|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| \\dsp-disk.eng.tau.ac.il\Documents\eeproj\Forms & Important\פרויקטים\הנחיות\Guidelines\Students\ENG_LOGO-01.png | | | **\\dsp-disk.eng.tau.ac.il\Documents\eeproj\Forms & Important\פרויקטים\הנחיות\Guidelines\Students\TAU_EngineeringENG.png** | |
| מערכת חישה לאלגוריתמיקה מתקדמת | | | |
| פרויקט מס' 22-1-1-2667  חוברת תדריכי התקנה עבור חיישני התשתית | | | |
| מבצעים: | | | |
|  | טל אריאלי | 318303268 | |
|  | שחר הנובר | 208685875 | |
| מנחה: | | | |
|  | שמחה ליבוביץ | אוניברסיטת ת"א | |
| מקום ביצוע הפרויקט: אוניברסיטת ת"א | | | |

**מדריך התקנת מצלמת intel realsense d435, הורדת intsl sdk וכל ספריות librealsense וpyrealsense2**

בעבור חיבור מצלמת intel realsense d435 נעזרנו במדריך הבא : <https://github.com/datasith/Ai_Demos_RPi/wiki/Raspberry-Pi-4-and-Intel-RealSense-D435>

דרישות להתקנה מראש:

* תחילה, ביצוע עדכון, שדרוג והתקנה של כלים שיהיו רלוונטים לפעולה התקינה של המצלמה:

sudo apt-get update && sudo apt-get dist-upgrade

sudo apt-get install automake libtool vim cmake libusb-1.0-0-dev libx11-dev xorg-dev libglu1-mesa-dev

* ישנה אפשרות להרחבת מערכת הקבצים על ידי שימוש בפקודה הבאה:

