Rapport de laboratoire

Ecole supérieure
Électronique

Laboratoire POBJ
Salle R112

Projet_Camera_2135

Réalisé par :

Nicolas Zaco

A l'attention de :

Mr. Bovey

Dates :

Début du laboratoire : 05.05.2022
Fin du laboratoire : 06.06.2022

Table des matières :

1 Cahier des charges

Voir cahier des charges sous annexe.

2 Etat initial du projet

Ce projet n'a pas été repris d'un précédent collègue, et a donc dû être élaboré en partant de zéro. Les seules spécifications de ce projet au début de son élaboration étaient :
De concevoir un système pouvant reconnaitre des formes simples à l'aide d'une caméra, et de la librairie OpenCV, et de pouvoir implémenter cela sur un Raspberry Pi.

3 Introduction à OpenCV en C# sur Visual studio.

OpenCV est une bibliothèque graphique de libre utilisation, qui peu nous permettre de faire du traitement d'image en temps réel. Cette librairie est très avantageuse lorsque l'on veut élaborer un système de reconnaissance d'image « simple » (comme une detection de forme dans notre cas) voire complexe, comme une detection de visages.

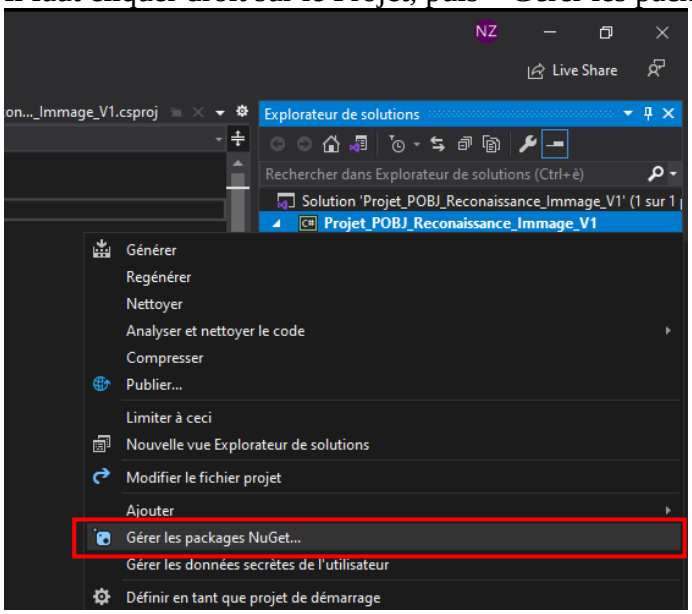
Pour utiliser plus facilement la OpenCV, nous créons simplement un projet console en C# sur Visual studio, et y installons un NuGet.

3.1 Installation de NuGet

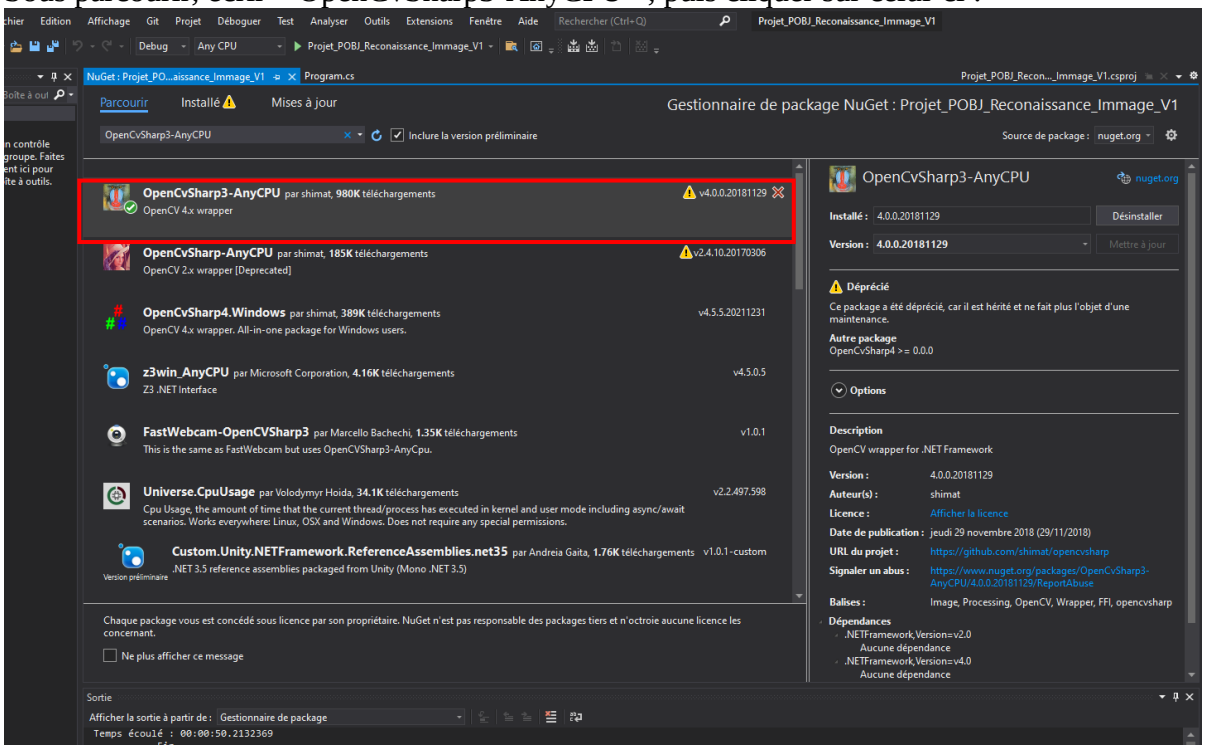
Pour pouvoir utiliser la librairie OpenCV en C# sur Visual studio, nous devons installer des NuGet (des packages / librairies) sur notre projet. Dans notre cas, nous devons installer le NuGet package « OpenCvSharp3-AnyCPU ».

