2014-2015

ENSG

Maxime QUENON
Soukaïna ZAROUGUI
Rudolf MILLET

[Modèle de classification d'efficience de la corrélation dense]

Télédétection

Projet Forcalquier

Sommaire

I. Introduction 3
II. Calibration 4
III. Prise de vue 9
IV. Traitement 11
V. Classification 13
1. Apprentissage 13
2. Tests 15
VI. Conclusion 18
Tables 19
Bibliographie 19
Annexes 21

I. Introduction

La reconstitution 3D est un processus en 2 étapes. Tout d'abord, elle effectue le calcul de pose par la détection automatique du nuage de points de liaison et des différents points de vue de l'appareil utilisé. Ensuite elle procède à une densification de ce nuage de points par la corrélation dite dense.

Cette corrélation dense étant sensible à plusieurs paramètres, il peut en résulter des zones bruitées sur le produit final, c'est-à-dire des zones où elle ne s'est pas bien déroulée.

Notre sujet a donc pour but de détecter automatiquement ces zones bruitées et de fournir, dans un cas pratique, à un utilisateur d'appareil photos, les informations nécessaires à la corrélation dense pour le succès d'une reconstitution 3D.

Comment donc anticiper a priori la qualité d'une corrélation ?

Nous nous sommes alors lancés dans la recherche de différentes textures (bois, bitume, carrosserie…) prise avec différents paramètres (luminosité, distance et angle de prise de vue…) pour déterminer dans chaque cas la qualité de la corrélation dense. S'en suivra une classification puis des tests pour vérifier nos résultats.

Nous présenterons dans la suite les méthodes de mise en place utilisées, de la calibration à la classification en passant par les prises de vue des textures, le stockage des images et des données sous forme de tableau et le traitement des cartes de confiance de la reconstitution 3D.

Figure 1. Reconstitution d'un nuage de points .ply

II. Calibration

L'Ecole Nationale des Sciences Géographiques a mis à notre disposition un appareil photo Canon EOS 5D MARK III que nous avons accompagné personnellement d'un Nikon D3200. Les deux appareils, possédant chacun une calibration différente, serviront à comparer les résultats obtenus en fonction de leur paramétrage.

Nous nous sommes également servis de clichés issus de la prise de vue par drone effectuée au-dessus de Peyruis et dont nous avons calculé ou extrait certains paramètres de manière approximative dans la mesure du possible à savoir la résolution, la focal et l'ISO.

La calibration est une étape longue et importante demandant beaucoup de minutie, car c'est le point de départ de toutes nos prises de vue de laquelle découleront les résultats de calibration.

Elle consiste à choisir un endroit très angulaire et assez texturé, qui ressemble à un tétraèdre ou à un escalier, que nous allons cibler pour une prise de vue.

Chaque prise de vue est caractérisée par les différents paramètres appliqués à l'appareil. Ainsi, nous jouons sur l'ouverture du diaphragme, l'ISO c'est-à-dire la sensibilité à la lumière et la distance à la cible visée.

La calibration a donc été précédée par plusieurs séries de photos. De celles-ci nous avons choisi les paramètres que nous allions fixer tels que l'ISO, l'ouverture et le temps de pose de chaque appareil afin d'éviter au maximum les flous de profondeur de champs. En prenant en compte la luminosité, nous nous sommes contraint à rester en dessous d'un temps de pose d'1/60ème de seconde pour éviter un autre flou gênant, le flou de bougé, pour le traitement qui suivra.

Tableau 1. Paramètres fixés

	Canon	Nikon JPG	Nikon NEF	Drone
Focal en mm	**35**	**18**	**18**	**27**
Ouverture	**11**	**9**	**9**	**?**
ISO	**200**	**100**	**100**	**200**

Maintenant que les paramètres sont fixés, nous figeons les objectifs à l'aide de ruban adhésif après avoir placé la mise au point presque vers l'infini de façon à ce que les prises de vue soient nettes de 5 à 20 m.

Nous nous sommes fixés un nombre minimum de 7 photos par prise de vue avec un recouvrement d'au moins 60%. Les schémas de prise de vue (en croix, en ligne, convergente, parallèle…) sont différents en fonction du contexte choisi lequel ne nous laissant parfois que peu de choix, l'endroit étant trop angulaire ou pas assez large.

C'est là qu'intervient le logiciel **Micmac** servant pour la reconstruction de scènes en 3 dimensions à partir d'images. Nous nous en servons par ligne de commande dans un terminal.

Pour la calibration nous lançons :

mm3d Tapioca All ".*JPG" 2500

Ceci va créer un fichier appelé « AutoCalXXX.xml » qui contiendra notamment la focal en pixel et les coefficients de distorsion.

Ces coefficients de distorsion sont utilisés dans la formule qui suit et qui donne en fonction de la distance au PPS (Projeté du sommet de la prise de vue sur l'image) la distorsion subit au point :

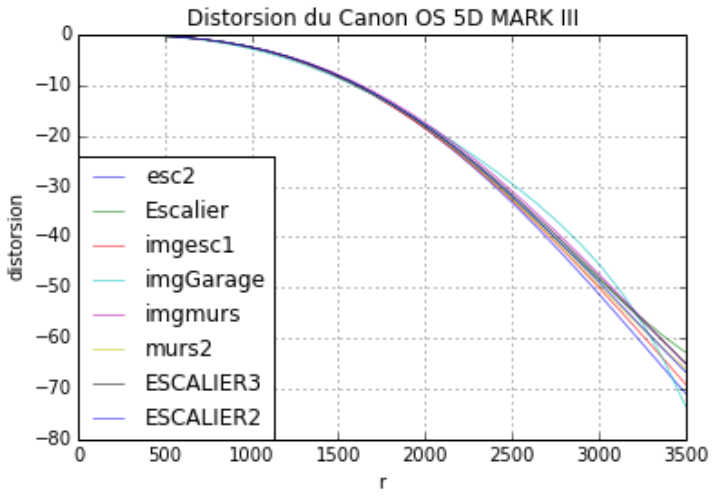
$$dr = ar^3 + br^5 + cr^7$$

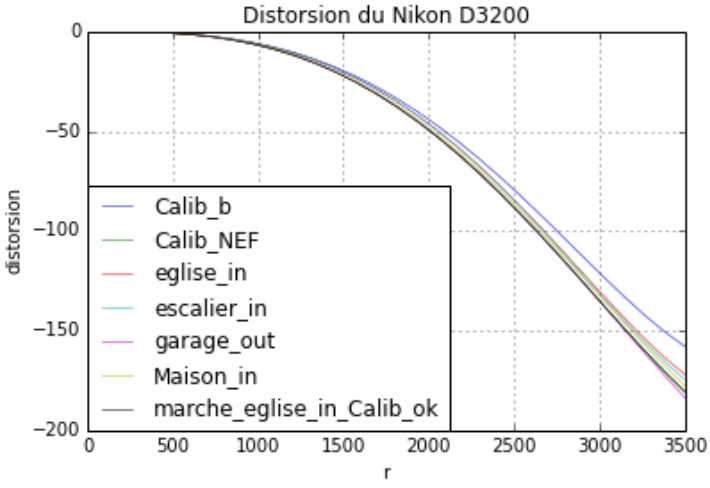
Tableau 2. Paramètres des fichiers .xml

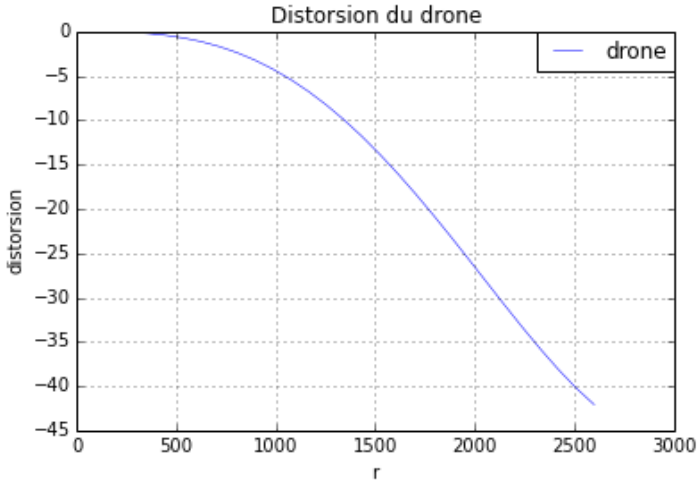
	Canon	Nikon JPG	Nikon NEF	Drone
Focal en pixel	**5494.73**	**4748.77**	**4748.81**	**3860.23**
a	**-2.66e-09**	**-7.08e-09**	**-7.07e-09**	**-4.79e-009**
b	**1.12e-16**	**2.28e-16**	**2.20e-16**	**3.86e-016**
c	**-1.45e-24**	**8.24e-25**	**1.10e-24**	**-4.77e-024**

Les différents fichiers .xml créés sont ensuite traités par un script python mis au point à l'aide du paquet numpy pour une meilleure visualisation et comparaison : « distorsionWithPolynom.py »

Graphiques :





