

# Stéréovision: Guide utilisateur

Annexe au travail d'automne

Sol Rosca (INF3b)

Automne 2019

# Table des matières

0.1. Matériel .....	1
0.2. Utilisation .....	3
0.2.1. Dépendances .....	3
0.2.2. Installation .....	3
0.2.3. Configuration .....	3
0.2.4. CLI .....	4
0.2.5. GUI .....	9

## 0.1. Matériel

Ce chapitre décrit le matériel utilisé. Il n'est pas nécessaire d'avoir le même, pour avoir un système de vision stéréoscopique fonctionnel ou pour utiliser le logiciel.



Figure 1 Matériel utilisé

- 2 x Logitech c920 HD Pro
- Dell XPS 15 7590 (i7-9750H @ max 4.5ghz)
- Coffee @100% arabica

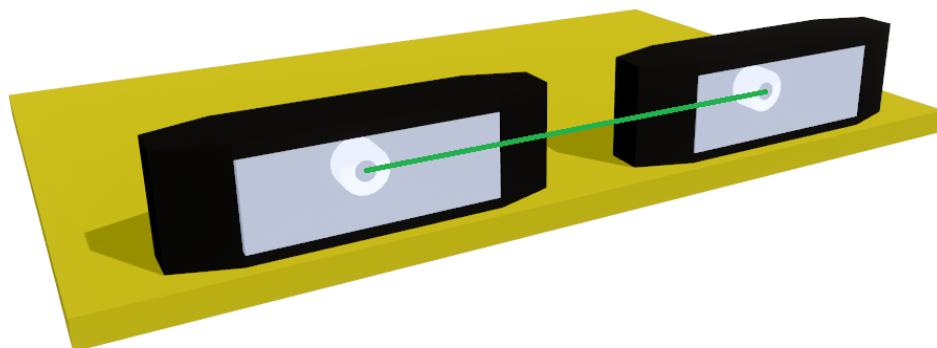
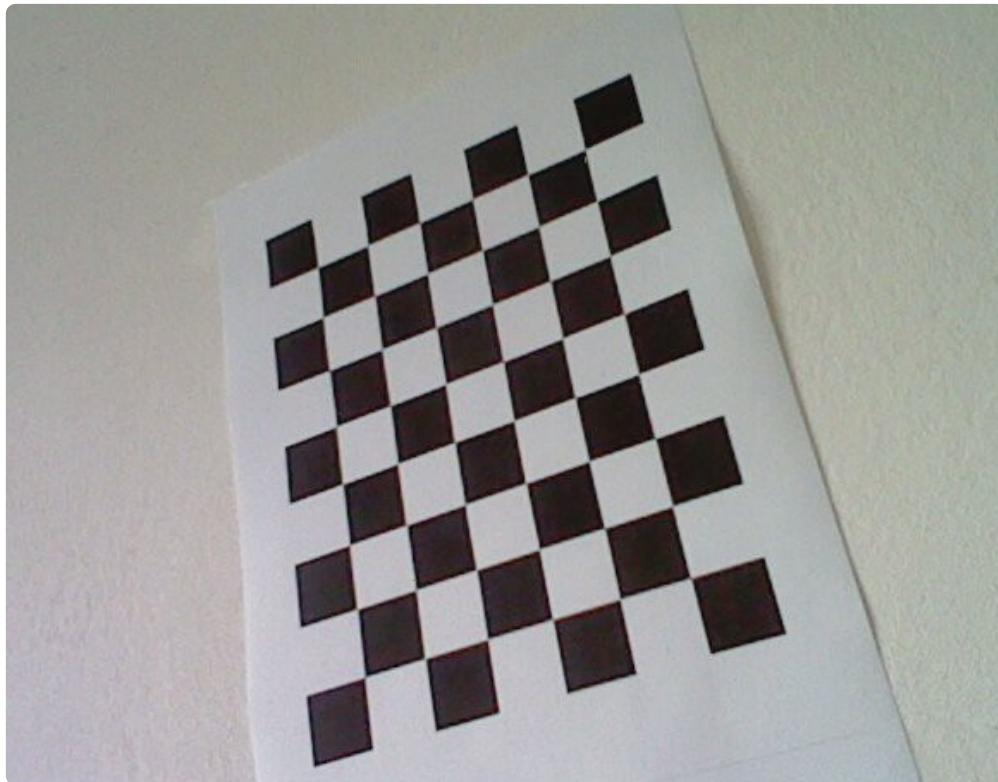