Voici comment procéder :

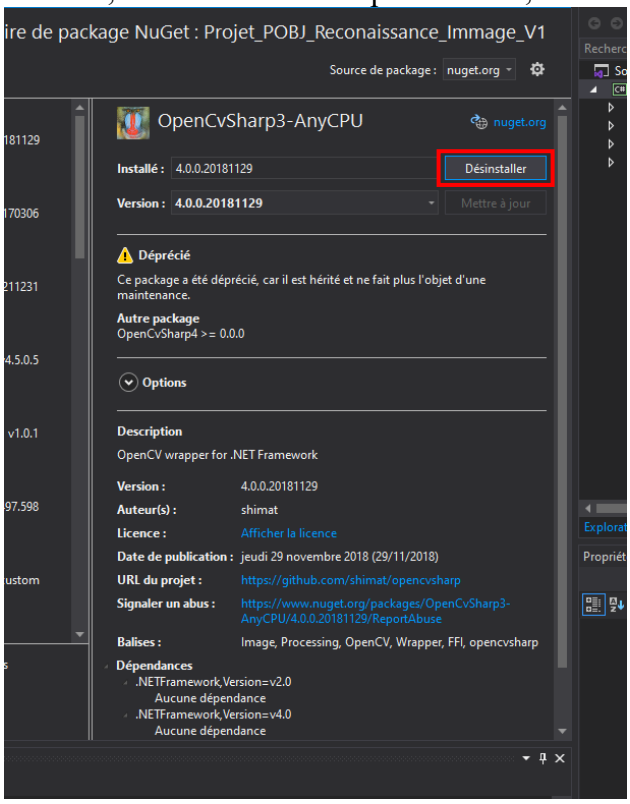
Il faut cliquer droit sur le Projet, puis « Gérer les packages NuGet » :



Sous parcourir, écrir « OpenCvSharp3-AnyCPU », puis cliquer sur celui-ci :



Pour installer le package, il fait cliquer sur la touche installer (dans mon cas le NuGet est déjà installer, la touche n'affiche pas installer, mais désinstaller.) :



4 Explication du code actuel

Le code actuel permet de pouvoir dessiner le contour d'une forme (les arêtes) en blanc, et de mettre en noir tout le reste de l'image.

Apres l'installation du package (NuGet) pour OpenCv, nous devons inclure la librairie dans notre projet :

```
using OpenCvSharp;      //inclus la lib OpenCv
```

Dans un premier temps nous allons chercher l'image à traiter à son chemin :

```
//Vas chercher l'immage et ls stock dans img
Mat img = Cv2.ImRead(@"D:\Utilisateurs\nicolas\Desktop\Projet_POBJ_Reconaissance_Immage_V1\Projet_POBJ_Reconaissance_Immag
```

L'image « img » est une image de type « Mat » (une matrice de points), à qui on assigne une image que l'on vas chercher avec la méthode statique « ImRead » à un certain chemin.

On affiche ensuite l'image originale que l'on vient d'aller chercher (résultat à la fin de l'explication du code) :

```
//affiche m'immage img originale
Cv2.ImShow("Original Image", img);
```

Nous clonons ensuite l'image originale dans une variable « cloneImg », cette image serra celle que l'on traitera. « cloneImg » serra de toute façon modifiée, mais doit être initialisée, c'est pour cella que l'on lui donne le clone de l'image « img ».

```
//Clone l'immage img dans "cloneImg"
Mat cloneImg = img.Clone();
```

Avec la méthode statique « Canny », nous faisons le contour de l'image « img » et le contour serra enregistrer dans la variable « cloneImg ».

```
//dessine le contour de la forme en blanc (garde que les contours)
Cv2.Canny(img, cloneImg, 100, 200, 3, false);
```

