Μάθημα

Εφαρμοσμένη Αριθμητική Ανάλυση

Επιβλέποντες καθηγητές:

Γιαννουτάκης Κωνσταντίνος

Χαλκίδης Σπυρίδων

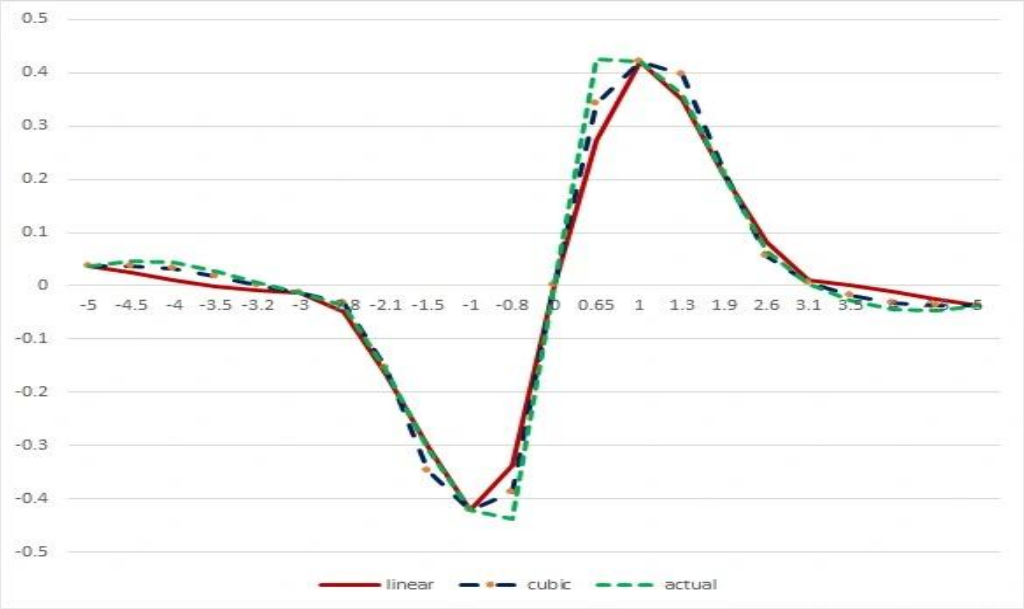
Τίτλος:

Μέθοδοι δικυβικής παρεμβολής για αλλαγή διαστάσεων εικόνας

Σιαλάκης Ραφαήλ Χρυσοβαλάντης ics22006
Χαλεπλής Διονύσης ics24060
Κευσενίδης Παρασκευάς ics21045

Εισαγωγή

Η παρεμβολή σε εικόνες είναι η διαδικασία που χρησιμοποιείται για την ανακατασκευή ή την αλλαγή μεγέθους μιας εικόνας, βασισμένη στη δειγματοληψία των εικονοστοιχείων (pixels). Η κυβική παρεμβολή, στην οποία αναφέρεται το παρόν άρθρο, είναι μια μέθοδος που χρησιμοποιείται ευρέως από προγράμματα επεξεργασίας εικόνων, καθώς προσφέρει πολύ καλύτερα αποτελέσματα από την παρεμβολή του κοντινότερου γείτονα, η οποία δεν υπολογίζει καινούργιες τιμές, καθώς και από την γραμμική παρεμβολή, η οποία ανακατασκευάζει την εικόνα χρησιμοποιώντας ευθείες. Η χρήση των κυβικών πολυωνύμων θεωρείται βέλτιστη, διότι ενέχει ισορροπία μεταξύ απόδοσης και πολυπλοκότητας. Προφανώς οι απαιτήσεις σε υπολογιστικούς πόρους αυξάνονται σε σχέση με μεθόδους που αξιοποιούν πολυώνυμα χαμηλότερου βαθμού, αλλα ταυτόχρονα μειώνεται το προσεγγιστικό σφάλμα του κάθε σημείου που υπολογίζουμε, γεγονός που έχει ως αποτέλεσμα μια πιο ακριβή παρεμβολή. Ο λόγος που χρησιμοποιούνται τα πολυώνυμα τρίτου βαθμού και όχι πολυώνυμα μεγαλύτερου βαθμού, είναι διότι τα πολυώνυμα μεγαλύτερου βαθμού παρουσιάζουν μια πολύ μεγάλη "ταλαντωτική" συμπεριφορά, η οποία δεν είναι επιθυμητή για προσέγγιση συναρτήσεων που έχουν ομαλή συμπεριφορά. Επίσης έχουμε και υπολογιστικά προβλήματα όταν ο αριθμός των δεδομένων είναι μεγάλος. Έτσι αν έχουμε 100 σημεία (x,y), i = 0,1,...,99 θα ήταν παράτολμη η αναζήτηση του πολυωνύμου με βαθμό 99, τέτοιο ώστε να ικανοποιείται η σχέση: $P(x) = f_i$, $i \in [0,100)$. Με σκοπό να ξεπεραστεί η ταλαντωτική συμπεριφορά των πολυωνύμων, χρησιμοποιούμε την παρεμβολή των splines, η οποία παρουσιάστηκε από τον Schoenberg το 1946, και από τότε αποτελεί αντικείμενο εκτεταμένης έρευνας.



