

---

# Vision, réalité virtuelle et augmentée

## Compte rendu - TP Poursuite de cible

*Clément Potin*

Master 2 Informatique, IMAGINE

Université de Montpellier

2021 - 2022

---



# SOMMAIRE

|   |           |
|---|-----------|
| <b>Choix techniques</b>                                 | <b>3</b>  |
| <b>Fonctionnement du logiciel</b>                       | <b>3</b>  |
| Préparation   | 3         |
| Lancement   | 4         |
| Utilisation   | 4         |
| Résultats   | 5         |
| <b>Décomposition du programme</b>                       | <b>6</b>  |
| <b>Mise en correspondance de motifs par corrélation</b> | <b>7</b>  |
| <b>Résultats</b>  | <b>8</b>  |
| <b>Lien du git</b>                                      | <b>10</b> |
| <b>Annexes</b>  | <b>10</b> |

## 1. Choix techniques

Le travail demandé portant sur du traitement des images, les possibilités quant aux choix du langage étaient l'utilisation du C/C++ avec les bibliothèques proposées, ou l'utilisation du Python, très adapté au travail sur des images.

J'ai fait le choix du Python qui, associé aux bibliothèques Numpy et OpenCV, me permettrait de progresser plus rapidement qu'un code C++, quitte à avoir à ré-implémenter certaines fonctions proposées dans les bibliothèques C/C++ liées au TP.

## 2. Fonctionnement du logiciel

### A. Préparation

Le projet s'organise de la façon suivante :

| Nom     | Type                |
|---------|---------------------|
| results | Dossier de fichiers |
| targets | Dossier de fichiers |
| main.py | Fichier PY          |

Les images sur lesquelles réaliser la poursuite de cible sont à placer dans le dossier **targets**, tel que :

| Nom          | Type               |
|--------------|--------------------|
| Image001.tif | IrfanView TIF File |
| Image002.tif | IrfanView TIF File |
| Image003.tif | IrfanView TIF File |
| Image004.tif | IrfanView TIF File |
| Image005.tif | IrfanView TIF File |

Les images peuvent être sous n'importe quel format (tant qu'elles restent sous un format reconnu comme image et donc pris en compte par *OpenCV*). Elles seront chargées par nom dans l'ordre alphabétique. La première image du dossier sera donc utilisée pour commencer la poursuite de cible.

Le dossier **results** contiendra les résultats de la poursuite de cible.

Enfin, le fichier **main.py** contient tout le code du projet. Il est codé en Python 3 ( $\geq 3.7$ ), et requiert l'installation des librairies *Numpy* et *OpenCV* (voir les liens [2] et [3] dans les annexes).

## **B. Lancement**

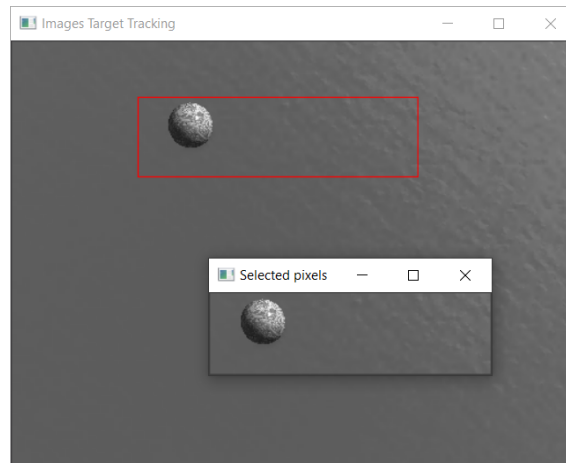
Pour lancer le logiciel, il devrait suffire d'ouvrir le dossier du projet (contenant le **main.py**) dans l'invite de commande de l'OS, et d'exécuter le programme **main.py** avec Python 3, sans argument supplémentaire requis. Exemples :

- `python3 main.py` (sous Windows)
- `./python3 main.py` (sous Linux)

## **C. Utilisation**

Après le lancement du programme, la fenêtre "Images Target Tracking" apparaît. Elle sert à visualiser la première image du dossier **targets**, et permet de dessiner (à l'aide de la souris) un rectangle (rouge) pour sélectionner la zone (cible) à poursuivre dans les images suivantes.

Lorsqu'une zone a été sélectionnée, une seconde fenêtre, nommée "Selected pixels", apparaît pour montrer à l'utilisateur les pixels (la zone/cible) qu'il a sélectionné. Cette fenêtre est uniquement visuelle, et aucune interaction n'est prévue avec celle-ci.








**Interface du logiciel**

Pour lancer le rendu (réaliser la poursuite de cible et enregistrer les résultats dans le dossier **results**), il suffit d'appuyer sur la touche **R**. La progression du rendu sera affichée dans la console.

Pour quitter le logiciel, il est possible d'appuyer sur la touche **Échap**.

