

# **MOBLABPY DOCUMENTATION**

*By Lewis Chan Chun Ming, Lawrence Tse Pui Yu*

# 1 INTRODUCTION

MobLabPy is a software toolbox designed to emulate the working of a communication channel and enable the conducting of communication-related experiments in associated courses, i.e., ELEC 1200: A System View of Communications: from Signals to Packets and ELEC 4150: Error Control Coding and Information Theory. MobLabPy gives users the ability to do experiments without dedicated hardware and software license, but with their laptop (or desktop) computers and mobile phones anywhere at any time. This toolbox is built upon Python as a module package. Python 3.9 and the MobLabPy module are required on the computer. “IP Camera Lite” for iPhones and iOS devices, or “IP Camera” for Android devices are also required (Both developed by ShenYao).

MobLabPy acts as a control panel of a visible light communication system. The computer display acts as a transmitter and the phone camera as a receiver. Users can experiment the effect of noise and error correcting codes have in communication.

Visit the MobLabPy Github/PyPI page for other resources.

<https://github.com/moblabpy/moblabpy>

<https://pypi.org/project/moblabpy/>



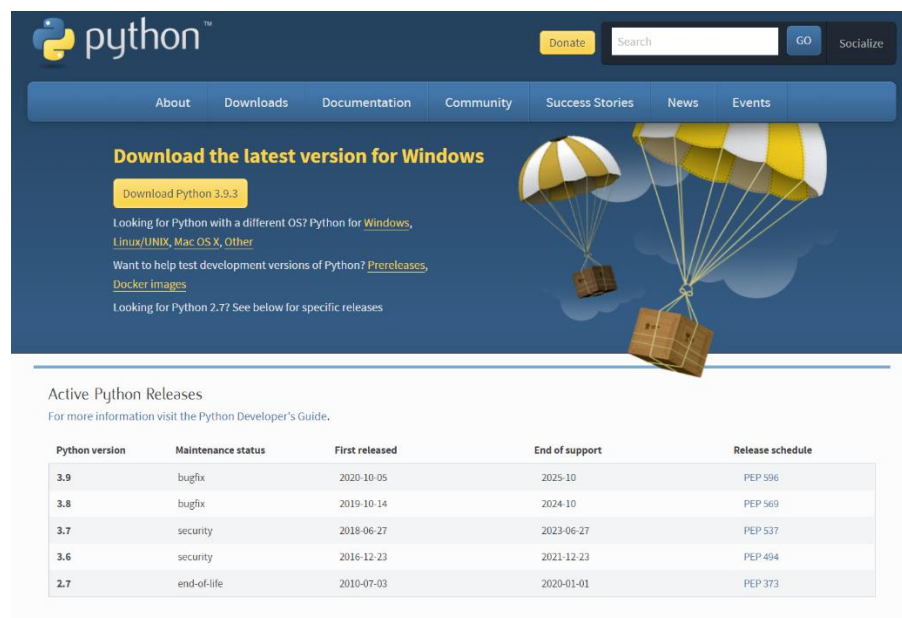
## 2 COMPUTER SOFTWARE INSTALLATION

### 2.1 INSTALLATION OF PYTHON

This section is for users without Python installed or with Python version older than 3.9. If you have Python 3.9 (or later) installed on your computer, please proceed to Section 2.2 (2.3 for Windows Users)

Python can be downloaded for free at:

<https://www.python.org/downloads/>



The screenshot shows the Python.org website. At the top, there is a navigation bar with links: About, Downloads, Documentation, Community, Success Stories, News, and Events. Below the navigation bar, there is a section titled "Download the latest version for Windows" with a button "Download Python 3.9.3". Below this, there are links for "Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other", "Want to help test development versions of Python? Prereleases, Docker images", and "Looking for Python 2.7? See below for specific releases". To the right of the text, there is an illustration of two parachutes with boxes hanging from them.

Active Python Releases

For more information visit the Python Developer's Guide.

| Python version | Maintenance status | First released | End of support | Release schedule |
|----------------|--------------------|----------------|----------------|------------------|
| 3.9            | bugfix             | 2020-10-05     | 2025-10        | PEP 596          |
| 3.8            | bugfix             | 2019-10-14     | 2024-10        | PEP 569          |
| 3.7            | security           | 2018-06-27     | 2023-06-27     | PEP 537          |
| 3.6            | security           | 2016-12-23     | 2021-12-23     | PEP 494          |
| 2.7            | end-of-life        | 2010-07-03     | 2020-01-01     | PEP 373          |

Select the respective operating system (Windows/Mac OS X)



This is a close-up of the "Download the latest version for Windows" section of the Python.org website. It features a yellow button labeled "Download Python 3.9.3". Below the button, there is a red rectangular box highlighting the text: "Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other". Below this box, there are links for "Want to help test development versions of Python? Prereleases, Docker images" and "Looking for Python 2.7? See below for specific releases". To the right of the text, there is an illustration of two parachutes with boxes hanging from them.