Σχήμα 1.1 Σύγκριση μεθόδων παρεμβολής

Περιεχόμενα

1. Δικυβική παρεμβολή

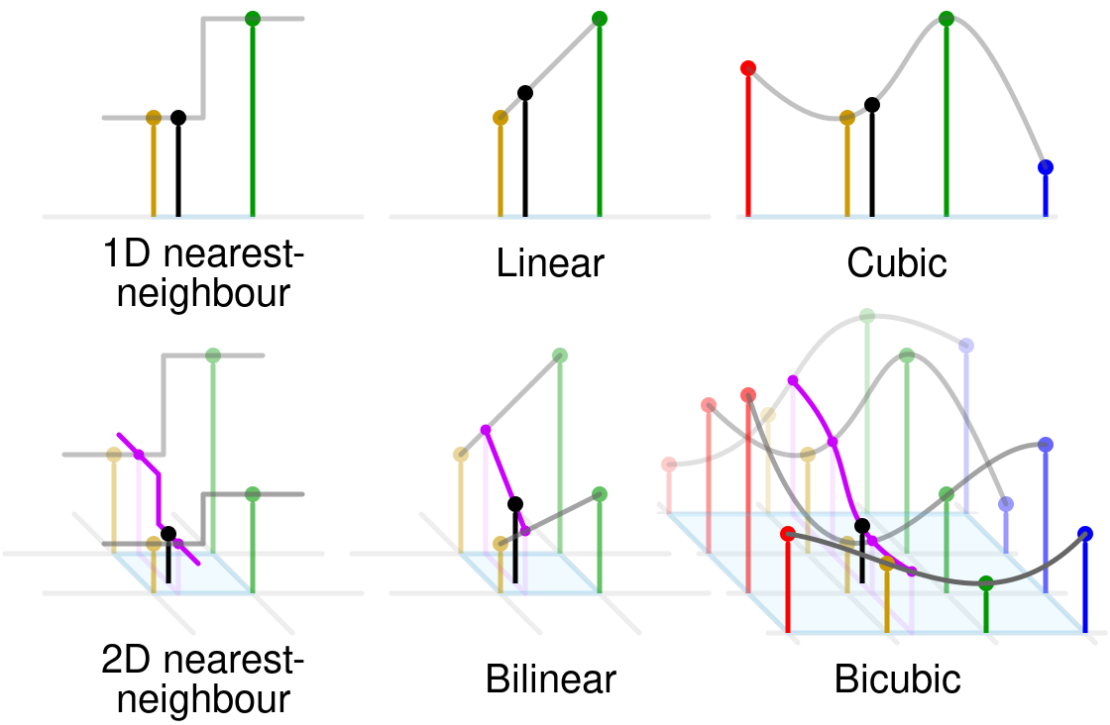
Η δικυβική παρεμβολή (bicubic interpolation) είναι μια μέθοδος πολυωνυμικής παρεμβολής που επεκτείνει την κυβική παρεμβολή σε δύο διαστάσεις. Η παρεμβολή της μεθόδου αυτής είναι πιο λεία από τους αλγορίθμους δι-γραμμικής και γειτονικού κόμβου παρεμβολής, γι' αυτό και προτιμάται στην επεξεργασία εικόνας. Ο αλγόριθμος λειτουργεί με την χρήση δύο συναρτήσεων:

1) Cubic Weight, με α=-0.5 (μια **τυπική τιμή** που έχει προταθεί και χρησιμοποιηθεί ευρέως στη βιβλιογραφία για δικυβική παρεμβολή)

$$W(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{for } |x| \le 1, \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{for } 1 < |x| < 2, \\ 0 & \text{otherwise,} \end{cases}$$

2) Τύπος επιφάνειας παρεμβολής (interpolation surface)

$$p(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j.$$



Εικόνα 1.1 Γραφική αναπαράσταση μονοδιάστατων και δισδιάστατων μεθόδων παρεμβολής

1.1 Κυβικό Βάρος

Το κυβικό βάρος είναι μια συνάρτηση που χρησιμοποιούμε στην δικυβική παρεμβολή για να υπολογίσουμε πόσο "συμβάλλει" κάθε γειτονικό pixel στη συνολική τιμή που θέλουμε να βρούμε. Η συνάρτηση αυτή εξαρτάται από την απόσταση του σημείου που υπολογίζουμε από κάθε γειτονικό pixel και περιγράφεται με τον παρακάτω μαθηματικό τύπο.

$$W(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{for } |x| \leq 1, \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{for } 1 < |x| < 2, \\ 0 & \text{otherwise,} \end{cases}$$

Αυτό σημαίνει ότι:

1. Αν η απόσταση από το σημείο είναι μικρότερη ή ίση από 1, τότε το βάρος είναι μεγαλύτερο.
2. Αν η απόσταση είναι μεταξύ 1 και 2, το βάρος γίνεται μικρότερο.
3. Αν η απόσταση είναι μεγαλύτερη από 2, το βάρος γίνεται μηδέν, δηλαδή το pixel δεν συμβάλλει καθόλου.

Η παράμετρος α στον τύπο καθορίζει το πόσο ομαλή ή αιχμηρή είναι η παρεμβολή. Συνήθως θέτουμε το α ίσο με τις τιμές -0.5 ή -0.75. Η χρήση του α = -0.5 έχει προταθεί από τον Keys, καθώς παράγει ένα πιο ισορροπημένο αποτέλεσμα. Η επιλογή ενός α > -0.5, αυξάνει την ένταση της ομαλότητας, αλλά μειώνει την ακρίβεια των παρεμβαλλόμενων τιμών, καθώς η καμπύλη τείνει να γίνεται πιο “επίπεδη” γύρω από τα δεδομένα. Αντιθέτως για α < -0.5, η καμπύλη τείνει να γίνεται πιο “αιχμηρή” και μπορεί να ταιριάζει καλύτερα με τα δεδομένα, αλλά ενδέχεται να εμφανίσει υπερβολές (overshooting). Μπορεί να είναι κατάλληλο για δεδομένα με έντονες μεταβολές, αλλά όχι για δεδομένα με θόρυβο.

