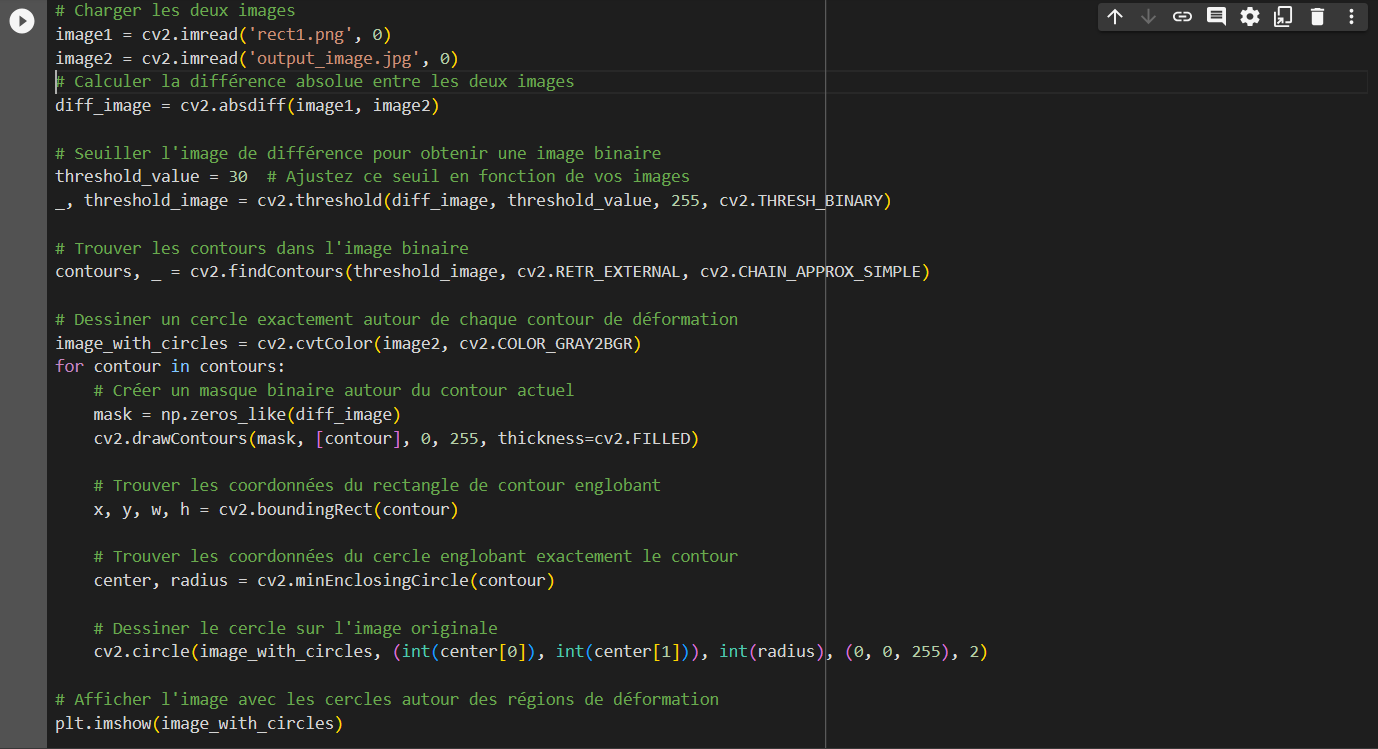
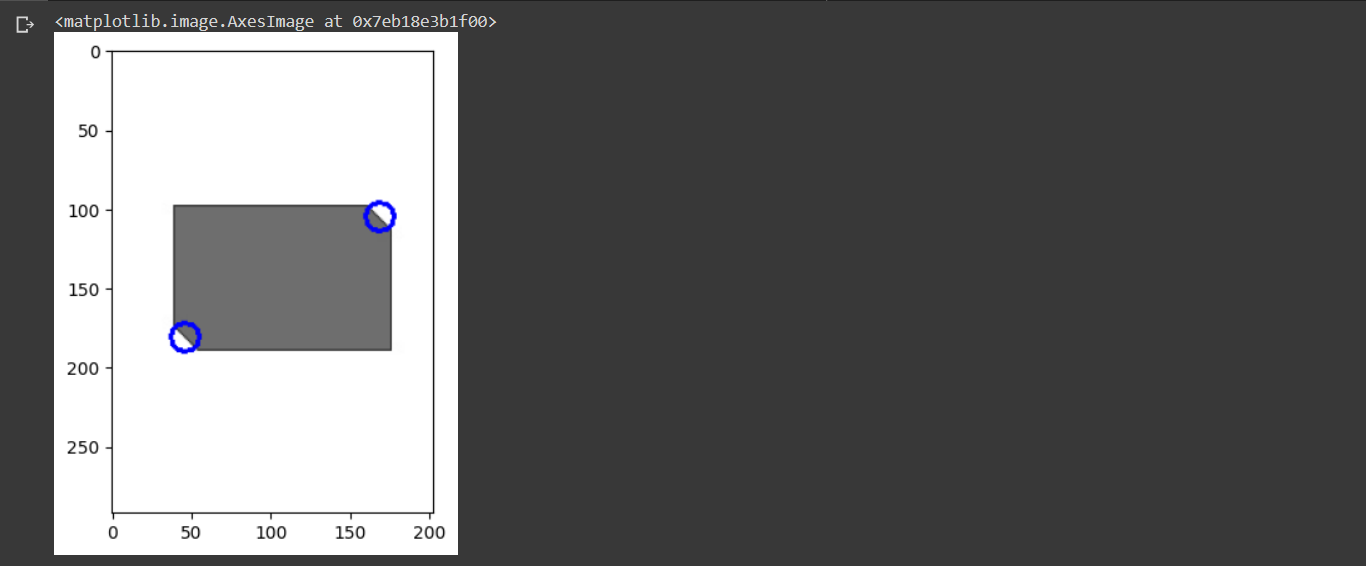
Explication du code pour la localisation