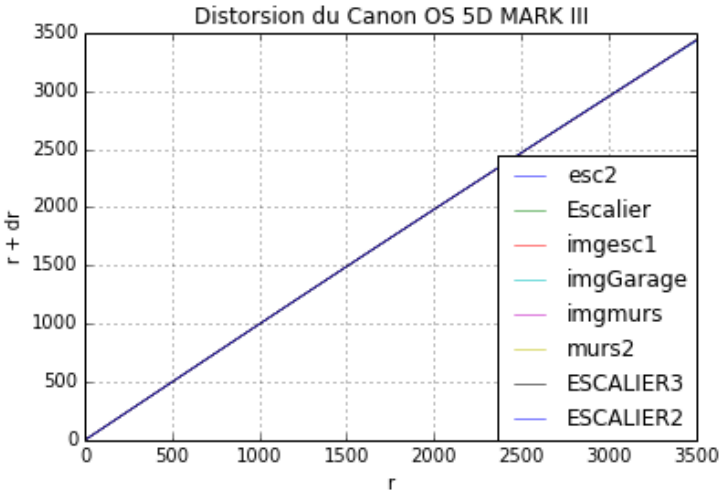


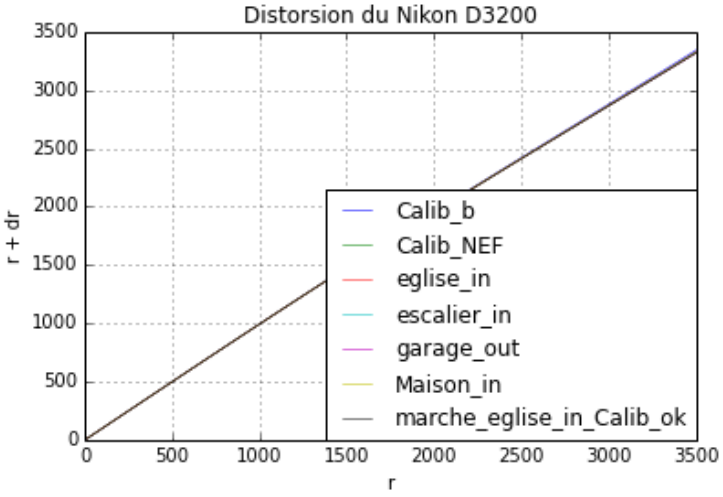
Ces graphes représentent les distorsions de chaque appareil en fonction de la distance au centre de distorsion (PPS).

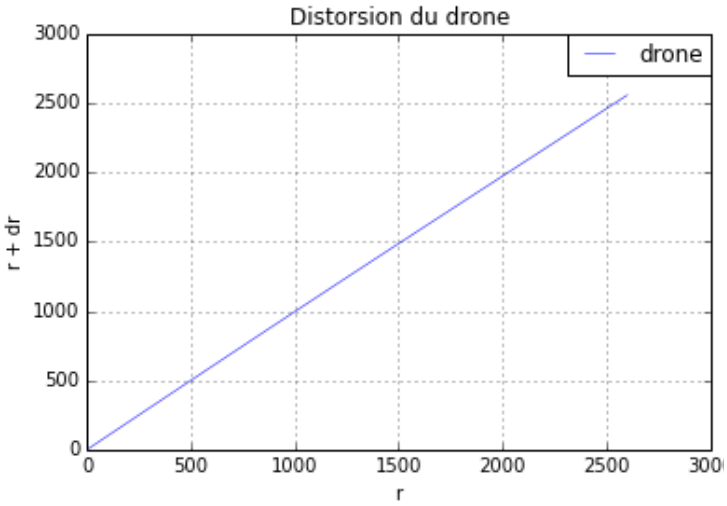
A partir d'une certaine distance, le résultat n'est plus physique car nous sortons des dimensions de l'image.

Bien que la meilleure calibration est choisie en fonction du schéma de prise de vue, de l'endroit choisi, en l'occurrence les grands escalier derrière la cathédrale de Forcalquier et du nombre de points de liaisons, ces graphes permettent de vérifier quelle calibration sort du lot et si le domaine « physique » se situe bien avant l'extrémum.

Finalement ce que nous pouvons dire c'est que le drone possède une distorsion moindre de l'ordre de 45 pixels certainement dû à la petite dimension de clichés. Le Nikon a une grande distorsion de l'ordre de 200 pixel contre 80 pixels pour le Canon qui a pourtant la même dimension d'image. La différence se joue au niveau des paramètres de chaque appareil.







Ces graphes représentent la distance avec la distorsion du point en pixels en fonction de sa distance théorique toujours à partir du même centre (PPS).

Ils permettent juste de vérifier que la distorsion n'est pas aberrante mais que la courbe est bien strictement monotone.

Ces vérifications sont appuyées par leur dérivé qui sont toutes positives.

Tableau 3. Distorsion en borure d'image de chaque calibration

	Dossiers	dr
CANON	ESC2	-69,37
	ESC1	-67,54
	garage	-70,45
	murs	-63,68
	MUR2	-64,87
	ESCALIER2	-65,30
	ESCALIER3	**-63,60**
NIKON	maison	-210,52
	escalier	-178,85
	eglise	-173,46
	calib_b	-159,70
	marche_eglise_in_Calib_ok	**-211,98**

A la suite de cette analyse nous avons donc choisi les fichiers .xml qui nous convenaient le mieux à savoir celui du dossier « **ESCALIER3** » pour le Canon et celui du dossier « **marche_eglise_in_Calib_ok** » pour le Nikon.

En effet, ces deux prises de vue ont été prises dans un endroit présentant différents plans. De plus, les schémas de prise de vue sont assez complètes et possède un bon recouvrement malgré la grande longueur des marches. Les photos ont été prises à des distances différentes. Et enfin nous avons obtenu un nombre suffisant de points de liaison bien placés sur les marches dû à leur texture.

Les graphes ont permis de constater que les courbes correspondantes n'étaient pas en dehors du lot et encore moins aberrantes.

III. Prise de vue

L'objectif premier est de mettre sur pied une base de donnés regroupant différentes textures pour ensuite les comparer. Nous avons donc cherché à prendre en photo des textures ayant leurs propres caractéristiques (plane, cylindrique, rugueuse, lisse…) puis en variant les paramètres propres à la prise de vue (angle, distance, ciblage…). D'autres paramètres visant à reproduire des contextes climatiques interviennent (sec, humide, ombragé…).

La prise de vue demande une mise en place délicate et réfléchie. Certaines ont nécessité un trépied à cause d'une luminosité faible ce qui aurait pu occasionner un flou de bougé si la vitesse d'obturation avait été trop long.

Au final, nous nous sommes servi d'un tableau pour répertorier toutes les textures, les paramètres et ainsi attribuer les résultats obtenus au bonne prise de vue.

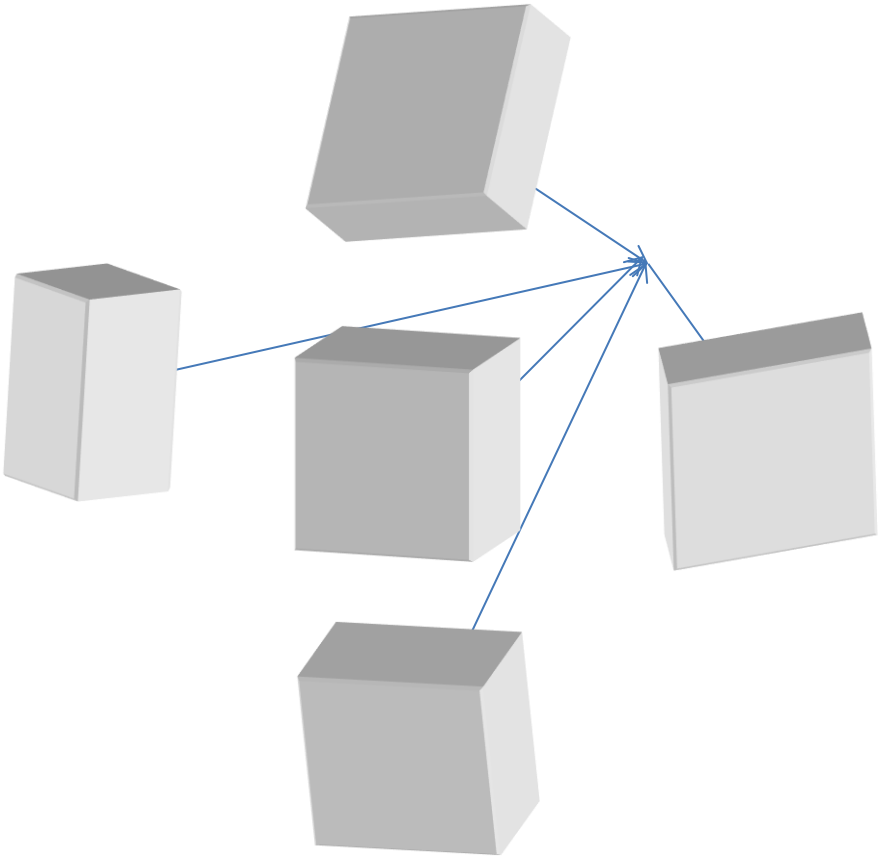


Figure 2. Prise de vue convergente en croix

Tableau 4. Présentation des différentes textures ainsi que des différents paramètres

