ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΤΜΗΜΑ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

ΕΡΓΑΣΤΗΡΙΟ ΤΗΛΕΠΙΣΚΟΠΗΣΗΣ ΚΑΙ ΓΣΠ

ΠΕΡΙΒΑΛΛΟΝΤΙΚΕΣ ΕΦΑΡΜΟΓΕΣ ΓΣΠ – 323Ε

Καθηγητής Ιωάννης Ν. Χατζόπουλος

Διευθυντής Εργαστηρίου Τηλεπισκόπησης και ΓΣΠ

Αυτοματισμός-Scripting-Python

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΤΜΗΜΑ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

ΕΡΓΑΣΤΗΡΙΟ ΤΗΛΕΠΙΣΚΟΠΗΣΗΣ ΚΑΙ ΓΣΠ

ΠΕΡΙΒΑΛΛΟΝΤΙΚΕΣ ΕΦΑΡΜΟΓΕΣ ΓΣΠ – 323Ε

Καθηγητής Ιωάννης Ν. Χατζόπουλος
Διευθυντής Εργαστηρίου Τηλεπισκόπησης και ΓΣΠ

Αυτοματισμός-Scripting-Python

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αιγαίου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Προγραμματισμός (scripting) με γλώσσα προγραμματισμού Πύθων (Python)

Ο πύθων είναι μια γλώσσα αντικειμενοστρεφούς προγραμματισμού ανεξαρτήτου πλατφόρμας, εύκολη στην εκμάθηση και έχει πρόσβαση στις επίσης ανοιχτού κώδικα βιβλιοθήκες γραφικών που χρησιμοποιούν τα ΓΣΠ και η τηλεπισκόπηση όπως είναι η GDAL (Geospatial Data Abstraction Library) για πλεγματικές επικαλύψεις και η OGR για διανυσματικές επικαλύψεις καθώς και πλήθος άλλων ανοιχτού κώδικα βιβλιοθηκών.
Είναι ελεύθερο κάτω από μια X/MIT είδους ανοιχτού κώδικα άδεια χρήσης από την Geospatial Foundation OSGeo. Τα περισσότερα πακέτα ΓΣΠ και τηλεπισκόπησης χρησιμοποιούν επίσης πύθων σαν γλώσσα προγραμματισμού (script) σε διαδικασίες αυτοματισμού.

Ο προγραμματισμός σε πύθων γίνεται συνεχώς περισσότερο δημοφιλής λόγω της απλότητας του, τη χρήση των περισσότερων βιβλιοθηκών λογισμικού και υπάρχουν πάρα πολλές τοποθεσίες στο διαδίκτυο με δωρεάν μεθόδους αυτοδιδασκαλίας προς εκμάθηση.

Ο πύθων επίσης μπορεί να χρησιμοποιηθεί σαν ανεξάρτητη γλώσσα προγραμματισμού για οποιαδήποτε εφαρμογή και υπάρχουν στο διαδίκτυο πάρα πολλά κάθε λογής έτοιμα προγράμματα.

Κάτω από τις λέξεις κλειδιά “PyPI - the Python Package Index” βρέθηκαν όταν γραφόταν οι παρούσες σημειώσεις 27194 τέτοια προγράμματα.

Προγράμματα όπως είναι τα ArcGis, Qgis, etc., χρησιμοποιούν πύθων σαν γλώσσα αυτοματισμού (scripting language).

Για εύκολη χρήση του πύθων σε εφαρμογές ΓΣΠ και τηλεπισκόπησης υπάρχει το πακέτο ανοιχτού κώδικα FWTools το οποίο περιέχει τον πύθων μαζί με τις βιβλιοθήκες γραφικών GDAL / OGR, τις βιβλιοθήκες του λειτουργικού συστήματος και τις βιβλιοθήκες για αριθμητική επεξεργασία.