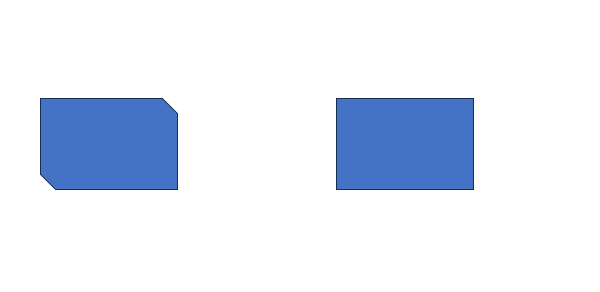
Le code :



Le résultat :



Les deux objets :



Explication :

* diff\_image = cv2.absdiff(image1, image2) : Cette ligne calcule la différence absolue pixel par pixel entre les deux images (image1 et image2) et stocke le résultat dans diff\_image. Cela donnera une image qui met en évidence les différences entre les deux images.
* threshold\_value = 30 : Ce seuil est utilisé pour binariser l'image de différence. Tous les pixels ayant une valeur supérieure à ce seuil seront définis à 255 (blanc), et les autres pixels seront définis à 0 (noir).
* \_, threshold\_image = cv2.threshold(diff\_image, threshold\_value, 255, cv2.THRESH\_BINARY) : Cette ligne applique le seuillage à l'image de différence (diff\_image) pour obtenir une image binaire où les régions de déformation sont mises en évidence.
* contours, \_ = cv2.findContours(threshold\_image, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE) : Cette ligne trouve les contours dans l'image binaire seuillée (threshold\_image).
* image\_with\_circles = cv2.cvtColor(image2, cv2.COLOR\_GRAY2BGR) : Cette ligne convertit l'image image2 en couleur en utilisant cv2.cvtColor(). Cela est nécessaire car nous allons dessiner des cercles colorés sur l'image pour mettre en évidence les régions de déformation.
* mask = np.zeros\_like(diff\_image) : créer un masque binaire initialisé avec des pixels noirs (valeurs de zéro) et ayant la même taille que l'image diff\_image. Ce masque sera utilisé plus tard pour représenter les zones à masquer ou à conserver lors d'opérations telles que le dessin de contours autour des régions de déformation dans l'image image\_with\_circles.
* cv2.drawContours(mask, [contour], 0, 255, thickness=cv2.FILLED) : Cette ligne dessine le contour actuel sur le masque en blanc (valeur 255). Cela créera un masque binaire autour du contour actuel.
* x, y, w, h = cv2.boundingRect(contour) : Ces lignes trouvent les coordonnées du rectangle englobant (rectangle délimitant) le contour actuel.
* center, radius = cv2.minEnclosingCircle(contour) : Ces lignes trouvent les coordonnées du cercle englobant exactement le contour actuel en utilisant la fonction cv2.minEnclosingCircle().
* cv2.circle(image\_with\_circles, (int(center[0]), int(center[1])), int(radius), (0, 0, 255), 2) : Cette ligne dessine un cercle autour du contour actuel sur l'image en couleur image\_with\_circles. Le cercle est dessiné en bleu ((0, 0, 255)), avec une épaisseur de 2 pixels.