Select and download Python version 3.9 (or later) installer.

## Python Releases for Windows Python Releases for Mac OS X

- [Latest Python 3 Release - Python 3.9.4](#)
- [Latest Python 2 Release - Python 2.7.18](#)

### Stable Releases

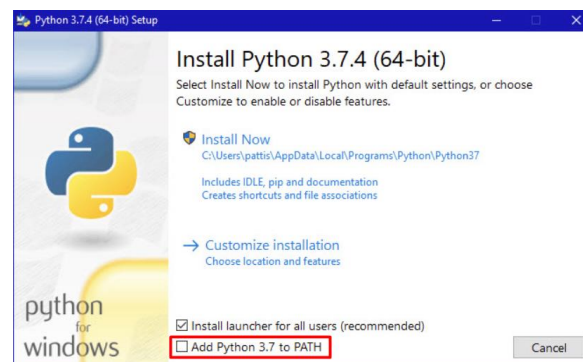
- [Python 3.9.4 - April 4, 2021](#)  
**Note that Python 3.9.4 cannot be used on Windows 7 or earlier.**
  - [Download Windows embeddable package \(32-bit\)](#)
  - [Download Windows embeddable package \(64-bit\)](#)
  - [Download Windows help file](#)
  - [Download Windows installer \(32-bit\)](#)
  - [Download Windows installer \(64-bit\)](#)
- [Python 3.9.3 - April 2, 2021](#)  
**Note that Python 3.9.3 cannot be used on Windows 7 or earlier.**
  - No files for this release.
- [Python 3.8.9 - April 2, 2021](#)  
**Note that Python 3.8.9 cannot be used on Windows XP or earlier.**
  - [Download Windows embeddable package \(32-bit\)](#)
  - [Download Windows embeddable package \(64-bit\)](#)
  - [Download Windows help file](#)
  - [Download Windows installer \(32-bit\)](#)
  - [Download Windows installer \(64-bit\)](#)
- [Python 3.9.2 - Feb. 19, 2021](#)  
**Note that Python 3.9.2 cannot be used on Windows 7 or earlier.**
  - [Download Windows embeddable package \(32-bit\)](#)
  - [Download Windows embeddable package \(64-bit\)](#)
  - [Download Windows help file](#)
  - [Download Windows installer \(32-bit\)](#)
  - [Download Windows installer \(64-bit\)](#)

- [Latest Python 3 Release - Python 3.9.4](#)
- [Latest Python 2 Release - Python 2.7.18](#)

