

Rapport PAN2

Choix du materiel

- **Ecran** (InnoLux N133HSE): Pour l'écran nous avons choisi un écran 13.3" (non tactile), nous avons estimé la taille suffisante pour un confort de lecture à distance raisonnable de la borne. De plus cet écran présente un système de montage à vis qui facilitera son intégration dans la borne.
- **Camera** : Nous avons le choix entre 2 camera
 - Une webcam Logitech c525
 - Grand choix de résolutions avec un framerate élevé par exemple 640x480@30fps, 960x720@30fps, 1920x1080@5fps
 - Mais champs de vision beaucoup trop faible pour notre application
 - Une camera de surveillance grand angle de la marque ELP
 - Choix de résolutions restreint, mais utilisable dans des bonnes conditions de luminosité

```
[1]: 'YUYV' (YUYV 4:2:2)
    Size: Discrete 1600x1200
        Interval: Discrete 0.200s (5.000 fps)
        Interval: Discrete 0.200s (5.000 fps)
    Size: Discrete 2592x1944
        Interval: Discrete 0.333s (3.000 fps)
    Size: Discrete 2048x1536
        Interval: Discrete 0.250s (4.000 fps)
    Size: Discrete 1920x1080
        Interval: Discrete 0.200s (5.000 fps)
    Size: Discrete 1280x1024
        Interval: Discrete 0.111s (9.000 fps)
    Size: Discrete 1280x720
        Interval: Discrete 0.200s (5.000 fps)
    Size: Discrete 1024x768
        Interval: Discrete 0.100s (10.000 fps)
    Size: Discrete 800x600
        Interval: Discrete 0.050s (20.000 fps)
    Size: Discrete 640x480
        Interval: Discrete 0.033s (30.000 fps)
    Size: Discrete 1600x1200
        Interval: Discrete 0.200s (5.000 fps)
        Interval: Discrete 0.200s (5.000 fps)
```

- Champs de vision adapté a notre utilisation
 - Cette camera apporte des distortions importantes sur les bords, cela complexifie la reconnaissance d'image.
- Nous avons choisi la deuxième camera pour les raisons citées ci-dessus
- **Ordinateur** : Nous avons à notre disposition un AMD Cubi doté de 4Go de RAM, 128Go de SSD et un Intel Core i3 5005U. L'objectif du reste de ce document est d'évaluer si les performances de cet ordinateur sont suffisantes pour notre application.

Choix du système d'exploitation

Nous avons choisi d'installer debian 11, une distribution de linux légère avec laquelle nous étions familiers. Pour l'environnement de bureau nous avons choisi LXDE là aussi pour le minimalisme qu'il présente au vu des performances de la machine. Nous avons par la suite désinstallé tout les packets installé par défaut par LXDE dont nous n'avions pas besoin. (Par exemple les jeux, la calculatrice, etc.)

Benchmarking

Pour évaluer la capacité du matériel à supporter la charge de notre application, nous avons exécuté en parallèle les applications qui seront utilisées pour le produit final ou une application aux besoins équivalents quand cela n'était pas possible. De ce fait ce benchmark n'est pas exact mais nous donnera un ordre d'idée sur les besoins de notre projet.

Liste des modules à exécuter

Pour ce faire nous avons mis en place un container Docker par module de notre application :

- **db** : un serveur mysql basique pour la base de donnée
- **phpmyadmin** : l'interface phpmyadmin pour gérer la base de donnée. Ce container n'est pas absolument nécessaire au produit final mais permet d'administrer facilement la base de donnée
- **review_api** : un serveur express pour servir l'api permettant de récupérer et ajouter des avis dans la base de donnée, ainsi que de calculer les statistiques.
- **interface_borne**: un serveur apache2 permettant de servir l'interface graphique de la borne.
- **interface_admin**: idem pour l'interface graphique, ce serveur pourrait être fusionné avec **interface_borne** mais dans le cadre de ce benchmarking on les gardera séparés.
- **backend_reconnaissance**: Ce container s'occupera de la reconnaissance audio et video de la borne. Cependant ces deux processus ne seront jamais actifs en même temps, de plus la reconnaissance d'image sera la plus coûteuse en terme de puissance de calcul. C'est pour cela qu'ici nous avons uniquement utilisé mediapipe hands (bibliothèque python de reconnaissance de mains) avec une implémentation de la communication avec l'interface de la borne comme programme équivalent.
- **video_loopback** : ce container sert à contourner un problème que nous avons rencontré avec la gestion des camera par linux. En effet seul un programme peut accéder au flux d'une camera à la fois. C'est pour cela que nous avons utilisé **v4l2loopback** avec **ffmpeg** pour dupliquer le flux de notre camera dans 2 cameras virtuelles.

En parallèle firefox est ouvert pour afficher l'interface graphique de la borne.

Résultats

En réglant la camera a 640x480@30fps, aucune perte d'image n'est observée dans le retour vidéo dans firefox.

