Picom

Mode d'emploi d'utilisation

Table des matières

Bibliothèque

1. Liste des bibliothèques utilisées :

Bibliothèques standards de Python (installées par défaut avec Python3) :

- **socket** : Permet la communication réseau entre les Raspberry Pi via des sockets.
- **struct** : Permet de manipuler des données en C, utile pour les communications binaires.
- **threading** : Gère l'exécution de plusieurs threads pour effectuer plusieurs tâches simultanément.
- **logging** : Utilisée pour l'enregistrement des événements, particulièrement utile pour le débogage.
- **json** : Utilisée pour la manipulation des données JSON, souvent utilisées pour les échanges de données entre systèmes.
- **time** : Fournit diverses fonctions pour manipuler le temps.
- **collections.deque** : Fournit une structure de données de type file pour la gestion d'un tampon circulaire.
- **queue** : Utilisée pour la gestion des files d'attente de manière thread-safe.

Bibliothèques tierces :

Ces bibliothèques doivent être installées séparément, car elles ne font pas partie de la bibliothèque standard de Python3.

- **cv2** (OpenCV) : Utilisée pour la capture, le traitement et l'affichage des images vidéo.
- **numpy** : Utilisée pour la manipulation des tableaux et des matrices, souvent utilisée avec OpenCV pour le traitement d'image.
- **pyaudio** : Permet la capture et la lecture de l'audio en temps réel.
- **tkinter** : Interface graphique native de Python, utilisée pour créer la fenêtre principale et les différents composants de l'interface utilisateur.

- **PIL (ou Pillow)** : Utilisée pour la manipulation des images, spécifiquement pour convertir les images OpenCV en format compatible avec Tkinter.

2. Comment installer les bibliothèques nécessaires :

Il suffit de taper ceci sur le terminal : sudo apt install pyhton3-'nom de la bibliothèque'.

On ne peut guère utiliser « pip install » car travaillant sur un Raspberry pi, l'environnement est traité en externe d'où l'importance d'utiliser « sudo » qui fait une demande en tant qu'administrateur. Voici ce qu'il faut écrire sur le terminal pour les installer :

sudo apt **update**
sudo apt install **python3**-opencv **python3**-numpy **python3**-pyaudio **python3**-pil

2.1 Détails des bibliothèques installées :

- **python3-opencv** : Installe OpenCV, qui est utilisé pour la capture et le traitement d'images et de vidéos.
- **python3-numpy** : Installe NumPy, une bibliothèque utilisée pour la manipulation efficace des tableaux, matrices et opérations mathématiques.
- **python3-pyaudio** : Installe PyAudio, nécessaire pour capturer et jouer de l'audio en temps réel.
- **python3-pil** : Installe Pillow, qui est une bibliothèque pour la manipulation d'images. Notez que sur certaines distributions, elle est encore référencée comme python3-pil.

Descriptif du code

Le script implémente une interface utilisateur pour capturer et diffuser de la vidéo et de l'audio entre deux Raspberry Pi. Il inclut aussi un système de chat textuel et une fonctionnalité d'alerte. Voici un descriptif détaillé de la structure du code et des points importants.

1. Importation des Bibliothèques

Le script commence par importer les bibliothèques nécessaires :

- **cv2 (OpenCV)** : Pour la capture et le traitement des vidéos.
- **numpy** : Pour manipuler les données numériques sous forme de tableaux.
- **pyaudio** : Pour la capture et la lecture de l'audio en temps réel.

- **socket** : Pour la communication réseau via des sockets UDP.
- **struct** : Pour la manipulation des données binaires.
- **tkinter et PIL (Pillow)** : Pour créer l'interface graphique (GUI).
- **threading et queue** : Pour gérer le multitâche et la communication inter-thread.
- **logging** : Pour enregistrer les événements importants et les erreurs.
- **json** : Pour manipuler les données au format JSON, utilisées pour les commandes entre les Raspberry Pi.