### Stable Releases

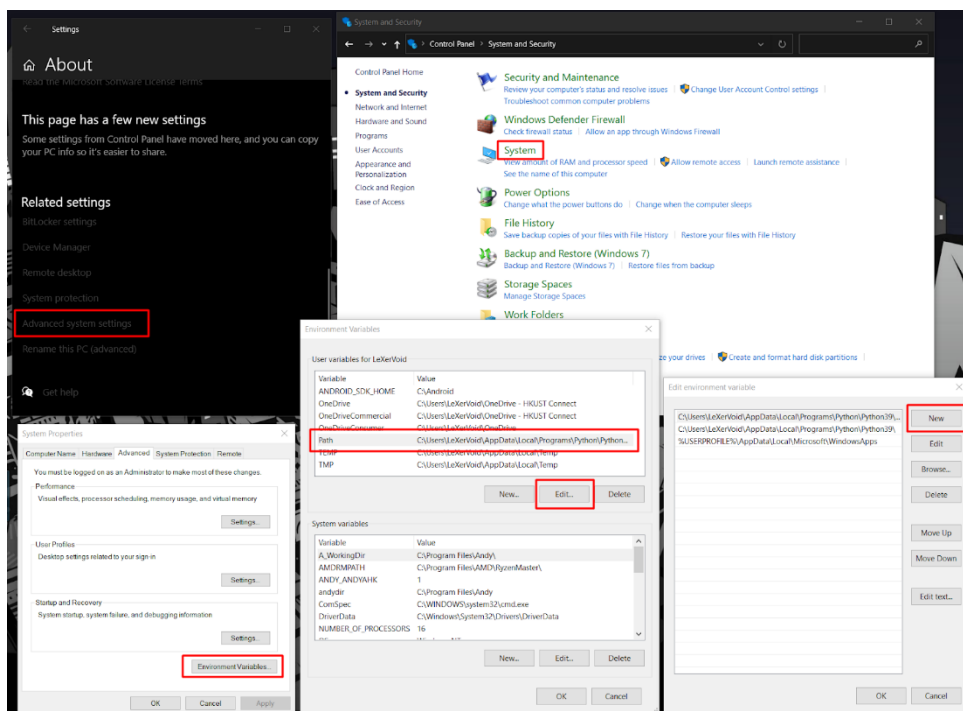
- [Python 3.9.4 - April 4, 2021](#)
  - [Download macOS 64-bit Intel installer](#)
  - [Download macOS 64-bit universal2 installer](#)
- [Python 3.9.3 - April 2, 2021](#)
  - No files for this release.
- [Python 3.8.9 - April 2, 2021](#)
  - [Download macOS 64-bit Intel installer](#)
- [Python 3.9.2 - Feb. 19, 2021](#)
  - [Download macOS 64-bit Intel installer](#)
  - [Download macOS 64-bit universal2 installer](#)
- [Python 3.8.8 - Feb. 19, 2021](#)
  - [Download macOS 64-bit Intel installer](#)
- [Python 3.6.13 - Feb. 15, 2021](#)
  - No files for this release.
- [Python 3.7.10 - Feb. 15, 2021](#)
  - No files for this release.
- [Python 3.8.7 - Dec. 21, 2020](#)
  - [Download macOS 64-bit Intel installer](#)
- [Python 3.9.1 - Dec. 7, 2020](#)
  - [Download macOS 64-bit Intel installer](#)
  - [Download macOS 64-bit universal2 installer](#)
- [Python 3.9.0 - Oct. 5, 2020](#)
  - [Download macOS 64-bit installer](#)

Follow the instruction of the installer and finish installation. Make sure to tick the option **“Add Python 3.9 to PATH”** at the start of the installer window.



In case you missed the option, either uninstall and reinstall Python, or follow the steps below:

1. Press **Win + R**, type **%AppData%** and hit **Enter** key
2. Back out one directory from Roaming
3. Navigate to **Local/Programs/Python/Python39/Scripts**
4. Copy the file path by selecting the whole directory in the address bar
5. Press **Win**, type **Control Panel** and hit **Enter** key
6. Navigate to **System and Security** → **System** → **Advanced System Setting** → **Environment Variables**
7. Select **Path** in the top box and **Edit**
8. Select **New** and paste the file path copied from step 4



## 2.2 INSTALLATION OF ZBAR ON MAC OS X

For Mac OS X, the zbar shared library needs to be installed.

Press **command+space**, type **terminal** and hit **enter** key

Install homebrew by typing the following command:

```
/BIN/BASH -C "\$\(CURL -FSSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh\)"
```

And install zbar:

```
brew install zbar
```

## 2.3 INSTALLATION OF MOBLABPY

For Windows users:

Press **Win**, type **cmd** and hit **Enter** key

Install MobLabPy by typing the following command:

[pip install moblabpy](#)

if the message “**‘pip’ is not recognized as an internal or external command**” appears, please refer to Section 2.1 for solution.

```
C:\Users\LeXerVoid>pip install moblabpy
Collecting moblabpy
  Downloading moblabpy-0.1.2.tar.gz (11 kB)
Requirement already satisfied: numpy in c:\users\lexervoid\appdata\local\programs\python\python39\lib\site-packages (from moblabpy) (1.19.5)
Requirement already satisfied: segno in c:\users\lexervoid\appdata\local\programs\python\python39\lib\site-packages (from moblabpy) (1.3.1)
Requirement already satisfied: pyzbar in c:\users\lexervoid\appdata\local\programs\python\python39\lib\site-packages (from moblabpy) (0.1.8)
Requirement already satisfied: Pillow in c:\users\lexervoid\appdata\local\programs\python\python39\lib\site-packages (from moblabpy) (8.1.0)
Requirement already satisfied: opencv-python in c:\users\lexervoid\appdata\local\programs\python\python39\lib\site-packages (from moblabpy) (4.5.1.48)
Using legacy 'setup.py install' for moblabpy, since package 'wheel' is not installed.
Installing collected packages: moblabpy
  Running setup.py install for moblabpy ... done
Successfully installed moblabpy-0.1.2
WARNING: You are using pip version 20.2.3; however, version 21.0.1 is available.
You should consider upgrading via the 'c:\users\lexervoid\appdata\local\programs\python\python39\python.exe -m pip install --upgrade pip' command.
```