#### **D. Résultats**

Lors du rendu (poursuite de cible), des images de résultats sont enregistrées dans le dossier **results**, tel que :

| Nom   | Type               |
|---|--------------------|
|  0.tif | IrfanView TIF File |
|  1.tif | IrfanView TIF File |
|  2.tif | IrfanView TIF File |
|  3.tif | IrfanView TIF File |
|  4.tif | IrfanView TIF File |

L'image **0.tif** correspond à l'image que l'utilisateur voit, et sur laquelle il a dessiné un rectangle de sélection. Les images suivantes correspondent aux images suivantes du dossier **targets** sur lesquelles ont été dessinés un rectangle rouge (de même taille que la sélection de l'utilisateur), après calcul de poursuite de la cible.

### 3. Décomposition du programme

Ordre d'implémentation des fonctionnalités :

- Vérification de l'existence (sinon création ou remise à zéro) des dossiers **targets** et **results**
- Récupération du chemin (path) de toutes les images contenues dans le dossier **targets**
- Affichage, avec OpenCV, de la première image du dossier **targets**
- Normalisation et enregistrement des coordonnées du clic et du relâchement de la souris
- Dessin d'un rectangle pour rendre visuellement compte à l'utilisateur de la zone qu'il a sélectionnée
- Enregistrement des données de cette zone (coordonnées limites, taille, valeur (niveaux de gris) des pixels)
- Affichage dans une nouvelle fenêtre de la zone sélectionnée, pour que le rectangle rouge ne gêne pas l'utilisateur dans son choix de la cible à poursuivre
- Génération des coordonnées des zones possibles dans laquelle la cible pourrait s'être déplacée dans l'image suivante, en fonction d'un rayon donné
- Récupération des informations d'une zone de l'image suivante en fonction de ses coordonnées
- Calcul de la moyenne et de l'écart-type d'une zone
- Calcul d'un score (coefficient de corrélation de Pearson) entre 2 zones, en utilisant les valeurs des pixels, les moyennes et les écarts-types
- Automatisation du calcul des scores entre la cible et toutes les zones possibles générées dans l'image suivante, et choix de la zone avec le score le plus élevé
- Remplacement de l'image courante par l'image suivante et de la cible par la zone avec le score le plus élevé, et automatisation de l'itération jusqu'à l'image finale
- Enregistrement de chaque image avec le dessin du rectangle représentant la zone au score le plus élevé dans le dossier **results**

#### **4. Mise en correspondance de motifs par corrélation**

Pour trouver la zone qui, dans l'image suivante, est la plus similaire à la cible dans l'image actuelle, j'ai utilisé la méthode de mise en correspondance par les coefficients de corrélation de Pearson.

Cette méthode, assez intuitive, vise à calculer un "score" de similarité entre 2 images (ou sous-ensembles d'images), via le calcul suivant :

$$\rho = \frac{1}{N_x * N_y} * \frac{\sum_x \sum_y (I1(x,y) - \mu_1) * (I2(x,y) - \mu_2)}{\sigma_1 * \sigma_2}$$

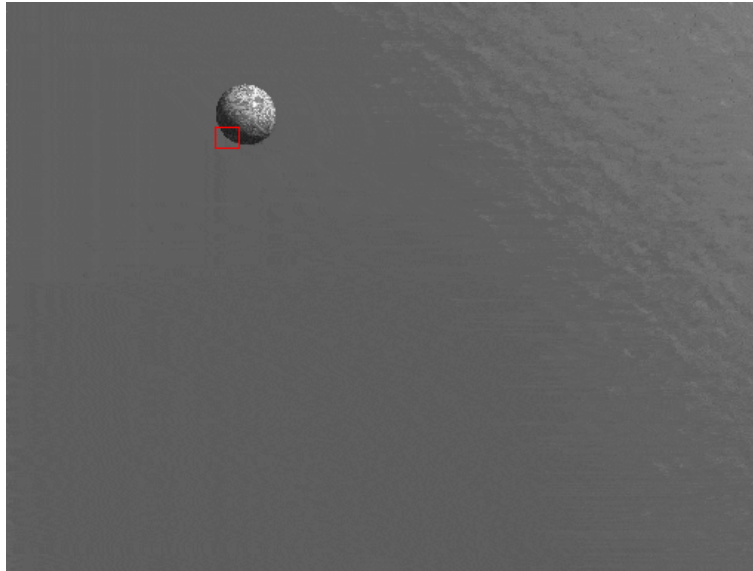
Où :

- $\rho$  est le coefficient de corrélation de Pearson
- $N_x$  et  $N_y$  la taille de la zone en largeur (x) et hauteur (y)
- $I1(x, y)$  et  $I2(x, y)$  les valeurs (en niveaux de gris) des pixels des images  $I1$  et  $I2$  pour les coordonnées  $x$  et  $y$  dans ces images
- $\mu_1$  et  $\mu_2$  les moyennes des niveaux de gris des images  $I1$  et  $I2$
- $\sigma_1$  et  $\sigma_2$  les écarts-types des niveaux de gris des images  $I1$  et  $I2$

Il suffit donc, à partir des coordonnées de la cible dans l'image  $I_n$ , de générer toutes les images  $I(n + 1)$  possibles dans un rayon prédéfini, et de comparer toutes ces images à  $I_n$  via le calcul du coefficient de corrélation de Pearson  $\rho$  entre chaque couple  $(I_n, I(n + 1))$ . Le couple ayant le meilleur score verra son image  $I(n + 1)$  sélectionnée comme cible pour l'itération suivante, car c'est celle qui, selon les coefficients de corrélation de Pearson calculés, est la plus similaire à la cible courante.

