Résumé du Script de Traitement d'Image

Ce document résume les étapes principales du script de traitement d'image, fournissant une vue d'ensemble du processus depuis la sélection de l'image jusqu'à l'affichage et l'interaction avec l'utilisateur.

---

1. **Segmentation Sémantique avec SAM**
   Une segmentation sémantique est réalisée sur l'image à l'aide du modèle SAM (Segment Anything Model) de Meta pour isoler les objets ou régions d'intérêt, facilitant les traitements suivants.

2. **Sélection de l'Image**
   L'utilisateur sélectionne une image via une boîte de dialogue de sélection de fichiers. Le chemin de l'image est enregistré et affiché dans l'interface.

3. **Chargement et Conversion de l'Image**
   L'image sélectionnée est chargée à l'aide d'OpenCV, puis convertie en niveaux de gris pour simplifier les traitements ultérieurs.

4. **Filtrage de l'Image**
   Un filtre médian (3x3) est appliqué pour réduire le bruit tout en préservant les bords. Ensuite, un filtre gaussien (3x3, sigma=0.25) adoucit l'image et réduit davantage le bruit, tout en conservant une certaine netteté.

5. **Binarisation**
   L'image lissée est convertie en une image binaire où les pixels sont soit noirs (0), soit blancs (255), selon un seuil automatique.

6. **Squelettisation**
   Une opération de squelettisation est appliquée à l'image binaire pour extraire la structure centrale (le 'squelette') de l'objet représenté.

7. **Épaississement du Squelette**
   Le squelette est dilaté pour le rendre plus visible en augmentant son épaisseur à l'aide d'un noyau de 3x3.

8. **Calcul des Points d'Intersection**
   Les points d'intersection du squelette avec deux lignes imaginaires (verticales ou horizontales, selon l'orientation de l'image) sont calculés. Ces lignes sont placées aux positions P1 et P3, définies initialement à un quart et trois quarts de la largeur ou de la hauteur de l'image.

9. **Affichage et Interaction**
   L'image est affichée avec les lignes et les points d'intersection superposés. L'utilisateur peut ajuster la position de P1 et P3 en utilisant les touches du clavier ('p', 'm', 'u', 'd'). En appuyant sur 'k', les points d'intersection sont envoyés à un module de calcul externe.

10. **Boucle d'Interaction**
    Le script entre dans une boucle où il attend l'entrée de l'utilisateur via le clavier. Les touches permettent de modifier les positions de P1 et P3, de quitter l'application ('q' ou 'Échap'), ou de valider les résultats ('k').

11. **Envoi des Résultats**
    Lorsque l'utilisateur appuie sur la touche 'k', les points calculés sont envoyés à un module de calcul externe, permettant ainsi d'afficher ou d'utiliser les résultats pour d'autres analyses.

12. **Chargement et Préparation de l'Image**
    Le script utilise une image en niveaux de gris (gray_image) et une version traitée appelée skeleton_overlay, probablement un squelette de l'image d'origine obtenu par traitement morphologique. Deux points spécifiques sur le squelette, P1_skeleton et P3_skeleton, sont identifiés comme points d'intérêt pour lesquels les tangentes seront trouvées.

13. **Recherche des Tangentes avec la Transformée de Hough**
    La fonction find_tangents_using_hough extrait une région d'intérêt (ROI) autour de chaque point d'intérêt (P1_skeleton et P3_skeleton) et utilise la transformée de Hough (cv2.HoughLinesP) pour détecter les lignes dans cette région. Les tangentes sont les vecteurs directionnels de ces lignes détectées.

14. **Calcul du Vecteur Moyenne**
    La fonction calculate_average_vector permet de calculer un vecteur de direction moyen, représentant la direction globale des lignes autour du point d'intérêt, à partir des tangentes détectées.

15. **Visualisation des Vecteurs sous forme de Lignes**
    La fonction draw_extended_line trace le vecteur moyen pour chaque point sous forme de ligne étendue sur l'image originale. Les lignes sont tracées depuis chaque point d'intérêt dans les directions des vecteurs moyens calculés.

16. **Calcul et Visualisation de l'Angle entre les Tangentes**
    La fonction calculate_angle_between_vectors calcule l'angle entre les deux vecteurs moyens (ceux de P1_skeleton et P3_skeleton) à l'aide du produit scalaire. Cet angle est ensuite arrondi et affiché. Un arc est dessiné pour représenter visuellement cet angle sur l'image.

17. **Calcul de l'Intersection des Lignes**
    La fonction find_intersection calcule le point d'intersection entre les deux lignes étendues depuis les points P1_skeleton et P3_skeleton, en résolvant un système d'équations linéaires.

18. **Dessin du Point d'Intersection**
    Si un point d'intersection est trouvé, la fonction draw_intersection_point le dessine sur l'image sous forme d'un petit cercle coloré.

19. **Affichage et Sauvegarde des Résultats**
    La fonction show_results affiche l'image résultante à l'écran (cv2.imshow) après avoir tracé toutes les lignes, l'arc et les points d'intersection, et sauvegarde l'image sur le disque sous le nom result_image.png.

Résumé Global

Ce script traite des images pour identifier des tangentes à des points spécifiques sur un squelette d'image, calculer l'angle entre ces tangentes, et visualiser les résultats sous forme de lignes et d'intersections sur l'image originale.

Les étapes incluent la détection des tangentes à l'aide de la transformée de Hough, le calcul des vecteurs moyens, la visualisation des angles et points d'intersection, et l'affichage des résultats.