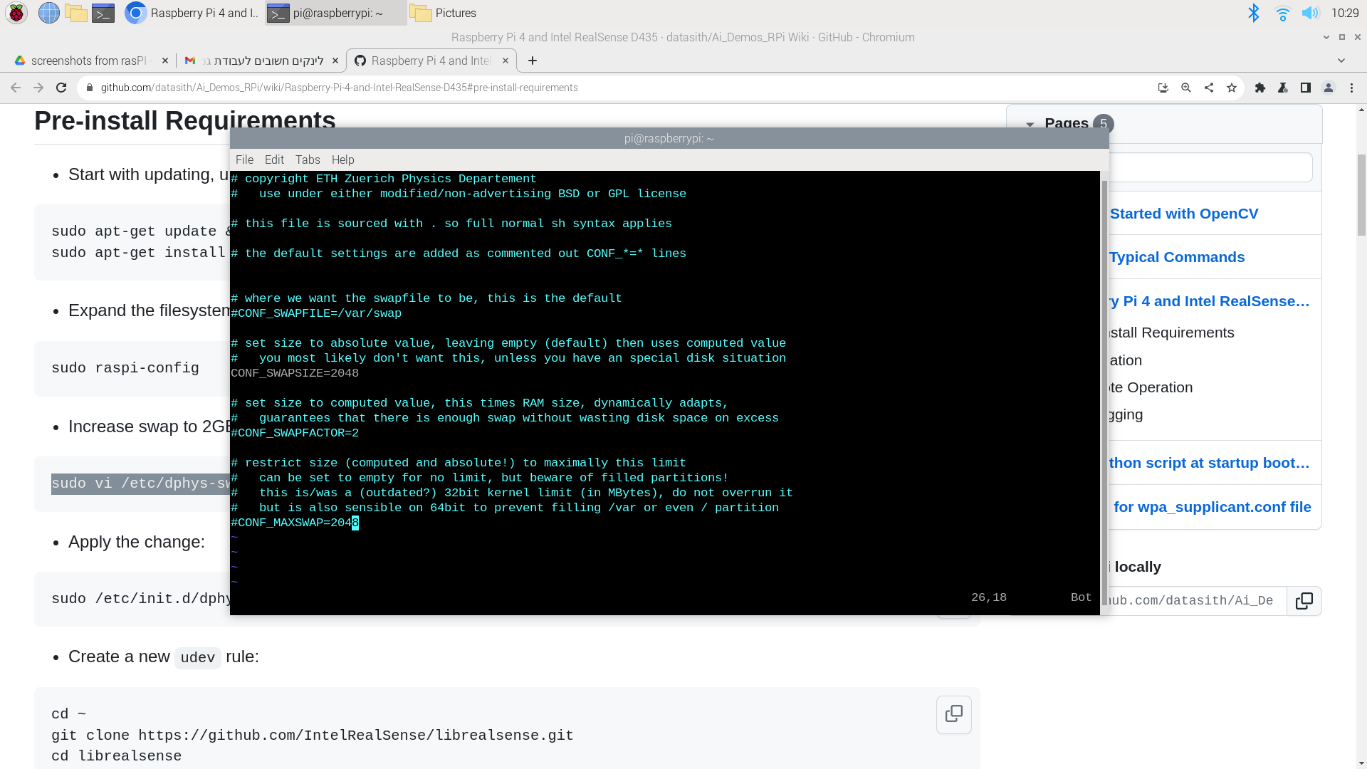
sudo raspi-config

בשלב זה יפתח התפריט הבא:

בחירת האפשרות 'אפשרויות מתקדמות' (Advanced options) ובחירת אתחול מחדש.

* הגדלת את ה-swap ל-2GB על ידי שינוי הקובץ מטה ל-CONF\_SWAPSIZE=2048:

sudo vi /etc/dphys-swapfile

קובץ vi הוא קובץ לעריכת טקסט ובו צריך לשנות את CONF\_SWAPSIZE ל 2048.

על מנת לצאת מהחלון של עריכת הטקסט, נכתוב ":qa" ונלחץ Enter.

* יישום השינויים באמצעות הפקודה:

sudo /etc/init.d/dphys-swapfile restart swapon -s

* יצירת udev Rule חדש:

cd ~

git clone https://github.com/IntelRealSense/librealsense.git

cd librealsense

sudo cp config/99-realsense-libusb.rules /etc/udev/rules.d/

* יישום השינויים (נדרשת שהריצה תהיה על ידי root):

sudo su

udevadm control --reload-rules && udevadm trigger

exit

* שינוי הנתיב על ידי הוספת השורה הבאה לקובץ .bashrc:

export LD\_LIBRARY\_PATH=/usr/local/lib:$LD\_LIBRARY\_PATH

* לבסוף, יישום השינויים על ידי הפקודה:

source ~/.bashrc

התקנה :

* התקנת protobuf - המנגנון הניטרלי לשפה, ניטרלי לפלטפורמה, וניתן להרחבה של גוגל ליצירת נתונים מובנים:

cd ~

git clone --depth=1 -b v3.10.0 https://github.com/google/protobuf.git

cd protobuf

./autogen.sh

./configure

make -j1

sudo make install

cd python

export LD\_LIBRARY\_PATH=../src/.libs

python3 setup.py build --cpp\_implementation

python3 setup.py test --cpp\_implementation

sudo python3 setup.py install --cpp\_implementation

export PROTOCOL\_BUFFERS\_PYTHON\_IMPLEMENTATION=cpp

export PROTOCOL\_BUFFERS\_PYTHON\_IMPLEMENTATION\_VERSION=3

sudo ldconfig

protoc --version

A screenshot of a computer

Description automatically generated

* התקנת הספריה המקבילה libtbb-dev עבור C++:

cd ~

wget https://github.com/PINTO0309/TBBonARMv7/raw/master/libtbb-dev\_2018U2\_armhf.deb

sudo dpkg -i ~/libtbb-dev\_2018U2\_armhf.deb

sudo ldconfig

rm libtbb-dev\_2018U2\_armhf.deb

* התקנת RealSense SDK librealsense:

cd ~/librealsense

mkdir build && cd build

cmake .. -DBUILD\_EXAMPLES=true -DCMAKE\_BUILD\_TYPE=Release -DFORCE\_LIBUVC=true

make -j1

sudo make install

* התקנת RealSense SDK pyrealsense2 Python bindings עבור librealsense:

cd ~/librealsense/build

cmake .. -DBUILD\_PYTHON\_BINDINGS=bool:true -DPYTHON\_EXECUTABLE=$(which python3)

make -j1

sudo make install

* שינוי הנתיב על ידי הוספת השורה הבאה לקובץ .bashrc:

export PYTHONPATH=$PYTHONPATH:/usr/local/lib

* יישום השינויים:

source ~/.bashrc

* הורדת OpenGL:

sudo apt-get install python-opengl

sudo -H pip3 install pyopengl

sudo -H pip3 install pyopengl\_accelerate==3.1.3rc1

* שינוי הגדרות הPI : אפשור OpenGL:

sudo raspi-config

"7. Advanced Options" – "A8 GL Driver" – "G2 GL (Fake KMS)"

דגשים עבור ההתקנה:

* הוספת הפקודה הבאה:

sudo apt-get install libssl-dev

לאחר מכן הוספת הפקודה:

cmake .. -DBUILD\_EXAMPLES=true -DCMAKE\_BUILD\_TYPE=Release -DFORCE\_LIBUVC=true

פקודה זאת נמצאת בInstall RealSense SDK librealsense:.