Επειδή τόσο ο πύθων όσο και τα FWTools είναι επεξεργαστές με γραμμή διαταγών (command line processing) χρησιμοποιούνται παράλληλα και κειμενογράφοι οθόνης για την ανάπτυξη πηγαίου κώδικα όπως είναι οι: Crimson Editor, PythonWin, DrPython, eclipse, etc. Στην ανάπτυξη των παραδειγμάτων που δίνονται πιο κάτω χρησιμοποιήθηκαν τα FWTools και ο Crimson Editor τα οποία έχουν εγκατασταθεί επίσης στο εργαστήριο GIS.

Στον Env-Server και στη διεύθυνση:
\\195.251.131.1\courses\Undergraduate courses\3rd YEAR\6th SEMESTER\ENVIRONMENTAL APPLICATIONS OF GIS\Software\Python
υπάρχουν δύο πακέτα αυτοδιδασκαλίας για πύθων, το ένα είναι γενικά για προγραμματισμό και το άλλο είναι ειδικά για ΓΣΠ και τηλεπισκόπηση.

Για την καλύτερη κατανόηση του πύθων δίνονται πιο κάτω δύο παραδείγματα το ένα χρησιμοποιώντας μια διανυσματική επικάλυψη πολυγώνου και τη βιβλιοθήκη OGR και ένα άλλο χρησιμοποιώντας μια δορυφορική εικόνα πλέγματος και τη βιβλιοθήκη GDAL.

Παράδειγμα διανυσματικής επικάλυψης χρησιμοποιώντας Πύθων

Δίνεται η διανυσματική επικάλυψη shape file <.shp> με πολύγωνα. Πρόκειται να γραφεί ένα πρόγραμμα (script) σε πυθων και θα ονομασθεί "PolyCoords" το οποίο όταν καλείται θα δέχεται τα εξής ορίσματα: (α) τη διαδρομή <fn> στην οποία βρίσκεται το διανυσματικό αρχείο, (β) το διανυσματικό αρχείο <InPoly>, και τον αριθμό του πολυγώνου <Npoly> που θέλουμε να τυπώσουμε τις συντεταγμένες του.

Το πρόγραμμα θα εντοπίζει τις συντεταγμένες των κορυφών του πολυγώνου και θα τις τυπώνει σε ένα αρχείο με όνομα "Xyz.txt" στον υποκατάλογο <fn>. Το πρόγραμμα θα μπορεί να εργάζεται μόνο για τα εξωτερικά πολύγωνα και όχι για τα εσωτερικά.

Ο συγκεκριμένος αλγόριθμος έχει ως εξής:

1. Καλέστε τις βιβλιοθήκες που θα χρησιμοποιηθούν: OGR για τα διανυσματικά δεδομένα, και τις βιβλιοθήκες του συστήματος OS και SYS για διαδικασίες εισόδου και εξόδου δεδομένων.
2. Δώστε τη διαδρομή για το αρχείο <shape file> που θα χρησιμοποιηθεί ή οποιοδήποτε άλλο αρχείο φόρμας OGR.
3. Ενεργοποιείστε τον οδηγό "ESRI Shapefile driver".
4. Συνδέστε τον οδηγό με το όνομα του shape το οποίο είναι η πηγή δεδομένων.
5. Πάρτε από την πηγή δεδομένων το χειριστήριο της επικάλυψης.
6. Με το χειριστήριο της επικάλυψης πάρτε το πολύγωνο με αριθμό <Npoly>.
7. Από το πολύγωνο πάρτε την αναφορά στη γεωμετρία. Να σημειωθεί ότι η γεωμετρία του πολυγώνου αποτελείται από δακτύλιους (ο εξωτερικός δακτύλιος με Id=0 και μηδέν ή περισσότερους εσωτερικούς δακτύλιους) και ο εκάστοτε δακτύλιος κρατά λογαριασμό για τα σημεία και τη γεωμετρία τους με την οποία συμμετέχει στη δομή του πολυγώνου. Οι δακτύλιοι συνεπώς είναι γεωμετρία μέσα σε γεωμετρία.
8. Πάρτε το χειριστήριο του δακτύλιου από τη γεωμετρία του χαρακτηριστικού.
9. Ετοιμάστε τρεις μεταβλητές σειράς X, Y, Z για να μεταφέρουν τις συντεταγμένες των σημείων του πολυγώνου.
10. Πάρτε από το δακτύλιο τον αριθμό σημείων του εξωτερικού δακτυλίου.
11. Ανοίξτε το ASCII αρχείο "Xyz.txt" για να γράψετε τις συντεταγμένες.
12. Ξεκινήστε επαναλήψεις (βρόχο) τόσες, όσος είναι ο αριθμός των σημείων.
13. Από τη γεωμετρία του πολυγώνου πάρτε τις X, Y, Z, συν/νες σημείου.
14. Τοποθετείστε τις X, Y, Z, συν/νες μέσα στις μεταβλητές σειράς
15. Τυπώστε τις X, Y, Z, συν/νες μέσα στο ASCII αρχείο "Xyz.txt"
16. Συνεχίστε την επανάληψη
17. Τυπώστε στην οθόνη τα πρώτα 5 σημεία.
18. Αδειάστε την πηγή δεδομένων και κλείστε το ASCII αρχείο.

