

Traitement et détection des images et vidéos d'éoliennes

Smati Daouia Ouissem

1. PARTIE I - TRAITEMENT D'IMAGES CLASSIQUES

Dans le cadre de la maintenance et du suivi des performances des éoliennes, la détection précise des pales est cruciale. Une méthode efficace pour identifier automatiquement le centre des pales permet de surveiller l'état des éoliennes et de diagnostiquer d'éventuels problèmes structurels ou d'alignement. Cependant, les méthodes de détection de contours traditionnelles peuvent se heurter à des limites, notamment dans des environnements complexes où l'arrière-plan contient de nombreux détails. Cela souligne la nécessité de développer des techniques de détection automatique plus sophistiquées, capables de différencier efficacement les pales des autres éléments de l'image.

Voici trois méthodes permettant de passer de l'image originale à l'image prétraitée :

1.1 Filtre de Canny (Canny Edge Detection)

Cette méthode détecte les contours dans une image en utilisant un algorithme de détection de bords qui trouve des gradients d'intensité élevés.

1.2 Filtre de Sobel

Un opérateur de gradient qui détecte les contours en calculant les gradients de l'intensité de l'image. Il produit des images des dérivées premières en x et en y.

1.3 Seuillage (Thresholding)

Transforme une image en niveaux de gris en une image binaire (noir et blanc) en fonction d'un seuil. Cela permet de segmenter les objets d'intérêt dans l'image.

2. PARTIE II - DÉTECTION DOBJETS

2.0.1 Algorithme YOLO pour Détection d'Objets

YOLO (You Only Look Once) est une méthode de détection d'objets en temps réel. Contrairement aux autres méthodes qui utilisent des pipelines en plusieurs étapes, YOLO traite l'image entière en une seule passe d'un réseau neuronal convolutionnel (CNN). Voici les principales étapes et concepts de YOLO :

- **Détection en une seule passe** : YOLO divise l'image en une grille $S \times S$. Chaque cellule de la grille est responsable de prédire un certain nombre de boîtes englobantes et leur score de confiance.
- **Prédictions multiples par cellule** : Pour chaque cellule, YOLO prédit plusieurs boîtes englobantes ainsi que des scores de classe pour ces boîtes.
- **Confiance et Scores de Classe** : La confiance de chaque boîte indique la probabilité que cette boîte contient un objet et la précision avec laquelle elle correspond à l'objet. Les scores de classe prédisent à quelle classe l'objet appartient.
- **Rapidité** : YOLO est conçu pour être rapide, permettant la détection en temps réel.

2.0.2 Suivi d'Objets avec YOLO

Le suivi d'objets avec YOLO implique généralement l'utilisation d'un algorithme de suivi pour suivre les objets détectés d'une image à l'autre dans une séquence vidéo. Voici une approche courante :

1. **Détection Initiale** : Utiliser YOLO pour détecter les objets dans la première image.
2. **Initialisation des Trackers** : Pour chaque objet détecté, initialiser un tracker d'objet (par exemple, KCF, CSRT, ou Deep SORT).
3. **Suivi à travers les Frames** : Pour chaque frame suivante, utiliser le tracker pour suivre les objets. YOLO peut être utilisé périodiquement pour corriger les trackers.
4. **Correction des Anomalies** : Si un tracker perd un objet, YOLO peut être utilisé pour redéetecter les objets et réinitialiser les trackers.

2.0.3 Intégration de YOLO pour la Détection de 3 Pales Éoliennes

Pour intégrer YOLO dans le cadre de la détection des 3 pales éoliennes, voici les étapes recommandées :

- **Préparation des Données**

- **Collecte d'Images** : Collecter un grand nombre d'images de turbines éoliennes avec des pales visibles.
- **Annotation** : Annoter les images avec des boîtes englobantes autour des pales éoliennes. Des outils comme LabelImg peuvent être utilisés pour cela.

- **Entraînement du Modèle YOLO**

- **Configuration** : Créer un fichier de configuration pour YOLO qui spécifie les paramètres du modèle et les classes d'objets (dans ce cas, une seule classe : "pale éolienne").
- **Entraînement** : Utiliser les images annotées pour entraîner le modèle YOLO. Cela peut être fait sur une machine locale avec GPU ou sur des plateformes cloud comme Google Colab.