2. Configuration des Paramètres

Le script définit plusieurs variables globales et constantes utilisées tout au long du code :

- **Dimensions de la vidéo (VIDEO_WIDTH, VIDEO_HEIGHT, FRAME_SIZE)** : Définissent la taille des vidéos capturées et affichées.
- **Paramètres audio (CHUNK, FORMAT, CHANNELS, RATE)** : Spécifient la configuration pour la capture et la lecture audio.
- **Adresses IP et ports (REMOTE_IP, SECONDARY_IP, VIDEO_PORT, AUDIO_PORT, COMMAND_PORT)** : Configurent la communication réseau entre les deux Raspberry Pi.
- **Variables de contrôle (stop_video_event, stop_secondary_video_event, stop_audio_event)** : Utilisées pour gérer l'arrêt des threads de capture/lecture audio et vidéo.
- **Tampons vidéo et audio (video_buffer, audio_buffer)** : Stockent temporairement les données audios et vidéo reçues.

3. Fonctions Principales

a. Capture et Diffusion Vidéo

- **send_video()** : Capture les images de la caméra locale, les redimensionne, et les envoie par paquets via UDP à l'autre Raspberry Pi.
- **receive_video_from_secondary()** : Reçoit des segments vidéo de l'autre Raspberry Pi, les réassemble, puis les affiche via l'interface Tkinter.

b. Capture et Diffusion Audio

- **send_audio()** : Capture l'audio depuis un microphone et l'envoie à l'autre Raspberry Pi via UDP.
- **receive_audio()** : Reçoit des données audio, ajuste le volume, et les joue via les haut-parleurs.

c. Gestion des Commandes

- **send_command(command)** : Envoie des commandes sous forme de messages JSON à l'autre Raspberry Pi, comme démarrer ou arrêter la vidéo/audio.
- **receive_command()** : Écoute et traite les commandes reçues de l'autre Raspberry Pi, comme afficher une alerte ou recevoir un message de chat.

d. Interface Utilisateur (Tkinter)

- **toggle_my_camera()** et **toggle_secondary_camera()** : Permettent à l'utilisateur de démarrer ou d'arrêter la capture vidéo sur l'un ou l'autre des Raspberry Pi.
- **toggle_audio()** : Permet à l'utilisateur de démarrer ou d'arrêter la capture audio.
- **adjust_volume(val)** : Ajuste le volume de la lecture audio.
- **send_chat_message()** et **receive_chat_message(command)** : Gèrent l'envoi et la réception de messages de chat entre les deux Raspberry Pi.
- **show_alert()** : Affiche un message d'alerte en plein écran.

4. Gestion des Threads

- Le script utilise des threads pour exécuter simultanément la capture/lecture vidéo et audio, ainsi que pour écouter les commandes reçues. Cela permet à l'interface graphique de rester réactive pendant que les opérations réseau et multimédia sont effectuées en arrière-plan.

5. Boucle Principale de l'Interface

- **root.mainloop()** : Lancement de la boucle principale de Tkinter, qui gère les événements de l'interface utilisateur.

Points Importants

1. **Multithreading** : Le script utilise le multithreading pour gérer simultanément la capture/lecture de la vidéo, de l'audio, et l'écoute des commandes. Cela permet d'éviter que l'interface utilisateur se bloque lors de l'exécution de ces tâches en arrière-plan.
2. **Communication Réseau via UDP** : Le choix d'UDP pour la transmission des vidéos et de l'audio est crucial pour minimiser la latence. Cependant, il faut noter que UDP ne garantit pas la livraison des paquets, ce qui pourrait entraîner des pertes de données vidéo/audio.