Figure 2 Écart entre les deux capteurs

L'écart de capteur à capteur est de 110 mm (trait vert sur la figure précédente). Cette distance est arbitraire mais les valeurs par défaut de certains paramètres accessibles dans le fichier `public/settings.py` sont calibrées sur cette base.

Une mire de calibration est nécessaire pour calibrer le système:



**Figure 3** Mire de calibration

Pour s'en procurer une, en l'absence d'un jeu d'échec, il est possible de simplement imprimer ce [pdf](#). Il est également conseillé de rigidifier cette mire en la collant sur une surface rigide.

## 0.2. Utilisation

Il est possible d'utiliser le logiciel avec deux interfaces différentes:

- **CLI** qui contient toutes les fonctionnalités du logiciel.
- **GUI** plus facile d'utilisation mais ne permet pas de calibrer les caméras. Elle nécessite une calibration préliminaire à l'aide du CLI.

### 0.2.1. Dépendances

- Python3.7

### 0.2.2. Installation

Le logiciel n'a pas été testé sous Windows et MacOS, aucun système de ce type n'étant disponible sur la machine de l'étudiant et le temps ne permettant pas de faire des tests extensifs sur différents systèmes. Néanmoins, il n'y a pas de raison particulière que ça ne fonctionne pas sur ces OS. Les précautions de base de portabilité ont été prises et généralement elles suffisent à assurer la portabilité du code Python.

Il est vivement conseillé de faire usage d'un environnement virtuel de type [virtualenvwrapper](#) ou [pyenv](#) pour ségréguer les dépendances du projet des dépendances de l'installation système de Python.

1. Téléchargez ou clonez les sources du projet sur [github](#).
2. Après décompression de l'archive, ouvrez un terminal à la racine du projet.
3. (optionnel) Créez un nouvel environnement virtuel et activez le.
4. Installation des dépendances avec `$ pip install -e .`

### 0.2.3. Configuration

Le fichier `public/settings.py` contient les configurations du logiciel. La majorité des valeurs par défaut sont valables mais il est nécessaire d'en adapter certaines:

- **DEVICE** : Les valeurs des champs `left` et `right` correspondent à l'id usb qu'occupent vos caméras.
- **CHESSBOARD** :
  - `rows` correspond au nombre de lignes -1 *nombre de lignes -1*<sup>1</sup>
  - `columns` correspond au nombre de colonnes - 1.
  - `square_size` représente la taille d'un carré en *cm*.

1. en réalité OpenCV compte les intersections et non pas les lignes ou colonnes

## 0.2.4. CLI

Pour lancer le CLI, À la racine du projet:

```
1 | $ python public/stereovision.py
```