- **Détection en Temps Réel**

- **Détection** : Déployer le modèle entraîné pour détecter les pales éoliennes dans des images ou des flux vidéo en temps réel.
- **Post-Processing** : Après la détection, appliquer des filtres comme le Non-Maximum Suppression (NMS) pour éliminer les boîtes englobantes redondantes.

- **Suivi des Pales**

- **Suivi** : Utiliser un algorithme de suivi, comme mentionné précédemment, pour suivre les pales éoliennes dans les séquences vidéo.
- **Analyse des Données** : Collecter et analyser les données de suivi pour surveiller l'état des pales éoliennes.

3. PARTIE III -RÉSULTATS PRATIQUES DES CODES UTILISÉS

3.1 Algorithme de Détection et Placement du Point

Nous avons développé un algorithme basé sur la détection des contours des pales d'éoliennes à partir d'images, utilisant une approche basique de traitement des images. Ce processus intègre des techniques de traitement d'images comme le flou gaussien, le seuillage adaptatif et la détection de contours via la méthode de Canny. L'objectif est d'identifier le contour le plus prononcé, celui de la pale, pour ensuite calculer son centroïde à l'aide des moments géométriques. Le centroïde est marqué sur l'image par un point, indiquant ainsi la position centrale estimée de la pale.

Comme montré dans la figure 1, qui illustre différentes étapes de traitement, nous avons opté pour une région d'intérêt (ROI) plus restrictive, centrée sur l'emplacement prévu de la pale. Cette approche a permis de limiter la détection à une zone réduite, minimisant ainsi l'inclusion de l'arrière-plan. De plus, les coordonnées du centroïde ont été ajustées pour tenir compte des déplacements résultant de la restriction du ROI.

Malgré l'efficacité de cette méthode dans des environnements simples, elle montre des limites dans des situations où l'arrière-plan est très complexe ou lorsque les contours de la pale ne sont pas bien définis. Pour résoudre ces problèmes, des modifications ont été nécessaires, notamment l'application d'une région d'intérêt (ROI) pour limiter l'analyse à la zone où la pale est attendue, ainsi que l'ajustement des paramètres de détection pour réduire les faux positifs issus de l'arrière-plan. Ces ajustements permettent d'améliorer la précision de la détection, mais restent insuffisants dans des scénarios plus complexes.



(a) Blurred image



(b) Binary image



(c) Closing image



(d) Contours image



(e) Final result image

Figure 1: Les images a, b, c illustrent les différentes étapes du traitement d'image visant à détecter la pale d'une éolienne, avec un point bleu placé au centre de la pale.

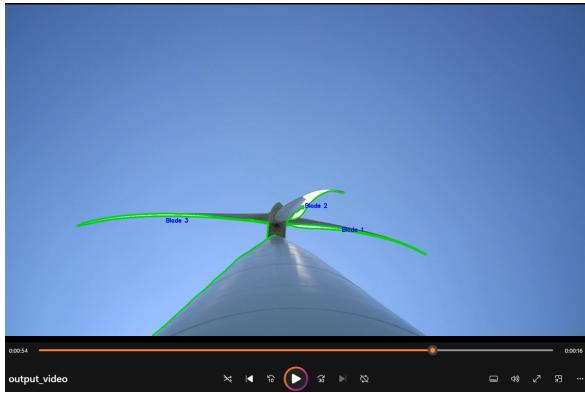
3.2 Détection Non Supervisée des Pales d'Éoliennes Basée sur le Calcul de l'Angle Arctan

Cette méthode est une approche non supervisée pour la détection des pales d'éoliennes dans une séquence vidéo. Elle ne nécessite pas de données d'entraînement étiquetées au préalable. Le processus repose sur la détection des contours dans chaque image vidéo, suivie de l'identification des contours qui correspondent aux pales, en supposant qu'elles ont une forme quadrilatérale. Une fois les contours des pales identifiés, l'algorithme calcule l'angle de chaque pale par rapport au moyeu de l'éolienne en utilisant la fonction arctan (arctangente). Cet angle est déterminé à partir des coordonnées du centre de la pale (son centroïde) et de la position centrale du moyeu. L'angle calculé aide ensuite à suivre les pales au fil des images, en leur attribuant des identifiants uniques pour suivre leur rotation. Cet algorithme est appliqué directement sur une vidéo, permettant ainsi de traiter chaque image successivement pour suivre la dynamique des pales en mouvement. La figure 2 montre les résultats de l'algorithme sur la vidéo, illustrant la précision de la détection et du suivi des pales dans le temps.

