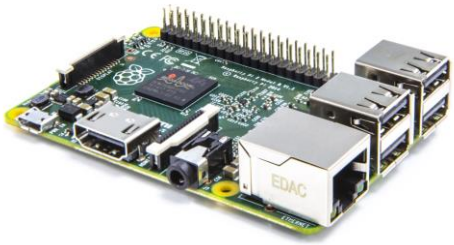
# Communiquer, contrôler à distance un système informatique autonome

Autres activités du TD : voir https://www.ensta-bretagne.fr/lebars/tutorials/TD_wifi_rtk_multiboot_rpi.pdf.

**Rendre sur Moodle dans un fichier de la forme DATE_NOM1_NOM2_..._NOMN.zip tous les documents utiles permettant de montrer le travail effectué : notes techniques, codes, fichiers de config, logs, données brutes et/ou traitées, infos précises sur les versions logicielles et matérielles utilisées, schémas en CAO ou manuels, captures d'écran, photos et vidéos à différentes étapes, etc.**

## Activité 3 : Préparation de Raspberry Pi

Nb max personnes : 4 quadrinômes
Matériel nécessaire par équipe : 1 Raspberry Pi 2, 3, 4 ou 5 et son alim et son ou ses adaptateurs pour écran, 1 adaptateur Wi-Fi USB pour les pi2, 1 micro SD 32 GB et son adapteur SD ou USB (si pas de port SD/micro SD sur PC portable), 1 câble eth, 1 clavier, 1 souris USB, 1 PC de salle info, 1 PC portable d'étudiant et 1 smartphone d'étudiant pouvant partager Internet, 1 camera USB ou pi (pour la fin, **attention à ne pas confondre le connecteur CSI avec DSI pour la caméra sur la pi !**)
Durée estimée : 1 demi-journée ou plus



Le but de cette activité va être de préparer une Raspberry Pi avec une configuration logicielle très proche de celle sur les PC sous Ubuntu. L'OS par défaut pour les Raspberry Pi est Raspberry Pi OS (anciennement nommé Raspbian), mais bien que la plupart des paquets soient similaires entre Raspberry Pi OS et Ubuntu, quelques-uns comme ceux de ROS sont souvent indisponibles (il est toutefois souvent possible de les compiler à partir des sources). Elle sera munie d'une carte Wi-Fi USB (pi2 uniquement, les pi3 et suivantes ayant déjà une carte Wi-Fi intégrée) et d'une carte micro SD qui aura pour but de contenir l'OS (un adaptateur SD ou USB permettra de la préparer à partir d'un PC sous Windows).

**Note sur les clés USB : bien que certains sites web indiquent qu'on peut désormais débrancher sans avoir cliqué sur Ejecter dans Windows, en pratique c'est plus compliqué vu que ça dépend de la version de Windows 10 et aussi des clés USB (ou adapteurs de cartes SD, etc.), certaines étant reconnues comme des disques durs. Pour éviter de se retrouver avec des données corrompues, toujours s'assurer que la clé ou le disque ont été éjectés proprement avant de les débrancher !**

## 3.1 Installation d'Ubuntu for ARM

Brancher la micro SD via son adaptateur USB sur un PC sous Windows avec les droits administrateurs, lancer une invite de commandes et taper :
**diskpart**
Puis dans diskpart :
**list disk**

En fonction de la liste qui s'affiche, déterminer le numéro du disque correspondant à la micro SD et le sélectionner avec la commande :
**select dis 9**
9 étant un exemple de numéro de disque. Effacer toutes les partitions dessus (**attention : toujours vérifier attentivement si c'est le bon disque !**) avec la commande :
**clean**

Taper:
**exit**
pour quitter diskpart.

Copier depuis un PC de salle info (disques réseau) **enseignement\rob\...\Robotique pratique\dd.exe** et **ubuntu-XXX-preinstalled-server-armhf+raspi.img.xz** de **…\Robotique pratique\pi2** (regarder dans **public\share** si **enseignement\rob** n'est pas disponible) ou **https://ubuntu-mate.org/download/armhf/jammy/** (pour une variante Ubuntu Mate) vers un dossier du PC. Extraire avec **7-Zip ubuntu-XXX.img.xz** et faire en sorte que **dd.exe** et **ubuntu-XXX.img** soient dans le même dossier, puis ouvrir une invite de commandes (avec les droits administrateurs) dans ce dossier et taper :
**dd if=ubuntu-XXX.img of="\\.\PhysicalDrive9" bs=2M --progress**
9 étant un exemple de numéro de disque (**attention : toujours vérifier attentivement si c'est le bon disque !**).