Bash

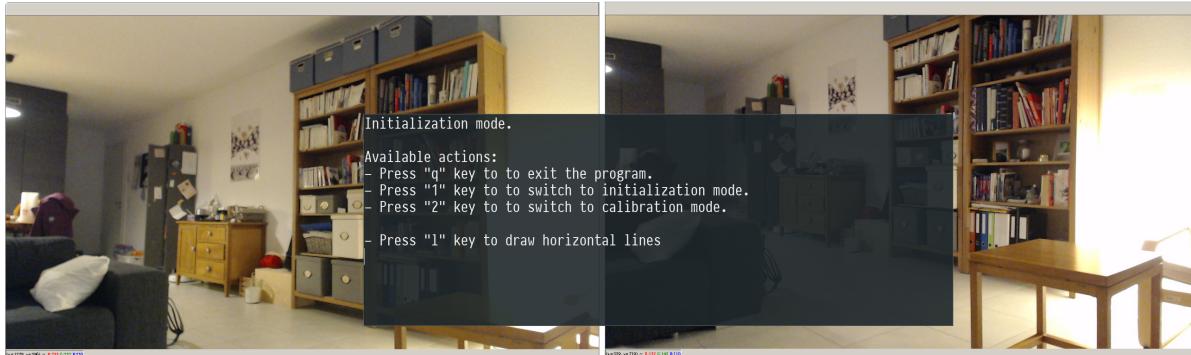


Figure 4 mode: initialization

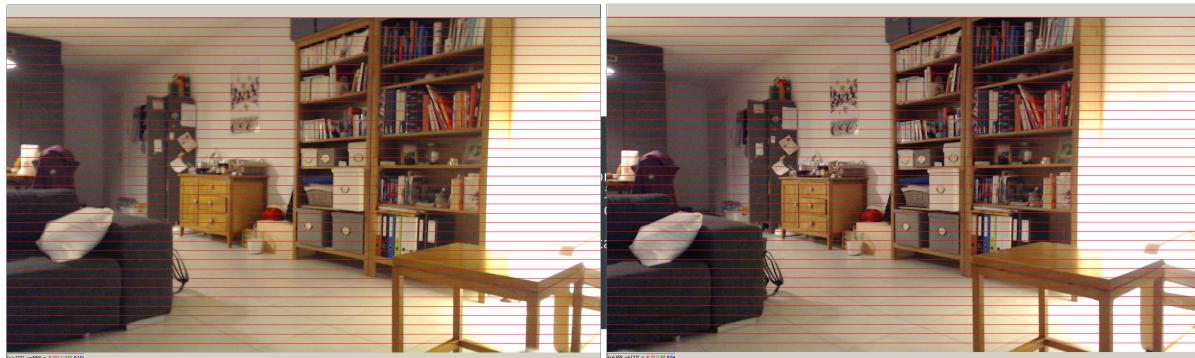
Les 4 actions possibles sont affichés. Les autres modes ne sont pas visibles car l'application ne détecte pas les fichiers contenant les données de calibration. Avant de lancer la procédure de calibration, il est utile de faire un pré-réglage manuellement des caméras. En pressant la touche "l" (L minuscule) du clavier, des lignes apparaissent:



Figure 5 Pré-réglage

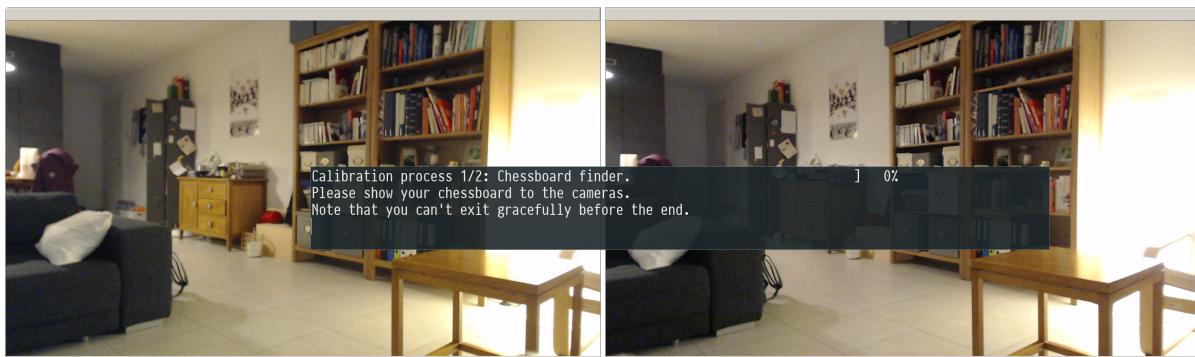
Ces lignes servent à régler grossièrement la position verticale des caméras. Sur la figure précédente, la caméra de gauche devrait être pivotée vers le bas, manuellement.