L'équation mathématique utilisée pour calculer l'angle θ d'une pale par rapport au moyeu est :

$$\theta = \arctan \left(\frac{y_{\text{centroid}} - y_{\text{hub}}}{x_{\text{centroid}} - x_{\text{hub}}} \right)$$

où $(x_{\text{centroid}}, y_{\text{centroid}})$ représente les coordonnées du centroïde de la pale, et $(x_{\text{hub}}, y_{\text{hub}})$ représente les coordonnées du moyeu de l'éolienne.



(a) Détection de pale sur la vidéo MAH02363.



(b) Détection de pale sur la vidéo MAH02363



(c) Détection de pale sur la vidéo MAH02371

Figure 2: Les images a, b, c montrent des captures d'écran de deux vidéos d'éoliennes sous différents angles de vue.

la méthode repose sur une détection de contours relativement simple, sans prise en compte de l'information contextuelle ou de la continuité temporelle d'une image à l'autre, ce qui peut entraîner des instabilités dans le suivi des pales lorsque les images consécutives présentent des variations significatives en termes de luminosité ou de perspective. Un problème majeur est la difficulté à distinguer de manière fiable les différentes pales (pale 1, pale 2, pale 3) en raison des variations d'angles de vue lors de l'enregistrement vidéo. En effet, les pales d'éoliennes apparaissent sous différents angles selon la position de la caméra, ce qui peut entraîner des erreurs dans l'estimation des contours et des angles. Par exemple, lorsqu'une pale est vue sous un angle plus aigu ou plus obtus, sa forme apparente peut changer de manière significative, rendant difficile son identification précise. Un autre problème notable est l'influence de l'arrière-plan, particulièrement dans des environnements où les contours de l'arrière-plan peuvent être confondus avec ceux des pales. L'algorithme, dans sa forme actuelle, peut détecter de nombreux faux positifs en raison des éléments du paysage, ce qui complique encore la distinction entre les pales.

En raison de ces limitations, cette méthode non supervisée basée sur l'angle arctan montre des performances limitées dans des situations réelles complexes. Pour améliorer la fiabilité de la détection et du suivi des pales, il serait nécessaire d'explorer des approches plus robustes, telles que des méthodes supervisées utilisant l'apprentissage profond, qui pourraient mieux gérer les variations de forme, d'angle et d'arrière-plan, et intégrer des informations temporelles pour stabiliser le suivi des pales à travers la séquence vidéo.

3.3 Détection supervisée des pales d'éoliennes basée sur le modèle YOLO-NAS sur Roboflow

3.3.1 Étape 1 : Préparation des Données

1. Utilisation de Roboflow pour la Préparation des Données

J'ai utilisée Roboflow pour gérer et préparer les datasets. Voici comment j'ai procédé :

- **Source Images** : J'ai uploadée un total de 265 images sur Roboflow, que j'ai ensuite annotées en trois classes : Pale 1, Pale 2, et Pale 3.
- **Annotation** : J'ai utilisé l'outil d'annotation de Roboflow pour étiqueter chaque image avec les trois classes de pales.

3.3.2 Étape 2 : Division des Données

1. Split des Données

J'ai divisée les données en trois ensembles distincts :

- **Training Set** : 186 images pour l'entraînement du modèle.
- **Validation Set** : 53 images pour valider les performances du modèle pendant l'entraînement.
- **Testing Set** : 26 images pour tester le modèle après l'entraînement.

3.3.3 Étape 3 : Prétraitement des Données

1. Prétraitement des Images

J'ai appliquée les étapes de prétraitement suivantes pour améliorer les performances du modèle et réduire le temps de préparation des données. Les données utilisées pour le traitement sont les frames de la vidéo MAH02363.MP4, extraites toutes les 1 seconde :

- **Auto-Orient** : Orientation automatique des images pour assurer une cohérence.
- **Resize** : Redimensionnement des images à une taille de 640x640 pixels.

