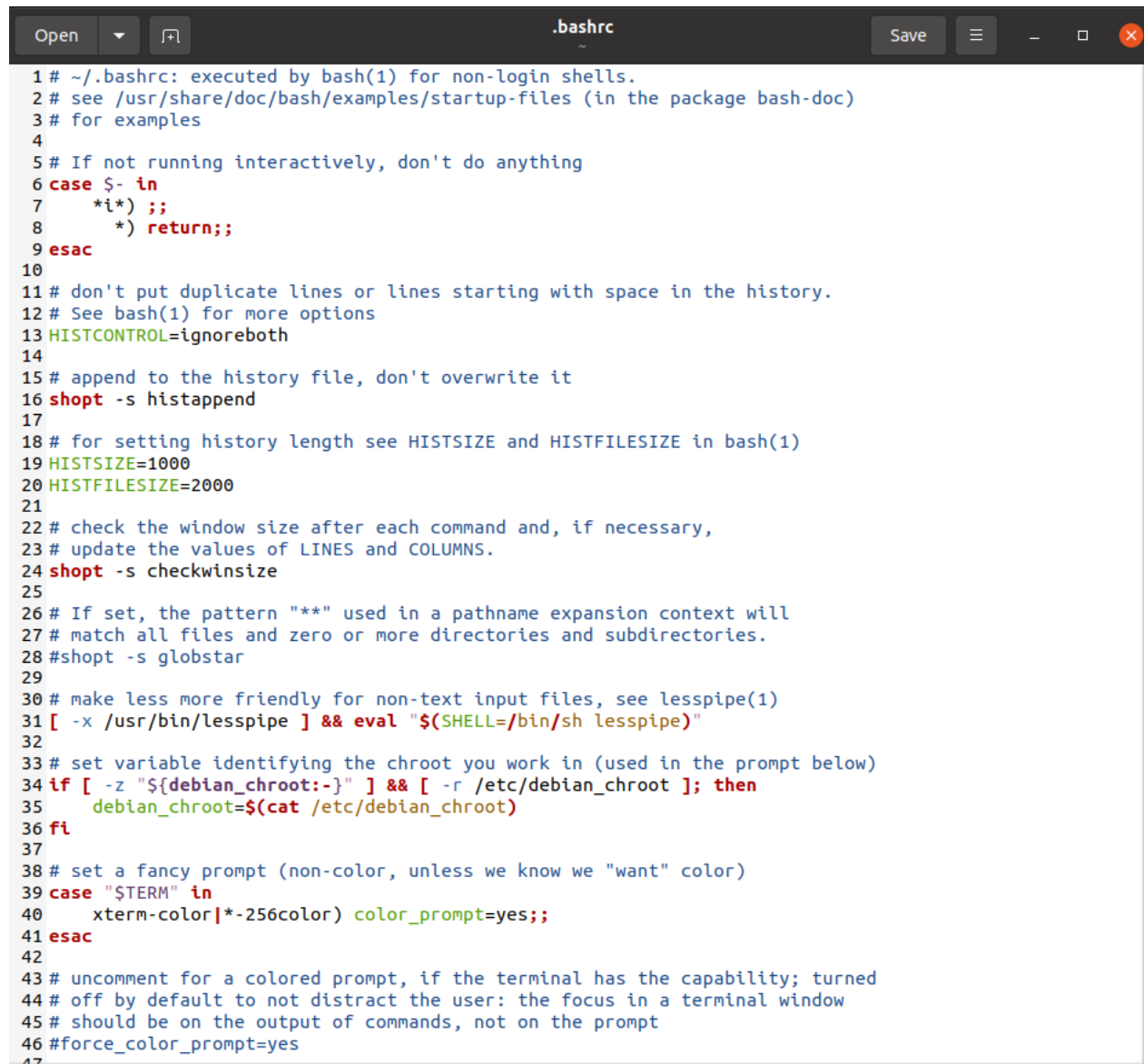
55) Right click on the demo example and click run



It might show you errors and might take some time but be patient as it can take more time to run at first time. As long as program is compiling you should be good 😊

## Appendix (bashrc)

I attach it here so you can see where I added the path to the bashrc.

