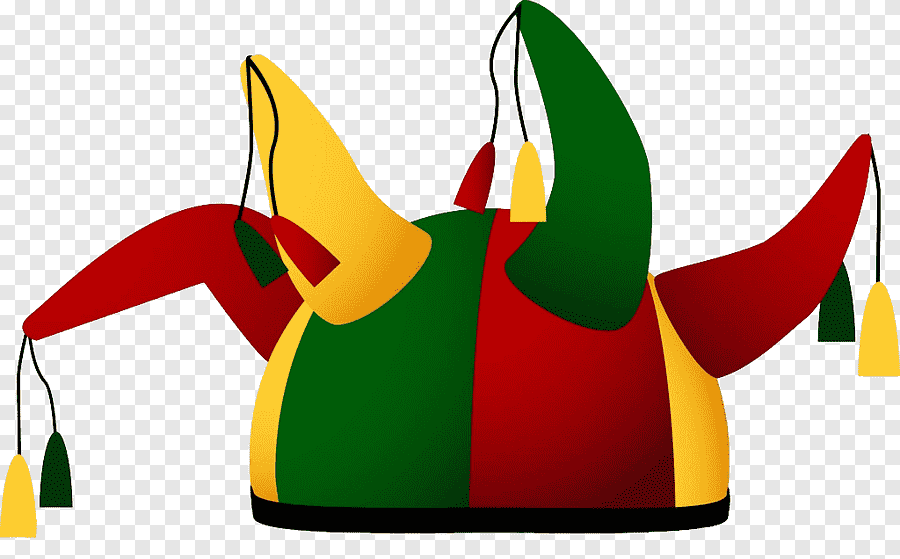
# Projet de traitement d’images

## Contenu

Pour ce projet, nous avons incrusté des images liées au cirque à notre webcam.

En ce qui concerne le filtre de la vidéo, nous avons réalisé un filtre sépia.

Une image contenant texte, équipement électronique

Description générée automatiquementNous avons sélectionné des images statiques (chapeau, œil, nez de clown) afin de les incruster sur les éléments détectés grâce aux Cascades Classifier.

Une image contenant ballon, aéronef, transport, boule

Description générée automatiquementEn plus de cela, nous avons incruster une image en mouvement : des balles de jonglage.

Une image contenant texte, conteneur, boîte

Description générée automatiquementEnfin nous avons pris une image de cirque en tant que fond d’écran dans le cas où l’utilisateur se situe devant un mur blanc.

Une image contenant texte

Description générée automatiquementToutes ces fonctionnalités peuvent être activées ou désactivées à l’aide de carrés cliquables. Le fond de ces carrés sont noires lorsque les fonctionnalités sont désactivées et blanches lorsqu’elles sont activées. Sur les carrés on retrouve les images incrustées afin de savoir quel carré correspond à quelle fonctionnalité.

Les carrés présents sur l’image ci-dessus gère l’activation/désactivation des fonctionnalités dans l’ordre suivant :

filtre sépia – chapeau – yeux – nez – balles qui tombent – fond d’écran

## Explications du code

Le début du code comporte les imports de modules (cv2, numpy, pyplot, math) puis les différentes fonctions que l’on a implémentées.

* Alpha : ressort une image binarisée à partir d’une image et d’un seuil
* Affichage : affiche une image choisie
* Test\_incrustation : affiche une image qui est filtré par son équivalent binarisée. On obtient donc l’image initiale et le fond est noir
* Fermeture : une image subit une dilatation suivie d’une érosion
* Filter\_sepia : application du filtre sépia sur une image
* Incrustation : Incrustation d’une image sur une autre. A partir d’une zone détectée, on redimensionne l’image à incruster ainsi que son alpha puis on applique les règles de multiplication et d’addition pour incruster l’image
* Rectangle : Trace les rectangles sur une image à partir des coordonnées données, le rectangle est blanc si le paramètre « enable » vaut 1 sinon le rectangle est noir
* On\_click : Permet de changer les variables qui indiquent si l’utilisateur veut activer ou désactiver chaque fonctionnalité lorsque des clics sont effectués sur certaines zones

Après nos fonctions on trouve les cascades classifier (détection des visages, des yeux et des nez) ainsi que les images à incruster (voir Contenu). On binarise les images pour pouvoir les incruster plus tard.

On a créé un dictionnaire qui permettra de retenir les choix d’activation ou de désactivation des fonctionnalités.

On génère un tableau composé des coordonnées des balles par frame pour pouvoir faire défiler les balles de haut en bas.

On définit les coordonnées des carrés qui permettent d’avoir un menu interactif. La couleur du fon de ces carrés est blanche ou noir selon si l’option a été activée ou non. Sur le fond on rajoute l’image correspondante à l’option.

On lance la webcam avec une boucle infini dont on peut sortir à l’aide d’un break en appuyant sur la touche «q».

Dans la boucle, chaque fonctionnalité est active si l’utilisateur a cliqué sur le bouton correspondant.

On enregistre le flux vidéo dans un fichier mp4.

L’ordre des incrustations n’est pas à négligé. En effet, si l’on incruste le fond d’écran après les yeux, les yeux seront considérés comme une partie du fond d’écran car ils sont blancs tout comme le mur qui devrait se trouver derrière l’utilisateur.

## Démo

Mode sépia : Incrustation chapeau :



Incrustation nez : Incrustation œil :



Incrustation balles : Incrustation cirque en fond :



## Problèmes rencontrés

Lors de ce projet, nous avons rencontré plusieurs problèmes.

Tout d’abord, nous avions eu des erreurs lors des incrustations de nos images lorsque celles-ci dépassaient les dimensions du frame. Après avoir découvert les raisons de cette erreur nous avons contraint l’image à ne pas dépasser ces limites.

Un deuxième problème était que les cascades classifier détectait des nez sur des objets extérieurs. Nous avons donc fait en sorte de ne conserver que les yeux et nez détectés qui sont à l’intérieur d’une tête détectée.

Nous avions au départ également une image d’œil qui n’était pas symétrique, nous avons donc inversé l’image avec la fonction flip de opencv puis nous incrustions l’image originale ou l’image inversée si l’œil se trouvait à gauche ou à droite du milieu de la tête détectée. Cependant nous avons par la suite trouvé un autre œil symétrique plus joli que le précédent, nous avons donc pu enlever ces lignes de code.

Autre mésaventure, pour incruster les images nous avions commencés par changer les pixels à l’aide de deux boucles imbriqués (une pour les colonnes, l’autre pour les lignes). Cette méthode était lourde pour le code et la vidéo rendue n’était donc pas fluide. Nous avons ensuite trouvé une solution pour remédier à ce problème, nous avons remplacé les boucles par le crop des frames.