Pour les autres paramêtres :

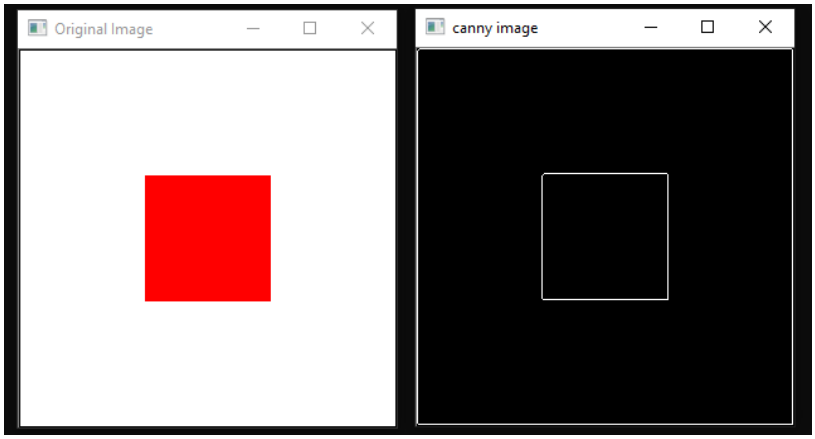
100 : correspond au premier seuil pour procédure d'histerisis.
200 : correspond au deuxième seuil pour procédure d'histerisis.
3 : selon des recherches en ligne « Taille d'ouverture pour l'opérateur Sobel », sa valeur par défaut est 3, j'ai donc laisser 3.
False : correspond à des normes « L2 ou L1 », la valeur par défaut est « false », je laisse donc « false »

On affiche ensuite l'image « cloneImg » qui serra le contour de notre forme :

```
//affiche l'immage des contours de l'immage
Cv2.ImShow("canny image", cloneImg);
```

Pour terminer on ajoute une méthode statique « WhaitKey » qui vas nous permetre de nous bloquer dans le code, afin de pouvoir observer le résultat de celui-ci.

Voici les résultats obtenus avec ce code :



Deux fenêtres s'ouvrent. La première est l'image originale « img » qui et un carré rouge sur un font blanc. La deuxième image est le résultat de la méthode « Canny » qui est le pourtour de notre carré.

Le code que j'ai élaboré est bien fonctionnel.

5 Etat final du projet et prochaines étapes du projet

Contrairement au requêtes du cahier des charges, je n'ai pas pu terminer le code pouvant me permettre à reconnaitre des formes simples, mais ai pu commencer à travailler sur celui-ci.

L'état final du projet est un code pouvant dessiner les bordures de formes.

La prochaine étape de ce projet serrait de pouvoir terminer le code permettant la reconnaissance de formes simples.
Ce code serrai capable de pouvoir reconnaitre des formes comme un rond, un carré, ou un triangle, et de pouvoir afficher leurs noms sois sur une interface ou sur une console.

Après l'élaboration de ce code, veuillez-vous référer au cahier des charges (voir point annexe) pour le reste des taches de ce projet.

6 Difficultés rencontrées et info sur la doc OpenCV

Lors de ce projet, j'ai rencontré plusieurs difficultés pour l'installation et l'utilisation de la librairie OpenCV. Cette librairie qui est très lourde, est peu évidente à prendre en main.

La documentation sur la librairie OpenCV en C# est assez maigre, contrairement au langage python ou OpenCV est beaucoup plus utilisé. Néanmoins la documentation en C++ peux s'avérer assez utile, car la librairie C++ est assez ressemblante à la librairie en C#.

Ces différentes difficultés sont la principale explication de la faible avancée de ce projet.

7 Conclusion

OpenCV est un outil open source très utile pour le traitement / reconnaissance d'images. Néanmoins, cette librairies, très vaste, est plutôt compliquée à prendre en main, et demande pas mal de temps (d'apprentissage) avant d'obtenir des résultats concrets.
De plus, la documentation OpenCV C# est assez maigre, il serrai en effet plus avantageux d'utiliser OpenCV en python.

Du coté du projet, le code actuel ne permet pas encore la reconnaissances de formes simples, mais le projet déjà existant permettra de pouvoir terminer ce code, avec tout les outils nécessaires à son élaboration.

Après l'élaboration du code de reconnaissance de formes simples, il faudra dans le futur pouvoir l'utiliser avec une caméra, et pouvoir utiliser le tout sur un système embarqué tel qu'un Raspberry PI.

Lausanne ETML-ES Nicolas Zaco
06.06.2022

8 Sources / Annexes

Cahiers des Charges :
Le cahier des charges se trouve dans le même dossier que ce fichier.

Documentation pour la prise en main de OpenCV (première application) :
https://www.tech-quantum.com/create-your-first-open-cv-module-in-csharp-net/

Documentation officielle OpenCV :
https://docs.opencv.org/4.x/