**Ceci va prendre quelques minutes.**

Il est possible que Windows ou d'autres applications cherchent automatiquement à accéder à la micro SD en même temps et fassent échouer la commande **dd**, éventuellement faire clic droit sur le bouton Start de Windows et lancer Disk Management puis enlever la lettre attribuée à l'une des partitions de la micro SD et refaire les commandes avec **diskpart** pour effacer la micro SD (on peut aussi utiliser **diskpart** pour retirer la lettre avec les commandes relatives aux **volumes**) puis retenter la commande **dd**, et/ou redémarrer, essayer de débrancher-rebrancher la micro SD et son adaptateur éventuel… A la place de **dd**, l'outil **balenaEtcher** (à télécharger sur Internet) peut aussi être utilisé.

Ejecter la micro SD, la mettre dans la Raspberry Pi, connecter tout le nécessaire (notamment l'écran en 1er avant de mettre sous tension la Raspberry Pi, ainsi que la clé Wi-Fi pour les pi2) et la démarrer (si la Raspberry Pi ne démarre pas, enlever et remettre la carte micro SD au cas où il y aurait un faux contact).

Si demandé, faire la **System Configuration** avec les paramètres suivants:
**English**
**French keyboard**
**Paris**

Si demandé, choisir un **Computer name** qui devra être unique sur le réseau de l'école…
A noter que du texte peut parfois s'afficher juste après l'invite de login, ce qui peut donner l'impression que l'invite de login n'est pas encore apparue…
A noter que CTRL+ALT+F1…CTRL+ALT+F7 permet de basculer entre plusieurs terminal/GUI en plein écran.

Identifiant : **user** (ou garder **ubuntu** pour les versions Server)
Password : **Workstation** (les versions Server ont **ubuntu** par défaut mais demandent de changer, à noter que le clavier peut être en QWERTY par défaut)

Si demandé, choisir **Log in automatically**

**L'installation peut prendre quelques minutes.**

Sur Ubuntu Server, le clavier est par défaut en QWERTY, pour le changer :
# Only for the current session
**sudo loadkeys fr**
# To make it permanent (configuration takes 5 min...)
**sudo dpkg-reconfigure keyboard-configuration**
# Select "Generic 105-key (Intl) PC", then "French" twice. For AltGr key, select "The default for the keyboard layout" then "no compose key"...

Une fois le bureau/terminal affiché, régler la date et l'heure (idéalement via l'interface graphique, sinon avec la commande **date**), désactiver les mises à jour automatiques (using GUI if available, otherwise choose **No** after **sudo dpkg-reconfigure unattended-upgrades**), puis redémarrer avec la commande **sudo reboot** (il y a parfois des problèmes réseaux au premier démarrage, à noter que la commande pour arrêter proprement est **sudo shutdown now** s'il n'y a pas de GUI…).

Facultatif: sur Ubuntu Mate, certaines commandes habituelles des versions standards d'Ubuntu sont différentes :
**gnome-terminal->mate-terminal**
**nautilus->caja**
**gedit->pluma**
**gnome-system-monitor->mate-system-monitor**
En tapant e.g. :
**sudo ln -s /usr/bin/mate-terminal /usr/bin/gnome-terminal**
**sudo ln -s /usr/bin/caja /usr/bin/nautilus**
**sudo ln -s /usr/bin/pluma /usr/bin/gedit**
**sudo ln -s /usr/bin/mate-system-monitor /usr/bin/gnome-system-monitor**
on peut par la suite réutiliser les mêmes commandes que sur un Ubuntu standard.

Note : sur Ubuntu Server, la commande **nano** permet de créer et éditer des fichiers textes (à noter que dans l'aide qui apparait parfois en bas, le caractère ^ correspond à la touche CTRL, e.g. CTRL+S pour enregistrer, CTRL+X pour quitter).

