|  |
| --- |
| **KIEFFER** |
| **Thomas** |

|  |
| --- |
| **CHEVALLIER** |
| **Arthur** |

**Noms : Prénoms : Gr : 31**

**Partie 1**

**Analyse des images :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Méthode de calcul – Outils utilisés | *Key\_16\_17\_D* | *Key\_16\_17* | *Key\_16\_17\_L* |
| Contraste | On utilise la formule Contraste = (Imax-Imin)/(Imax+Imin) avec I le niveau de gris | 1 | 1 | 0.68 |
| Dynamique | Dynamique = niveau gris max – niveau gris min (avec l’histogramme) | 135 – 0 = 135 | 255-0 = 255 | 255-48 = 207 |

**Seuils manuels :** (min-max)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | *Key\_16\_17\_D* | *Key\_16\_17* | *Key\_16\_17\_L* | *Key\_8\_9* | *Key\_12\_13* | *Key\_14\_15* | *Key\_20\_22* |
| Valeurs | 3-135 | 20-255 | 150-255 | 15-255 | 10-255 | 113-255 | 108-255 |

**Seuils automatiques :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Méthode(s) | *Key\_16\_17\_D* | *Key\_16\_17* | *Key\_16\_17\_L* | *Key\_8\_9* | *Key\_12\_13* | *Key\_14\_15* | *Key\_20\_22* |
| Menu Image > Adjust > Auto Thresold | Otsu | Otsu | Otsu | Huang | MaxEntropy | Moments | Li |
| Valeurs | 37-135 | 101-255 | 157-255 | 15-255 | 11-255 | 108-255 | 74-255 |

**Calibration :**

*Résolution en x :* 1.956px/mm *Résolution en y :* 1.956px/mm *Précision : 0.5mm*

**Prétraitement(s) :**

|  |
| --- |
| Avant d’appliquer l’algorithme, on étalonne la mesure avec l’image Etalon\_45 (implémenté dans la macro) et on a  mesuré la résolution (pour en faire une constante dans la macro de mesure).  Ensuite, on utilise la méthode de Otsu afin de faire une détection de contour puis un seuillage pour binariser l’image.  Hypothèses :   * Etalonnage avant la mesure * Résolution constante * Les clefs doivent être positionnés à peu près au même endroit (horizontalement, d’après l’énoncé). |

**Méthode de calcul (algorithme) :**

|  |
| --- |
| * On trace une ligne sur l’image. * On se rend compte que la ligne traverse différents contours qui induisent des pics à 255 sur le profil de ligne. * On différencie les groupes de contour : bords extérieurs de la clef et bords intérieurs. * On mesure la distance entre les 2 bords intérieurs (bordures de l’arrondi de la clef) : c’est la largeur d’écartement.   La distance est normalement obtenue avec x1-x2 les coordonnées des bords, mais on obtient un résultat en pixel : il faut la diviser par la résolution de l’image.  On a aussi remarqué que l’image pouvait contenir des pixels en escalier (2pixels à 255 à la suite sur la ligne) dans le contour. On doit donc les prendre en compte dans l’algorithme en ne gardant que le premier pixel afin de calculer le bon écart. |

**Résultats :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | *Key\_16\_17\_D* | *Key\_16\_17* | *Key\_16\_17\_L* | *Key\_8\_9* | *Key\_12\_13* | *Key\_14\_15* | *Key\_20\_22* |
| Valeur mesurée | 15.5mm | 15.5mm | 13.5mm | 7,5mm | 13mm | 14mm | 20mm |

|  |
| --- |
| Nous avons effectué certaines hypothèses afin de simplifier le fonctionnement de l’algorithme. Dans le cadre d’une industrialisation du processus, elles sont faciles à suivre si on fait défiler les clefs sous une caméra linéaire par exemple.  Pour conclure, les résultats de notre algorithme sont plutôt très bons, en l’état nous ne pouvons pas savoir si les quelques écarts constatés sont dus à une imprécision de l’algorithme ou dans le nom de la clef.  Nous pouvons toutefois rajouter une hypothèse sur la luminosité : on peut voir dans les valeurs obtenues qu’avec un éclairage trop lumineux les résultats sont plus éloignés de la réalité (car les niveaux de gris sont faussés, en particulier aux contours). Cela reste une contrainte peu restrictive puisque la luminosité peut être contrôlable dans l’environnement d’acquisition. |