Textures	Ciblage	Angle de PDV	Eclairage	Etat
Bitume	Parallèle	Oblique	Ensoleillé	Humide
Bois de l'armoire	Convergente	Face	Sombre	Sec
Carrosserie			Ombragé	
Ecorce			Légèrement sombre	
Ecorce moussue				
Gazon				
Mur crépis				
Muret				
Panneau en liège				
Rouille				
Sol cailloux				
Sol carrelage				
Tuile				
Végétation basse				
Végétation haute				
Encre tableau				
Sol en béton				
Mur blanc peinture				
Tableau blanc sale				

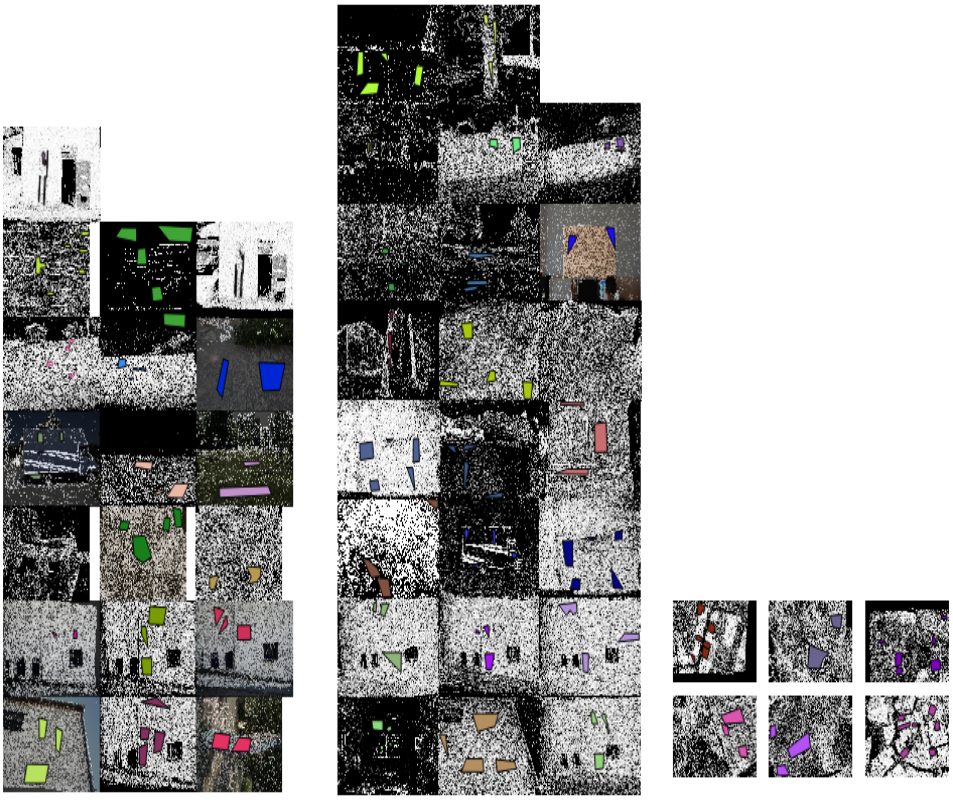


Figure 3. Base de données d'images de texture et de corrélation sur QGIS

IV. Traitement

Une fois la prise de vue terminée, nous passons à la phase de traitement avec Micmac. Nous nous plaçons dans le dossier où se trouve les images et nous définissons son image maîtresse. Le traitement suit le schéma suivant :

-Recherche de points de liaison :

mm3d Tapioca All ".*JPG" 2500

-Mise en place des images :

mm3d Tapas RadialStd ".*JPG" InCal=Ori-calib/ Out=MEP

-Création des nuages de points de liaison :

mm3d Apericloud ".*JPG" MEP

-Création des cartes de corrélation :

mm3d Malt GeomImage ".*(AA|BB|CC|00|DD|EE).JPG" OriMEP/ Master="XXXX0000.JPG"

Ensuite nous nous plaçons dans le dossier MM-Malt-Img-AAAA0000 et nous lançons :

-Création de l'image avec le relief ombragé :

mm3d GrShade Z_Num8_DeZoom1_STD-MALT.tif ModeOmbre=IgnE Mask=AutoMask_STD-MALT_Num_7.tif

-Création du nuage dense de points :

mm3d Nuage2Ply NuageImProf_STD-MALT_Etape_8.xml Attr=../AAAA0000.JPG

La deuxième étape avec "Tapas" nous fournit un résidu que nous devons qualifier en fonction du pourcentage de points de liaison conservé et du nombre de points de liaison. Par exemple, il se peut que nous ayons un bon résidu mais avec très peu de points de liaison, de l'ordre de la centaine.

Ensuite arrivent les résultats de la corrélation normalisée, calculés à partir de l'histogramme d'une zone jugée bien corrélée, à savoir la moyenne qui doit être la plus élevée possible et le sigma, c'est-à-dire l'écart type, qui lui doit être le plus bas possible.

Et en ce qui concerne les surfaces planes, nous nous efforçons de calculer le bruit du nuage de points en construisant un plan moyen dans ce nuage. Nous en déduisons l'écart à ce plan à 2 sigma prêt. Ce processus s'effectue avec le logiciel **Cloud Compare** qui prend en entrée un fichier contenant un nuage de points. Nous y créons dans la surface intéressante, un plan à partir duquel nous mesurerons l'écart aux points. Le résultat peut s'observer sur la figure 5, les zones rouges sont les plus éloignées du plan contrairement aux zones vertes.

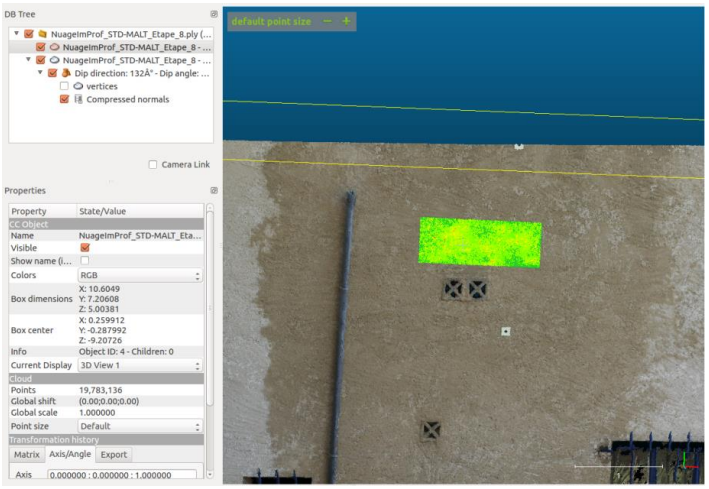


Figure 4. Cloud Compare (mur mouillé)

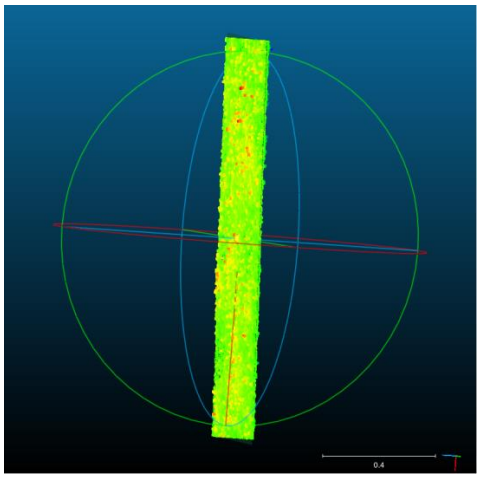


Figure 5. Cloud Compare écart au plan (mur mouillé)

Figure 6. Carte de corrélation (végétation basse) .tif

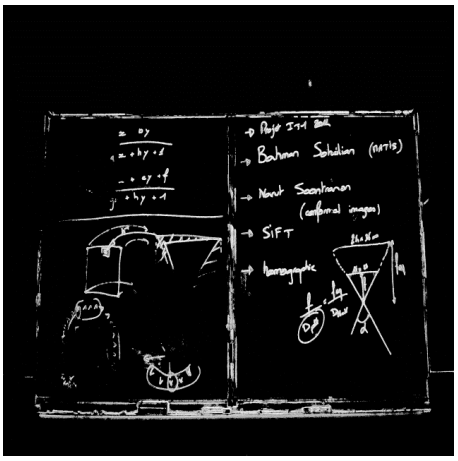


Figure 7. Carte de corrélation (tableau blanc) .tif

Tous ces résultats sont stockés dans un tableau et associés à chaque texture, appareil et paramètres respectifs. S'en suit une analyse relative de ces résultats les uns par rapport aux autres en prenant compte de tous les différents paramètres. En effet nous pouvons avoir un excellent résultat de corrélation mais un très grand écart du plan associé. C'est le cas notamment avec la texture de l'armoire prise par le CANON, il possède un résultat moyen de corrélation de 0,81 mais un écart au plan de 140 mm à 2 sigma près.

Cette organisation des résultats sous forme de base de données permet par la suite, une classification de manière méthodique.

V. Classification

La classification s'effectue en 2 étapes et nécessite l'utilisation de 2 logiciels :

-**QGIS** pour la mosaïque d'images. C'est avec ce logiciel qu'on fait l'apprentissage de nos couches shapefile.