1.2 Υπολογισμός Παρεμβολής

Ο τύπος της δικυβικής παρεμβολής στη δικυβική μέθοδο είναι ο τρόπος που βρίσκουμε την τιμή ενός σημείου ανάμεσα σε άλλα σημεία, βασισμένοι στις τιμές των γειτονικών pixels. Στη δικυβική παρεμβολή, αυτό γίνεται με βάση τα βάρη που υπολογίζονται από το κυβικό βάρος. Η διαδικασία ξεκινάει με την επιλογή των γειτονικών pixel, ορίζοντας ένα 4x4 πλέγμα γύρω από το σημείο που μας ενδιαφέρει. Ακολουθεί ο υπολογισμός της απόστασης του σημείου από κάθε γειτονικό pixel στις διαστάσεις (x,y). Εφαρμόζεται στη συνέχεια το κυβικό βάρος για την εύρεση της συμβολής του κάθε pixel στην ανάθεση τιμών στα καινούργια pixels και τέλος το σταθμισμένο άθροισμα χρησιμοποιείται για τον υπολογισμό του αθροίσματος όλων των τιμών

του πλέγματος, ως γινόμενο με τα αντίστοιχα βάρη τους, τόσο στον άξονα x'x, όσο και στον y'y.

Ο μαθηματικός τύπος είναι:

$$p(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} w(x - x_i) w(y - y_j)$$

Όπου:

1. Τα a_{ij} είναι οι τιμές των pixels.
2. Τα $w(x - x_i) * w(y - y_j)$ είναι τα βάρη στις δύο διαστάσεις.

Αυτό σημαίνει ότι για να βρούμε την παρεμβολημένη τιμή p(x,y), κοιτάμε τα 16 πιο κοντινά pixels, βλέπουμε πόσο σημαντικό είναι το καθένα (ανάλογα με την απόσταση) και συνδυάζουμε τις τιμές τους με βάση τα βάρη.

Είναι σαν να "ζυγίζουμε" τη συμβολή του κάθε pixel, με τα πιο κοντινά να έχουν μεγαλύτερη επίδραση και τα πιο μακρινά μικρότερη.

1.3 Κλιμάκωση εικόνας

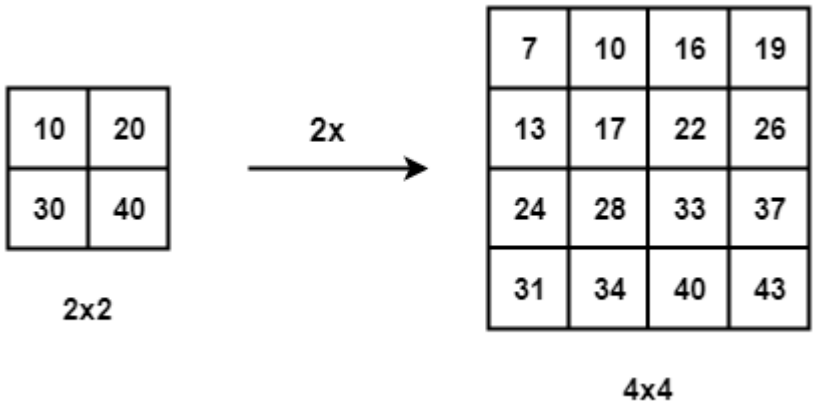
Η κλιμάκωση αναφέρεται στη διαδικασία αύξησης ή μείωσης των διαστάσεων μιας εικόνας μέσω της αλλαγής του αριθμού των pixels που την αποτελούν. Πρακτικά, αυτό σημαίνει ότι αλλάζουμε το μέγεθος της εικόνας, είτε μεγαλώνοντάς την για να εμφανιστεί σε υψηλότερη ανάλυση, είτε μικραίνοντάς την για να καταλάβει λιγότερο χώρο ή να προσαρμοστεί σε συγκεκριμένες ανάγκες.

Για να επιτευχθεί αυτή η διαδικασία, χρησιμοποιούμε έναν **συντελεστή κλιμάκωσης** (scale factor), ο οποίος καθορίζει πόσο θα αλλάξουν οι διαστάσεις της εικόνας. Ο συντελεστής αυτός εφαρμόζεται σε κάθε σημείο της νέας εικόνας για να βρούμε τη θέση του αντίστοιχου σημείου στην αρχική εικόνα. Η σχέση αυτή περιγράφεται από τον τύπο:

$$oldpixel = \frac{newpixel}{scale\ factor}$$

Με άλλα λόγια, για κάθε pixel της νέας εικόνας, υπολογίζουμε πού θα αντιστοιχούσε στην αρχική εικόνα, λαμβάνοντας υπόψη τον συντελεστή κλιμάκωσης. Αυτό μας βοηθά να διατηρήσουμε την οπτική ποιότητα και να εξασφαλίσουμε ότι τα νέα pixels αντανακλούν τις σωστές πληροφορίες. Η διαδικασία της κλιμάκωσης μπορεί να περιλαμβάνει είτε μεγέθυνση είτε σμίκρυνση:

- **Μεγέθυνση**: Σε αυτή την περίπτωση, η εικόνα αποκτά περισσότερα pixels από την αρχική της, γεγονός που επιτρέπει την εμφάνιση περισσότερων λεπτομερειών ή την προσαρμογή σε μεγαλύτερες αναλύσεις. Για τη μεγέθυνση, τα νέα pixels υπολογίζονται με παρεμβολή από τα γειτονικά pixels της αρχικής εικόνας.
- **Σμίκρυνση**: Εδώ, η εικόνα μειώνεται σε μέγεθος, και επομένως, πολλά από τα αρχικά pixels "συγχωνεύονται". Αυτό απαιτεί την κατάλληλη μεθοδολογία για να αποφύγουμε την απώλεια σημαντικών πληροφοριών.



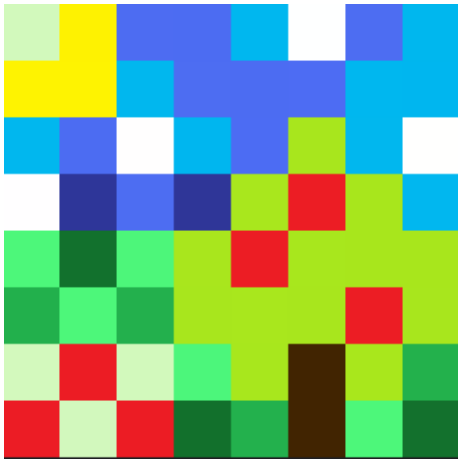
1.4 Υλοποίηση Αλγορίθμου

Βάσει όσων αναλύθηκαν παραπάνω, αναπτύξαμε και τον κώδικα σε python. Για την υλοποίηση της μεθόδου χρησιμοποιήθηκαν δύο εξωτερικές βιβλιοθήκες της python: η numpy, για την μαθηματική πράξη floor και η cv2 για ανάγνωση και εγγραφή στην εικόνα.