A screenshot of a text editor window titled ".bashrc". The window has a dark theme with a light-colored text area. The text is a bashrc configuration file with various comments and commands. The editor has a menu bar with "Open", "Save", and other icons. The text is as follows:

```
1 # ~/.bashrc: executed by bash(1) for non-login shells.
2 # see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
3 # for examples
4
5 # If not running interactively, don't do anything
6 case $- in
7     *) ;;
8     *) return;;
9 esac
10
11 # don't put duplicate lines or lines starting with space in the history.
12 # See bash(1) for more options
13 HISTCONTROL=ignoreboth
14
15 # append to the history file, don't overwrite it
16 shopt -s histappend
17
18 # for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
19 HISTSIZE=1000
20 HISTFILESIZE=2000
21
22 # check the window size after each command and, if necessary,
23 # update the values of LINES and COLUMNS.
24 shopt -s checkwinsize
25
26 # If set, the pattern "*" used in a pathname expansion context will
27 # match all files and zero or more directories and subdirectories.
28 shopt -s globstar
29
30 # make less more friendly for non-text input files, see lesspipe(1)
31 [ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"
32
33 # set variable identifying the chroot you work in (used in the prompt below)
34 if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
35     debian_chroot=$(cat /etc/debian_chroot)
36 fi
37
38 # set a fancy prompt (non-color, unless we know we "want" color)
39 case "$TERM" in
40     xterm-color|*-256color) color_prompt=yes;;
41 esac
42
43 # uncomment for a colored prompt, if the terminal has the capability; turned
44 # off by default to not distract the user: the focus in a terminal window
45 # should be on the output of commands, not on the prompt
46 #force_color_prompt=yes
47
```

```

45 # should be on the output of commands, not on the prompt
46 #force_color_prompt=yes
47
48 if [ -n "$force_color_prompt" ]; then
49     if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
50         # We have color support; assume it's compliant with Ecma-48
51         # (ISO/IEC-6429). (Lack of such support is extremely rare, and such
52         # a case would tend to support setf rather than setaf.)
53         color_prompt=yes
54     else
55         color_prompt=
56     fi
57 fi
58
59 if [ "$color_prompt" = yes ]; then
60     PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\-\
61     [\033[00m\]\$ '
62 else
63     PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
64 fi
65 unset color_prompt force_color_prompt
66
67 # If this is an xterm set the title to user@host:dir
68 case "$TERM" in
69     xterm*|rxvt*)
70         PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
71         ;;
72     *)
73         ;;
74 esac
75
76 # enable color support of ls and also add handy aliases
77 if [ -x /usr/bin/dircolors ]; then
78     test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
79     alias ls='ls --color=auto'
80     #alias dir='dir --color=auto'
81     #alias vdir='vdir --color=auto'
82
83     alias grep='grep --color=auto'
84     alias fgrep='fgrep --color=auto'
85     alias egrep='egrep --color=auto'
86 fi
87
88 # colored GCC warnings and errors
89 export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'
90
91 # some more ls aliases

```

```

90 # some more ls aliases
91 alias ll='ls -lF'
92 alias la='ls -A'
93 alias l='ls -CF'
94
95 # Add an "alert" alias for long running commands. Use like so:
96 # sleep 10; alert
97 alias alert='notify-send --urgency=low -i "${[ $? = 0 ]}&& echo terminal || echo error)" "${history|
tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[\&|]\s*alert$//'\`}'"
98
99 # Alias definitions.
100 # You may want to put all your additions into a separate file like
101 # ~/.bash_aliases, instead of adding them here directly.
102 # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103
104 if [ -f ~/.bash_aliases ]; then
105     . ~/.bash_aliases
106 fi
107
108 # enable programmable completion features (you don't need to enable
109 # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
110 # sources /etc/bash.bashrc).
111 if ! shopt -oq posix; then
112     if [ -f /usr/share/bash-completion/bash_completion ]; then
113         . /usr/share/bash-completion/bash_completion
114     elif [ -f /etc/bash_completion ]; then
115         . /etc/bash_completion
116     fi
117 fi
118
119 export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/daniel/.mujoco/mujoco200/bin
120
121
122 # >>> conda initialize >>>
123 # !! Contents within this block are managed by 'conda init' !!
124 __conda_setup="$('/home/daniel/anaconda3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
125 if [ $? -eq 0 ]; then
126     eval "$__conda_setup"
127 else
128     if [ -f "/home/daniel/anaconda3/etc/profile.d/conda.sh" ]; then
129         . "/home/daniel/anaconda3/etc/profile.d/conda.sh"
130     else
131         export PATH="/home/daniel/anaconda3/bin:$PATH"
132     fi
133 fi
134 unset __conda_setup
135 # <<< conda initialize <<<
136

```