-**Orfeo ToolBox** pour le calcul des coefficients des images Haralick et la classification.

1. Apprentissage

Nous avons donc tout d'abord procédé à un seuillage dans le tableau des données de prises de vue. A l'aide de graphiques qui représentaient la moyenne du coefficient de corrélation de chaque appareil en fonction de la texture, et du résultat de l'écart au plan, pour consolider notre premier choix, nous avons pu déterminer de manière relative pour chaque appareil, un seuillage.

Tableau 5. Seuillages

	Seuillage de la moyenne du coefficient de corrélation
Canon EOS 5D MARK III	**0,83**
Nikon D3200	**0,79**
Drone	**0,7**

La détermination des seuillages s'est faite de manière relative car chaque prise de vue a sa propre résolution ce qui, pour le drone qui en possédait une entre 10 et 21 mm, explique pourquoi nous avons choisi un seuil plus bas que les autres. De plus, nous pouvons trouver dans le tableau, des textures exclues malgré leur bonne moyenne de corrélation à cause de leur trop grand écart au plan. C'est en effet le cas du bois de l'armoire. Il faut savoir que rien n'est absolu dans l'analyse de nos résultats.

D'autres textures possédant une bonne moyenne de corrélation n'ont pas été retenues à cause de leur type de fichier .nef non géré par QGIS et OTB.

Pour en revenir à QGIS, nous avons au début organisé notre environnement de travail en créant des fichiers .tfw et .jgw respectivement pour les fichiers de corrélation (.tif) et les fichiers images des clichés maîtres de nos prises de vue (.jpg). Ceux-ci nous ont permis de jouer sur les coordonnées de manière à les placer en mosaïque en coordonnées positives et de les regrouper par appareil en plus de superposer les corrélations et les images couleurs. Le but est de pouvoir dessiner des polygones délimitant des zones caractéristiques à l'aide des cartes de

corrélation sur les images couleurs. Grâce au seuillage, les zones blanches sur les corrélations correspondent aux zones qui ont bien corrélées et les plus sombres, le contraire. Nous créons des couches .shp aussi bien pour des zones qui marchent que pour des zones qui ne marchent pas en attribuant aux couches un attribut « class » 1 ou 0.

La classification effectuée par le logiciel Orfeo ToolBox a également besoin de fichiers **Haralick** qui apportent des critères supplémentaires permettant de mieux la discriminer. La création du fichier Haralick se lance à partir de la commande suivante :

otbgui_HaralickTextureExtraction

Elle s'effectue à partir de l'image maîtresse de chaque dossier, et sur le canal 2, c'est-à-dire le canal vert qui possède plus de texture. Ainsi le fichier image_maîtresse_haral.tif, est un fichier image texturé à 8 canaux. Et comme pour les fichiers image maîtresse .jpg/nef et les cartes de corrélation .tif, nous avons dû créer des fichiers de géoréférencement .tfw car Orfeo ToolBox n'accepte que des coordonnées positives.

Nous établissons ensuite 2 listes de fichiers. Une pour les fichiers Haralick et une autre pour les fichiers .shp. Puis ces 2 listes sont traitées par un script python « **python_rf.py** » écrit pour l'occasion. Ce script effectue un apprentissage avec la méthode des **random forests** et à l'aide du logiciel OTB. Ce processus a également en sortie 2 fichiers: la matrice de confusion en .csv et le fichier modèle d'apprentissage en .txt.

La matrice de confusion est une matrice carrée :

Tableau 6. Matrice de confusion

Estimés / Réels	0	1
0	$\begin{pmatrix} TN & FP \\ FN & TP \end{pmatrix}$	
1		

TN : True Negative occurrence de vrai dans la class dans les non corrélables

FN : False Negative occurrence de faux dans la class dans les non corrélables

FP : False Negative occurrence de faux dans la class dans les corrélables

TP : True Positive occurrence de vrai dans la class dans les corrélables

La précision est la proportion de solutions trouvées qui sont pertinentes.

Formule : $P = \frac{TP}{TP+FP}$

Le rappel est la proposition des solutions pertinentes qui sont trouvées.

Formule : $R = \frac{TP}{TP+FN}$

Le F-mesure est la moyenne harmonique de la précision et du rappel.

Formule : $F = \frac{2PR}{P+R}$

Le Kappa est le coefficient qui sert à évaluer la matrice de confusion par rapport à une matrice de confusion dont les valeurs sont aléatoires.

Formule : $Kappa = \frac{N(\sum_1^r x_{ii}) - \sum_1^r (x_{i+} * x_{+i})}{N^2 - \sum_1^r (x_{i+} * x_{+i})}$

où N est le nombre total d'observation, r est le nombre de rangée de la matrice, x_{i+} et x_{+i} sont les totaux marginaux respectivement de la rangée j et de la colonne i et les x_{ii} sont le nombre d'observation de la colonne i à la rangée i.

2. Tests

Les tests consistent à classifier des images Haralick. Nous ne pouvons classifier d'autres types d'image car nous entrons le fichier modèle d'apprentissage crée précédemment contenant 8 canaux comme l'image Haralick.

Le logiciel OTB correspondant se lance avec la commande qui suit :

otbgui_ImageClassifier

Cela nous renvoie en sortie une image classifiée contenant les classes établies à savoir les « class » 0 et 1, respectivement les zones susceptibles d'être mal corrélées et bien corrélées.

Nous pouvons considérer cela comme un test car au final nous comparons les cartes de corrélation des textures déjà existantes dans la base de données avec ces images classifiées et nous vérifions si les zones blanches (1) et noires (0) correspondent.

Une bonne correspondance résulte d'une bonne vectorisation sur QGIS des cartes de corrélation et d'un bon apprentissage sur OTB. Le contraire peut demander de reprendre tout le processus ou simplement de suréchantilloner nos couches .shp.

Malheureusement, la classification ne s'est pas déroulée comme prévu. La capacité des ordinateurs mis à notre disposition ne nous a pas permis de classifier tous nos Haralick et shapefile en une seule fois. Pour remédier à cela, nous avons écrit un script python qui les traite indépendamment. Finalement, nous avons pu classifier 10 Haralicks et 10 shapefiles en un seul processus. Nous avons donc dû nous résigner à établir les différents jeux d'apprentissage suivant :

- **NIKON** :
  - **Texture de mur unique** : [crépis ; humide/sec ; muret en pierre éclairé/ombragé ; peinture intérieur]
  - **Mix** : [bois ; bitume ; carrosserie ; muret ensoleillé ; mur crépis humide/sec ; sol gravier ; sol béton ; tronc à l'ombre]
  - **Spécifique à l'image** : [mur crépis sec ; bois ; gravier ; béton ; végétation basse]
  - **Arbre** : [mur crépis sec ; bois ; gravier ; béton ; végétation basse ; détection d'arbre]

Figure 8. Correl armoire_soleil.tif

Figure 9. test_arbre_S.png

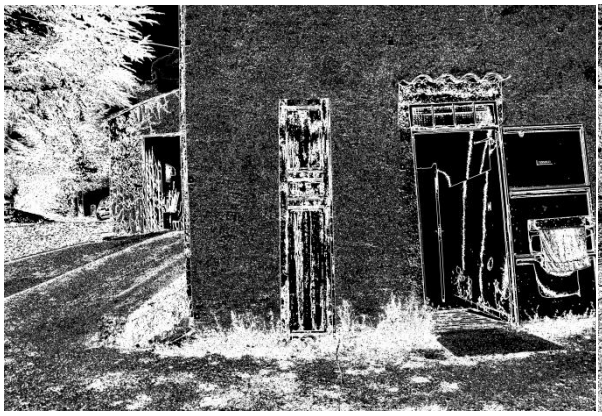


Figure 10. essai_mix_S.png

Figure 11. essai_mur_S.png

Tableau 7. Jeux d'apprentissage testés sur armoire_soleil.tif

	Précision	Rappel	F-score	Kappa	Matrice de confusion
Arbre	0.75	0.46	0.57	0.31	$\begin{pmatrix} 4611 & 5317 \\ 1503 & 8464 \end{pmatrix}$
Spécifique à l'image	0.78	0.57	0.66	0.40	$\begin{pmatrix} 5592 & 4178 \\ 1552 & 7483 \end{pmatrix}$
Mix	0.85	0.71	0.78	0.48	$\begin{pmatrix} 7032 & 2861 \\ 1196 & 478 \end{pmatrix}$
Texture de mur unique	0.79	0.82	0.81	0.56	$\begin{pmatrix} 5798 & 1285 \\ 1513 & 44 \end{pmatrix}$

- **CANON** :
  - **classif1** : [camion ; muret ; herbe ; écorce ombre]
  - **classif2** : [camion ; muret ; herbe ; écorce ; tableau]
  - **classif3** : [camion ; muret ; herbe ; tableau ; écorce ; mur]
  - **classif4** : [camion ; muret ; herbe ; tableau ; écorce moussue ; mur]
  - **classif5** : [camion ; herbe ; écorce moussue ; mur ; écorce ; porte soleil]

VI. Conclusion

Bien qu'il s'agisse d'un sujet léger sans grande difficulté majeure, ce fut une recherche qui nous a demandé beaucoup de patience, d'imagination et de persévérance. En effet, à commencer par la calibration, nous avons dû mettre en place différentes techniques devant répondre à la grande diversité que devait offrir notre base de données.