Ο πηγαίος κώδικας του προγράμματος δίνεται πιο κάτω και αποθηκεύεται σε ένα αρχείο με όνομα "CoordsPoly.py". Το πρόγραμμα ακολουθεί τον πιο πάνω αλγόριθμο έχοντας ενδιάμεσα πολλές επεξηγηματικές προτάσεις με #.

```
def PolyCoords(fn,InPoly,Npoly):
        # Prof. John N. Hatzopoulos
        # Open Shapefile
        import ogr, os, sys
        os.chdir(fn)
        driver = ogr.GetDriverByName('ESRI Shapefile')
        dataSource = driver.Open(InPoly, 0)
        # Get Layer
        print 'The file contains ', dataSource.GetLayerCount(), 'Layers'
        layer = dataSource.GetLayer(0)
        # Get Features
        print layer.GetName(), ' contains ', layer.GetFeatureCount(), 'features'
        feature = layer.GetFeature(Npoly)
        # Get Geometry
        geometry = feature.GetGeometryRef()
        print ' Feature contains the Geometry', geometry.GetGeometryName(), ' It contains'
        print geometry.GetGeometryCount()
        print geometry.GetGeometryName()
        # Get Geometry inside Geometry
        ring = geometry.GetGeometryRef(0)
        print geometry.GetGeometryName(), ' contains the Geometry', ring.GetGeometryName()
        print ' It contains', ring.GetPointCount(), ' points in a ', ring.GetGeometryName()
        # Write points in Vectors and Textfile
        pointsX = []; pointsY = []; pointsZ = []; k = ', '
        numpoints = ring.GetPointCount()
        file = open(r'Xyz.txt','w')
        for p in range(numpoints):
                        lon=ring.GetX(p)
                        lat=ring.GetY(p)
                        z = ring.GetZ(p)
                        # lon, lat, z = ring.GetPoint(p)
                        pointsX.append(lon)
                        pointsY.append(lat)
                        pointsZ.append(z)
                        x = [str(lon), k, str(lat), k, str(z), '\n']
                        file.writelines(x)
        print ' the first 5 points ', pointsX[0:5], pointsY[0:5] , pointsZ[0:5]
        dataSource = None
        file.close()
        #end
```

```
def PolyCoords(fn,InPoly,Npoly):
    # Prof. John N. Hatzopoulos
    # Open Shapefile
    import ogr, os, sys
    os.chdir(fn)
    driver = ogr.GetDriverByName('ESRI Shapefile')
    dataSource = driver.Open(InPoly, 0)
    # Get Layer
    print 'The file contains ', dataSource.GetLayerCount(), 'Layers'
    layer = dataSource.GetLayer(0)
    # Get Features
    print layer.GetName(), ' contains ', layer.GetFeatureCount(), 'features'
    feature = layer.GetFeature(Npoly)
```

```
# Get Geometry
geometry = feature.GetGeometryRef()
print ' Feature contains the Geometry', geometry.GetGeometryName(), ' It contains'
print geometry.GetGeometryCount()
print geometry.GetGeometryName()

# Get Geometry inside Geometry
ring = geometry.GetGeometryRef(0)
print geometry.GetGeometryName(), ' contains the Geometry', ring.GetGeometryName()
print ' It contains', ring.GetPointCount(), ' points in a ', ring.GetGeometryName()

# Write points in Vectors and Textfile
pointsX = []; pointsY = []; pointsZ = []; k = ', '
numpoints = ring.GetPointCount()
file = open(r'Xyz.txt','w')
```

```
for p in range(numpoints):
        lon=ring.GetX(p)
        lat=ring.GetY(p)
        z = ring.GetZ(p)
        # lon, lat, z = ring.GetPoint(p)
        pointsX.append(lon)
        pointsY.append(lat)
        pointsZ.append(z)
        x = [str(lon), k, str(lat), k, str(z), '\n']
        file.writelines(x)
print ' the first 5 points ', pointsX[0:5], pointsY[0:5] , pointsZ[0:5]
dataSource = None
file.close()
#end
```