Conclusions – Commentaires :

**Partie 2**

**Binarisation (si utilisée) :**

Méthode choisie : Otsu

|  |  |
| --- | --- |
| *Images testées* | *Valeur de seuil calculée* |
| Clef.tif | 139 |
| Rondelle.tif | 147 |

**Pré-traitements éventuels (et paramètres de réglage) :**

|  |
| --- |
| On a fait une binarisation en utilisant un seuillage automatique en utilisant la méthode de Li Dark pour séparer les objets du fond (en gardant un fond noir). Ensuite, nous avons également effectué un Outline et une Dilatation afin de récupérer tous les contours d’objets (en blanc sur fond noir).  Pour les caractéristiques, on a utilisé le Shape Filter fourni par IJ Blob. |

**Caractéristiques choisies :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Feature | *Clé* | | *Rondelle* | | *Pièce* | | *Dé* | |
| Min | Max | Min | Max | Min | Max | Min | Max |
|  |  |  |  |  |  |  |  |  |
| Périmètre | 2,382 | 3,813 | 0,614 | 0,754 | 0,614 | 1,108 | 0,555 | 0,582 |
| Surface | 0,071 | 0,210 | 0,027 | 0,043 | 0,031 | 0,100 | 0,022 | 0,025 |
| Circularité | 66,701 | 80,218 | 13,230 | 16,674 | 12,154 | 12,497 | 13,356 | 13,817 |
| Nombre de trous | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 6 |
| Maximum inscriped circle diameter | 0,97 | 0,196 | 0,061 | 0,094 | 0,192 | 0,350 | 0,050 | 0,105 |

**Lot de données :** L’étalon sera toujours considéré comme étant une pièce. En sachant cela, on pourrait ne pas le compter comme une erreur dans les résultats suivants.

*Images testées :* Objet1 à 7 (dans l’ordre)

Nombre d’objets : détectés / présents :