For Mac OS X users

Press **command+space**, type **terminal** and hit **enter** key

Install pip by typing the following commands (if you haven't):

[curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py](#)

[python get-pip.py](#)

Install MobLabPy by typing the following command:

[pip install moblabpy](#)

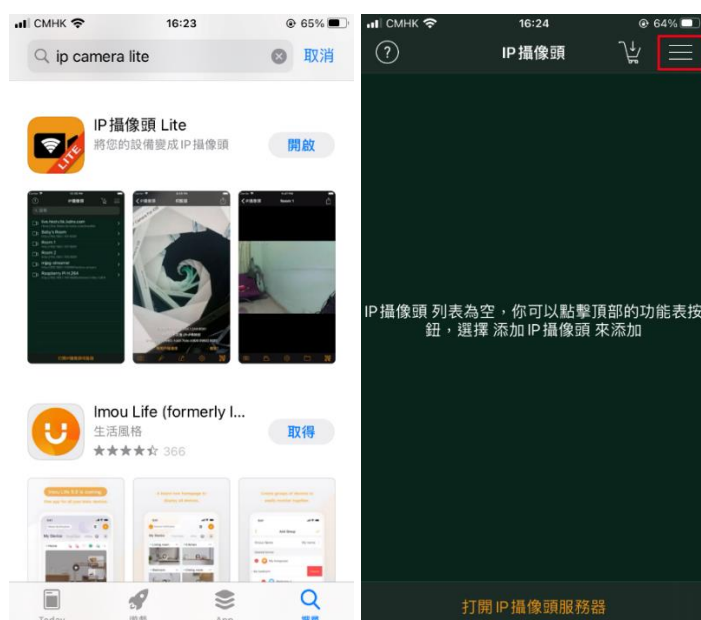


### 3 PHONE SOFTWARE INSTALLATION AND SET UP

#### 3.1 IPHONE AND IOS DEVICES

Look for “IP Camera Lite” by ShenYao China on Google Play or visit the following link:

<https://itunes.apple.com/us/app/ip-camera-lite/id1013455241?mt=8>



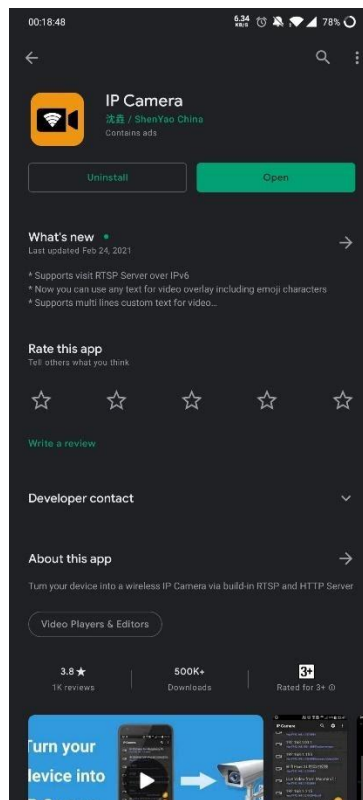
After Installation, open the app and go to “**Setting**”. Navigate to “**User**” and “**Password**” under the **Server** Category and empty them.