* כאשר מתקינים את realsense sdk pyrealsense 2, מוסיפים את הפקודות הבאות:

sudo apt-get install gcc-7 g++-7

sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 60 --slave /usr/bin/g++ g++ /usr/bin/g++-7

sudo update-alternatives --set gcc "/usr/bin/gcc-7"

ההוספה נעשתה לפני ההתקנה.

פתרון הבעיה מופיע בכתובת: [Raspberry pi 4 Install Realsense SDK make -j1 issues · Issue #10033 · IntelRealSense/librealsense · GitHub](https://github.com/IntelRealSense/librealsense/issues/10033)

* בחלק של opengl:   
  בוצע רק החלק הבא:

sudo apt-get install -y python3-opengl

* לאחר ההתקנה נתקלנו בשגיאה no module named pyrealsense2 שמצביעה על בעיה בהורדת ספריית הpyrealsense2: קבצי ה.so הורדו למקומות מסויימים במחשב שלא מקושרים עם הפייתון שלו ולכן היינו צריכים לערוך את קובץ ה .bashrc בפקודה הבאה:

nano ~/.bashrc

לאחר מכן הוספנו את השורות הבאות בסוף קובץ הbashrc:

export LD\_LIBRARY\_PATH=/usr/local/OFF:$LD\_LIBRARY\_PATH

export PYTHONPATH=$PYTHONPATH:/usr/local/OFF

export PYTHONPATH=$PYTHONPATH:/home/serveradmin/librealsense/build/Release

(בשורות אלו מופיע הנתיב לתיקיות החשובות של הספרייה), לאחר מכן שמרנו שינויים באמצעות פקודת :

source ~/.bashrc.

(מצאנו פתרון לזה בסוף הדיון בgithub הבא: <https://github.com/IntelRealSense/librealsense/issues/3062> )

* כעת נתקלנו בשגיאה הבאה:

ImportError: /usr/local/lib/librealsense2.so.2.28: undefined symbol: \_\_atomic\_fetch\_add\_8

שגיאה זו התרחשה מכיוון שהייתה בעיה בקובץ הcmake ולכן על מנת לפתור את בעיה זו, מחקנו את ספריית librealsense ולפי מדריך ההתקנה ביצענו

Install libtbb-dev parallelism library for C++

Install RealSense SDK librealsense

לאחר מכן, שינינו את הקובץ cmake שנמצא ב

~/librealsense/wrappers/python/CMakeLists.txt

לפי הפתרון בדיון הgithub הבא:

[Import Error SDK 2.24.0 Python3.7 on Raspberry Pi 4 Raspbian 10 Buster · Issue #4375 · IntelRealSense/librealsense · GitHub](https://github.com/IntelRealSense/librealsense/issues/4375)

לאחר כן, המשך ההתקנה בוצעה על פי המדריך המצורף.

כעת ניתן להריץ סקריפטים מה-shell ולייבא את pyrealsense2 ללא בעיות.

**התקנת ספריות פייתון אחרות שרלוונטיות לנו:**

* הורדת numpy ע"י פקודת:

sudo apt install python3 python3-numpy

* הורדת opencv ע"י פקודת:

pip3 install opencv-python

* עדכון הספריות ע"י פקודת :

pip3 install --upgrade numpy opencv-python

**התקנת סנסור RPLIDAR SLAMTEC A2**

התקנת ספריות sdk של SLAMTEC RPLIDAR :

* <https://github.com/Slamtec/rplidar_sdk> כפי שכתוב במדריך ביצענו את הפקודות הבאות :

sudo apt-get update

sudo apt-get install build-essential

git clone <https://github.com/slamtec/rplidar_sdk.git>

cd rplidar\_sdk

make DEBUG=1

כעת ניתן לראות את הקבצים המקומפלים ב:

output/Linux/Release or output/Linux/Debug directory

* בשלב זה נצטרך privileges to access the serial port ולכן נבצע את הפקודות הבאות:

sudo nano /etc/udev/rules.d/99-usb-serial.rules

תוך הוספת השורה הבאה:

KERNEL=="ttyUSB\*", MODE="0666"

נשמור את השינויים ונצא מהקובץ.