Creating New Version

Prepare your images and data for training by compiling them into a version. Experiment with different configurations to achieve better training results.



Source Images

Images: 265

Classes: 3

Unannotated: 0



Train/Test Split

Training Set: 186 images

Validation Set: 53 images

Testing Set: 26 images



Preprocessing

[What can preprocessing do?](#)

Decrease training time and increase performance by applying image transformations to all images in this dataset.

Auto-Orient	Edit	x
Resize Stretch to 640×640	Edit	x
+ Add Preprocessing Step		

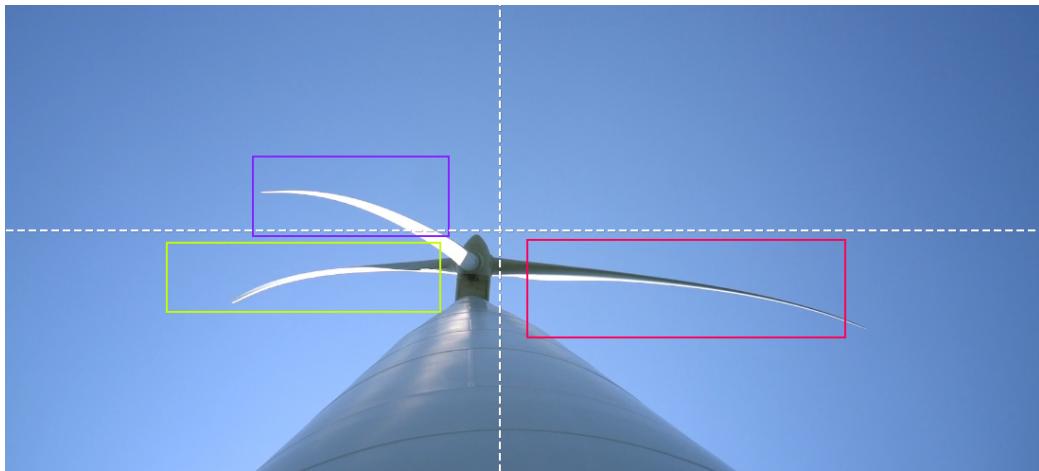
(a) Les détails du prétraitement des données sur Roboflow.

2. Images Annotées

• Visualisation des Annotations

J'ai utilisée Roboflow pour annoter manuellement les images, en employant des outils qui facilitent et accélèrent le processus d'annotation.

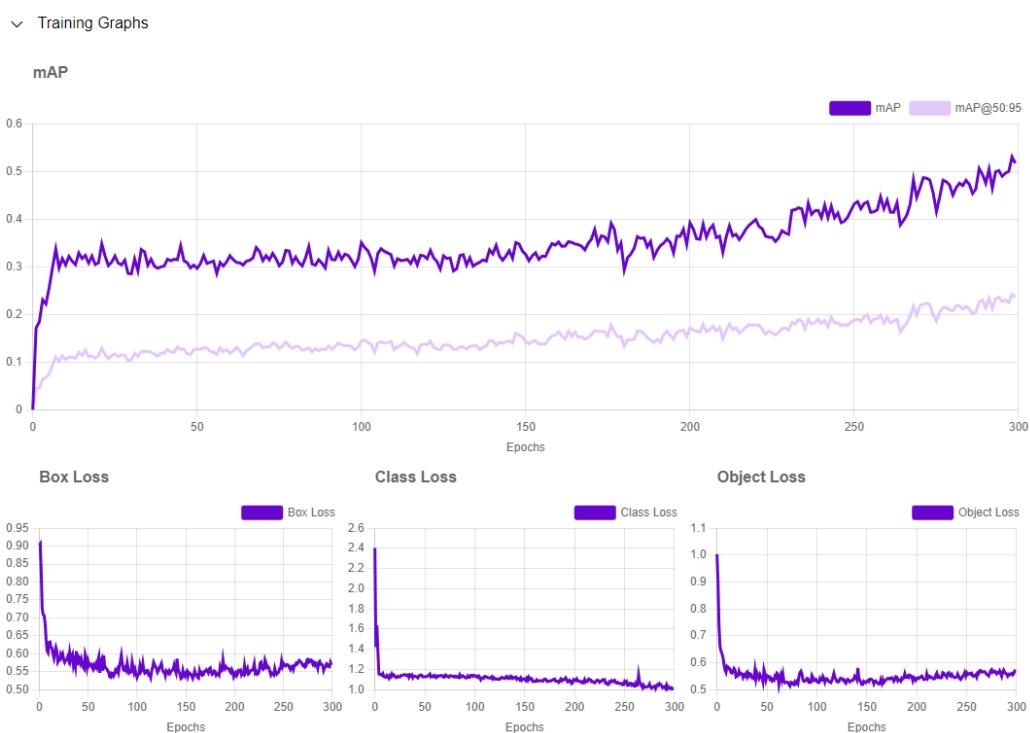
- Cette image montre les annotations avec des boîtes englobantes de différentes couleurs représentant les trois classes de pales :
 - Pale 1 : Encadrée en jaune
 - Pale 2 : Encadrée en violet
 - Pale 3 : Encadrée en rouge