Αρχικά, το πρόγραμμα, δέχεται 3 παραμέτρους: το όνομα της αρχικής εικόνας, τον συντελεστή κλιμάκωσης και το όνομα της εικόνας, που θα δημιουργηθεί μετά την επεξεργασία της αρχικής. Έχοντας αυτά τα δεδομένα, το πρόγραμμα εκτελεί τη μέθοδο scale_image. Αυτή με τη σειρά της υπολογίζει το νέο ύψος και το νέο πλάτος της εικόνας, βάσει των αρχικών αυτών τιμών και του συντελεστή κλιμάκωσης. Το νέο ύψος και πλάτος συνδυάζονται σε έναν διπλά εμφωλευμένο βρόχο, ο οποίος είναι υπεύθυνος για τον υπολογισμό κάθε φορά της τιμής του oldpixel, βάσει του τύπου που περιγράφηκε στην υποενότητα 1.3. Στο τέλος κάθε βρόχου, καλεί τη μέθοδο bicubic_interpolate.

Η μέθοδος bicubic_interpolate υλοποιεί τον τύπο υπολογισμού παρεμβολής. Εδώ, γίνεται ο υπολογισμός του βάρους των 16 γειτονικών pixel του oldpixel, το οποίο η μέθοδος δέχτηκε ως είσοδο. Στο τέλος επιστρέφεται το αποτέλεσμα της συνάρτησης υπολογισμού παρεμβολής, πίσω στην καλούσα μέθοδο.

Με το τέλος της scale_image, η εικόνα πλέον έχει αλλάξει μέγεθος με βάση τον συντελεστή κλιμάκωσης και έχει επιστραφεί στον χρήστη, για επισκόπηση. Παρακάτω παρουσιάζεται η εικόνα που χρησιμοποιήθηκε ως είσοδος στο πρόγραμμα, καθώς και η διαμορφωμένη εικόνα που το πρόγραμμα δημιούργησε ως έξοδο:



Εικόνα 1.1 Εικόνα 8x8 στην οποία υλοποιήθηκε η δικυβική παρεμβολή

Εικόνα 1.2 Εικόνα που μεγεθύνθηκε με μεγενθυτικό παράγοντα 20

2. Δικυβική τμηματική παρεμβολή

Η δικυβική τμηματική παρεμβολή (bicubic spline interpolation) είναι μια αριθμητική μέθοδος πολυωνυμικής παρεμβολής η οποία βασίζεται στα splines για τα οποία ο μαθηματικός ορισμός παρουσιάζεται παρακάτω. Η συγγραφή του κώδικα για το συγκεκριμένο πρόβλημα έγινε διαιρώντας το, σε υποπροβλήματα. Πιο συγκεκριμένα στην αρχή το πρόβλημα επιλύθηκε σε μια διάσταση, με εικόνες της μορφής 16x1, οι οποίες ήταν ασπρόμαυρες, ώστε η παρεμβολή να εφαρμοστεί μόνο σε μια τιμή που ανήκει στο διάστημα [0,255], και όχι σε ένα tuple της μορφής (x_a,x_b,x_c), $x_a,x_b,x_c \in [0,255]$, που προκύπτει από τις έγχρωμες εικόνες. Έχοντας επιλύσει το πρόβλημα σε μια διάσταση, χρησιμοποιήθηκε ο αλγόριθμος της δικυβικής τμηματικής παρεμβολής, αρχικά σε κάθε γραμμή της εικόνας και στη συνέχεια σε κάθε στήλη της εικόνας, δεδομένου ότι μια εικόνα αποτελεί έναν πίνακα mxn, όπου $m,n \in [0,255]$. Τέλος, έχοντας ολοκληρώσει την παρεμβολή των στηλών, πραγματοποιούμε μετασχηματισμό του πίνακα πάλι σε μορφή γραμμών, όπως ήταν εξαρχής. Δηλαδή, για κάθε στήλη που έχει δημιουργηθεί μετά την κυβική παρεμβολή, δημιουργούμε νέες γραμμές, τέτοιες ώστε τα πρώτα στοιχεία κάθε στήλης να συγχωνευθούν στην πρώτη γραμμή, τα δεύτερα στοιχεία κάθε στήλης να συγχωνευθούν στη δεύτερη γραμμή κλπ.

2.1 Κυβικές splines

Έστω $a = x_0 < \cdots < x_n = b$ ένας διαμερισμός του διαστήματος $[a, b]$ και m θετικός ακέραιος. Μία συνάρτηση $s{:}[a, b] \to \mathbb{R}$, θα ονομάζεται πολυώνυμο spline βαθμού m ως προς τον διαμερισμό
$x_0 < \cdots < x_n$, αν και μόνο αν • Η s είναι $m - 1$ φορές συνεχώς παραγωγίσιμη στο $[a, b]$
($s \in C^{m-1}[a, b]$) Η s αποτελεί πολυώνυμο m βαθμού για κάθε $x \in x_i$, x_{i+1} , $i = 0,1, \dots , n - 1$
Θέλουμε να κατασκευάσουμε μια spline 3ου βαθμού, που να παρεμβάλει την f στα σημεία
$x_0, x_1, \dots , x_n$. Συνολικά οι συνθήκες που έχουμε να υπολογίσουμε είναι:

1) $s_i(x_i) = f(x_i)$, $s_i(x_{i+1}) = f(x_{i+1})$, $x_i \in [x_i,x_{i+1}]$, που δίνουν 2n συνθήκες,
2) $s_i'(x_i) = s_i'(x_{i+1})$, $s_i''(x_i) = s_i''(x_{i+1})$, $x_i \in [x_i,x_{i+1}]$, λόγω συνέχειας, που δίνει 2n-2 συνθήκες
3) Οριακές συνθήκες $s_0''(x_0) = 0$ και $s_{n-1}''(x_n) = 0$ (φυσικές κυβικές splines)

Επομένως συνολικά προκύπτουν 4n συνθήκες.