* נוסיף Dialout Group Membership ע"י הפקודה:

sudo usermod -aG dialout $USER

ואכן כעת אם נבצע groups כפקודה בshell, נראה שאנו חברים ב dialout.

* נבצע reboot
* ניתן לראות שאפשרנו Correct Serial Port אם נבצע את פקודת:

sudo chmod 0666 /dev/ttyUSB0

ls /dev/ttyUSB\*

נקבל : /dev/ttyUSB0

* כעת ניתן לראות שlidar עובד ע"י הפקודות הבאות:

נעבור לספריית rplidar\_sdk:

cd rplidar\_sdk

* נריץ את הטסט של הסנסור שהורדנו בספריית ההתקנה שלו ע"י הפקודה:

./output/Linux/Release/ultra\_simple --channel --serial /dev/ttyUSB0 115200

ניתן לראות שהחיישן מסתובב וקיבלו את הoutput הבא:

תמונה שמכילה טקסט, צילום מסך, תוכנה, תכונות מולטימדיה

התיאור נוצר באופן אוטומטי

כאשר:

* + Theta: הזווית שבה הLIDAR ביצע את המדידה.
  + Dist: המרחק שנמדד על ידי הLIDAR בזווית המדידה.
  + Q: איכות המדידה
  + Version: מציינת את הגרסה של תוכנית איסוף הנתונים של ה ultra-simple LIDAR.
  + SLAMTEC LIDAR S/N: המספר הסידורי של חיישן הLIDAR.
  + Firmware Ver: גרסת החומרה ששעובדת כעת בLIDAR/
  + SLAMTEC Lidar health status: מדד התקינות של החיישן. עבור ערך "0" החיישן תקין.
* בשלב זה התקנו Wrapper ספריית פייתון שבאמצעותה אנו מתקשרים בין החיישן ובין הRaspberry Pi בפייתון.

pip install module-wrapper

* כעת נתקין סביבת עבודה python לlidar:

ראשית, נתקין סביבה וירטואלית לחבילות הרלוונטיות שנקראת: lidarvenv נעשה זאת ע"י הפקודות הבאות:

mkdir lidarvenv

cd lidarvenv

python3 -m venv venv

**בדיקות:**

נפעיל את הסביבה הוירטואלית ע"י הפקודה:

source venv/bin/activate

בתוך הסביבה הוירטואלית נוריד את החבילות ע"י הפקודות הבאות: (מתבצע בשימוש הראשוני בלבד)

pip install numpy pygame

pip install rplidar

pip3 install rplidar-roboticia

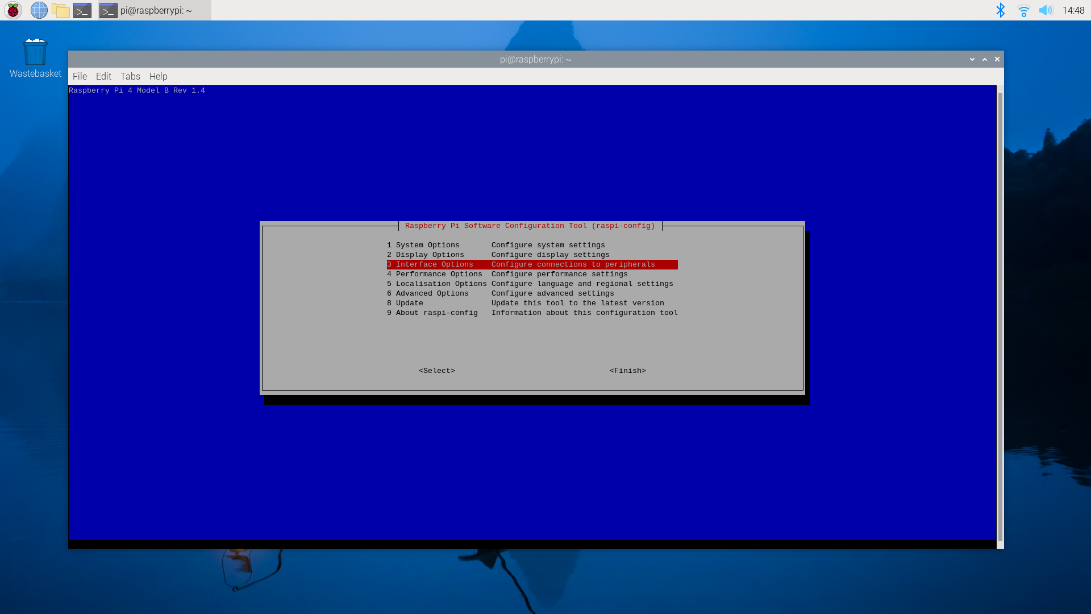
בחרנו את מודל פייתון זה מ Roboticiaמכיוון שמצאנו כי הממשק שלו עם LIDAR שלנו טוב.