## 3.2 Configuration du réseau

Se connecter au réseau Wi-Fi **iot** (ou celui de votre smartphone).

En général, Ubuntu Server n'a pas d'interface graphique par défaut et utilise **Netplan** comme outil de gestion du réseau, contrairement aux versions Ubuntu Desktop qui utilisent plutôt **NetworkManager**. S'inspirer des infos dans https://www.ensta-bretagne.fr/lebars/Share/Ubuntu.txt (chercher section évoquant Netplan) pour créer un fichier **/etc/netplan/01-netcfg.yaml** contenant les paramètres pour se connecter en DHCP au réseau Wi-Fi **iot** (ou celui de votre smartphone, à noter qu'il peut y avoir des incompatibilités avec certains types de cryptage Wi-Fi ou de fréquence/canal, si besoin changer les paramètres du mode Point d'accès Wi-Fi sur le smartphone pour qu'il soit plutôt en 2.4 GHz WPA2) ou au réseau Ethernet (filaire) de l'école. Mettre à jour le nom réseau de la pi avec un nom unique avec **sudo nano /etc/hostname** et changer aussi si nécessaire le nom à la ligne 127.0.1.1 avec le nouveau nom réseau dans **sudo nano /etc/hosts**. Redémarrer et essayer de se connecter en SSH à la pi (un serveur SSH est actif par défaut pour Ubuntu Server, par contre il se peut que l'authentification par mot de passe soit désactivée par défaut, voir les fichiers de config dans **/etc/ssh** pour la réactiver si nécessaire).

Facultatif : Si possible, pour chaque connexion réseau, changer **IPv4 Settings\Additional search domains** avec **ensta-bretagne.fr,ensieta.fr,ensieta.ecole**.

OLD : Si problèmes d'accès à Internet liés au proxy de l'école, dans **Control Center\Internet and Network\Network proxy** choisir **Manual** et mettre les réglages suivants et cliquer sur **Apply system wide** :
**http 192.168.1.17 8080**
**https 192.168.1.17 8080**
**ftp 192.168.1.17 8080**
**socks 192.168.1.10 822**
ou
**http 192.168.1.10 3128**
**https 192.168.1.10 3128**
**ftp 192.168.1.10 3128**
**socks 192.168.1.10 822**
(il y a potentiellement plusieurs serveurs proxy à l'école selon les zones, etc.).

Malgré ces réglages, certaines applications particulières peuvent ne pas réussir à se connecter correctement à Internet à cause du proxy de l'école, dans ce cas il faut les traiter au cas par cas. Par exemple, la commande **apt-get** a son propre fichier de configuration
**sudo nano /etc/apt/apt.conf**
(il y a aussi parfois /etc/apt/apt.conf.d/proxy) et mettre ce qui est dans http://www.ensta-bretagne.fr/lebars/Share/apt.conf . Pour y accéder, il faudra lancer **Firefox** et dans **Options\Advanced\Network\Settings**, choisir **Auto-detect proxy settings for this network** (**Firefox** aussi ne prend pas toujours en charge les réglages de proxy du système et nécessite sa propre configuration).

De plus, il peut être nécessaire de taper :
**export https_proxy=192.168.1.10:3128**
**export http_proxy=192.168.1.10:3128**
**export ftp_proxy=192.168.1.10:3128**
**export socks_proxy=192.168.1.10:822**
**export no_proxy=localhost,127.0.0.0/8,ensieta.ecole,ensieta.fr,ensta-bretagne.fr**

dans le terminal dans lequel on travaille pour que les commandes suivantes tapées dans ce terminal puissent utiliser les réglages de proxy. Si les commandes utilisent **sudo**, il faudra peut-être utiliser **sudo -E** pour qu'elles prennent en compte les variables définies.

Vérifier que :
**sudo apt-get update**
se passe correctement, sinon vérifier les différents réglages réseau.