Το πρόγραμμα έχει γραφεί για να λειτουργήσει σαν ανεξάρτητη μονάδα επεξεργασίας και μπορεί να κληθεί από οποιαδήποτε πλατφόρμα πύθων, αρκεί να υπάρχουν εκεί οι βιβλιοθήκες GDAL και OGR. Να σημειωθεί ότι τόσο το QGis όσο και τα FWTools περιέχουν τις βιβλιοθήκες αυτές και το πρόγραμμα αυτό έχει δοκιμασθεί και στα δύο αυτά συστήματα. Από το Qgis python console με πλατφόρμα Widows το πρόγραμμα καλείται και εκτελείται ως εξής:

```
>>> import sys
>>> sys.path.append('διαδρομή /για το /πρόγραμμα script /πυθων/')
>>> import CoordsPoly
>>> fn = ('διαδρομή /για την /επικάλυψη πολυγώνου/')
>>> CoordsPoly.PolyCoords(fn, 'Samithraki.shp',16)
>>> reload(CoordsPoly)
>>> CoordsPoly.PolyCoords(fn, 'Samothraki.shp',5)
```

Προσοχή: στις εντολές πυθων η διαδρομή έχει / και όχι \ το οποίο χρησιμοποιείται στο πυθων για συνέχεια μιας πρότασης σε άλλη γραμμή.

Η πρώτη εντολή φορτώνει τη βιβλιοθήκη του συστήματος.

Η δεύτερη εντολή θέτει μια προσωρινή διαδρομή της τοποθεσίας που βρίσκεται το πρόγραμμα πυθων (script) που πρόκειται να εκτελεσθεί, η διαδρομή αυτή διαγράφεται αυτόματα όταν βγούμε από το σύστημα των εντολών πυθων.

Η Τρίτη εντολή φορτώνει το αρχείο με το πρόγραμμα "CoordsPoly.py ".

Η τέταρτη εντολή βάζει τη διαδρομή που βρίσκεται η πολυγωνική επικάλυψη στη μεταβλητή <fn>.

Η πέμπτη εντολή εκτελεί το πρόγραμμα, να σημειωθεί το όνομα του αρχείου του προγράμματος είναι CoordsPoly.py ενώ το όνομα του προγράμματος όπως φαίνεται στην πρώτη γραμμή του πηγαίου κώδικα πιο πάνω είναι PolyCoords.

Τα ορίσματα δίνουν τη διαδρομή <fn>, το όνομα της επικάλυψης και τον αριθμό του πολυγώνου (16) που θέλουμε τις συντεταγμένες των κορυφών του.