La calibration nous a demandé deux bonnes journées et plusieurs séries de photos. Les traitements sur ordinateur de la calibration et des textures ont pris un certain temps que nous avons dû gérer dans notre emploi du temps. Ce dernier étant optimisé avec le contexte climatique de tous les jours.

D'autres difficultés sont apparues au cours du processus. En effet, le fait de manipuler toutes les images couleurs et les cartes de corrélation dans QGIS ralentissait beaucoup l'affichage. Ce problème de matériel est arrivé à son sommet au moment où nous ne disposions plus d'assez de mémoire vive pour traiter toutes les classifications ensemble. Nous avons donc dû mettre au point un script python et plusieurs jeux de données spécifiques devant traiter moins d'images. Ceux-ci ont quand même pu présenter des résultats satisfaisants, au niveau de la comparaison avec les cartes de corrélation, mais en ne faisant pas intervenir autant d'informations que pouvait offrir notre base de données.

Pour finir avec les difficultés, nous pouvons ajouter que nous avons vu trop grand. En effet, nous aurions dû traiter les données par blocs plus restreints au lieu de vouloir les traiter toutes ensemble. Et ainsi éviter des problèmes de surcharge.

La répartition du travail s'est faite de manière équitable et prenait en compte les compétences et aptitudes de chacun.

Enfin ce que nous retenons de ce projet de fin de stage, c'est la manière à analyser une série de résultats. En effet, notre sujet ne nous permettait pas de présenter des résultats de façon absolue mais bien de manière relative en les comparant les uns aux autres. Un résultat n'a d'intérêt que par rapport à un autre, d'où la présence de différents paramètres pour une même texture.

Il en va de même pour nos différents jeux de données avec OTB. Ayant dû en faire plusieurs, nous ne nous retrouvons pas avec un bon et unique jeu de données mais avec des jeux relativement bons, tout dépend de l'image Haralick à classifier. La technique repose donc dans le choix et la construction des jeux de données.

Tables

Tableaux

Tableau 1. Paramètres fixés 4
Tableau 2. Paramètres des fichiers .xml 5
Tableau 3. Distorsion en borure d'image de chaque calibration 8
Tableau 4. Présentation des différentes textures ainsi que des différents paramètres 10
Tableau 5. Seuillages 13
Tableau 6. Matrice de confusion 14
Tableau 7. Jeux d'apprentissage testés sur armoire_soleil.tif 17

Illustrations

Figure 1. Reconstitution d'un nuage de points .ply 3
Figure 2. Prise de vue convergente en croix 9
Figure 3. Base de données d'images de texture et de corrélation sur QGIS 10
Figure 4. Cloud Compare (mur mouillé) 12
Figure 5. Cloud Compare écart au plan (mur mouillé) 12
Figure 6. Carte de corrélation (végétation basse) .tif 12
Figure 7. Carte de corrélation (tableau blanc) .tif 12
Figure 8. Correl armoire_soleil.tif 16
Figure 9. test_arbre_S.png 16
Figure 10. essai_mix_S.png 16
Figure 11. essai_mur_S.png 16

Bibliographie

Sites Internet :

-Robert Haralick (https://en.wikipedia.org/wiki/Robert_Haralick)
-Comment changer de répertoire courant / se déplacer dans l'arborescence (http://www.developpez.net/forums/d696854/systemes/windows/vos-contributions/faqs-windows/batch-changer-repertoire-courant-se-deplacer-l-arborescence/)
-FAQ MATLAB (http://matlab.developpez.com/faq/?page=fileio)
-GTiff -- GeoTIFF File Format (http://www.gdal.org/frmt_gtiff.html)
-Support Options GNU (http://www.gnu.org/software/octave/support.html)
-imagemagick (http://doc.ubuntu-fr.org/imagemagick)
-matplotlib pyplot (http://matplotlib.org/api/pyplot_api.html)
-Micmac (http://logiciels.ign.fr/?-Micmac,3-)
-5.8.8 Train a classifier from multiple images (https://www.orfeo-toolbox.org/CookBook/CookBooksu122.html#x157-8830005.8.8)

Documentation Micmac :

-MicMac, Apero, Pastis and Other Beverages in a Nutshell!
April 1, 2015

Annexes

1. «Tri_donnees.ods» :

Appareil	Focale (mm)	Résolution (mm)	Dossier	Nb. d'images	Type	Distance PDV (mm)	Etat	Eclairage	Angle de PDV	Ciblage	Texture	Résidu	Résolution (mm)	Corrélation (moy)	Corrélation (σ)	Ecart au plan à 2 σ (mm)	Commentaires
NIKON	*18,28*	*1,285*	*mur+tableau_blanc*	*6*	*JPG*	*[2200;6100]*	*Sec*	*Légèrement sombre*	*Face*	*Convergente*	*Tableau blanc sale*	*0,60*	*1,28*	*0,60*	*0,08*	*4*	*ok*
											Mur blanc peinture			*0,64*	*0,11*	*3*	*ok*
CANON	*34,34*	*0,400*	*TABLEAU*	*6*	*JPG*	*[2200;6100]*	*Sec*	*Légèrement sombre*	*Face*	*Convergente*	*Encre tableau*	*1,08*	*0,40*	*0,10*	*0,27*	*2*	*très peu de points de liaison*
NIKON	18,28	1,432	sol_en_beton	5	JPG	6800	Sec	Eclairé/Ombragé	Face	Parallèle	Sol en béton	0,41	1,43	0,93	0,08	2	ok
	18,28	1,432	sol_en_beton_NEF	5	NEF							0,38	1,43	0,97	0,04	3	ok
CANON	34,34	0,291	SOL	22	JPG	[1500;6800]	Sec	Ombragé	Face/Oblique 47°	Parallèle	Sol en béton	0,69	0,29	0,86	0,15	2	ok
NIKON	*18,28*	*0,232*	*sol_carrelage*	*5*	*JPG*	*1100*	*Sec*	*Légèrement sombre*	*Oblique*	*Convergente*	*Sol carrelage*	*0,60*	*0,23*	*0,81*	*0,39*	*5*	*à voir*
CANON	34,34	0,237	SOLCLASSE	6	JPG	[1300;1850]	Sec	Légèrement sombre	Oblique	Parallèle	Sol carrelage	0,56	0,24	0,84	0,13	4	ok
NIKON	18,28	1,306	parallèle_perpendiculaire_sec	5	JPG	6200	Sec	Ombragé	Face	Parallèle	Mur crépis	0,35	1,31	0,95	0,06	5	ok
CANON	34,34	1,128	mur_garage_sec_parallèle	5	JPG	6200	Sec	Ombragé	Face	Parallèle	Mur crépis	0,52	1,13	0,93	0,08	4	ok
NIKON	18,28	1,306	parallèle_centre_sec	5	JPG	6200	Sec	Ombragé	Face	Convergente	Mur crépis	0,36	1,31	0,95	0,06	4	ok
CANON	34,34	1,128	mur_garage_sec_centré	5	JPG	6200	Sec	Ombragé	Face	Convergente	Mur crépis	0,60	1,13	0,92	0,08	3	ok
NIKON	18,28	0,674	oblique_centre_sec	5	JPG	[3200;5000]	Sec	Ombragé	Oblique 30°	Convergente	Mur crépis	0,45	0,67	0,90	0,10	3	ok
CANON	34,34	0,728	mur_garage_sec_oblique	5	JPG	[4000;5000]	Sec	Ombragé	Oblique 21°	Convergente	Mur crépis	0,47	0,73	0,91	0,10	5	ok
NIKON	18,28	1,306	parallèle_perpendiculaire_humide	5	JPG	6200	Humide	Ombragé	Face	Parallèle	Mur crépis	0,46	1,31	0,94	0,06	5	ok
CANON	34,34	1,128	mur_garage_humide_parallèle	5	JPG	6200	Humide	Ombragé	Face	Parallèle	Mur crépis	0,58	1,13	0,93	0,08	4	ok
NIKON	18,28	1,306	parallèle_centré_humide	5	JPG	6200	Humide	Ombragé	Face	Convergente	Mur crépis	0,47	1,31	0,94	0,06	4	ok
CANON	34,34	1,128	mur_garage_humide_centré	5	JPG	6200	Humide	Ombragé	Face	Convergente	Mur crépis	0,64	1,13	0,92	0,08	4	ok