La figure suivante montre le résultat, après une correction grossière:



**Figure 6** Correction manuelle

Le système est maintenant prêt à être calibré. Pour ce faire, il est nécessaire de changer de mode pour le mode 2, qui est dédié à la calibration, en pressant sur la touche "2" du clavier:



**Figure 7** mode: calibration

Le mode calibration est un mode en deux procédures:

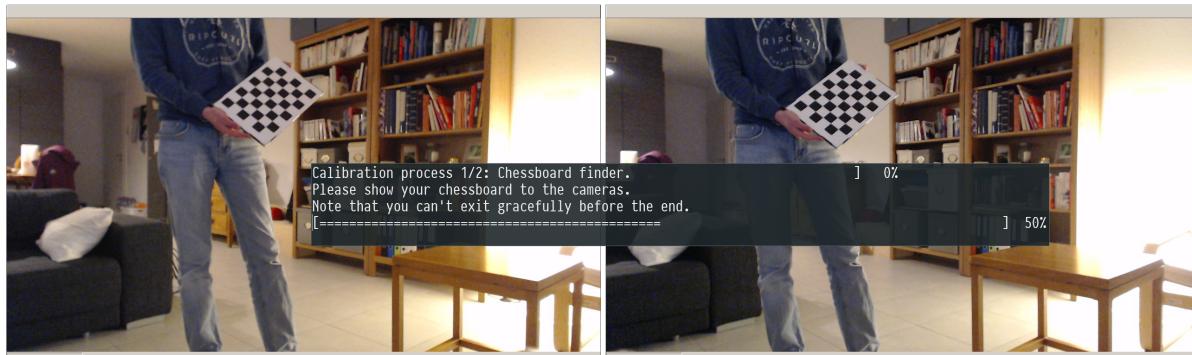
1. Le logiciel va capturer un *certain nombre*<sup>1</sup> de paires d'images.
2. Le logiciel utilise les images pour déterminer un certain nombre de paramètres, notamment les paramètres intrinsèques, les données de distorsion, les matrices fondamentale et essentielle.

Pour effectuer les captures, il est nécessaire de se munir de la mire de calibration et de se balader avec, devant les caméras. Les captures se font automatiquement lors de la détection de la mire dans les deux images. Après une capture, le logiciel laisse 2 secondes à l'utilisateur pour changer de position, avant de recommencer à chercher la mire.

---

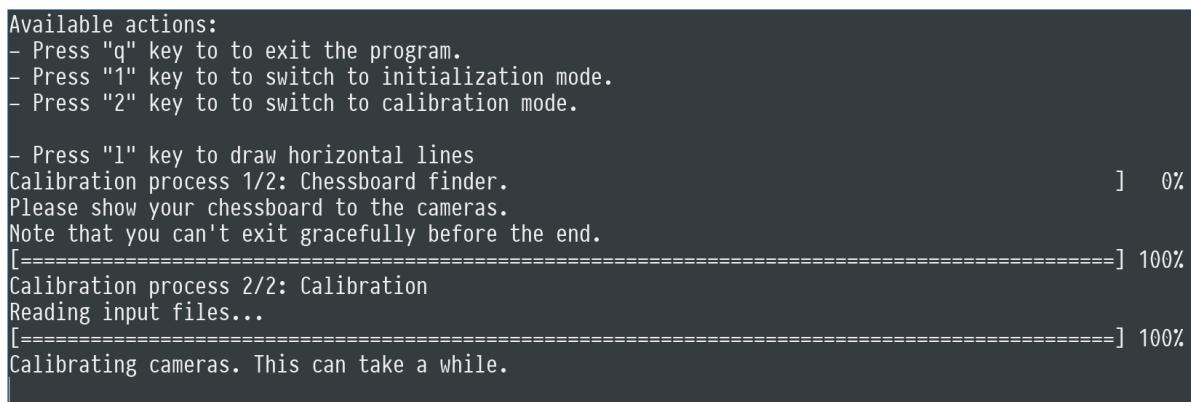
1. définit dans le module public/settings.py