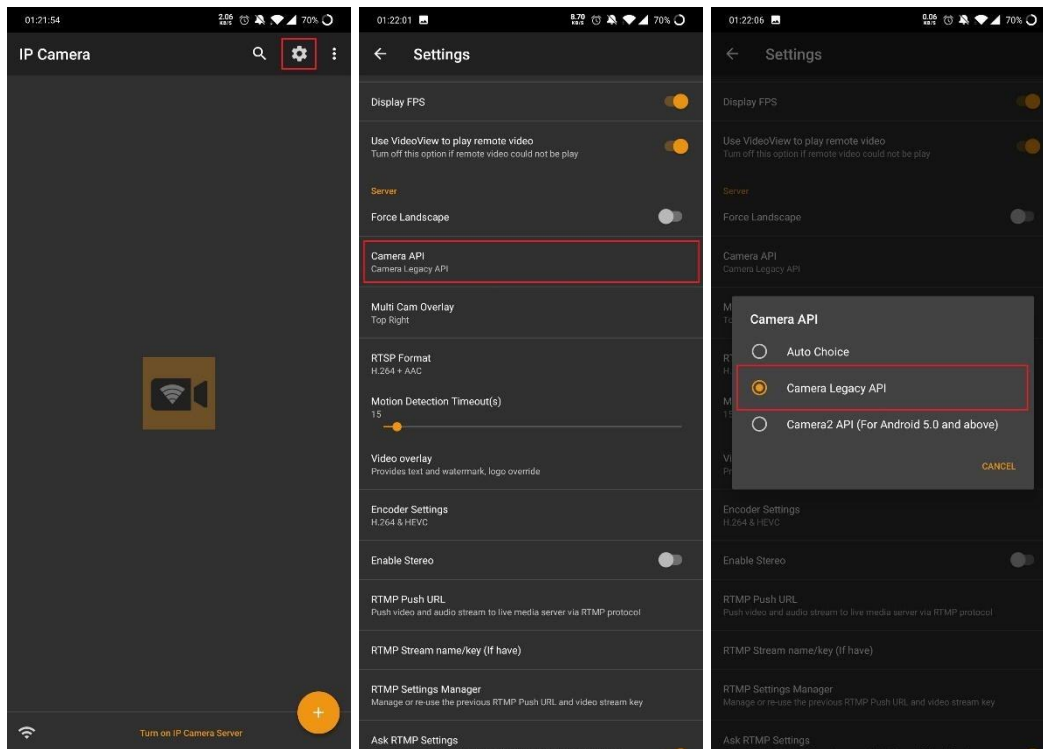
## 3.2 ANDROID DEVICES

Look for “IP Camera” by ShenYao China on Google Play or visit the following link:

<https://play.google.com/store/apps/details?id=com.shenyaocn.android.WebCam>



After installation, open the app and go to “**Setting**”. Navigate to “**Camera API**” under the **Server** category and set to “**Camera Legacy API**”



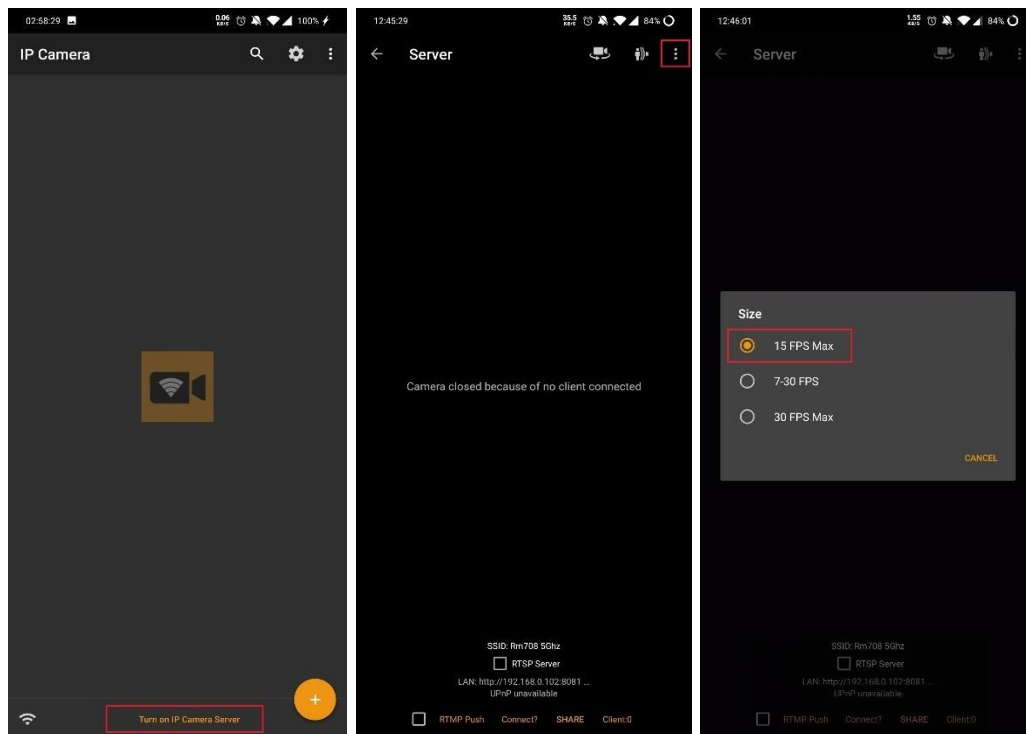
Scroll down and find “**User**” and “**Password**”. Click and empty them.

|               |       |
|---------------|-------|
| RTSP Port     | 8554  |
| User          | admin |
| Password      | ***** |
| Mail Settings |       |

## 4 USING MOBLABPY

### 4.1 CONFIGURING IP CAMERA

Make sure the computer is connected to the same Wi-Fi network when using IP camera. Open IP Camera and turn on the server. The IP address of the phone is displayed at the bottom of the screen. The framerate and resolution setting can be found on the upper right corner. Set the framerate to “**15 FPS Max**”.