NIKON	**18,28**	**0,505**	**Herbe_ombre**	**5**	**JPG**	**2400**	**Sec**	**Ombragé**	**Oblique 35°**	**Parallèle**	**Végétation basse**	**0,52**	**0,51**	**0,82**	**0,13**	**////**	**ok**
CANON	**34,34**	**0,291**	**C-vegetation_basse_ombre_face**	**5**	**JPG**	**1600**	**Sec**	**Ombragé**	**Face**	**Parallèle**	**Végétation basse**	**0,63**	**0,29**	**0,88**	**0,13**	**////**	**ok**
NIKON	*18,28*	*0,505*	*Herbe_soleil*	*4*	*JPG*	*2400*	*Sec*	*Ensoleillé*	*Oblique 35°*	*Parallèle*	*Végétation basse*	*1,30*	*0,51*	*////*	*////*	*////*	*aucun point de liaison sur l'herbe*
	18,28	*0,505*	*Herbe_soleil_NEF*	*5*	*NEF*							*0,60*	*0,51*	*0,82*	*0,16*	*////*	*ok*
CANON	**34,34**	**0,291**	**C_vegetation_basse_ensoileille_face**	**5**	**JPG**	**1600**	**Sec**	**Ensoleillé**	**Face**	**Parallèle**	**Végétation basse**	**0,60**	**0,29**	**0,90**	**0,12**	**////**	**98,13 % des 375 points de liaison sur la FG0A6628,JPG**
NIKON	**18,28**	**0,842**	**Tronc_ombre**	**6**	**JPG**	**4000**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Ecorce**	**0,55**	**0,84**	**0,87**	**0,13**	**////**	**ok**
CANON	**34,34**	**0,673**	**C_ecorce-ombre-centre**	**6**	**JPG**	**3700**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Ecorce**	**0,47**	**0,67**	**0,87**	**0,13**	**////**	**ok**
NIKON	**18,28**	**0,695**	**Tronc_ombre_mousse**	**6**	**JPG**	**3300**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Ecorce moussue**	**0,51**	**0,69**	**0,88**	**0,12**	**////**	**ok**
CANON	**34,34**	**0,601**	**C_ecorce_moussue_ombre_centre**	**6**	**JPG**	**3300**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Ecorce moussue**	**0,36**	**0,60**	**0,89**	**0,12**	**////**	**ok**
NIKON	**18,28**	**1,242**	**muret_soleil**	**5**	**JPG**	**5900**	**Sec**	**Ensoleillé**	**Face**	**Convergente**	**Muret**	**0,69**	**1,24**	**0,96**	**0,05**	**6**	**ok**
CANON	**34,34**	**1,074**	**Muret Soleil**	**5**	**JPG**	**5900**	**Sec**	**Ensoleillé**	**Face**	**Convergente**	**Muret**	**0,69**	**1,07**	**0,93**	**0,12**	**5**	**ok**
NIKON	**18,28**	**1,242**	**muret_ombre**	**5**	**JPG**	**5900**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Muret**	**0,48**	**1,24**	**0,91**	**0,12**	**9**	**ok**
CANON	**34,34**	**1,074**	**Muret_ombre**	**5**	**JPG**	**5900**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Muret**	**0,65**	**1,07**	**0,92**	**0,13**	**17**	**ok**
NIKON	*18,28*	*1,348*	*herbe_verte_ombre*	*5*	*JPG*	*6400*	*Humide*	*Ombragé*	*Face*	*Parallèle*	*Gazon*	*0,59*	*1,35*	*0,52*	*0,37*	*////*	*aucun point de liaison sur l'herbe*
CANON	*34,34*	*1,165*	*Herbre_ombre*	*5*	*JPG*	*6400*	*Humide*	*Ombragé*	*Face*	*Parallèle*	*Gazon*	*////*	*1,16*	*////*	*////*	*////*	*NULL*
NIKON	*18,28*	*1,348*	*herbe_verte_soleil*	*3*	*JPG*	*6400*	*Sec*	*Ensoleillé*	*Oblique 15°*	*Convergente*	*Gazon*	*0,69*	*1,35*	*////*	*////*	*////*	*aucun point de liaison sur l'herbe*
CANON	*34,34*	*1,165*	*herbre_Soleil*	*5*	*JPG*	*6400*	*Sec*	*Ensoleillé*	*Oblique 15°*	*Convergente*	*Gazon*	*1,06*	*1,16*	*0,75*	*0,12*	*////*	*résidu trop élevé*
NIKON	*18,28*	*2,611*	*metal_rouille*	*4*	*JPG*	*12400*	*Sec*	*Ombragé*	*Face*	*Parallèle*	*Rouille*	*0,72*	*2,61*	*0,79*	*0,17*	*////*	*nombre de points de liaison faible et mal réparti*
CANON	*34,34*	*2,257*	*Metal*	*5*	*JPG*	*12400*	*Sec*	*Ombragé*	*Face*	*Parallèle*	*Rouille*	*13,00*	*2,26*	*////*	*////*	*////*	*résidu trop élevé*
NIKON	*18,28*	*4,212*	*sol_gravier*	*5*	*JPG*	*20000*	*Sec*	*Ensoleillé*	*Oblique 5°*	*Convergente*	*Sol cailloux*	*0,73*	*4,21*	*0,66*	*0,21*	*////*	*ok*
CANON	**34,34**	**3,640**	**Gravier**	**5**	**JPG**	**20000**	**Sec**	**Ensoleillé**	**Oblique 4°**	**Convergente**	**Sol cailloux**	**0,84**	**3,64**	**0,83**	**0,13**	**24**	**pas assez de points de liaison**
NIKON	*18,28*	*2,085*	*carrosserie*	*5*	*JPG*	*9900*	*Sec*	*Ensoleillé*	*Face*	*Convergente*	*Carrosserie*	*0,60*	*2,08*	*0,69*	*0,11*	*16*	*à voir*
CANON	*34,34*	*1,802*	*Camion*	*5*	*JPG*	*9900*	*Sec*	*Ensoleillé*	*Face*	*Convergente*	*Carrosserie*	*0,95*	*1,80*	*0,68*	*0,12*	*350*	*pas assez de points de liaison*
NIKON	*18,28*	*2,801*	*bitume*	*4*	*JPG*	*13300*	*Sec*	*Ensoleillé*	*Oblique 7°*	*Convergente*	*Bitume*	*0,58*	*2,80*	*0,73*	*0,16*	*////*	*ok*
NIKON	**18,28**	**0,526**	**bois_armoire**	**5**	**JPG**	**2500**	**Sec**	**Assez ensoleillé**	**Face**	**Convergente**	**Bois de l'armoire**	**0,50**	**0,53**	**0,86**	**0,11**	**2**	**ok**
NIKON	**18,28**	**0,442**	**liege_vive_la_belgique**	**4**	**JPG**	**2100**	**Sec**	**Assez ombragé**	**Face**	**Convergente**	**Panneau en liège**	**0,36**	**0,44**	**0,89**	**0,11**	**2**	**ok**
Drone	**27,41**	**10,000**	**D_bitume_ensoleille**	**5**	**JPG**	**?**	**Sec**	**Ensoleillé**	**Face**	**Parallèle**	**Bitume**	**0,48**	**10,00**	**0,84**	**0,11**	**29**	**ok**

Drone	**27,41**	**10,000**	**D_bitume_sombre**	**6**	**JPG**	**?**	**Sec**	**Sombre**	**Face**	**Parallèle**	**Bitume**	**0,42**	**10,00**	**0,79**	**0,12**	**27**	**ok**
Drone	**27,41**	**15,000**	**D_tuiles_ensoleilles**	**9**	**JPG**	**?**	**Sec**	**Ensoleillé**	**Face**	**Parallèle**	**Tuile**	**0,36**	**15,00**	**0,95**	**0,08**	**////**	**ok**
Drone	**27,41**	**19,000**	**D_tuiles_sombres**	**5**	**JPG**	**?**	**Sec**	**Sombre**	**Face**	**Parallèle**	**Tuile**	**0,40**	**19,00**	**0,96**	**0,06**	**////**	**ok**
Drone	**27,41**	**21,000**	**D_vegetation_ensoleillee**	**5**	**JPG**	**?**	**Sec**	**Ensoleillé**	**Face**	**Parallèle**	**Végétation haute**	**0,50**	**21,00**	**0,74**	**0,15**	**////**	**ok**
											Végétation basse			**0,85**	**0,12**	**////**	**ok**
Drone	**27,41**	**10,000**	**D_vegetation_sombre**	**5**	**JPG**	**?**	**Sec**	**Sombre**	**Face**	**Parallèle**	**Végétation haute**	**0,42**	**10,00**	**0,70**	**0,19**	**////**	**ok**
											Végétation basse			**0,82**	**0,13**	**////**	**ok**
NIKON	**18,28**	**1,000**	**armoire_ombre**	**5**	**JPG**	**4749**	**Sec**	**Ombragé**	**Face**	**Convergente**	**Bois de l'armoire**	**0,49**	**1,00**	**0,79**	**0,14**	**7**	**ok**
CANON	*34,34*	*1,000*	*porte_ombre*	*5*	*JPG*	*5495*	*Sec*	*Ombragé*	*Face*	*Parallèle*	*Bois de l'armoire*	*0,58*	*1,00*	*0,81*	*0,13*	*140*	*ok*
NIKON	**18,28**	**1,000**	**armoire_soleil**	**5**	**JPG**	**4749**	**Sec**	**Ensoleillé**	**Face**	**Convergente**	**Bois de l'armoire**	**0,51**	**1,00**	**0,86**	**0,11**	**4**	**ok**
CANON	*34,34*	*1,000*	*porte_soleil*	*5*	*JPG*	*5495*	*Sec*	*Ensoleillé*	*Face*	*Parallèle*	*Bois de l'armoire*	*0,73*	*1,00*	*0,85*	*0,11*	*133*	*ok*

2. «distorsionWithPolynom.py» :