Η έκτη εντολή επαναρχίζει τη λειτουργία του προγράμματος αν θέλουμε να λειτουργήσει για το επόμενο πολύγωνο της ίδιας επικάλυψης. Η έβδομη εντολή εκτελεί το πρόγραμμα για το πολύγωνο 5.

Ο πηγαίος κώδικας λειτουργεί με αντικείμενα από τη διανυσματική βιβλιοθήκη OGR και μέλη αντικειμένων αυτών όπως είναι οι ιδιότητες και οι μέθοδοι τα οποία χωρίζονται από τα αντικείμενα με τελίτσα (.). Οι μέθοδοι απαιτούν παρενθέσεις (), ενώ οι ιδιότητες δεν χρησιμοποιούν. Μερικά από αυτά τα μέλη αντικειμένων έχουν ως εξής:

Βήμα 3: driver = ogr.GetDriverByName('ESRI Shapefile')
Βήμα 4: dataSource = driver.Open(InPoly, 0)
…
Βήμα 10: numpoints = ring.GetPointCount()
…
Βήμα 13: lon=ring.GetX(p)
lat=ring.GetY(p)
z = ring.GetZ(p)
Βήμα 14: pointsX.append(lon)
pointsY.append(lat)
pointsZ.append(z)
Βήμα 15: x = [str(lon), k, str(lat), k, str(z), '\n']
file.writelines(x)

Παράδειγμα πλεγματικής επικάλυψης χρησιμοποιώντας Πύθων

Δίνεται μια πολυφασματική πλεγματική επικάλυψη.

Το πρόγραμμα πυθων δημιουργεί μια καινούρια πλεγματική επικάλυψη μιας μπάντας με το δείκτη βλάστησης NDVI στην ίδια φόρμα με την εικόνα εισόδου χρησιμοποιώντας τις φασματικές ζώνες 3- του κόκκινου (band3- red) και 4- του κοντινού υπέρυθρου (band4- NIR).

Ο δείκτης βλάστησης προκύπτει από τον υπολογισμό:

NDVI = (band4 - band3) / (band4 + band3)

Ο αλγόριθμος εδώ έχει ως εξής:

Καλέστε τις βιβλιοθήκες που θα χρησιμοποιηθούν: gdal, gdalconst για πλεγματικές επικαλύψεις και τις βιβλιοθήκες του συστήματος OS και SYS για διαδικασίες εισόδου και εξόδου δεδομένων.

2. Δώστε τη διαδρομή για το πλεγματικό αρχείο που θα χρησιμοποιηθεί ή οποιοδήποτε άλλο αρχείο φόρμας GDAL.
3. Φορτώστε όλους τους καταγραφείς της GDAL.
4. Πάρτε τον οδηγό για "raster format code" και καταγράψτε τον.
5. Διασυνδέστε τον οδηγό με το πλεγματικό αρχείο που περιέχει την πηγή δεδομένων.
6. Αν δεν υπάρχει το αρχείο με τα δεδομένα να τερματιστεί το πρόγραμμα.
7. Πάρτε τον αριθμό γραμμών και στηλών της πηγής δεδομένων.
8. Πάρτε περισσότερες πληροφορίες για την πηγή δεδομένων όπως: τις συντεταγμένες του πρώτου εικονοστοιχείου της εικόνας και τις διαστάσεις του εικονοστοιχείου (width, height).
9. Πάρτε τη φασματική ζώνη 3 και 4 (band3, band4) της εικόνας.
10. Μετατρέψτε τις band3, band4 σε δεκαδικό αριθμητικό φορμάτ.
11. Δημιουργείστε μια μάσκα που θα συγκρατεί τις τιμές (band4 + band3 = 0) για να αποφύγουμε τη διαίρεση με μηδέν.
12. Δημιουργείστε την επικάλυψη του NDVI στη μνήμη αφαιρώντας τις μηδενικές τιμές της μάσκας:

$$NDVI = (band4 - band3) / (band4 + band3)$$

13. Πάρτε την προβολή από τη γεωαναφορά της εικόνας εισόδου.
14. Διασυνδέστε το NDVI από τη μνήμη σε μια πλεγματική επικάλυψη δηλώνοντας δεκαδικό αριθμητικό φορμάτ.
15. Μεταφέρετε την προβολή γεωαναφοράς στην επικάλυψη εξόδου.
16. Μεταφέρετε τις τιμές του NDVI από τη μνήμη στην επικάλυψη εξόδου.
17. Υπολογίστε τις στατιστικές της επικάλυψης εξόδου.
18. Γράψτε την επικάλυψη εξόδου στο δίσκο.
19. Τυπώστε τις στατιστικές στην οθόνη.
20. Ελευθερώστε τις κατειλημμένες θέσεις της μνήμης.
21. Τέλος.

Ο πηγαίος κώδικας του προγράμματος δίνεται πιο κάτω και αποθηκεύεται σε ένα αρχείο με όνομα “NdviDef.py”. Το πρόγραμμα ακολουθεί τον πιο πάνω αλγόριθμο έχοντας ενδιάμεσα πολλές επεξηγηματικές προτάσεις με #.

```
def Ndvid(fn, multband, ndvimg, codeimg):
    # Prof. John N. Hatzopoulos
    # script to take one raster multiband
    # and do NDVI calculations - map algebra
    # import the GDAL libraries
    import os, sys
    import Numeric
    import gdal, gdalconst
    from gdalconst import *
    os.chdir(fn)
```

```
# ***********************************************
# Register all drivers
gdal.AllRegister()
# Get the driver of codeimg and register
driver = gdal.GetDriverByName(codeimg)
driver.Register()
# ***********************************************
# Open the raster data set
ds = gdal.Open(multband, GA_ReadOnly)
if ds is None:
        print 'Could not open ' + fn
        sys.exit(1)
# ***********************************************
```

```
# Get image dimensions
cols = ds.RasterXSize
rows = ds.RasterYSize
bands = ds.RasterCount
# Get more information on Geotransform
# adfGeoTransform[0] /* top left x */
# adfGeoTransform[1] /* w-e pixel resolution */
# adfGeoTransform[2] /* rotation, 0 if image is "north up"
*/
# adfGeoTransform[3] /* top left y */
# adfGeoTransform[4] /* rotation, 0 if image is "north up"
*/
# adfGeoTransform[5] /* n-s pixel resolution */
```

```
geotransform = ds.GetGeoTransform()
originX = geotransform[0]
originY = geotransform[3]
pixelWidth = geotransform[1]
pixelHeight = geotransform[5]
# ************************************
#Get individual raster bands to compute NDVI
band3 = ds.GetRasterBand(3)
band4 = ds.GetRasterBand(4)
# NDVI = (nearInfrared - Red) / (nearInfrared + Red)
# change the array data type from integer to float to allow decimals
data4 = band4.ReadAsArray(0, 0, cols, rows).astype(Numeric.Float16)
data3 = band3.ReadAsArray(0, 0, cols, rows).astype(Numeric.Float16)
# if data3 + data2 = 0 then create mask for those pixels
mask = Numeric.greater(data3 + data4, 0)
# For masked pixels assign value of zero
ndvi = Numeric.choose(mask, (0, (data4 - data3) / (data3 + data4)))
```

```
# ***********************************************
# these variables will get information about the input codeimg so we can
# write out our new codeimg into the correct geographic space
# and with correct row/column dimensions
geo = ds.GetGeoTransform()  # get the datum
proj = ds.GetProjection()   # get the projection
#shape = ds.GetRasterBand(3).shape # get the image dimensions - format (row, col)
# ***********************************************
# here we write out the new image, only one band to write out in this case
# driver = gdal.GetDriverByName(codeimg)
Outds = driver.Create(ndvimg, cols, rows, 1, gdal.GDT_Float32)
# here we set the variable dst_ds with destination filename, number of columns and rows
# 1 is the number of bands we will write out gdal.GDT_Float32 is the data type - decimals
```

```
        Outds.SetGeoTransform( geo ) # set the datum
        Outds.SetProjection( proj )  # set the projection
        # write numpy array band1 as the first band of the
multiTiff - this is the blue band
        Outds.GetRasterBand(1).WriteArray(ndvi)
        stat = Outds.GetRasterBand(1).GetStatistics(0,1)
        Outds.FlushCache()
        # get the band statistics (min, max, mean, standard
deviation)
        # set the stats we just got to the band
        # Outds.GetRasterBand(1).SetStatistics(stat[0], stat[1],
stat[2], stat[3])
        print stat
        ds = None
        outds = None
        #end
```