La barre de progression permet de connaître l'état d'avancement de la procédure:



**Figure 8** Capture de la mire

La transition vers la seconde procédure se fait automatiquement lorsque la barre de progression est complète. La seconde procédure ne demande *aucune action de l'utilisateur*<sup>1</sup> et prend un *certain temps*<sup>2</sup> pour faire ses calculs:



**Figure 9** Calcul des données de calibration

Si la première procédure échoue pour une raison ou pour une autre ou si l'utilisateur souhaite faire une nouvelle calibration, il lui suffit de relancer ce mode.

- 
1. Sauf si l'option CALIBRATION.show\_chessboard est à True dans le module public/settings.py
  2. Dépend de la résolution et de la puissance de la machine

Une fois les calculs effectués, l'application change automatiquement de mode pour aller dans le numéro 3, c'est à dire le mode distorsion:



Figure 10 Mode: distorsion

Dans ce mode il est possible d'afficher les lignes d'aide (touche "l") qui permettent de visualiser la qualité de la rectification ainsi que de la suppression de la distorsion. Pour alterner entre images normales et images réctifiées, il est possible de presser la touche "d".

La touche "4" active le mode depth:

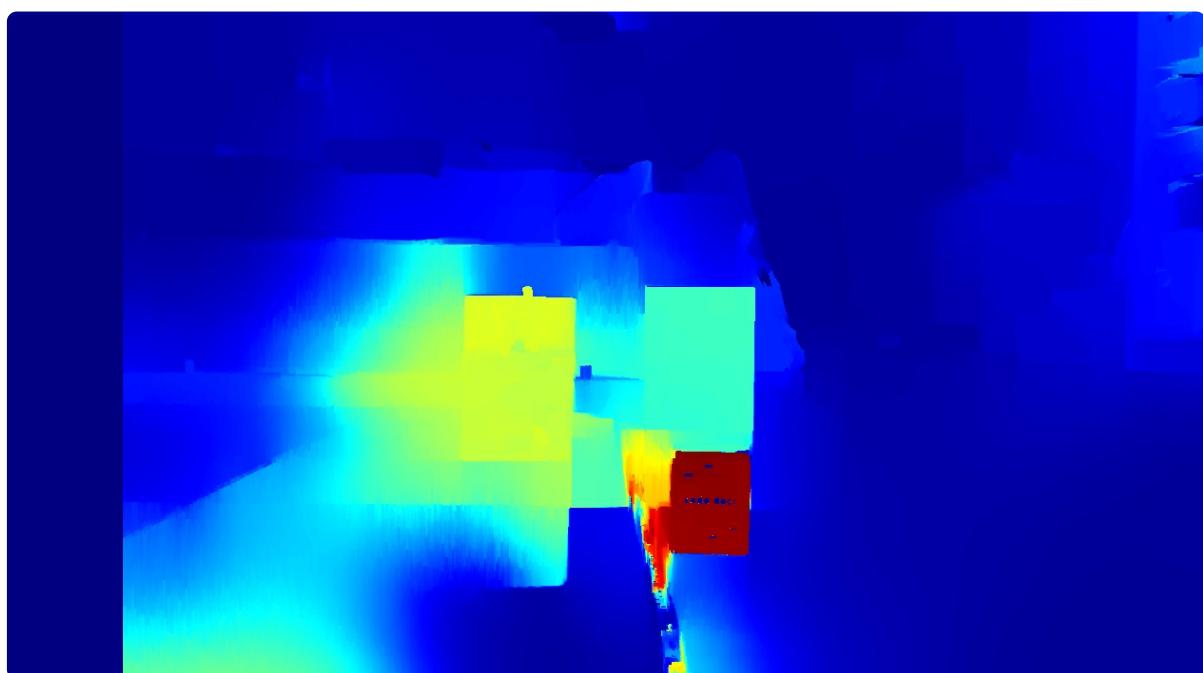


Figure 11 Mode: depth

Dans ce mode, plus la couleur tend vers le rouge, plus l'objet est proche. Et dans la précédente capture (qui correspond à la scène de la **figure 21**), il est facile de voir que l'objet au premier plan est rouge vif, celui au second jaune et celui dans le 3e bleu clair.