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import xml.etree.ElementTree as ET
import numpy as np
import matplotlib.pyplot as plt
#from pylab import *
from scipy import misc

calibs_nikon =
[{"name":"Calib_b","path":"/media/synology/projet_T1/Calibrations/Calibr
ation_Nikon_D3200/Nikon/Calib_b/Ori-Calib_b/AutoCal180.xml"}, #0

{"name":"Calib_NEF","path":"/media/synology/projet_T1/Calibrations/Cal
ibration_Nikon_D3200/Nikon/Calib_NEF/Ori-MEP/AutoCal180.xml"}, #1

{"name":"eglise_in","path":"/media/synology/projet_T1/Calibrations/Calib
ration_Nikon_D3200/Nikon/eglise_in/Ori-eglise_Calib/AutoCal180.xml"},
#2

{"name":"escalier_in","path":"/media/synology/projet_T1/Calibrations/Cali
bration_Nikon_D3200/Nikon/escalier_in/Ori-
escalier_Calib/AutoCal180.xml"}, #3

{"name":"garage_out","path":"/media/synology/projet_T1/Calibrations/Ca
libration_Nikon_D3200/Nikon/garage_out/Ori-
garage_Calib/AutoCal180.xml"}, #4

{"name":"Maison_in","path":"/media/synology/projet_T1/Calibrations/Cali
bration_Nikon_D3200/Nikon/Maison_in/Ori-
maison_Calib/AutoCal180.xml"}, #5

#{"name":"marche_eglise_in","path":"/media/synology/projet_T1/Calibrati
ons/Calibration_Nikon_D3200/Nikon/marche_eglise_in/Ori-
marche_eglise/AutoCal180.xml"},

{"name":"marche_eglise_in_Calib_ok","path":"/media/synology/projet_T1
/Calibrations/Calibration_Nikon_D3200/Nikon/marche_eglise_in_Calib_o
k/Ori-marche_eglise/AutoCal180.xml"}] #6

calibs_canon =
[{"name":"esc2","path":"/media/synology/projet_T1/Calibrations/Calibrati
on_Canon_OS_5D_MARK_III/Canon/esc2/Ori-
MEPCALIBRATIONESC2/AutoCal350.xml"}, #0

{"name":"Escalier","path":"/media/synology/projet_T1/Calibrations/Calibr
ation_Canon_OS_5D_MARK_III/Canon/Escalier/Ori-
MEPCALIBRATIONESCALIER1/AutoCal350.xml"}, #1

{"name":"imgesc1","path":"/media/synology/projet_T1/Calibrations/Calibr
ation_Canon_OS_5D_MARK_III/Canon/imgesc1/Ori-
MEPCALIBRATION/AutoCal350.xml"}, #2

{"name":"imgGarage","path":"/media/synology/projet_T1/Calibrations/Cal
ibration_Canon_OS_5D_MARK_III/Canon/imgGarage/Ori-
MEPCALIBRATIONGARAGE/AutoCal350.xml"}, #3

{"name":"imgmurs","path":"/media/synology/projet_T1/Calibrations/Calibr
ation_Canon_OS_5D_MARK_III/Canon/imgmurs/Ori-
MEPCALIBRATIONMURS/AutoCal350.xml"}, #4

{"name":"murs2","path":"/media/synology/projet_T1/Calibrations/Calibrati
on_Canon_OS_5D_MARK_III/Canon/murs2/Ori-
MEPCALIBRATIONMURS2/AutoCal350.xml"}, #5

{"name":"ESCALIER3","path":"/media/synology/projet_T1/Calibrations/C
alibration_Canon_OS_5D_MARK_III/Canon/ESCALIER3/Ori-
MEPCALIBRATIONESCALIER3/AutoCal350.xml"}, #6

{"name":"ESCALIER2","path":"/media/synology/projet_T1/Calibrations/C
alibration_Canon_OS_5D_MARK_III/Canon/ESCALIER2/Ori-
MEPCALIBRATIONESCALIER2/AutoCal350.xml"}] #7

calibs_drone =
[{"name":"drone","path":"/media/synology/projet_T1/Calibrations/Calibrati
on_drone/AutoCal183.xml"}] #0

epais=0.5
symb=''
inter=30

inter_min_distortion = 0
inter_max_distortion = 3500
inter_min_r_dr = 0
inter_max_r_dr = 3500
inter_max_drone = 2600

# distortion
############################################################
############################################################
############################################################

for calib in calibs_nikon:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_distortion,inter_max_distortion,inter)
    plt.plot(r,a*r**3+b*r**5+c*r**7, label=calib["name"], linewidth=epais,
marker=symb)
    plt.ylabel('distorsion')
    plt.xlabel("r")
    plt.title("Distorsion du Nikon D3200")
#plt.axis('equal')
plt.legend(loc=3, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du Nikon D3200')
plt.show()

# r + dr
############################################################
############################################################
############################################################
###

for calib in calibs_nikon:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_r_dr,inter_max_r_dr,inter)
    dr = a*r**3+b*r**5+c*r**7
    plt.plot(r, r + dr, label=calib["name"], linewidth=epais, marker=symb)
    plt.ylabel('r + dr')
    plt.xlabel("r")
    plt.title("Distorsion du Nikon D3200")
#plt.axis('equal')
plt.legend(loc=4, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du Nikon D3200 r + dr')
plt.show()

# derivation #

calib = calibs_nikon[6] # Choisir la fonction !
tree = ET.parse(calib["path"])
liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
a = liste_coeff[0]
b = liste_coeff[1]
c = liste_coeff[2]
r = arange(inter_min_r_dr,inter_max_r_dr,inter)
def fonction(r):
    return r+(a*r**3+b*r**5+c*r**7)
y = fonction(r)
#plt.plot(r,y)
yp = misc.derivative(fonction,y)
plt.plot(r, yp, label=calib["name"])
plt.ylabel("d( r + dr)")
plt.xlabel("r + dr")
plt.title("Distorsion du Nikon D3200")
#plt.axis('equal')
plt.legend(loc=1, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du Nikon D3200 d( r + dr)')
plt.show()

# distortion
############################################################
```

```
############################################################
############################################################

for calib in calibs_canon:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_distortion,inter_max_distortion,inter)
    plt.plot(r,a*r**3+b*r**5+c*r**7, label=calib["name"], linewidth=epais,
marker=symb)
    plt.ylabel('distorsion')
    plt.xlabel("r")
    plt.title("Distorsion du Canon OS 5D MARK III")
#plt.axis('equal')
plt.legend(loc=3, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du Canon OS 5D MARK III')
plt.show()

# r + dr
############################################################
############################################################
############################################################
###

for calib in calibs_canon:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_r_dr,inter_max_r_dr,inter)
    dr = a*r**3+b*r**5+c*r**7
    plt.plot(r, r + dr, label=calib["name"], linewidth=epais, marker=symb)
    plt.ylabel('r + dr')
    plt.xlabel("r")
    plt.title("Distorsion du Canon OS 5D MARK III")
#plt.axis('equal')
plt.legend(loc=4, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du Canon OS 5D MARK III r + dr')
plt.show()

# derivation #

calib = calibs_canon[6] # Choisir la fonction !
tree = ET.parse(calib["path"])
liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
a = liste_coeff[0]
b = liste_coeff[1]
c = liste_coeff[2]
r = arange(inter_min_r_dr,inter_max_r_dr,inter)
def fonction(r):
    return r+(a*r**3+b*r**5+c*r**7)
y = fonction(r)
#plt.plot(r,y)
yp = misc.derivative(fonction,y)
plt.plot(r, yp, label=calib["name"])
plt.ylabel("d( r + dr)")
plt.xlabel("r + dr")
plt.title("Dérivée du Canon OS 5D MARK III")
#plt.axis('equal')
plt.legend(loc=1, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Dérivée du Canon OS 5D MARK III d( r + dr)')
plt.show()

# distortion
############################################################
############################################################
############################################################

for calib in calibs_drone:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_distortion,inter_max_drone,inter)
    plt.plot(r,a*r**3+b*r**5+c*r**7, label=calib["name"], linewidth=epais,
marker=symb)
    plt.ylabel('distorsion')
    plt.xlabel("r")
    plt.title("Distorsion du drone")
#plt.axis('equal')
plt.legend(loc=1, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du drone')
plt.show()

# r + dr
############################################################
############################################################
############################################################
###

for calib in calibs_drone:
    tree = ET.parse(calib["path"])
    liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
    a = liste_coeff[0]
    b = liste_coeff[1]
    c = liste_coeff[2]
    r = np.linspace(inter_min_r_dr,inter_max_drone,inter)
    dr = a*r**3+b*r**5+c*r**7
    plt.plot(r, r + dr, label=calib["name"], linewidth=epais, marker=symb)
    plt.ylabel('r + dr')
    plt.xlabel("r")
    plt.title("Distorsion du drone")
#plt.axis('equal')
plt.legend(loc=1, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Distorsion du drone r + dr')
plt.show()