*#Clés : 3/3 #Rondelles : 2/2 #Pièces : 2/1 #Dés : 0/0*

*#Clés : 3/3 #Rondelles : 3/3 #Pièces : 3/2 #Dés : 0/0*

*#Clés : 3/3 #Rondelles : 2/2 #Pièces : 2/1 #Dés : 0/1*

*#Clés : 2/2 #Rondelles : 2/2 #Pièces : 2/1 #Dés : 3/3*

*#Clés : 1/1 #Rondelles : 2/2 #Pièces : 2/1 #Dés : 1/1*

*#Clés : 1/1 #Rondelles : 1/1 #Pièces : 2/1 #Dés : 2/2*

*#Clés : 1/1 #Rondelles : 1/1 #Pièces : 2/1 #Dés : 0/1*

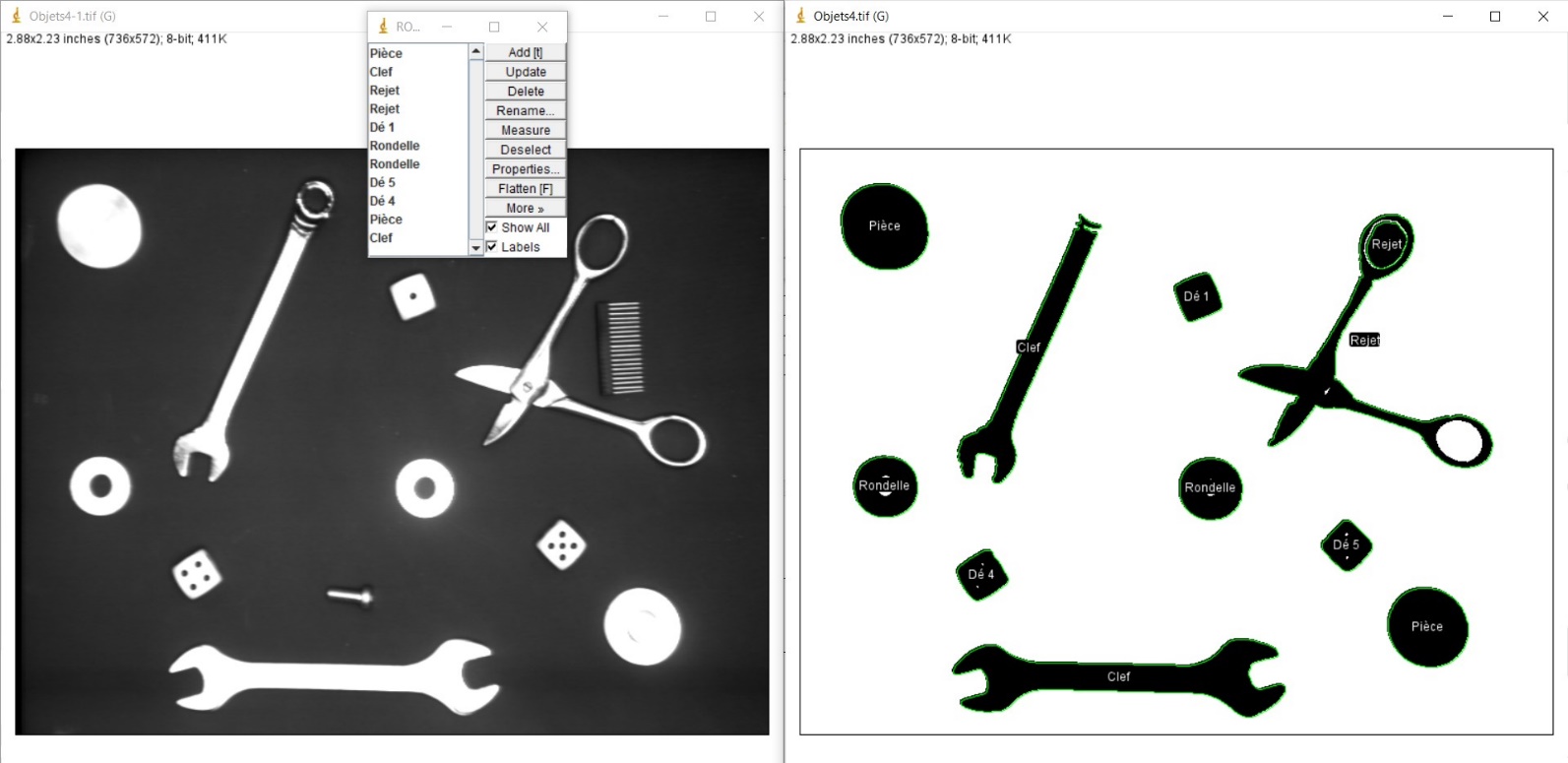
**Matrice de confusion : Pour l’image Objet2\_L.tif. On considère que l’étalon est bien placé dans pièce.**