Ce mode offre de nombreuses options qui seront détaillées dans le prochain sous-chapitre:

```
Depth mode.

Available actions:
- Press "q" key to exit the program.
- Press "1" key to switch to initialization mode.
- Press "2" key to switch to calibration mode.
- Press "3" key to distortion mode.
- Press "4" key to depth mode

- Press "b" key to change between blockmatcher mode
- Press "d" key to show raw disparity map
- Press "c" key to show fixed and colored disparity map
- Press "w" key to show a WLS filtered map
In this mode you can double click to get a conversion
from disparity to distance. (Needs to be setup)
|
```

**Figure 12** Depth: options

Les dernières lignes de la figure précédente laissent sous entendre qu'il est possible d'obtenir la distance en double cliquant sur un point de l'image. Cette fonctionnalité est expérimentale et est décrite dans le chapitre suivant.

## 0.2.5. GUI

Lancer le programme en mode GUI donne accès à une interface graphique. Pour lancer ce mode, à la racine du projet:

```
1 | $ python public/stereovision_gui.py
```

Bash

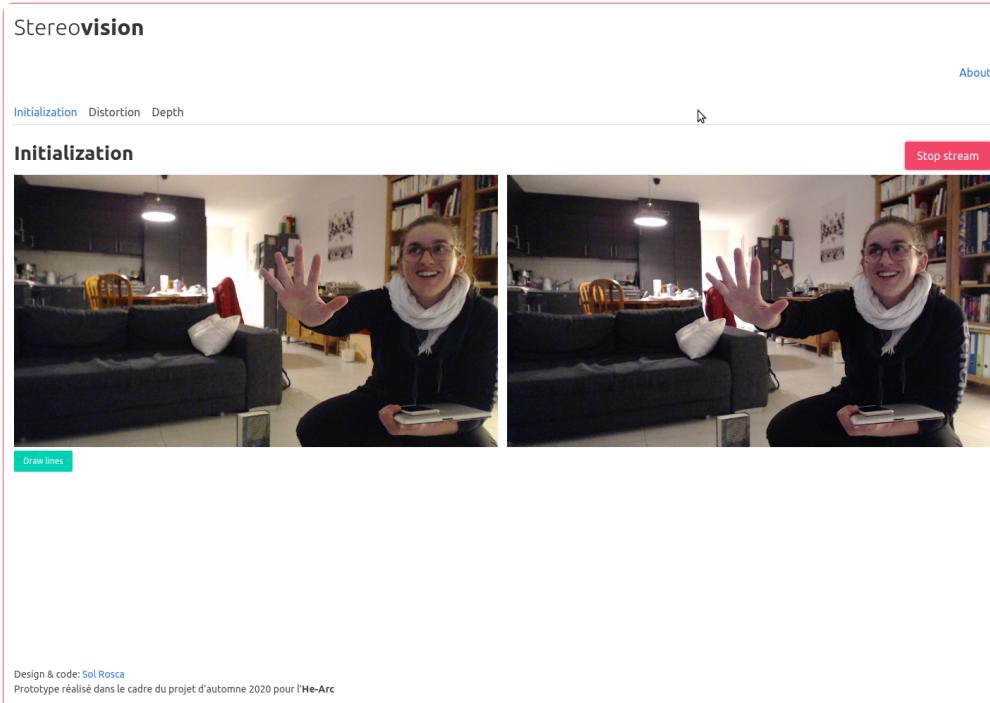
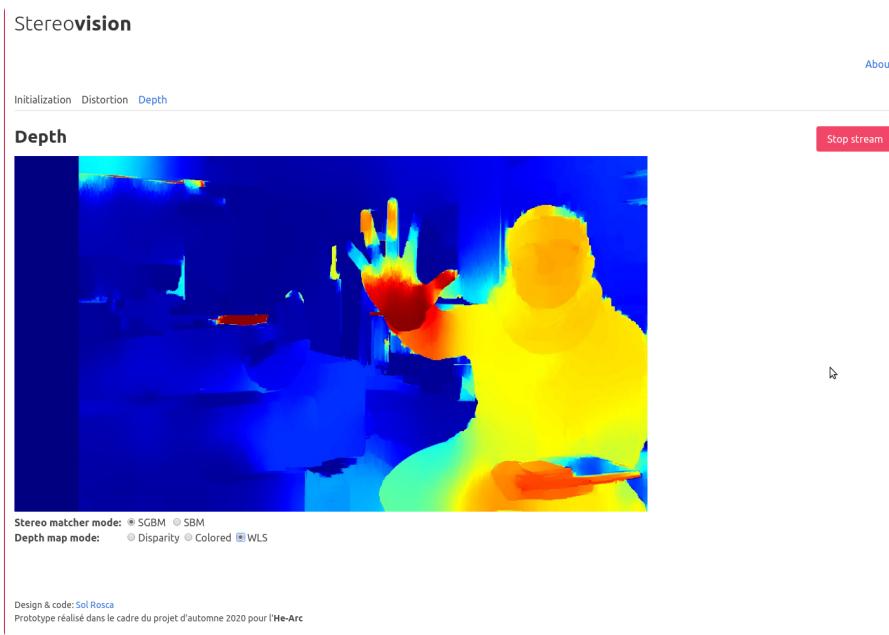