## 3.3 Commande raspi-config, paramètres spécifiques aux pi, serveur SSH

Pour redimensionner la partition du système et la faire utiliser toute la micro SD ainsi qu'activer un serveur SSH, utiliser la commande **raspi-config** (suivre les instructions sur: http://ubuntu-mate.org/raspberry-pi/ si besoin, il se peut que ce soit déjà fait automatiquement). Vérifier le redimensionnement à l'aide de la commande **df** et le bon fonctionnement du serveur SSH (e.g. d'abord en local avec la commande **ssh user@127.0.0.1**, puis à distance à partir d'un PC connecté sur le même réseau), s'il ne fonctionne pas désactiver l'option dans **raspi-config** et essayer (redémarrer si nécessaire) :
**sudo apt-get -y install openssh-client openssh-server**

Warning: the SSH server might be installed and working but for security reasons (e.g. well-known default password), it might refuse connections using password authentication. In that case, either check if **PasswordAuthentication** setting is enabled in **/etc/sshd_config** (or in any included file) or see part **Ne plus avoir besoin de taper de mot de passe en SSH**.

Si la commande **raspi-config** n'existe pas, chercher comment l'installer en regardant dans https://www.ensta-bretagne.fr/lebars/Share/unattended.sh , à défaut la plupart des réglages peuvent être changés en éditant **/boot/firmware/config.txt**.

Facultatif : check e.g. with **top** command if the pi has at least 4 GB of RAM or if there is swap, otherwise configure a swap file with **sudo apt install dphys-swapfile**.

RAM dedicated to GPU could be increased:
**sudo nano /boot/firmware/config.txt**
and in **[all]** section add
**gpu_mem=256**
(check also settings in included files, if any).

## 3.4 Facultatif : GUI et NoMachine

Pour Ubuntu Server, il devrait être possible d'installer les paquets nécessaires pour obtenir l'interface graphique d'un Ubuntu version Destkop (cependant elle risque d'être assez lente et saccadée) :
**sudo apt-get install ubuntu-desktop**

**L'installation de ce paquet prendra environ 30 min.**

Pendant ce temps, il peut être éventuellement possible de faire certains éléments des parties suivantes qui n'interfèrent a priori pas avec les commandes **apt** ou **dpkg**…

If GUI looks too slow, changing the display manager may help (not for pi4 with Ubuntu 22.04?):
**sudo apt-get install lightdm**
**sudo dpkg-reconfigure gdm3**
and choose **lightdm**.

Pour Ubuntu Server et même après installation du paquet ubuntu-desktop et redémarrage, il se peut que le réseau ne puisse pas être configuré facilement via l'interface graphique à cause de Netplan, voir la section évoquant NetworkManager dans https://www.ensta-bretagne.fr/lebars/Share/Ubuntu.txt pour essayer de corriger cela.
Il peut aussi être intéressant d'activer l'autologin dans **Settings\Users**.

Pour pouvoir configurer la Raspberry Pi plus confortablement si elle a une GUI, on peut installer **NoMachine**, qui permettra d'accéder au bureau à distance avec le protocole NX. Aller sur https://www.nomachine.com/download pour télécharger et installer sur la Raspberry Pi la partie serveur. En parallèle sur le PC de salle info, installer **NoMachine Enterprise Client** (voir https://www.nomachine.com/download-enterprise ) dans **C:\Temp** et trouver comment l'utiliser pour se connecter au bureau de la Raspberry Pi…

Sur la Raspberry Pi, il est possible que le clavier ne soit plus reconnu avec la bonne disposition après connexion au bureau à distance. Une solution possible est d'aller dans **Control Center\Keyboard\Layouts** et d'ajouter le clavier **English (US)** pour par la suite pouvoir confirmer le bon choix de clavier dans la barre de notification (il faut parfois recliquer dessus).

You might want to change default framebuffer resolution e.g. **framebuffer_width=1024** and **framebuffer_height=768** when no screen is connected, and if needed enable **hdmi_force_hotplug=1** in **/boot/firmware/config.txt**.

## 3.5 Manipulations avec SFTP, SCP

Lancer **FileZilla** depuis un PC et trouver comment se connecter en SFTP à la pi pour copier des fichiers dans les 2 directions. Faire de même via terminal avec la commande SCP.

## 3.6 Facultatif : Ne plus avoir besoin de taper de mot de passe en SSH

Si on se connecte souvent via SSH depuis un PC considéré comme « sûr », il est possible d'éviter de taper son mot de passe à chaque connexion en indiquant au serveur SSH sa « **public key** » et en s'assurant que le serveur est bien un « **known host** ».