# derivation #

calib = calibs_drone[0] # Choisir la fonction !
tree = ET.parse(calib["path"])
liste_coeff = [float(elt.text) for elt in
tree.findall('CalibrationInternConique/CalibDistortion/ModRad/CoeffDist')
]
a = liste_coeff[0]
b = liste_coeff[1]
c = liste_coeff[2]
r = arange(inter_min_r_dr,inter_max_drone,inter)
def fonction(r):
    return r+(a*r**3+b*r**5+c*r**7)
y = fonction(r)
#plt.plot(r,y)
yp = misc.derivative(fonction,y)
plt.plot(r, yp, label=calib["name"])
plt.ylabel("d( r + dr)")
plt.xlabel("r + dr")
plt.title("Dérivée du drone")
#plt.axis('equal')
plt.legend(loc=1, borderaxespad=0.)
#plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.grid(True)
plt.savefig('Dérivée du drone d( r + dr)')
plt.show()
```

3. «python_rf.py» :

```
#!/usr/bin/python

# Import the otb applications package
import otbApplication

# The following line creates an instance of the TrainImagesClassifier
application
TrainImagesClassifier =
otbApplication.Registry.CreateApplication("TrainImagesClassifier")

# The following lines set all the application parameters:
images =
[#"/home/prof/Bureau/texture/herbe_verte_ombre/DSC_0011_haral.tif",
        #"/home/prof/Bureau/texture/bitume/DSC_0036_haral.tif",
        #"/home/prof/Bureau/texture/bois_armoire/DSC_0006_haral.tif",
        #"/home/prof/Bureau/texture/carosserie/DSC_0031_haral.tif",
        #"/home/prof/Bureau/texture/herbe_ombre/DSC_0001_haral.tif",

#"/home/prof/Bureau/texture/herbe_verte_soleil/DSC_0017_haral.tif",

#"/home/prof/Bureau/texture/liege_vive_la_belgique/DSC_0001_haral.tif
",
        #"/home/prof/Bureau/texture/metal_rouille/DSC_0024_haral.tif",
        "/home/prof/Bureau/texture/muret_ombre/DSC_0010_haral.tif",
        "/home/prof/Bureau/texture/muret_soleil/DSC_0001_haral.tif",

"/home/prof/Bureau/texture/mur+tableau_blanc/DSC_0001_haral.tif",

"/home/prof/Bureau/texture/oblique_centre_sec/DSC_0011_haral.tif",

"/home/prof/Bureau/texture/parallele_centre_humide/DSC_0022_haral.tif
",

"/home/prof/Bureau/texture/parallele_centre_sec/DSC_0008_haral.tif",

"/home/prof/Bureau/texture/parallele_perpendiculaire_humide/DSC_001
9_haral.tif",

"/home/prof/Bureau/texture/parallele_perpendiculaire_sec/DSC_0002_h
aral.tif"]
        #"/home/prof/Bureau/texture/sol_carrelage/DSC_0004_haral.tif",
        #"/home/prof/Bureau/texture/sol_en_beton/DSC_0002_haral.tif",
        #"/home/prof/Bureau/texture/sol_gravier/DSC_0026_haral.tif",
        #"/home/prof/Bureau/texture/tronc_ombre/DSC_0015_haral.tif",

#"/home/prof/Bureau/texture/tronc_ombre_mousse/DSC_0017_haral.tif"]

vectors =
[#"/home/prof/Bureau/texture/herbe_verte_ombre/DSC_0011_herbe_ver
te_ombre.shp",
        #"/home/prof/Bureau/texture/bitume/DSC_0036_bitume.shp",

#"/home/prof/Bureau/texture/bois_armoire/DSC_006_bois_armoire.shp",

#"/home/prof/Bureau/texture/carosserie/DSC_0031_carosserie.shp",

#"/home/prof/Bureau/texture/herbe_ombre/DSC_0001_herbe_ombre.sh
p",

#"/home/prof/Bureau/texture/herbe_verte_soleil/DSC_0017_herbe_verte
_soleil.shp",

#"/home/prof/Bureau/texture/liege_vive_la_belgique/DSC_0001_liege.sh
p",

#"/home/prof/Bureau/texture/metal_rouille/DSC_0024_metal_rouille.shp"
,

"/home/prof/Bureau/texture/muret_ombre/DSC_0010_muret_ombre.shp"
,

"/home/prof/Bureau/texture/muret_soleil/DSC_001_muret_soleil.shp",

"/home/prof/Bureau/texture/mur+tableau_blanc/DSC_0001_mur_tableau
_blanc.shp",

"/home/prof/Bureau/texture/oblique_centre_sec/DSC_0011_OBLIQUE_
CENTRE-SEC.shp",

"/home/prof/Bureau/texture/parallele_centre_humide/DSC_0022_parallel
e_centre_humide.shp",

"/home/prof/Bureau/texture/parallele_centre_sec/DSC_0008_parallele_c
entre_sec.shp",

"/home/prof/Bureau/texture/parallele_perpendiculaire_humide/DSC_001
9_parallele_perpendiculaire_humide.shp",

"/home/prof/Bureau/texture/parallele_perpendiculaire_sec/DSC_0002_P
ARALLELE_PERPENDICULAIRE_SEC.shp"]

#"/home/prof/Bureau/texture/sol_carrelage/DSC_0004_sol_carrelage.sh
p",

#"/home/prof/Bureau/texture/sol_en_beton/DSC_002_sol_en_beton.shp
",

#"/home/prof/Bureau/texture/sol_gravier/DSC_0026_sol_gravier.shp",

#"/home/prof/Bureau/texture/tronc_ombre/DSC_0015_tronc_ombre.shp"
,

#"/home/prof/Bureau/texture/tronc_ombre_mousse/DSC_0017_tronc_o
mbre_mousse.shp"]

TrainImagesClassifier.SetParameterStringList("io.il", images)

TrainImagesClassifier.SetParameterStringList("io.vd", vectors)

#TrainImagesClassifier.SetParameterString("io.imstat",
"EstimateImageStatisticsQB1.xml")

TrainImagesClassifier.SetParameterInt("sample.mv", 1000)

TrainImagesClassifier.SetParameterInt("sample.mt", 1000)

TrainImagesClassifier.SetParameterFloat("sample.vtr", 0.7)

#TrainImagesClassifier.SetParameterString("sample.edg","1")

TrainImagesClassifier.SetParameterString("sample.vfn", "class")


# Random forest
TrainImagesClassifier.SetParameterString("classifier","rf")
TrainImagesClassifier.SetParameterString("classifier.rf.max","5")
TrainImagesClassifier.SetParameterString("classifier.rf.min", "10")
TrainImagesClassifier.SetParameterString("classifier.rf.ra","0")
TrainImagesClassifier.SetParameterString("classifier.rf.cat","10")
TrainImagesClassifier.SetParameterString("classifier.rf.var","0")
TrainImagesClassifier.SetParameterString("classifier.rf.nbtrees","100")
TrainImagesClassifier.SetParameterString("classifier.rf.acc","0.01")

#SVM
#TrainImagesClassifier.SetParameterString("classifier","svm")
#TrainImagesClassifier.SetParameterString("classifier.svm.k", "linear")
#TrainImagesClassifier.SetParameterString("classifier.svm.c","1")
#TrainImagesClassifier.SetParameterString("classifier.svm.nu", "0")
#TrainImagesClassifier.SetParameterString("classifier.svm.coef0","0")
#TrainImagesClassifier.SetParameterString("classifier.svm.gamma","1")
#TrainImagesClassifier.SetParameterString("classifier.svm.degree","1")


TrainImagesClassifier.SetParameterString("io.out", "rfModel.txt")

TrainImagesClassifier.SetParameterString("io.confmatout",
"rfConfusionMatrix.csv")

# The following line execute the application
TrainImagesClassifier.ExecuteAndWriteOutput()
```

4. «run_project.m» :

```
clear all
close all
clc
format bank
R=[];
for i=1:4
        for j=i+1:5
                h=metascript(i,j);
                R(i,j)=h;
        end
end
R(5,:)=[0 0 0 0 0];
R
```

5. «new_run_project.m» :

```
clear all
close all
clc
format bank


cd ..
cd('Classification/Nikon')

liste1=['arbre';'image';'mixer';'muret';'test2';'test3'];
matrix=[];
for l=1:6
        cd ([liste1(l,:)])
        file_csv=(['rfConfusionMatrix',liste1(l,:),'.csv']);
        N=0;
        B=0;
        E=0;
        A=load(file_csv);
        precision=A(1,1)/(A(1,1)+A(2,1));
        rappel=A(1,1)/(A(1,1)+A(1,2));
        fscore=(2*precision*rappel)/(precision+rappel);
        for i=1:2
                C=0;
                D=0;
                for j=1:2
                        N=N+A(i,j);
                        C=C+A(i,j);
                end
                for k=1:2
                        D=D+A(k,i);
                end
                B=B+A(i,i);
                E=E+(C*D);
        end
        K=((N*B)-E)/((N*N)-E);
        matrix=[matrix;precision,rappel,fscore,K];
        cd ..
end
cd ..
cd ..
cd octave
matrix
```

6. «metascript.m» :

```
function h=metascript(m,n)
dossier_cible=[num2str(m),'_',num2str(n)];
cd(dossier_cible)
h=resultat(['DSC_000',num2str(m),'.JPG.tx
t'],['DSC_000',num2str(n),'.JPG.txt']);
cd ..
```