## 5. Résultats

Certains résultats obtenus avec cette méthode sont prometteurs et prouvent que la méthode est efficace pour faire de la poursuite de cible :



***SequenceSansVariation***

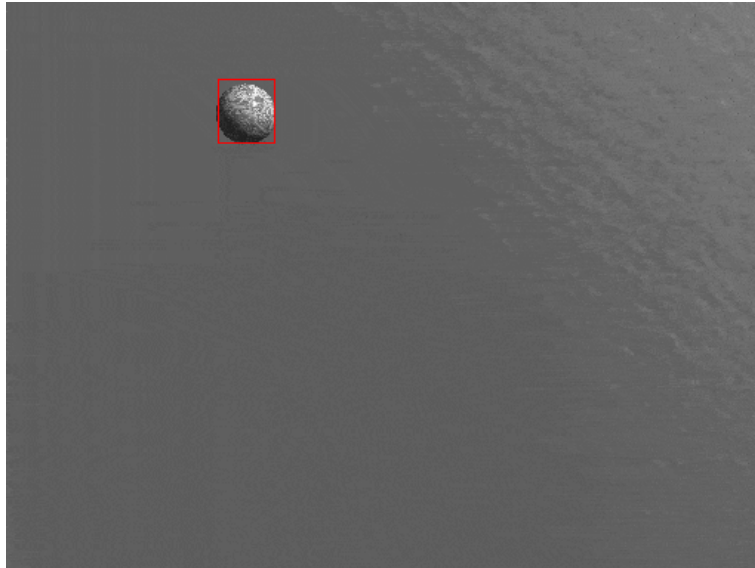
D'autres montrent que la méthode a ses failles. Notamment, si la zone sélectionnée est décalée lors d'une itération, la cible sera elle aussi décalée, et donc toutes les cibles suivantes également.

C'est ce qui arrive dans les séquences suivantes :



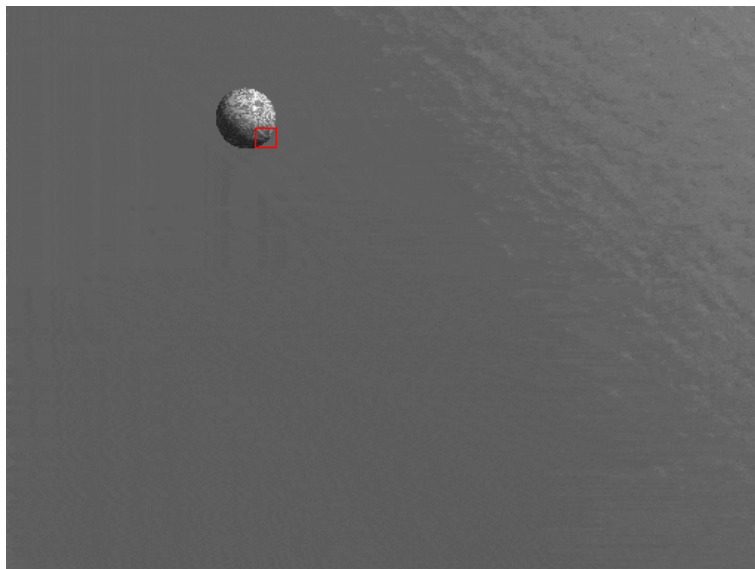
***Ghost2***





***SequenceSansVariation***

Néanmoins, alors qu'on pourrait penser certains résultats faussés (comme dans les cas suivants), la méthode des coefficients de corrélation de Pearson s'avère tout de même assez efficace car si on regarde la rotation de la sphère sur cette séquence, la cible suit bien la zone pointée par la cible d'origine (et pas simplement le "côté haut-gauche"). La sphère en marbre ayant une texture très bruitée, cette méthode réussit malgré tout à produire des résultats satisfaisants.



***SequenceSansVariation***

## 6. Lien du git

Le projet est trouvable sur GitHub, à l'adresse suivante :

<https://github.com/FeuFeve/Pictures-Target-Tracking>

## 7. Annexes

1. Lien du sujet de TP :

[https://www.lirmm.fr/~strauss/Master2Vision/TP\\_Tracking\\_Info/PoursuiteDeCibleInfo.pdf](https://www.lirmm.fr/~strauss/Master2Vision/TP_Tracking_Info/PoursuiteDeCibleInfo.pdf)

2. Installation de la librairie Numpy :

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

3. Installation de la librairie OpenCV :

<https://pypi.org/project/opencv-python/>

4. Lien du rapport avec les gifs animés :

<https://docs.google.com/document/d/1Mi-qFG1L2xE2tKEnpht12R5k7ZWlyDykXYzDJb2i7Tw/edit?usp=sharing>