Depuis le PC client, vérifier dans **~/.ssh** s'il n'y a pas déjà un fichier **id_rsa.pub**, sinon exécuter :
**ssh-keygen -t rsa -b 2048**
pour générer une clé publique dans le fichier **id_rsa.pub**. L'envoyer au serveur avec :
**ssh-copy-id -i .ssh/id_rsa.pub pi@pi**
Si la commande **ssh-copy-id** n'est pas disponible, on peut faire :
**scp .ssh/id_rsa.pub pi@pi:Downloads**
**ssh pi@pi**
**cat Downloads/id_rsa.pub>>.ssh/authorized_keys**
**rm Downloads/id_rsa.pub**

Si de plus l'authentification par mot de passe est désactivée (see **PasswordAuthentication** setting in **/etc/sshd_config** or in any included file), il faudra rajouter la clé publique au fichier **.ssh/authorized_keys** du serveur par d'autres moyens.

En revenant sur un terminal du PC client, vérifier qu'aucun mot de passe n'est demandé avec :
**ssh pi@pi**
(normalement la toute première fois qu'on se connecte en SSH, le client demande si on veut rajouter le serveur dans le fichier **~/.ssh/known_hosts**, l'idée étant de vérifier si le serveur auquel on se connecte un bien un serveur habituel, et non un pirate qui se fait passer pour le serveur avec le même nom réseau, adresse IP, etc.).

Note : si les droits d'accès sont changés au niveau du dossier utilisateur ou du dossier .ssh ou ses fichiers, la connexion SSH peut ne plus fonctionner.

## 3.7 Facultatif : accès aux lecteurs réseaux de l'école

Pour se connecter au dossier **public/share** de l'école et accéder à vos dossiers réseaux habituels (nécessite d'être connecté à un réseau où ils sont accessibles), s'inspirer de https://www.ensta-bretagne.fr/lebars/Share/remote_ensieta.sh : changer **lebarsfa** par votre **identifiant** et **utilisateurs1** par utilisateurs2,3,4,5,6,21,22,23 ou 24 (à tester ou chercher dans les propriétés de **U:** sur un PC de salle info sous Windows), de plus **webperso**, **recherche** et **labos** ne sont peut-être pas accessibles par des étudiants. Les identifiants à utiliser seront vos identifiants habituels sur les PC de l'école. C'est normal si le mot de passe ne s'affiche pas du tout à mesure qu'on le tape.

**Note de sécurité informatique : toujours être vigilants quand on vous demande de rentrer vos identifiants dans un contexte inhabituel !**

## 3.8 Réglages divers et installation d'applications

Voici quelques applications/prérequis souvent utiles en robotique (voir aussi https://www.ensta-bretagne.fr/lebars/Share/unattended.sh ) :

```
# Might prompt to accept a license or change configuration files...
sudo apt-get -y install ttf-mscorefonts-installer davfs2 samba
# Main packages...
sudo apt-get -y install \
 ttf-mscorefonts-installer davfs2 samba nfs-common cifs-utils smbclient wget curl net-tools nmap wakeonlan openvpn sshfs \
 build-essential autoconf automake libtool gdbserver \
 zip unzip p7zip-full xz-utils lzma \
 subversion git git-core cvs valgrind electric-fence dos2unix dialog doxygen xdot texinfo libacl1-dev libattr1-dev augeas-tools whois debconf-utils dpkg-dev libcppunit-dev \
 libmodbus-dev libusb-1.0-0-dev gpsd-clients ser2net socat librxtx-java \
 minicom gtkterm setserial i2c-tools screen tmux gpart gparted partimage mtools fsarchiver zerofree bmap-tools gddrescue dcfldd procinfo iotop htop atop sysstat \
 apt-transport-https ca-certificates gnupg gnupg-agent dirmngr software-properties-common \
 cmake cmake-gui cmake-curses-gui libcurl4-openssl-dev libavformat-dev libswscale-dev pkg-config \
 bison flex m4 libgtkglext1-dev libplib-dev libgtk2.0-dev libncurses-dev freeglut3-dev \
 vlc filezilla audacity usb-modeswitch usb-modeswitch-data
```