3. **Tampons Circulaires** : Les tampons circulaires (comme audio_buffer) sont utilisés pour lisser la lecture audio en cas de petites interruptions dans la réception des paquets, aidant à éviter les coupures audios.
4. **Interface Utilisateur** : Tkinter est utilisé pour créer une interface utilisateur simple mais fonctionnelle, permettant aux utilisateurs de contrôler facilement les flux vidéo et audio, ainsi que d'envoyer/recevoir des messages de chat.
5. **Volume Audio** : Le script inclut une fonctionnalité de contrôle du volume, ce qui est essentiel pour ajuster la lecture audio selon les besoins des utilisateurs.

Explication de la Connexion entre les Deux Raspberry Pi

La connexion des deux Raspberry Pi se fait via un réseau Wi-Fi local ou par Ethernet, avec un Raspberry Pi principal qui gère l'interface administrateur (y compris les commandes) et un Raspberry Pi secondaire qui interagit avec le premier pour la capture et la diffusion de l'audio, de la vidéo, et des messages de chat. Voici une explication détaillée de cette connexion.

1. Configuration du Réseau Wi-Fi Local

- **Raspberry Pi Principal (Serveur/Admin)** : Ce Raspberry Pi crée un point d'accès Wi-Fi local, auquel le deuxième Raspberry Pi se connecte. Ce Raspberry Pi est aussi celui qui exécute le code de gestion de l'interface utilisateur, permettant à l'utilisateur de contrôler les flux vidéo, audio et les commandes via une interface graphique (Tkinter).
- **Raspberry Pi Secondaire (Client)** : Ce Raspberry Pi se connecte au réseau Wi-Fi local créé par le Raspberry Pi principal. Il envoie et reçoit des flux audio et vidéo selon les commandes reçues du Raspberry Pi principal.

  Il faut nécessairement alors connectés les Raspberry pi au même wifi.

  Dans le fichier du script nommé « PicomAdmin/Client.py », effacer l'hashtag correspond au type de connexion que vous utilisé sur la variable nommé « REMOTE_IP ». Une adresse IP correspondant à une connexion WI-FI est attribuée par défaut, idem pour l'Ethernet. Ces adresse IP reste donc inchangé sauf sur une migration de l'application sur un autre appareil.

  Si aucun wifi local n'a été créer, il est possible d'en crée 1 en accédant au paramètre wifi via la barre latérale, puis aller à « Advanced Options » enfin « Create Wi-Fi Hotspot... ». Vous aurez le choix d'émettre un mot de passe ou non mais si c'est le cas, utiliser la

sécurité Wi-Fi « WPA et WPA2 personnel » pour qu'il soit détectable. Lier à un nouveau Wi-Fi, l'adresse IP changera. Il est alors nécessaire de modifier dans le script l'adresse IP de REMOTE_IP. L'adresse IP des Raspberry pi sont visible en glissant la souris sur le logo WI-FI sur la barre latérale.

2. Configuration du Réseau Ethernet

Concernant la connexion Ethernet, il suffit de brancher un câble Ethernet entre les deux Raspberry pi. Après cela il faut attendre 1 à 2 min pour que le Raspberry pi trouve l'IP correspondant au script. Pour savoir si les Raspberry pi sont connectés, un logo de deux flèches bleues dans un sens vertical opposé serra visible. Dans le cas contraire, elles seront grises avec des croix rouges. Ils se peut qu'après avoir débranché ou rebranché le câble Ethernet, cette liaison et de recherche de l'adresse IP ne s'effectue pas. Il faut alors redémarrer celui qui n'est pas connecté, ou redémarrer les deux Raspberry pi en même temps puis rebranché les deux câbles.