Nous avons observé la capacité de Mediapipe l'image de la camera et de la communiquer à l'interface web qui affiche le résultat. Nous avons obtenu les résultats suivants pour 2 modes de gestion de fréquence du processeur (la gestions automatique de base schedutil et performance qui utilise la fréquence maximale)

CPU Scheduler	FPS moyens	Ecart type
schedutil	10.4	1.4
performance	10.1	0.9

Ce taux de rafraichissement est suffisant pour que l'application paraisse relativement réactive à l'utilisateur, bien que cela ne soit pas du temps réel.

Pendant ce temps l'api de traitement des avis et la base de donnée fonctionnent correctement en affichant une latence de 8ms pour une récupération de la liste d'avis.

Pendant ce temps l'utilisation du processeur qui varie de 250% à 280% (sur 400% pour les 4 coeurs) et une utilisation de la RAM de 50% (1.96Go) ce qui nous laisse de la marge en cas d'ajout imprévu.

Les processus utilisant le plus de CPU sont la reconnaissance d'image (70%) et firefox pour afficher l'interface de la borne (70-80%). En cas de besoin ces valeurs pourront être diminuées au prix de la fluidité du retour vidéo. Pour la RAM c'est le serveur mysql (10%) et firefox (10%) qui consomment le plus.

Pour ce qui est de la température, comme la borne sera dans un environnement fermé, il était important de tester le bon fonctionnement du materiel dans ces conditions. Nous avons laissé tourner l'application pendant 2h dans une boîte en carton fermée. Au début du test la température du processeur était de 50°C, au bout de 2h la température était montée à 70°C, ce qui reste assez faible pour ne pas limiter les performances du CPU.

```
$ sensors
acpitz-acpi-0
Adapter: ACPI interface
temp1:      +27.8°C  (crit = +110.0°C)
temp2:      +29.8°C  (crit = +110.0°C)

coretemp-isa-0000
Adapter: ISA adapter
Package id 0: +69.0°C  (high = +105.0°C, crit = +105.0°C)
Core 0:      +69.0°C  (high = +105.0°C, crit = +105.0°C)
Core 1:      +68.0°C  (high = +105.0°C, crit = +105.0°C)
```

Impact de l'utilisation de Docker

Nous nous sommes également posé la question de l'impact de l'utilisation de docker dans les performances de notre projet. Pour mesurer cela, nous avons effectué des benchmark directement sur le système puis dans un container Docker pour mesurer la différence. Nous avons utilisé sysbench pour évaluer les performances du CPU, de la RAM et du disque (écriture/lecture aléatoire).

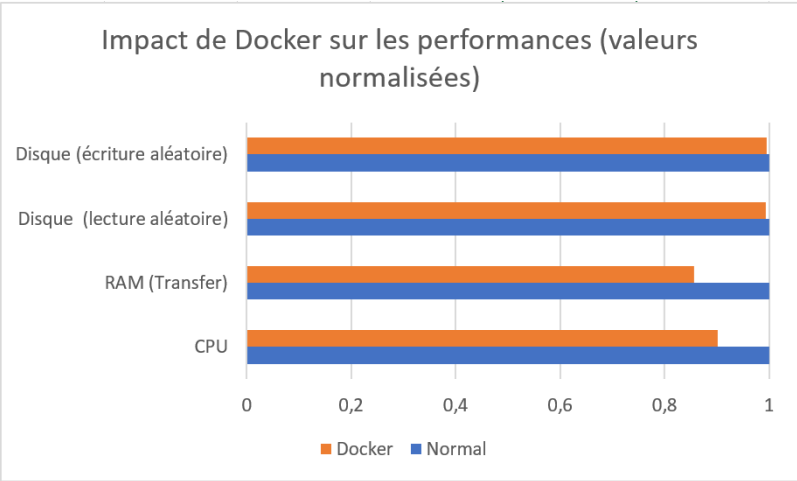
Le script permettant de faire le benchmark

```
sysbench --test=cpu run >>sysbench.log
sysbench --test=memory run >>sysbench.log
sysbench --test=fileio --file-test-mode=rndrw prepare
sysbench --test=fileio --file-test-mode=rndrw run >>sysbench.log
sysbench --test=fileio cleanup
```

Le Dockerfile du container dans lequel nous avons exectué le même script

```
FROM alpine:latest
RUN apk add --no-cache sysbench
WORKDIR /app
COPY benchmark_script.sh /app/benchmark_script.sh
CMD ["sh", "benchmark_script.sh"]
```

Les résultats de ce test on permis de conclure que l'impact de docker était négligeable.



Type	CPU (Evts/s)	RAM (Mbi/s)	Disque lecture(Mbi/s)	Disque écriture (Mbi/s)
Normal	613.14	3259.84	12.34	8.22
Docker	552.76	2794.21	12.27	8.18

Conclusion

Au vu des tests effectués le materiel dont nous disposons semble adapté à notre projet. Il serait cependant possible de réduire l'utilisation faite de l'ordinateur embarqué dans la borne en déplaçant la partie

stockage et traitement des avis sur un autre serveur.