Figure 13 GUI

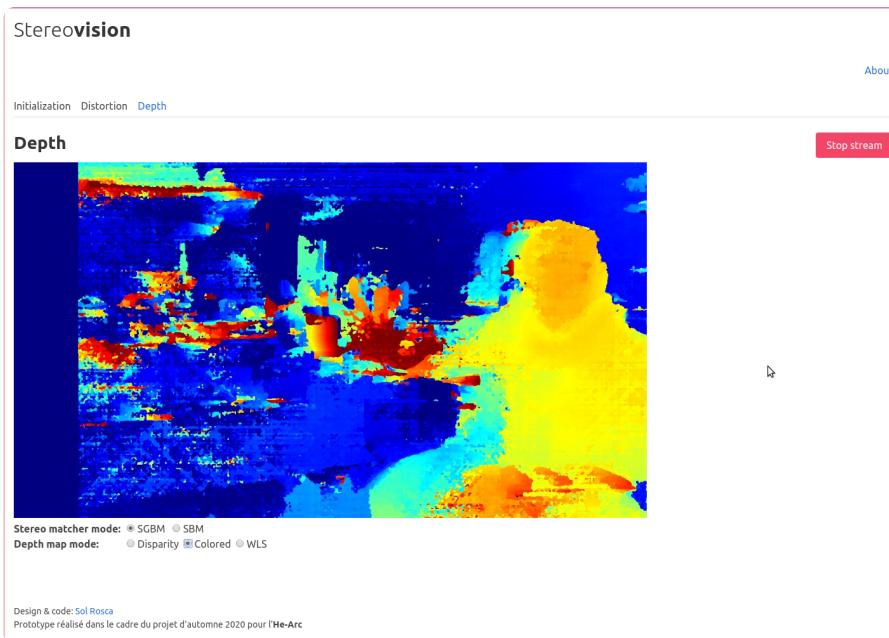
Dans la GUI, on retrouve les modes de l'application sous forme d'onglets, en haut à gauche. Il y en a un de moins qu'en mode CLI car il n'est pas possible pour le moment de calibrer dans ce mode. Dans chaque onglet, en haut à droite se trouve un bouton pour lancer / stopper la capture. Si une capture est en cours et que l'onglet est changé, la capture cesse automatiquement et doit être relancée dans le nouveau mode. Et finalement, en bas à gauche des images, il y a les différentes options du mode courant, qui sont les mêmes que celles du CLI. Sur la précédente figure, on peut voir un bouton pour activer l'affichage des lignes horizontales d'aide.