2.2 Αλγόριθμος του Thomas

Στη γραμμική άλγεβρα, ένας τριδιαγώνιος πίνακας είναι ένας πίνακας ζώνης που έχει μη μηδενικά στοιχεία μόνο στην κύρια διαγώνιο στην πρώτη διαγώνιο κάτω από αυτήν και στην πρώτη διαγώνιο πάνω από την κύρια διαγώνιο. Ένα τριδιαγώνιο γραμμικό σύστημα είναι ένα σύστημα της μορφής:

$$\begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & 0 \\ & a_3 & b_3 & \ddots & \\ & 0 & \ddots & \ddots & c_{n-1} \\ & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix}$$

το οποίο μπορεί να λυθεί αποδοτικά με τον αλγόριθμο του Thomas. Κατά τη μέθοδο αυτή εφαρμόζουμε απαλοιφή Gauss, ώστε να μηδενίσουμε διαδοχικά τις στήλες του επαυξημένου πίνακα εφαρμόζοντας γραμμοπράξεις, και στη συνέχεια επιλύουμε το σύστημα με προς τα πίσω αντικατάσταση.

2.3 Θεωρητική υλοποίηση

Η υλοποίηση της μεθόδου που περιγράφηκε στο 2.1 μπορεί να υλοποιηθεί με μια μέθοδο που μας επιτρέπει να υπολογίσουμε τις φυσικές κυβικές splines για οποιαδήποτε σημεία παρεμβολής. Αρχικά αν θεωρήσουμε τα πολυώνυμα παρεμβολής της μορφής:

$$s_i(\mathrm{x}) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i$$

και ορίσουμε:

$$\mathrm{h} = x_{i+1} - x_i \quad \text{και} \quad M_i = s_i''(x_i)$$

τότε μπορούμε να υπολογίσουμε τις δεύτερες παραγώγους των πολυωνύμων λύνοντας την εξίσωση:

$$\frac{h}{6}M_{i-1} + \frac{2h}{3}M_i + \frac{h}{6}M_{i+1} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h}, i = 1,2, \dots, n-1$$

Τέλος, γνωρίζουμε ότι από τις οριακές συνθήκες των φυσικών splines προκύπτει ότι:

$$\mathrm{M_0} = \mathrm{M_n} = 0$$

Ενώ η παραπάνω εξίσωση μπορεί να γραφτεί στη μορφή ενός τριδιαγώνιου γραμμικού συστήματος:

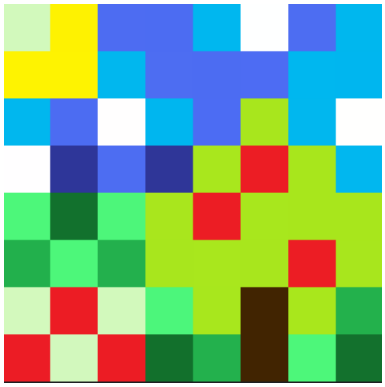
$$\begin{bmatrix} \frac{2h}{3} & \frac{h}{6} & 0 & & & 0 \\ \frac{h}{6} & \frac{2h}{3} & \frac{h}{6} & & & \\ 0 & \frac{h}{6} & & & & 0 \\ & & & & \frac{2h}{3} & \frac{h}{6} \\ 0 & & & \frac{h}{6} & \frac{2h}{3} & \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n-1} \end{bmatrix} = \begin{bmatrix} \frac{y_2 - 2y_1 + y_0}{h} \\ \frac{y_3 - 2y_2 + y_1}{h} \\ \vdots \\ \frac{y_n - 2y_{n-1} + y_{n-2}}{h} \end{bmatrix}$$

Το γραμμικό αυτό σύστημα επιλύεται με τη μέθοδο που αναπτύξαμε στο **2.2**, ενώ οι συντελεστές του πολυωνύμου προσδιορίζονται από τις σχέσεις:

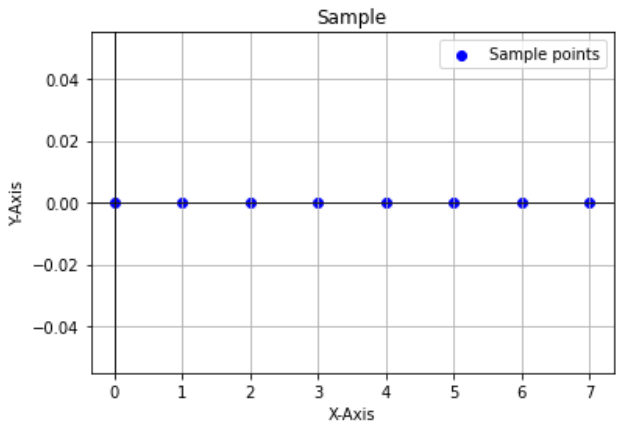
$$a_i = \frac{M_{i+1} - M_i}{6h}, b_i = \frac{M_i}{2}, c_i = \frac{y_{i+1} - y_i}{h} - \frac{h}{6}(M_{i+1} + 2M_i), d_i = y_i$$