(a) Image annotée extraite de la vidéo MAH02363.MP4

3.3.4 Étape 4 :Entraînement du Modèle sur Roboflow

Une fois que j'ai préparé et annoté mes données, j'ai utilisé Roboflow pour entraîner un modèle YOLO-NAS sur les images des pales d'éoliennes. L'entraînement a duré environ 1 heure. Pendant cette période, j'ai configuré les paramètres d'entraînement tels que le nombre d'époques, la taille des images, et le batch size pour optimiser les performances du modèle.



(a) Détails de l'entraînement avec les courbes de perte.

Pendant l'entraînement, Roboflow a fourni des visualisations en temps réel des courbes de perte, me permettant de suivre les progrès du modèle et de m'assurer qu'il convergeait correctement.

Average Precision by Class



(a) Détails de la Précision moyenne par catégorie.

3.3.5 Étape 5: Test du Modèle

Après l'entraînement, j'ai utilisé Roboflow pour tester les performances du modèle sur l'ensemble de test, composé de 26 images. Le modèle a été évalué en termes de précision, de rappel, et de mAP (mean Average Precision).

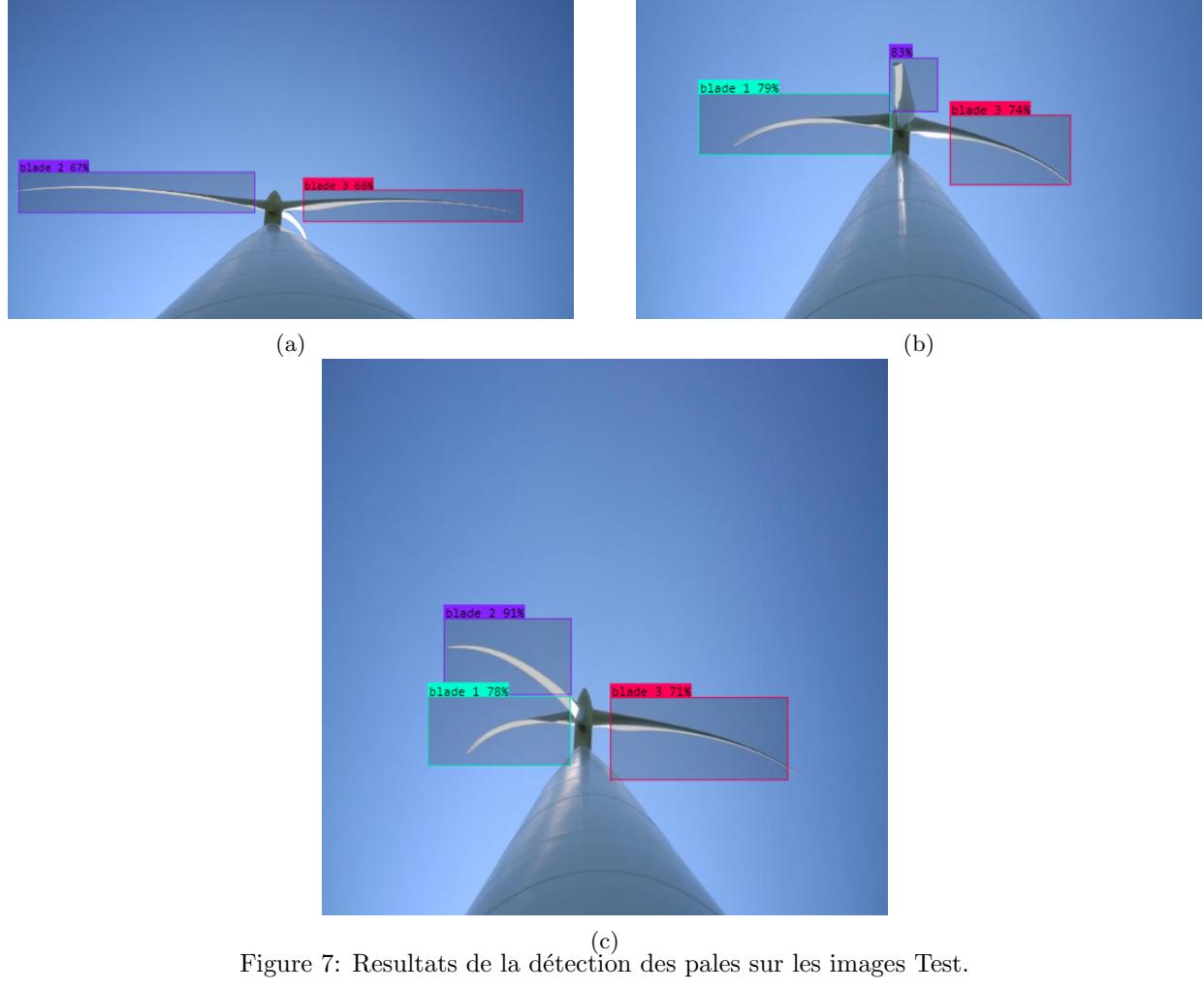


Figure 7: Resultats de la détection des pales sur les images Test.

3.4 Détection supervisée des pales d'éoliennes basée sur le modèle YOLOv5, entraînée en Python et implémentée en C++

3.4.1 Entraînement en Python