**L'installation de ces paquets prendra environ 15 min.**

Pendant ce temps, il est possible de faire divers réglages inspirés de l'installation d'Ubuntu fournie pour les PC (voir https://www.ensta-bretagne.fr/lebars/Share/Ubuntu.txt à titre indicatif, qui fait notamment appel à https://www.ensta-bretagne.fr/lebars/Share/unattended.sh , ainsi que certains réglages spécifiques aux Raspberry Pi sur https://www.ensta-bretagne.fr/lebars/Share/Raspbian.txt ). En voici une sélection.

**# Disable sudo password...**
**sudo visudo**
**# Add at the end :**
**user ALL=NOPASSWD: ALL**

**Control Center\Personal\Screensaver** : disable screen saver and lock screen

**nautilus**: in **Edit\Preferences\Behavior\Executable Text Files** set **Ask each time** enable, show hidden files (in menu or buttons), show link creation action
**gnome-terminal**: in **Edit\Preferences**, set unlimited history scrolling...

Pour permettre l'utilisation des ports série :
**sudo adduser $USER dialout**
# If problems with COM ports?
#sudo apt-get remove modemmanager

Après installation des paquets, changer le groupe de travail réseau WORKGROUP à ROBOTICS :
**sudo nano /etc/samba/smb.conf**

Pour y rajouter un dossier partagé, s'inspirer de https://www.ensta-bretagne.fr/lebars/Share/smb_Share.txt et faire :
**mkdir ~/Share**
**sudo chmod -R 777 ~/Share**
**sudo /etc/init.d/samba restart**

Prérequis pour pouvoir utiliser les programmes de configuration des cartes Pololu Maestro (commande d'ESC et servomoteurs, changer les réglages de proxy si nécessaire, optionnel) :
**sudo apt-key adv --keyserver-options http-proxy=$http_proxy --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 3FA7E0328081BFF6A14DA29AA6A19B38D3D831EF**
**echo "deb https://download.mono-project.com/repo/ubuntu stable-$(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/mono-official-stable.list**
**sudo apt-get update**
**sudo apt-get -y install mono-complete**

**L'installation de ces paquets prendra environ 10 min.**

Configuration de la Pololu Maestro (optionnel) :
**wget https://www.pololu.com/file/download/maestro-linux-150116.tar.gz?file_id=0J315**
**mv maestro-linux-150116.tar.gz?file_id=0J315 maestro-linux-150116.tar.gz**
**tar xvfz maestro-linux-150116.tar.gz**
**cd maestro-linux**

**sudo cp 99-pololu.rules /etc/udev/rules.d/**
**#sudo mono ./UscCmd --servo 0,4000**
**#sudo mono ./UscCmd --servo 0,8000**
**#sudo mono ./MaestroControlCenter**

**pigpio** library can generate good software PWM (as well as simple digital inputs and outputs) from almost any pin of a pi, check with a servomotor.

MAVLink (programmation de la communication avec l'ArduPilot, optionnel) :
**wget https://github.com/mavlink/c_library_v2/archive/master.zip**
**unzip master.zip**
**sudo mv -f c_library_v2-master/ /usr/local/include/mavlink**

Pour information, les paquets .deb téléchargés par la commande **apt-get** sont parfois sauvegardés dans **/var/cache/apt/archives** (if not, **echo 'Binary::apt::APT::Keep-Downloaded-Packages "1";' | sudo tee /etc/apt/apt.conf.d/10apt-keep-download**). Si on souhaite libérer de l'espace disque, les commandes :
**sudo apt-get clean**
**sudo apt-get autoremove**
peuvent les effacer, ainsi que les paquets inutiles.

De plus, il n'est parfois pas toujours possible d'installer tous les paquets qu'on souhaite en une seule commande. Décomposer en plusieurs commandes peut parfois résoudre certains types de problèmes d'installation.