Dans l'onglet "Depth", on retrouve l'image résultante du traitement sur les deux caméras qui affiche une carte de profondeur. Par défaut c'est le mode WLS qui est activé et qui propose un filtre particulier qui permet de repliquer le résultat de la carte de profondeur sur une image filtrée:



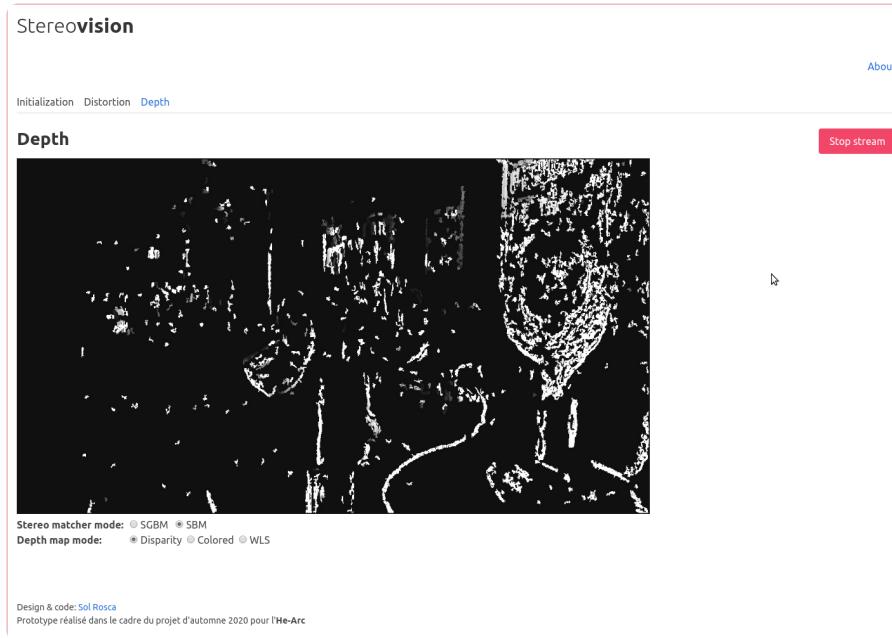
**Figure 14** Carte de profondeur avec filtre WLS

Si on le retire en cliquant sur la radio "Colored", l'image résultante n'est plus traitée avec le filtre WLS et affiche juste une carte de profondeur colorée:



**Figure 15** Carte de profondeur colorée

Et si on retire la couleur, en cliquant sur la radio "Disparity", il ne reste que la carte de disparité sans aucun traitement autre qu'une normalisation ramenant la disparité maximum à 0.



**Figure 16** Carte de disparité avec blockmatcher simple

Sur la figure précédente, le mode du *blockmatcher*<sup>1</sup> a également été changé. Il existe deux modes:

- **SBM** (StereoBlockMatcher): Un mode qui crée des cartes moins riches mais permet d'avoir une image fluide.
- **SGBM**: Un mode qui crée des cartes disparité riches mais très coûteux en puissance de calcul. Entraine une réduction conséquente du nombre d'images par secondes.

Il est possible de changer les paramètres du blockmatcher dans le module `public/settings.py` dans la partie `DEPTH_MAP_DEFAULTS`.

---

1. Algorithme de recherche de correspondance entre les deux images



Figure 17 SBM + color map

La capture précédente montre une carte de profondeur colorée (plus c'est rouge, plus c'est proche) mais avec le blockmatcher **SBM**. Dans ce mode, pour une résolution de capture HD (1280 x 920), le nombre d'images par seconde est proche de 60 alors qu'avec le mode **SGBM** et le filtre WLS il oscille entre 2 et 3 images par secondes.