Le processus débute avec l'entraînement d'un modèle YOLOv5 en Python, utilisant un jeu de données annotées et prétraitées par Roboflow. Les paramètres du modèle, incluant la taille des images et les hyperparamètres, sont ajustés pour optimiser l'apprentissage. Ceci permet au modèle de détecter efficacement des objets spécifiques, tels que les pales d'éoliennes.

3.4.2 Exportation du Modèle

Une fois l'entraînement terminé, le modèle est exporté dans un format compatible avec C++, tel que ONNX, facilitant ainsi l'utilisation du modèle dans un environnement C++.

3.4.3 Prétraitement des Images en C++

Le code C++ commence par prétraiter les images extraites de la vidéo. Ce prétraitement inclut le redimensionnement et la normalisation des images pour les adapter aux spécifications du modèle YOLOv5.

3.4.4 Détection des Objets

Les images prétraitées sont ensuite passées à travers le modèle YOLOv5 dans le code C++, qui génère des sorties sous forme de boîtes englobantes, identifiant les objets détectés dans chaque frame de la vidéo.

3.4.5 Traitement Postérieur

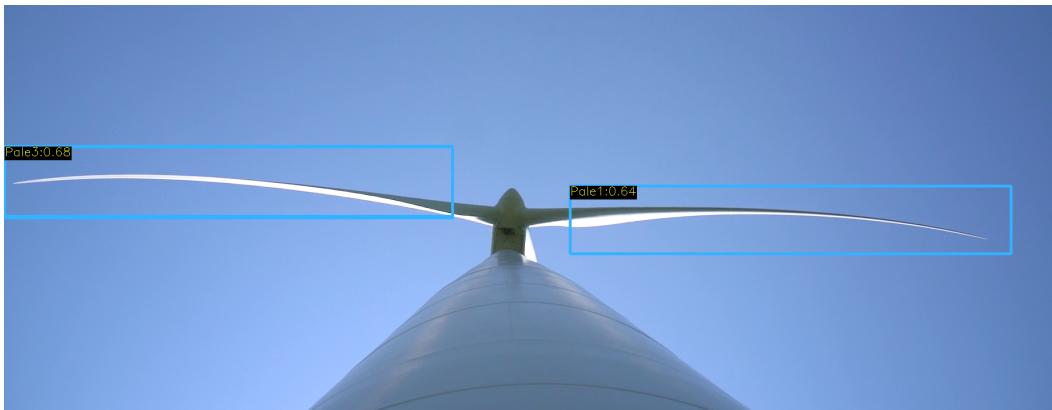
Une attention particulière est portée au traitement postérieur pour s'assurer que les trois pales d'éolienne (Pale 1, Pale 2, et Pale 3) sont détectées correctement. Si une pale, telle que "Pale 1", n'est pas détectée, un traitement secondaire est effectué en ajustant les paramètres pour améliorer la précision.