## 4.2 SETTING UP CHANNEL IN PYTHON

To learn about the input and output parameter of any mentioned functions, please visit MobLabPy's GitHub page for more detail

Before starting the channel, A few variables are needed.

```
IP_ADDRESS = http://192.168.0.102:8081
message = "Hello World!"
```

The **IP address** should be replaced by the one obtained from the IP Camera App from Section 4.1.

The **PROPS** class contains changeable settings. The density of the square, or the Bit Per Slide value and the size scaling of the video player can be changed by the following code:

```
PROPS.set_BPS(16)
PROPS.SCALE = 2
```

The **BPS** should be a square number, 16, 25 and 36 are recommended, and **SCALE** can be any positive numbers.

Prepare the message by turning it to a bit sequence with the code below:

```
ascii_seq = str_to_ascii_seq(message)
bit_seq = ascii_seq_to_bit_seq(ascii_seq)
bit_mess = append_zeros(bit_seq, props = PROPS,
is_str = False)
```

This will turn the message to a bit sequence representing the binary ascii number of each character, appended to a multiple of the BPS.

```
IP_ADDRESS = "http://192.168.0.102:8081"
message = "Hello World!"
PROPS.set_BPS(16)
PROPS.SCALE = 2
ascii_seq = str_to_ascii_seq(message)
bit_seq = ascii_seq_to_bit_seq(ascii_seq)
bit_mess = append_zeros(bit_seq, props = PROPS, is_str = False)
```

To start the channel, the message has to be generated first. Run the following code:

```
generate_video(mystr = bit_mess, if_bin = True)
```

This will generate a video which contains multiple slides of image with QR codes and black-and-white square matrices. The matrices represent the bit sequence inputted as bit\_mess and is our means of communicating data.

In MobLabPy, major functions are contained and handled using a class MobLabPy. Set up the class by running the following code:

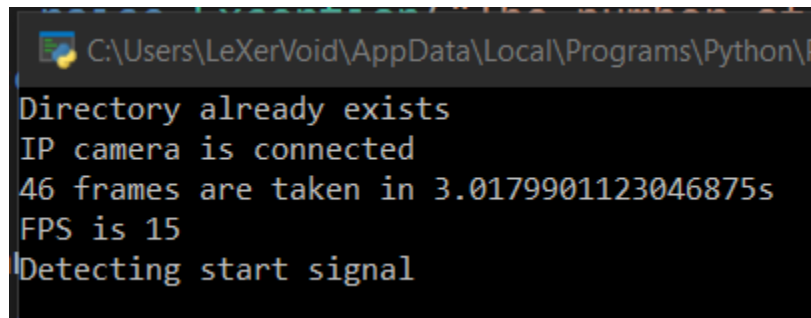
```
cam = MobLabPy(IP_ADDRESS, append_zeros(bit_seq,
props = PROPS, is_str = True), PROPS)

MobLabPy.start(cam)
```

This will start the process of setting up the channel and start the transmission.

```
generate_video(mystr = bit_mess, if_bin = True)
cam = MobLabPy(IP_ADDRESS, append_zeros(bit_seq, props = PROPS, is_str = True), PROPS)
MobLabPy.start(cam)
```

The console output displays the status of the process, which shows the FPS of the camera measured by the program, and whether the IP camera has been connected. The phone's screen should display the video feed of its camera, which indicates the phone is connected. The line "Detecting start signal" will appear when everything is ready.