2.4 Αναδειγματοληψία

Μια ακόμη πρόκληση που αντιμετωπίσαμε κατά την υλοποίηση της συγκεκριμένης μεθόδου ήταν η αναδειγματοληψία, δηλαδή εφόσον έχουμε ορίσει τους συντελεστές και τα κυβικά πολυώνυμα, πόσα και ποιά νέα σημεία θα λάβουμε από κάθε διάστημα $[x_i,x_{i+1}]$ με την όσο το δυνατόν καλύτερη κατανομή των σημείων στο διάστημα αυτό. Έστω λοιπόν ότι έχουμε αρχικά ένα δείγμα των 8 σημείων και θέλουμε να παράγουμε ένα δείγμα των 16 σημείων. Το δείγμα Α ανήκει στο διάστημα [0,7] και η γραφική του αναπαράσταση παρουσιάζεται παρακάτω. Τονίζεται ότι στα παρακάτω σχήματα τα σημεία αναπαρίστανται πάνω στον οριζόντιο άξονα, ή στην ευθεία y = 0, καθώς ο σκοπός της υποενότητας είναι οι μέθοδοι κατανομής των σημείων και όχι η αναπαράσταση τους σε πολυώνυμα.



Εικόνα 2.1 Εικόνα 8x8 στην οποία υλοποιήθηκε η δικυβική τμηματική παρεμβολή



Σχήμα 2.1 Αναπαράσταση δείγματος Α στον οριζόντιο άξονα

Έχοντας ως σημείο αναφοράς το παραπάνω δείγμα θα παρουσιάσουμε τις δύο μεθόδους που χρησιμοποιήθηκαν κατά τη δειγματοληψία, καθώς και τα προβλήματα που παρουσιάστηκαν στην πορεία. Για να συζητήσουμε για τη δημιουργία νέου δείγματος θα πρέπει να ορίσουμε το Δx το οποίο προκύπτει από τον εξής τύπο:

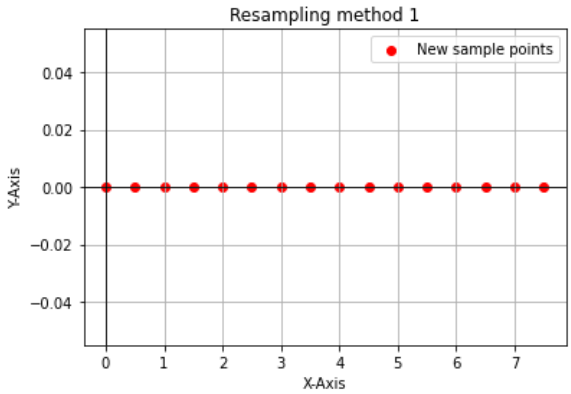
$$\Delta x = \frac{sample\_size}{new\_sample\_size}$$

Μέθοδος 1

Έστω n το πλήθος των σημείων ενός νέου δείγματος A που θέλουμε να ορίσουμε. Το δείγμα μας ορίζεται από το σύνολο A = $\{x_0, x_1,..,x_n\}$. Το σύνολο του νέου δείγματος ορίζεται από τη σχέση:

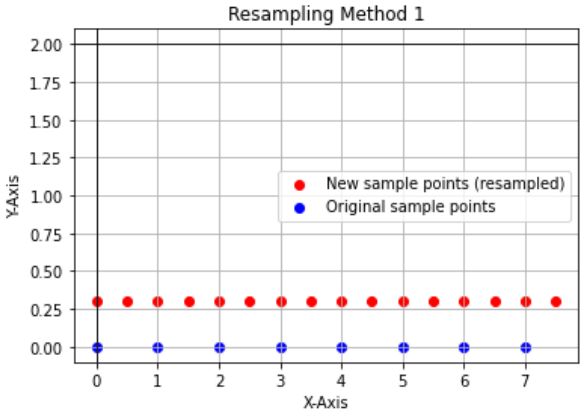
$$x_{resampled} = \Delta x * x_i$$

Έστω το δείγμα A, το οποίο είναι ένα σύνολο που αποτελείται από 8 στοιχεία, A = {0,1,...,7}. Θέλουμε να δημιουργήσουμε ένα νέο δείγμα το οποίο αποτελείται από 16 στοιχεία βάσει της **Μεθόδου 1** που ορίσαμε παραπάνω. Παρακάτω παρουσιάζεται το γράφημα του νέου δείγματος:



Σχήμα 2.2 Αναπαράσταση αναδειγματοληψίας με τη μέθοδο 1 στον οριζόντιο άξονα

Παρατηρούμε αρχικά ότι οι τιμές δεν είναι ισότιμα κατανεμημένες στο διάστημα [0,7], και υπάρχει μια οριακή τιμή $x_n > 7$, επομένως δεν υπάρχει κάποιο spline s_i τέτοιο ώστε να μπορούν να υπολογιστούν οι τιμές στο διάστημα [7, x_n).



Σχήμα 2.3 Σύγκριση του αρχικού δείγματος με το νέο δείγμα

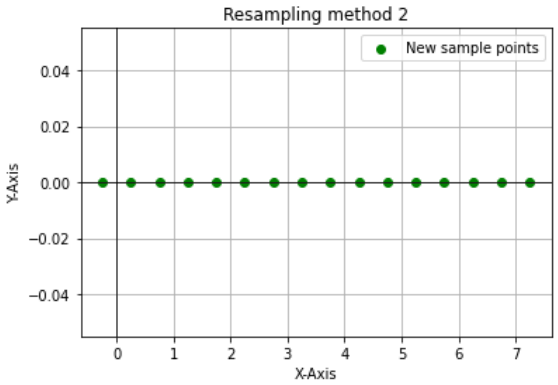
Η ανισοκατανομή των τιμών στο διάστημα [0,7] μας ανάγκασε να δοκιμάσουμε μια διαφορετική μέθοδο, η οποία περιγράφεται παρακάτω:

Μέθοδος 2

Έστω n το πλήθος των σημείων ενός νέου δείγματος A που θέλουμε να ορίσουμε. Το δείγμα μας ορίζεται από το σύνολο $A = \{x_0, x_1,..,x_n\}$. Το σύνολο του νέου δείγματος ορίζεται από τη σχέση:

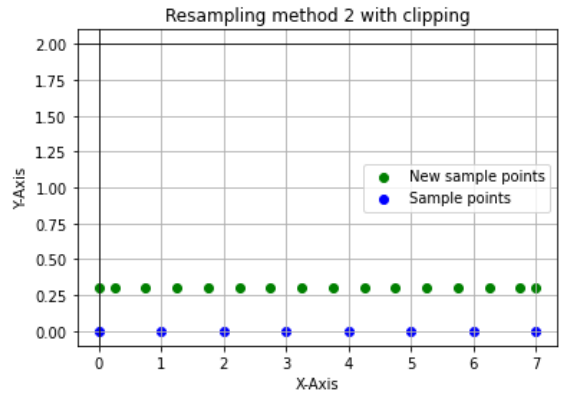
$$x_{resampled} = (n + 0.5) * \Delta x - 0.5, \quad n \in [x_0, x_n]$$

Έστω το δείγμα A, το οποίο είναι ένα σύνολο που αποτελείται από 8 στοιχεία, $A = \{0,1,...,7\}$. Θέλουμε να δημιουργήσουμε ένα νέο δείγμα το οποίο αποτελείται από 16 στοιχεία με βάση τη **Μέθοδο 2** που ορίσαμε παραπάνω. Παρακάτω παρουσιάζεται το γράφημα του νέου δείγματος:



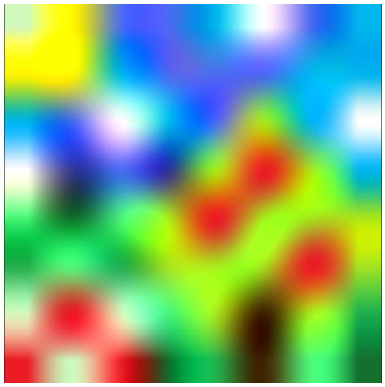
Σχήμα 2.4 Αναπαράσταση αναδειγματοληψίας με τη μέθοδο 2 στον οριζόντιο άξονα

Η συγκεκριμένη μέθοδος προσφέρει μια πιο ίση κατανομή των σημείων στο διάστημα [0,7], παρόλα αυτά προκύπτουν 2 οριακές τιμές $x_0 < 0$ και $x_n > 7$, επομένως δεν υπάρχει κάποιο spline s_i τέτοιο ώστε να μπορούν να υπολογιστούν οι τιμές στα διαστήματα $(x_0, 0]$ και $[7, x_n)$. Επομένως η διαχείριση των σημείων αυτών έγινε με τη μέθοδο της αποκοπής, όπου αν $x_i < 0$ τότε $x_i = \min(x_i,0)$, ενώ αν $x_i > x_n$ τότε $x_i = \max(x_n,x_i)$.



Σχήμα 2.5 Σύγκριση του αρχικού δείγματος με το νέο δείγμα με τη **Μέθοδο 2**.

Το πρόβλημα με τη συγκεκριμένη μέθοδο είναι ότι η παρεμβολή στο border της εικόνας δεν είναι ακριβής και εμφανίζονται artifacts περιμετρικά της εικόνας, τα οποία είναι ορατά με γυμνό μάτι.



Εικόνα 2.2 Εικόνα με artifacts περιμετρικά λόγω της **Μεθόδου 2** με μεγενθυτικό παράγοντα 20

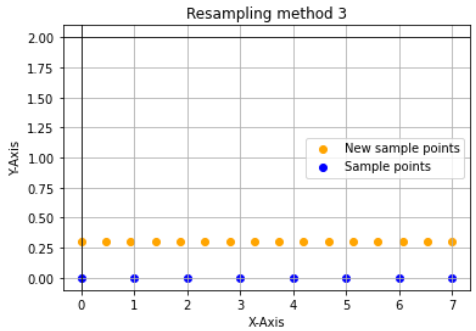
Τα artifacts εμφανίζονται διότι όπως βλέπουμε στο **Σχήμα 2.5**, τα οριακά σημεία A = (0, y_0) και B = (7, y_7) "αποκόπτονται" από τις πραγματικές τους θέσεις, διότι δεν υπάρχει κάποιο κυβικό πολυώνυμο που μπορεί να προσεγγίσει τις τιμές εκτός του διαστήματος όπου ορίστηκε το αρχικό δείγμα. Παρακάτω παρουσιάζουμε μια τρίτη μέθοδο η οποία θα λύσει όλα αυτά τα προβλήματα:

Μέθοδος 3

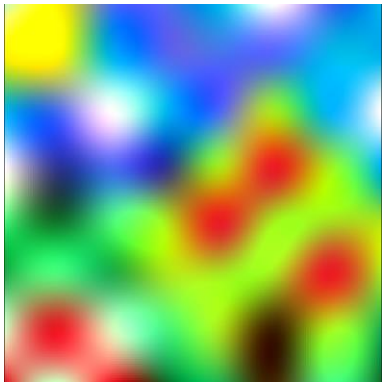
Έστω n το πλήθος των σημείων ενός νέου δείγματος A που θέλουμε να ορίσουμε. Το δείγμα μας ορίζεται από το σύνολο A = {x_0, x_1,..,x_n}. Το σύνολο του νέου δείγματος ορίζεται από τη σχέση:

$$x_{resampled} = n * \frac{sample\_size-1}{new\_sample\_size-1}, n \in [x_0, x_n]$$

Η συγκεκριμένη μέθοδος παράγει την βέλτιστη κατανομή των τιμών στο διάστημα [x_0,x_n], καθώς σε κάθε διάστημα υπάρχει και ένα κυβικό spline που μπορεί να προσεγγίσει οποιαδήποτε τιμή της συνάρτησης.



Σχήμα 2.6 Σύγκριση του αρχικού δείγματος με το νέο δείγμα με τη **Μέθοδο 3**.