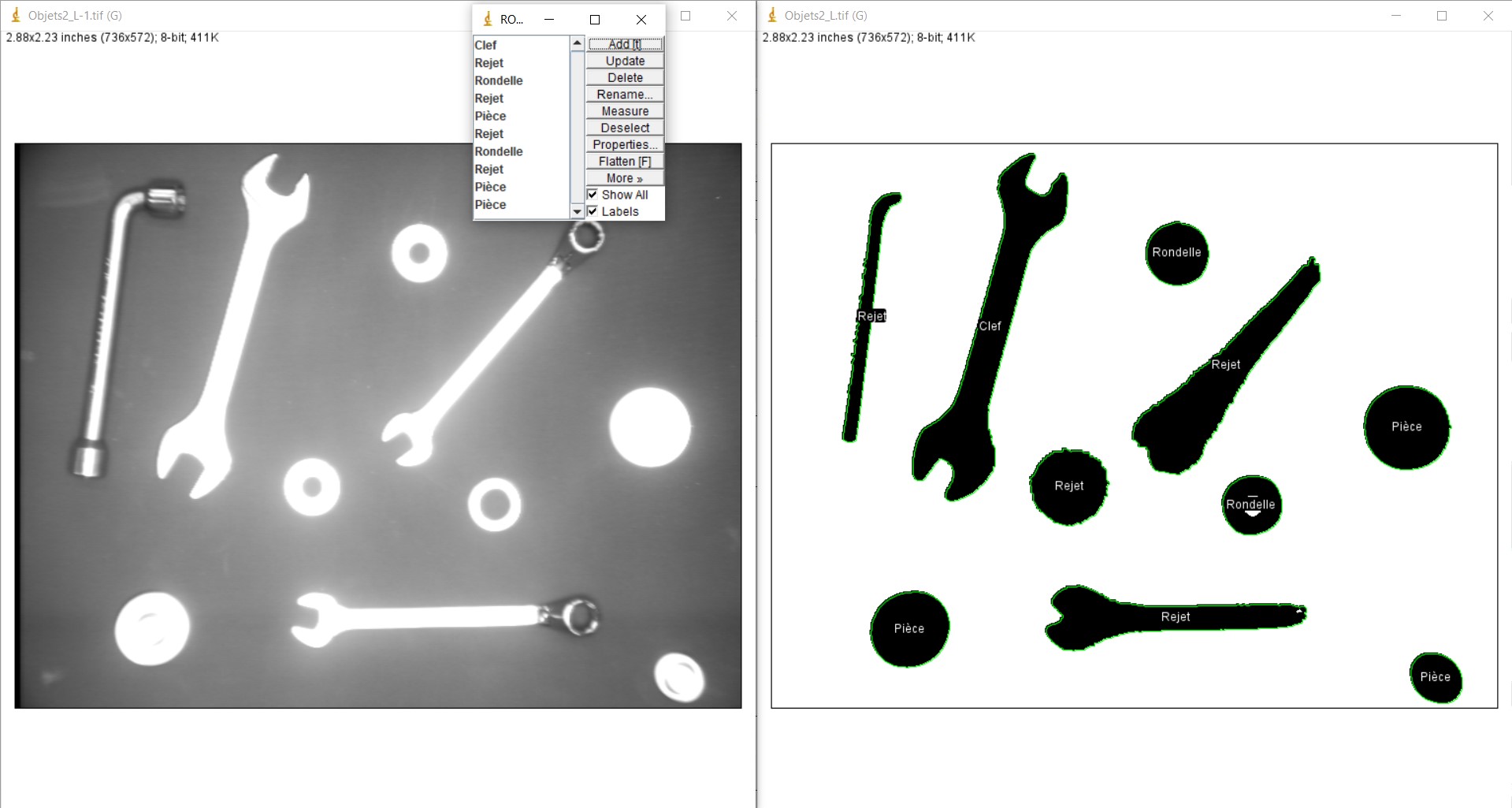
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Clé** | **Rondelle** | **Pièce** | **Dé** | **Rejet** |
| **Clé** | **1** | **0** | **0** | **0** | **2** |
| **Rondelle** | **0** | **2** | **0** | **0** | **1** |
| **Pièce** | **0** | **0** | **3** | **0** | **0** |
| **Dé** | **0** | **0** | **0** | **0** | **0** |
| **Rejet** | **0** | **0** | **0** | **0** | **1** |

|  |
| --- |
| Nous pouvons calculer la précision :   * Pour les clefs : 100% * Pour les rondelles : 100% * Pour les dés : 100% - l’échelon (soit 66%) * Pour le rejet : 1/3 = 33%   Les objets sont toujours soit mis dans la bonne classe soit dans la classe de rejet, ce qui permet d’éviter de faux positifs et donc par exemple un mauvais tri des éléments**.** |

Commentaires :

**Exemples d’images résultats (si noms de classes affichés) : Pour Objet4 puis Objet2\_L**





**Limites de votre méthode :**

|  |
| --- |
| L’étalon pose toujours problème pour la classification puisque ce n’est pas possible de le distinguer par rapport aux pièces (en tout cas avec nos caractéristiques). Cela n’est cependant pas important si l’étalonnage se fait sur une image à part, ce qui permettrait d’enlever l’étalon des images à analyser.  Les résultats sont également moins bons avec une luminosité élevée, ce qui est aussi gérable dans un environnement industriel. |

**Conclusion :**

|  |
| --- |
| Pour conclure, notre classificateur semble assez performant. Les résultats sur les images acquises sont très bons.  Si on imagine que ce classificateur sera utilisé dans une optique de tri des objets, la classe de rejet permet de ne pas attribuer des éléments dans la mauvaise classe et pourrait signifier la nécessité d’une vérification humaine, sur un nombre très réduit d’objets.  Dans un milieu adapté, avec une bonne luminosité et un environnement d’acquisition constant, les résultats de classement seront encore meilleurs. |