A screenshot of a Windows command prompt window. The title bar shows the path "C:\Users\LeXerVoid\AppData\Local\Programs\Python\Python39\python.exe". The command prompt displays the following text:

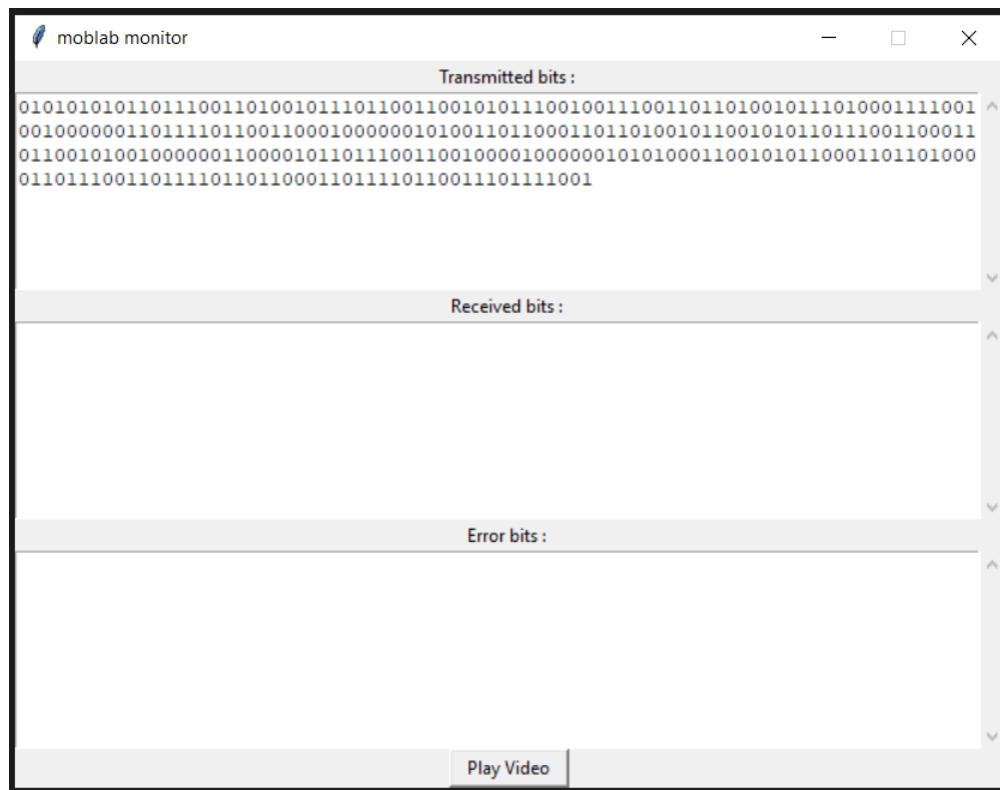
```
Directory already exists  
IP camera is connected  
46 frames are taken in 3.0179901123046875s  
FPS is 15  
Detecting start signal
```

If at any point of the transmission the message **"Synchronization failed. Please restart the program"** appears in the console, please stop the program and retry, make sure other configurations are correct.

### 4.3 USER INTERFACE AND VIDEO PLAYER

A window will be shown after the execution of the previous line.

The three text boxes display the transmitted, received and error bits of the current transmission. At the start of the transmission, only the first box is filled.





A button “**Play Video**” is located at the bottom of the window. Pressing it will open a new window, which plays the video generated in Section 4.1. The video player is built independently and does not require external software. A “**Replay**” button is located at the bottom to reset the video and play from the beginning.



## 4.4 TRANSMIT AND RECEIVE

Do not turn off your phone or exit IP Camera while the transmission is undergoing. This will cut the connection between the computer and the server.

Align the phone camera with the computer display, either by hand or preferably by placing the phone on a podium. **Make sure the phone is at the same angle as the display, such that the video will be captured as a square, or it may cause a decoding error.**



Press the “**Play Video**” to start the video at a new window. The video starts with a countdown, and then a big QR code appears (see above). This is the pilot signal which indicates the start of the transmission.

When this QR code is detected by the camera, a line confirming the pilot signal appears in the console.

```
C:\Users\LeXerVoid\AppData\Local\Programs\Python\Python39\python.exe
IP camera is connected
46 frames are taken in 3.0109333992004395s
FPS is 15
Detecting start signal
Start signal is detected. Start transmission...
█
```

This means the transmission has begun. **Do not move the camera, video window and let the video play until the end.**

The image below is the format of the bit matrix. It consists of four smaller QR codes used to locate the video and rotate the image. The square in the middle is filled with black-and-white square which represents the bit sequence.