Εικόνα 2.3 Εικόνα χωρίς artifacts περιμετρικά λόγω της **Μεθόδου 3** με μεγενθυτικό παράγοντα 20

2.5 Προγραμματιστική υλοποίηση

Η προγραμματιστική υλοποίηση της μεθόδου της δικυβικής τμηματικής παρεμβολής σε python έγινε βάσει της υποενότητας 2.4. Αρχικά προσεγγίσαμε το πρόβλημα στην απλούστερη μορφή του, έχοντας ώς δεδομένο μια μονοδιάστατη ασπρόμαυρη εικόνα.

Η μοναδική εξωτερική βιβλιοθήκη που χρησιμοποιήθηκε για την υλοποίηση της μεθόδου είναι η pillow της python, η οποία μας δίνει την δυνατότητα προβολής και λήψης των εικονοστοιχείων μιας εικόνας, καθώς και των χρωματικών καναλιών της με ευκολία.

Αρχικά ανοίγουμε την εικόνα και την μετατρεπουμε σε ασπρόμαυρη με τη μέθοδο Image.convert("L"), όπου Image το αντικείμενο της εικόνας. Στη συνέχεια χρησιμοποιήσαμε τη μέθοδο list(Image.getData()), η οποία επιστρέφει μια λίστα με κάθε εικονοστοιχείο της εικόνας. Αν επιλέξουμε την ανάγνωση μιας ασπρόμαυρης εικόνας επιστρέφει μια λίστα από τιμές που ανήκουν στο διάστημα [0,255], ενώ η ανάγνωση μιας εικόνας που έχει μετατραπεί σε RGB, επιστρέφει μια λίστα από τριπλέτες της μορφης (x_a,x_b,x_c), με τα στοιχεία της τριπλέτας να ανήκουν στο διάστημα [0,255].

Στη συνέχεια ξεκινήσαμε με την αρχικοποίηση των τριδιαγώνιων πινάκων για τον υπολογισμό του M_i με τον αλγόριθμο του Thomas. Για τη βέλτιστη διαχείριση μνήμης, η αναπαράσταση του συντελεστή πίνακα γίνεται με διανύσματα αντί για έναν $(n \times n)$ πίνακα. Έχοντας υπολογίσει λοιπόν την δεύτερη παράγωγο, υπολογίζουμε τα πολυώνυμα παρεμβολής σε κάθε διάστημα $[x_i, x_{i+1}]$ βάσει των σχέσεων που ορίσαμε παραπάνω. Εφαρμόζουμε την μέθοδο αναδειγματοληψίας που ορίσαμε στο 2.4, και υπολογίζουμε τις νέες τιμές $y_i : s_i(x_i) = f(x_i)$, $x_i \in [x_0,x_n]$.

Έχοντας υπολογίσει τις τιμές του πίνακα y, χρησιμοποιούμε τη μέθοδο Image.new(), ώστε να δημιουργήσουμε μια καινούργια εικόνα, και τη μέθοδο Image.putData(), όπου με τη μέθοδο αυτή περνάμε τα δεδομένα του πίνακα y στην νέα εικόνα που ορίσαμε.

Η μετατροπή της μονοδιάστατης μεθόδου σε δισδιάστατη απαιτεί τη μέθοδο που ορίσαμε παραπάνω να εφαρμοστεί σε κάθε γραμμή και κάθε στήλη του πίνακα. Το σημείο που χρήζει προσοχή είναι μετά τη λήξη της παρεμβολής των στηλών, πρέπει να αναδιατάξουμε τα στοιχεία του πίνακα y με τέτοιο τρόπο ώστε το πρώτο στοιχείο κάθε στήλης να μπει στην πρώτη γραμμή, το δεύτερο στοιχείο κάθε στήλης να μπει στη δεύτερη γραμμή κλπ. Εφόσον αυτό υλοποιηθεί, μένει το πέρασμα των δεδομένων σε μια νέα εικόνα και η προβολή της εικόνας που έχει υποστεί μεγέθυνση.

Τέλος με σκοπό η μέθοδος να μην έχει εφαρμογή μόνο σε ασπρόμαυρες εικόνες, δημιουργήσαμε μια μέθοδο extract_colour_channel(), η οποία εξάγει το κάθε χρωματικό κανάλι, (κόκκινο, πράσινο, μπλε) και πραγματοποιεί δικυβική τμηματική παρεμβολή στις τιμές του. Έτσι αυτό απαιτούσε την κατασκευή μιας άλλης συνάρτησης createNewImage(), η οποία ενώνει τις τιμές που προκύπτουν από τις παρεμβολές κάθε χρωματικού καναλιού και παράγει το τελικό αποτέλεσμα.

Βιβλιογραφία

1. Γιαννουτάκης, Κ. (2024). Εφαρμοσμένη Αριθμητική Ανάλυση [Γραμμικά Συστήματα, Πολυωνυμική Παρεμβολή] [Διαφάνειες PowerPoint]. Πανεπιστήμιο Μακεδονίας.
2. Μισυρλής. Ν. (2022). Αριθμητική Ανάλυση, μια αλγοριθμική προσέγγιση. Εκδόσεις Τσότρας.
3. Gao, S., & Gruev, V. (2011). Bilinear and bicubic interpolation methods for division of focal plane polarimeters.

Ιστοσελίδες

1. https://dsp.stackexchange.com/questions/20368/what-is-the-difference-between-cubic-interpolation-and-cubic-spline-interpolat [Προβλήθηκε 24/12/2024]
2. https://www.geeksforgeeks.org/python-opencv-bicubic-interpolation-for-resizing-image/ [Προβλήθηκε 24/12/2024]
3. https://www.geeksforgeeks.org/cubic-spline-interpolation/ [Προβλήθηκε 24/12/2024]
4. https://www.youtube.com/watch?v=syH8ASkotFg&t=274s [Προβλήθηκε 25/12/2024]
5. https://www.youtube.com/watch?v=pBtqaK0PzrA&t=215s [Προβλήθηκε 25/12/2024]