## 3.9 Facultatif : transmission vidéo avec gstreamer

**gstreamer** (**sudo apt-get -y install gstreamer1.0-libav gstreamer1.0-plugins-good gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly gstreamer1.0-plugins-base-apps libx264-dev libjpeg-dev**) permet de faire différents types de conversion et transmissions vidéos, sous Windows ou Linux, et prend un charge un grand nombre de codecs, dont certains sont purement logiciels et d'autres utilisent des fonctions spécifiques matérielles de certains CPU et GPU. La Raspberry Pi est notamment supportée avec sa caméra optionnelle (**attention à ne pas confondre le connecteur CSI avec DSI sur la pi**) et les fonctions OpenMAX (OMX) de la GPU. En particulier, l'OMX de la Raspberry est capable de faire de la compression/décompression JPEG et H264 bien plus rapidement que si on utilisait juste que la CPU. **gstreamer** est très flexible mais du coup assez complexe (parfois peut-être aussi à cause de bugs…) à utiliser au premier abord : toute la chaine de récupération d'un flux vidéo, réglages de paramètres éventuels, conversions de formats, envoi dans une certaine destination doit être spécifiée dans ses paramètres en ligne de commande, on appelle ceci souvent le **pipeline**.

La commande **gstreamer** que l'on utilise le plus souvent est **gst-launch-1.0**. Voici un exemple pouvant a priori fonctionner sur la plupart des ordinateurs (si **x264enc** n'est pas reconnu, essayer de trouver une alternative, e.g. avec **gst-inspect-1.0 | grep 264**, voir aussi https://qengineering.eu/install-gstreamer-1.18-on-raspberry-pi-4.html ) :
**gst-launch-1.0 -e -v videotestsrc ! x264enc ! filesink location=test.avi**
**-e** est un paramètre de **gst-launch-1.0** qui permet une finalisation correcte du fichier video lorsqu'on fait CTRL+C pour arrêter, **-v** affiche des informations de status (on peut notamment rajouter **GST_DEBUG=2** juste avant **gst-launch-1.0** pour augmenter le nombre

d'informations). Le pipeline commence avec la spécification d'une source (**src**), **videotestsrc**, qui est un flux video auto-généré pour les tests, le symbole **!** est comme une sorte de connecteur branchant la sortie de ce qui est à gauche à l'entrée de ce qui est à droite. La partie gauche doit pouvoir notamment fonctionner comme une source (**src**), alors que la partie droite doit pouvoir fonctionner notamment comme une destination (**sink**). Si de plus le format de la sortie n'est pas compatible avec les différents formats d'entrée supportés par ce qui est à droite, un message d'erreur de type **erroneous pipeline: could not link XXX to YYY** s'affichera probablement, dans ce cas il faut essayer de voir si la **src** a des paramètres permettant de changer son format (vérifier aussi les paramètres de la **sink** au cas où…), sinon on peut essayer de rajouter **! videoconvert ! videoscale** entre les 2, mais il se peut que ce soit couteux en utilisation processeur. Ici, en lançant la commande on peut déduire que le flux video sortant de **videotestsrc** est de type **video/x-raw** avec code de codec FOURCC **Y444**. **x264enc** semble lui pouvoir fonctionner en **sink** avec le même type de flux et comme **src** de type **video/x-h264**. **x264enc** est a priori un codec purement logiciel permettant de convertir un flux video brut en flux **H264**, on peut consulter sa doc officielle sur https://gstreamer.freedesktop.org/documentation/x264/ (ou utiliser la commande **gst-inspect-1.0 x264enc**) pour voir ce qu'il peut prendre en entrée et ses paramètres possibles (e.g. **tune=zerolatency bitrate=512** s'il y a trop de latence et qu'on veut aussi limiter le débit, à spécifier entre **x264enc** et le prochain **!**, suivi du format e.g. **video/x-h264, profile=constrained-baseline** peut être parfois nécessaire…). **filesink** est ensuite une **sink** qui a pour paramètre obligatoire **location** pour spécifier un nom de fichier.

Pour tenter de forcer un format de flux particulier entre **videotestsrc** et **x264enc**, on peut faire :
**gst-launch-1.0 -e -v videotestsrc ! video/x-raw, format=BGR, width=320, height=240 ! x264enc ! filesink location=test.avi**
On devrait alors recevoir un message d'erreur assez explicite. Vérifier que le rajout de **videoconvert** résout le problème (au prix d'utilisation supplémentaire probablement inutile dans ce cas du processeur).