Align the video player with the camera feed. IP camera shows what the camera is capturing and you can place the window in the middle based on it. **Have sufficient lighting and avoid reflection from the computer screen.** If outer interference is too heavy, it may disrupt the transmission entirely. When the video is playing, and the receiver is taking captures, **do not block any of the QR codes.**

The transmission can be interrupted by placing obstacles between the camera and the monitor to block the black-and-white matrixes, or brighten/dim the light to emulate different signal strengths. As the video plays, the received bits will be displayed on the monitor window at the received bits text box. The error bits text box also updates and compares the received bits with the original sequence. It displays '0' for every correctly transmitted bit and '1' for any error. Any '1' in the error bit text box is printed in red for increased readability.



At the end of the video, the pilot signal will appear again, indicating the end of the transmission. The line “**Transmission finished**” will be printed on the console. The IP Camera is disconnected and the received bits are available.

To access the result, run the following code:

```
result = cam.get_bit_seq()
```

The bit sequence will be stored in the result variable.

## 5 SUPPLEMENTARY FUNCTIONS

For extra detail on every function, visit the Github page for docstring documents.

MobLabPy includes a number of extra functions for users to use in manipulating string and bit sequences.

### **str\_to\_ascii\_seq(my\_str)**

- Turns string of character to an array of ascii numbers, each representing a letter of the original string.
- Example input: “Hi” (type: string)
- Example output: [72 105] (type: int array)

### **ascii\_seq\_to\_str(ascii\_arr)**

- Turns an array of ascii numbers to string of character
- Example input: [72 105] (type: integer array)
- Example output: “Hi” (type: string)

### **ascii\_seq\_to\_bit\_seq(ascii\_arr)**

- Turns an array of ascii numbers to string of bit sequence
- Example input: [72 105] (type: integer array)
- Example output: [0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1] (type: int array)

### **bit\_seq\_to\_ascii\_seq(bit\_str)**

- Turns a string of bit sequence to an array of ascii numbers
- Example input: [0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1] (type: int array)
- Example output: [72 105] (type: integer array)

### **append\_zeros(bit\_seq, props, is\_str)**

- Add zeros to a string of bit sequence until the length is a multiple of props.BPS. Returns string if is\_str is True and integer array if False
- Example input: “01001000”, props, True (type: string, PROPS, bool)
- Example output: “0100100000000000” (type: string)
- Example input: “01001000”, props, True (type: string, PROPS, bool)
- Example output: [0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1] (type: int array)

### **encode(D, G)**

- Perform matrix multiplication on D and G, i.e.  $D \times G$
- Example input: [1 1 1],  $\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$  (type: list, ndarray)
- Example output: [1 1 1 0 0 0] (type: ndarray)

### **syndrome(R, H)**

- Calculate the syndrome of the received bit sequence
- Example input: [1 0 0 0 1 1],  $\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$  (type: list, ndarray)
- Example output: [0 0 0] (type: ndarray)

## 6 EXAMPLE USAGE

The following code is an example usage of MobLabPy

```
IP_ADDRESS = "http://192.168.0.102:8081"
message = "University of Science and Technology"
PROPS.set_BPS(25)
PROPS.SCALE = 1.5
ascii_seq = str_to_ascii_seq(message)
bit_seq = ascii_seq_to_bit_seq(ascii_seq)
bit_mess = append_zeros(bit_seq, props = PROPS, is_str =
False)

# Add coding here

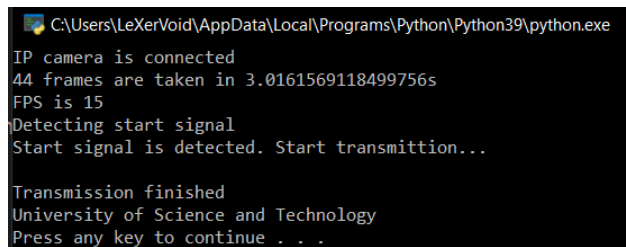
generate_video(mystr = bit_mess, if_bin = True)
cam = MobLabPy(IP_ADDRESS, append_zeros(bit_seq, props =
PROPS, is_str = True), PROPS)
MobLabPy.start(cam)

result = cam.get_bit_seq()

# Add decoding here

result_ascii = bit_seq_to_ascii_seq(result)
result_string = ascii_seq_to_str(result_ascii)
print(result_string)
```

The console output is as following:



```
C:\Users\LeXerVoid\AppData\Local\Programs\Python\Python39\python.exe
IP camera is connected
44 frames are taken in 3.0161569118499756s
FPS is 15
Detecting start signal
Start signal is detected. Start transmission...

Transmission finished
University of Science and Technology
Press any key to continue . . .
```