3. Communication entre les Deux Raspberry Pi

La communication entre les deux Raspberry Pi se fait via des sockets UDP, une méthode de transmission rapide mais non sécurisée qui convient bien aux flux en temps réel comme la vidéo et l'audio. Voici comment cette communication est organisée :

- **Adresses IP et Ports** :
  - **REMOTE_IP** : C'est l'adresse IP du Raspberry Pi secondaire telle qu'elle est vue par le Raspberry Pi principal. Dans notre cas, c'est 10.42.0.147.
  - **SECONDARY_IP** : Adresse IP que vous utiliserez si vous souhaitez différencier les rôles des Raspberry Pi dans le réseau s'il y a plusieurs Raspberry pi connecté.
  - **Ports (VIDEO_PORT, AUDIO_PORT, COMMAND_PORT)** : Trois ports différents sont utilisés pour transmettre la vidéo, l'audio, et les commandes respectivement. Cela permet de segmenter les types de données échangées, réduisant ainsi les risques de collisions et facilitant la gestion des flux.
- **Transmission Vidéo** :
  - **Envoi de Vidéo (send_video())** : Le Raspberry Pi principal capture la vidéo via sa caméra, puis envoie les segments vidéo au Raspberry Pi secondaire via le port UDP spécifié. Les segments sont ensuite réassemblés pour être affichés.

  - **Réception de Vidéo (receive_video_from_secondary())** : Le Raspberry Pi principal peut aussi recevoir des flux vidéo du Raspberry Pi secondaire, si configuré ainsi. Ces segments vidéo sont reçus, réassemblés, et affichés via l'interface utilisateur.

- **Transmission Audio** :
  - **Envoi d'Audio (send_audio())** : Le Raspberry Pi principal capture l'audio en temps réel via un microphone, puis envoie les données audio au Raspberry Pi secondaire via un port UDP.
  - **Réception d'Audio (receive_audio())** : Le Raspberry Pi principal peut recevoir des flux audio du Raspberry Pi secondaire. Les données sont reçues, ajustées en volume, puis jouées.

- **Gestion des Commandes** :
  - **Envoi de Commandes (send_command())** : Le Raspberry Pi principal envoie des commandes au Raspberry Pi secondaire sous forme de messages JSON, pour contrôler des actions comme démarrer ou arrêter la capture vidéo/audio, ou encore afficher des alertes.
  - **Réception de Commandes (receive_command())** : Le Raspberry Pi principal écoute en permanence les commandes provenant du Raspberry Pi secondaire, et exécute les actions correspondantes.

4. Fonctionnement de l'Interface Administrateur

- **Raspberry Pi Principal** : L'interface utilisateur permet de contrôler l'ensemble du système. L'utilisateur peut démarrer ou arrêter les flux vidéo/audio, envoyer des commandes au Raspberry Pi secondaire, et afficher des messages de chat. C'est le centre de contrôle, ce qui fait de ce Raspberry Pi l'administrateur du réseau.
- **Raspberry Pi Secondaire** : Ce Raspberry Pi exécute les commandes envoyées par le Raspberry Pi principal, telles que démarrer la capture vidéo ou audio. Il n'a pas d'interface utilisateur, il est principalement un exécutant des instructions reçues.

  Un fichier nommé « config.json » à été créer pour permettre a un utilisateur de référer dans ce fichier l'adresse correspondant sans ouvrir le script.

Installation OS et Emplacement du code

Utiliser l'application « Raspberry pi Imager » pour pouvoir formater et installer l'os de son choix, par rapport à la version de son Raspberry pi. Un lien est disponible pour plus d'information :

https://www.raspberrypi.com/software

Concernant le code, il faut se rendre Dans : /Documents/Picom/PicomAdmin.py

Dans le fichier Picom vous pourriez y voir un fichier nommé « start_app.sh » qui permet lancer le code sans éditeur de texte. Un autre fichier dans le répertoire Desktop permet via le biais de « start_app.sh » et une image pour l'afficher en raccourci et exécuter dès le bureau.

Astuce

1.Copier les données de la carte SD

Il y a la possibilité de copier l'intégralité des fichiers et données d'une carte SD et de la transférer à un autre. Cela permettra d'exécuter l'application sans réaliser les étapes émises plus haut. Pour cela utiliser SD-Card Copier, une application intégrer au Raspberry pi qui permet de faire ce travail.