Το πρόγραμμα έχει γραφεί για να λειτουργήσει σαν ανεξάρτητη μονάδα επεξεργασίας και μπορεί να κληθεί από οποιαδήποτε πλατφόρμα πύθων, αρκεί να υπάρχουν εκεί οι βιβλιοθήκες GDAL και OGR.

Να σημειωθεί ότι τόσο το QGis όσο και τα FWTools περιέχουν τις βιβλιοθήκες αυτές.

Το παρόν πρόγραμμα έχει δοκιμασθεί και τρέχει στο FWTools 2.4.7. Στο Qgis έχει πρόβλημα και πρέπει να γίνουν κάποιες αλλαγές στον πηγαίο κώδικα όπως αντί του Numeric πρέπει να χρησιμοποιηθεί το numpy.

Στο παρόν παράδειγμα χρησιμοποιείται μια εικόνα Landsat-TM σε φόρμα GeoTiff. Από το Qgis python console με πλατφόρμα Widows το πρόγραμμα καλείται και εκτελείται ως εξής:

```
>>> import sys
>>> sys.path.append('διαδρομή /για το /πρόγραμμα script /πυθων/')
>>> import NdviDef
>>> fn = ('διαδρομή /για την /πλεγματική επικάλυψη/")
>>>NdviDef.Ndvid(fn,'LandsatLesvosOnly-Egsa.tif','LesvosNDVImg.tif','GTiff')
```

Προσοχή: στις εντολές πυθων η διαδρομή έχει / και όχι \ το οποίο χρησιμοποιείται στο πυθων για συνέχεια μιας πρότασης σε άλλη γραμμή.

Η πρώτη εντολή φορτώνει τη βιβλιοθήκη του συστήματος.

Η δεύτερη εντολή θέτει μια προσωρινή διαδρομή της τοποθεσίας που βρίσκεται το πρόγραμμα πυθων (script) που πρόκειται να εκτελεσθεί, η διαδρομή αυτή διαγράφεται αυτόματα όταν βγούμε από το σύστημα των εντολών πυθων.

Η Τρίτη εντολή φορτώνει το αρχείο με το πρόγραμμα "NdviDef.py".

Η τέταρτη εντολή βάζει τη διαδρομή που βρίσκεται η πλεγματική επικάλυψη στη μεταβλητή <fn>.

Η πέμπτη εντολή εκτελεί το πρόγραμμα, να σημειωθεί το όνομα του αρχείου του προγράμματος είναι NdviDef.py ενώ το όνομα του προγράμματος όπως φαίνεται στην πρώτη γραμμή του πηγαίου κώδικα πιο πάνω είναι Ndvid.

Τα ορίσματα δίνουν τη διαδρομή <fn>, το όνομα της επικάλυψης εισόδου 'LandsatLesvosOnly-Egsa.tif', το όνομα της επικάλυψης εξόδου 'LesvosNDVImg.tif' και τον κωδικό της φόρμας των πλεγματικών αρχείων 'GTiff'.