3.4.6 Estimation Manuelle

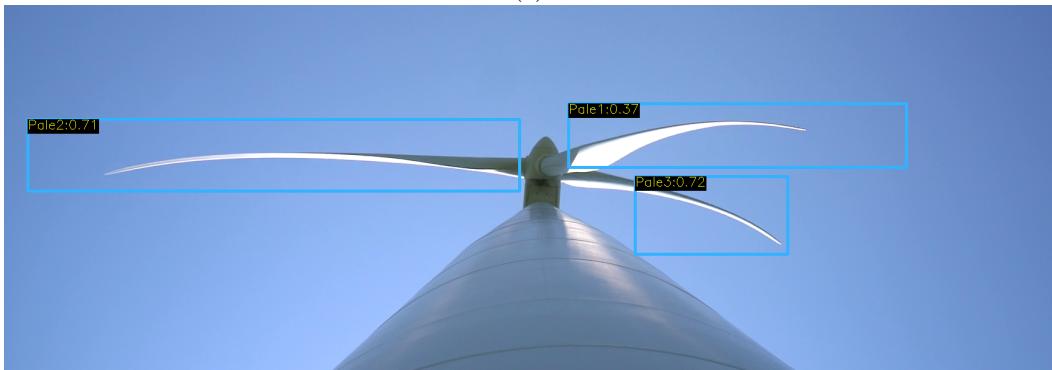
Si la détection reste infructueuse après le traitement secondaire, une estimation manuelle de la position de "Pale 1" est réalisée, en se basant sur les positions détectées de "Pale 2" et "Pale 3".

3.4.7 Finalisation des Résultats

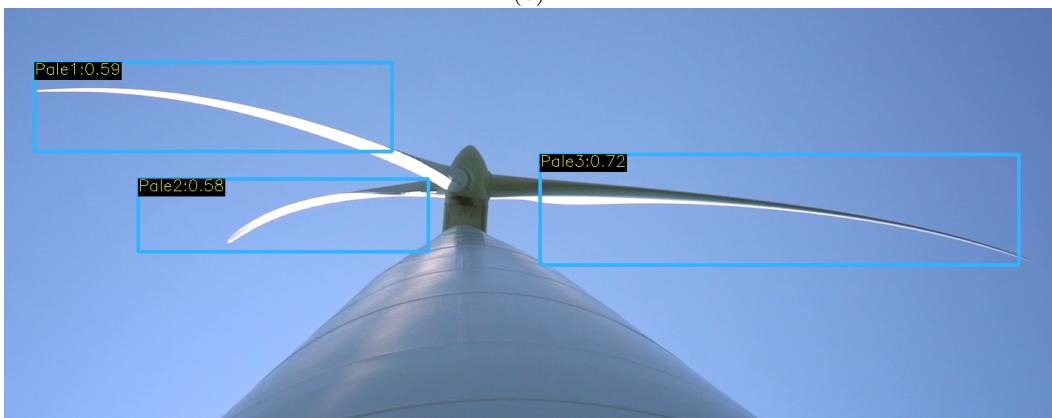
Ce processus de traitement postérieur renforce la robustesse du modèle, assurant ainsi une détection fiable des objets d'intérêt dans chaque image de la vidéo. Les résultats finaux sont ensuite sauvegardés dans un fichier vidéo de sortie.



(a)



(b)



(c)

Figure 8: Résultats du code C++ appliqué sur la vidéo MAH02363.MP4.