Bien que cette commande nous génère bien un fichier avec de la vidéo dedans, nous ne respectons pas ici vraiment le format **avi** car à part dans l'extension du fichier, à aucun moment on ne spécifie qu'il faut faire une conversion au format **avi** (et sur https://gstreamer.freedesktop.org/documentation/coreelements , rien ne dit que **filesink** prend en compte l'extension de fichier pour faire une conversion automatique). En effet, **VLC** ne semble pas le lire, par contre **Media Player Classic** y arrive parfois mais ne respecte pas l'échelle de temps … Il faut probablement au moins rajouter **h264parse ! avimux** entre **x264enc** et **filesink** pour augmenter les chances que les lecteurs vidéo classiques interprètent bien le fichier.

Sur une pi, la commande devrait marcher mais prendre beaucoup de processeur. Pour utiliser la GPU à la place, remplacer **x264enc** par **omxh264enc** (ou par un autre encodeur H264 (**v4l2h264enc ! 'video/x-h264,level=(string)4'** pour Raspberry Pi OS Bullseye), vérifier aussi si les formats d'entrée-sortie sont toujours compatibles).

Sur portable ou avec une pi avec écran, essayer de remplacer le **filesink** par **autovideosink** pour afficher le flux vidéo en direct (consulter la doc si besoin). Il faudra a priori supprimer tout ce qui est relatif à la compression **H264** et conteneur **avi**, ou rajouter le nécessaire pour la décompression. Côté **src**, sous Windows on peut remplacer **videotestsrc** par **ksvideosrc device-index=0** pour récupérer le flux d'une webcam, sous Linux ce sera **v4l2src**

**device=/dev/video0**, si on a la caméra intégrée à la Raspberry il y a aussi **rpicamsrc** (si https://github.com/thaytan/gst-rpicamsrc est installé, ou **libcamerasrc** pour Raspberry Pi OS Bullseye)

En robotique, envoyer à distance le flux vidéo d'une webcam embarquée avec une faible latence, un faible débit et une bonne qualité est un besoin courant, que **gstreamer** peut souvent gérer en nous évitant d'apprendre à coder/décoder divers formats et codecs vidéos complexes. En s'aidant de la partie **Remote video and telemetry** dans https://www.ensta-bretagne.fr/lebars/tutorials/TD_ARDU_PARROT.pdf , mettre en place une transmission vidéo temps réel avec enregistrement dans un fichier. Dans ce document, **QGroundControl** est un programme tout fait qui attend un flux compressé selon le protocole **H264 via RTP** sur le port **UDP 5600**. A la place de **QGroundControl**, on devrait pouvoir aussi afficher le flux vidéo avec :
**gst-launch-1.0 -e -v udpsrc port=5600 close-socket=false multicast-iface=false auto-multicast=true ! application/x-rtp, payload=96 ! rtpjitterbuffer ! rtph264depay ! avdec_h264 ! queue ! autovideosink**

Note: dans certaines situations, il semblerait que **gstreamer** ne fonctionne pas avec localhost…

## 3.10 Facultatif : accès à distance avec le protocole RDP

On peut aussi se connecter à distance avec le protocole RDP (en pratique, il n'est pas toujours bien supporté sous Linux) mais l'avantage est qu'un client RDP est présent par défaut sous Windows. Voir https://www.ensta-bretagne.fr/lebars/Share/Ubuntu.txt pour installer le nécessaire si besoin et sur le PC de salle info, exécuter **mststc /admin** et taper **PC-NAME** dans **Computer** (remplacer **PC-NAME** par le nom réseau défini plus haut, éventuellement rajouter le numéro de port séparé par « : » si le serveur n'est pas configuré sur le port par défaut).

Note de sécurité informatique : sur un PC partagé, il peut être nécessaire d'effacer les identifiants enregistrés. Certains (lecteurs réseaux, RDP) sont regroupés sous Windows et gérés en tapant la commande **control userpasswords2**, dans **Advanced\Manage Passwords**.

**Some other activities that were previously in this document have been moved to https://www.ensta-bretagne.fr/lebars/tutorials/TD_bridge_route.pdf .**