Η έκτη εντολή επαναρχίζει τη λειτουργία του προγράμματος αν θέλουμε να ξανατρέξουμε το πρόγραμμα.

Ο πηγαίος κώδικας λειτουργεί με αντικείμενα από τη διανυσματική βιβλιοθήκη GDAL και μέλη αντικειμένων αυτών όπως είναι οι ιδιότητες και οι μέθοδοι τα οποία χωρίζονται από τα αντικείμενα με τελίτσα (.).

Οι μέθοδοι απαιτούν παρενθέσεις (), ενώ οι ιδιότητες δεν χρησιμοποιούν. Μερικά από αυτά τα μέλη αντικειμένων έχουν ως εξής:

Βήμα 3: gdal.AllRegister()
Βήμα 4: driver = gdal.GetDriverByName(codeimg)
…
Βήμα 10: data4 = band4.ReadAsArray(0, 0, cols, rows).astype(Numeric.Float16)
data3 = band3.ReadAsArray(0, 0, cols, rows).astype(Numeric.Float16)
…
Βήμα 13: geo = ds.GetGeoTransform() # get the datum
proj = ds.GetProjection() # get the projection
Βήμα 14: Outds = driver.Create(ndvimg, cols, rows, 1, gdal.GDT_Float32)
Βήμα 15: Outds.SetGeoTransform(geo) # set the datum
Outds.SetProjection(proj) # set the projection

Ζητούνται τα εξής:

1. Πληκτρολογήσετε σε επεξεργαστή κειμένου το πιο πάνω πρόγραμμα PolyCoords και να το αποθηκεύσετε με το όνομα CoordsPoly.py.
2. Χρησιμοποιείστε τα διανυσματικά δεδομένα <Samothraki.shp> και χρησιμοποιείστε είτε το QGis ή τα FWTools και τρέξτε σύμφωνα με τις πιο πάνω οδηγίες το πρόγραμμα CoordsPoly.py και τυπωστε τις συντεταγμένες των κορυφών στα εξής πολύγωνα: 3, 6, 11, 14, 15. Πόσα πολύγωνα έχει η επικάλυψη αυτή;
3. Πληκτρολογήσετε σε επεξεργαστή κειμένου το πιο πάνω πρόγραμμα Ndvid και να το αποθηκεύσετε με το όνομα NdviDef.py.
4. Χρησιμοποιείστε τα πλεγματικά δεδομένα <LandsatLesvosOnly-Egsa.tif> και χρησιμοποιείστε είτε το QGis ή τα FWTools και τρέξτε σύμφωνα με τις πιο πάνω οδηγίες το CoordsPoly.py και δημιουργείστε μια επικάλυψη με το NDVI με όνομα μέρος από το επώνυμο σας.
5. Παραδώστε αναφορά με περίληψη της διαδικασίας και με αποτελέσματα με τις συντεταγμένες των κορυφών των πολυγώνων με κατάλληλο τίτλο για κάθε πολύγωνο, απάντηση το ερώτημα, και αρχείο με το δείκτη βλάστησης καθώς και αρχεία με τα αντίστοιχα προγράμματα πυθων (scripts)

Προσοχή: Στα FWTools τυπώστε python στη γραμμή εντολών για να ενεργοποιηθεί ο πυθων με τα >>> που είναι το σύμβολο εισαγωγής εντολών του πυθων (command prompt).

Σημείωση: Στα προγράμματα ανοιχτού κώδικα χρειάζεται λίγη εμπειρία για τις βιβλιοθήκες (import) που χρησιμοποιούνται και αν αυτές είναι πρόσφατες ή παλιές ώστε να ταιριάζουν με το πρόγραμμα καθώς και βιβλιοθήκες .dll.

Επίσης για οποιοδήποτε πρόβλημα βάζουμε το μήνυμα λάθους (error message) στη μηχανή αναζήτησης στο διαδίκτυο και συνήθως παίρνουμε την απάντηση.