ניתן לראות פירוט של המודל בגיט הבא: <https://github.com/Roboticia/RPLidar>

**התקנה ושימוש בסיסי במסך הOLED:**

* פתיחת טרמינל
* הזנת הפקודה :

sudo raspi-config

* נפתח תפריט ובו מאפשרים תקשורת I2C: **הוספת צילום מסך**
  + תפריט ראשון: בחירת האפשרות "configure connections to peripher" עבור "Interface Option":
  + A computer screen shot of a computer screen

    Description automatically generatedתפריט שני: בחירת האפשרות "I2C":

הערה: הממשק ישאל:

“Would you like the ARM I2C interface to be enabled?”

אחרי האישור יופי:

“ARM I2C interface is enabled”

* הספריות הבאות אמורות להיות מותקנות אך אם הן לא- הרצת הפקודות הבאות:

sudo apt install -y python3-dev

sudo apt install -y python3-smbus i2c-tools

sudo apt install -y python3-pil

sudo apt install -y python3-pip

sudo apt install -y python3-setuptools

sudo apt install -y python3-rpi.gpio

sudo apt install -y python3-venv

* מציאת הכתובת של מודול התצוגה OLED: כשספריות תקשורת הI2C מותקנות, נשתמש בפקודה i2cdetect על מנת למצוא את המודול באפיק (Bus) הI2C .

הזנת הפקודה הבאה:

i2cdetect -y 1

A computer screen with a black screen

Description automatically generatedומתקבל:

קיבלנו שהמכשיר זוהה עם כתובת של "0x3c". זוהי כתובת ה-hex ברירת המחדל עבור סוג זה של מכשיר. במידה ונעשה שימוש בדגם Model B Rev 1 Pi, ניתן להקליד את הפקודה הבאה במקום:

i2cdetect -y 0

* יצירת תיקית פרויקט: כעת, כאשר החומרה פועלת, ניצור תיקיה שתכיל את כלל הפריטים התוכנתיים למסך.

נזין את הפקודות הבאות:

cd ~

mkdir oled-screen

cd oled-screen

* יצירת סביבת הרצה בpython: על מנת להתקין את ספריית luma.oled ניצור תחילה סביבה וירטואלית של Python. זה יאפשר להכיל את כל הספריות והתלות בספרייה אחת, תוך הגנה על התקנת הליבה של Python מפני התנגשויות.

כדי ליצור סביבה וירטואלית Python 3 בתיקיית משנה בשם "luma-env" נשתמש בפקודה הבאה:

python3 -m venv venv

פעולה זו יצרה מערכת Python 3 עצמאית. נפעיל אותה באמצעות הפקודה הבאה:

source venv/bin/activate

ניתן לשים לב ששם הסביבה הוירטואלית מופיע מול שורת הפקודה. בזמן ההפעלה, כל ספריות Python שמותקנות יותקנו בסביבה וירטואלית זו ולא יפריעו להתקנת Python המוגדרת כברירת מחדל.

הערה: יצרנו סביבת Python 3 ולכן כל שימוש ב-"Python" יפנה אוטומטית ל Python 3.

* התקנת ספריית Luma OLED – Python: על מנת להתקין את הספריה, נריץ את הפקודה הבאה:

python -m pip install --upgrade luma.oled

* הורדה ושימוש בדוגמאות בסיסיות של LUMA: על מנת לבדוק את המסך מומלץ לבדוק כמה מהסקריפטים הפשוטים (מפורט פה בהמשך איך ניגשים אליהם וישנה דוגמה)

נוודא שאנחנו נמצאים בתיקית הפרויקט

cd ~/oled-screen

לאחר מכן נוודא ש-git מותקן:

sudo apt install -y git

מאגר הדוגמאות:

git clone https://github.com/rm-hull/luma.examples.git

יתכן שיהיה צורך בהתקנות הבאות עבור עבודה תקינהשל הדוגמאות:

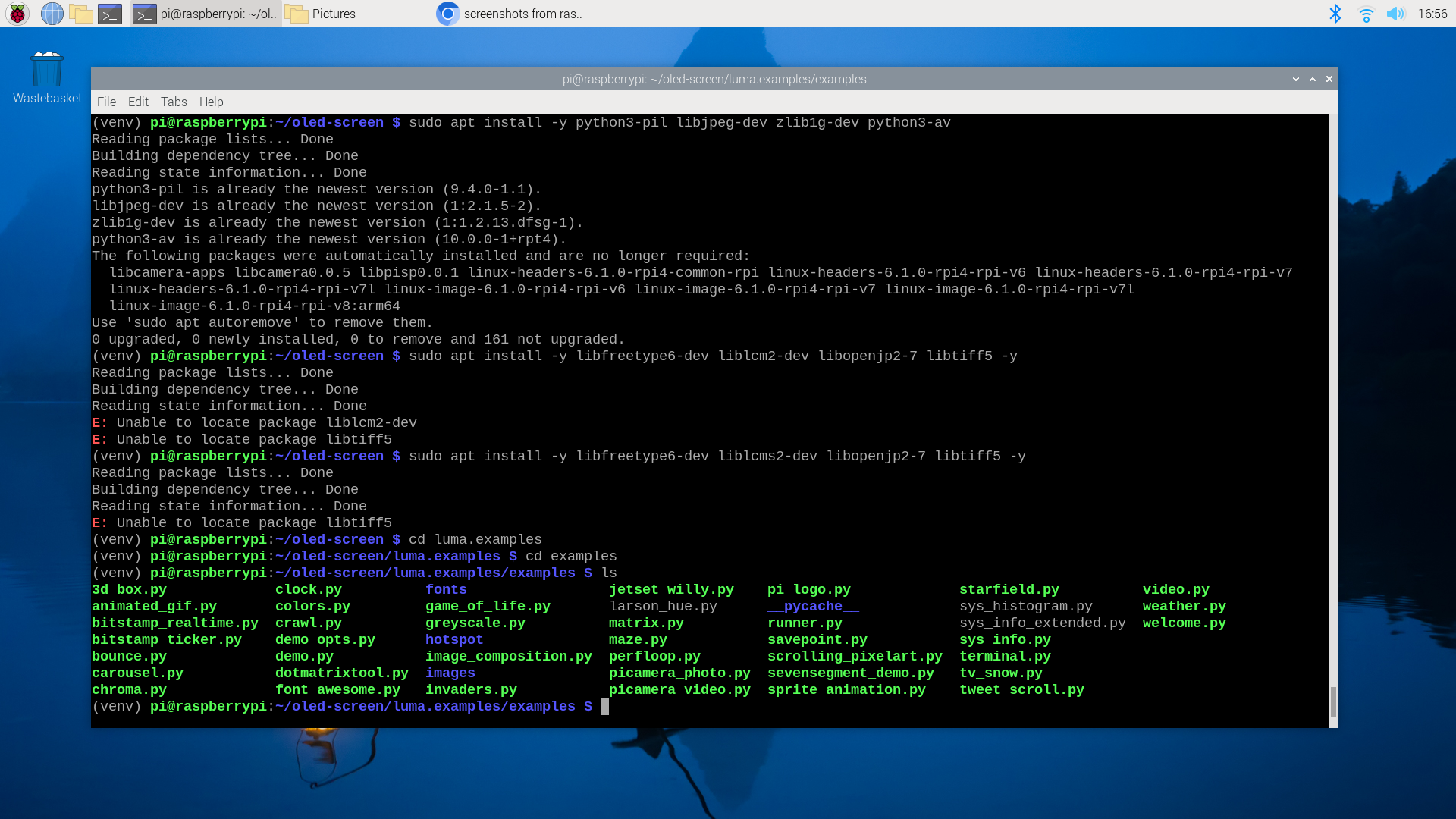
sudo apt install -y python3-pil libjpeg-dev zlib1g-dev python3-av

sudo apt install -y libfreetype6-dev liblcms2-dev libopenjp2-7 libtiff5 -y

עכשיו ניתן להריץ דוגמאות:

cd luma.examples  
cd examples

באמצעות ls נוכל לראות את מאגר הסקריפטים של luma:



* דוגמה להרצה של סקריפט ניסיון :

python 3d\_box.py

