Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής

Αναγνώριση τυπωμένων μουσικών συμβόλων και μετατροπή τους σε μουσικό περιεχόμενο

Διπλωματική Εργασία

του

Διονύση Ρεβήσιου

Επιβλέπων: Γεώργιος Καμπουράκης

Αναπληρωτής Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ..... Μαρτίου 2013.

……..	……..	……..
Γεώργιος Καμπουράκης	**Ελευθέριος Καγιάφας**	**Βασίλειος Λούμος**
Αναπληρωτής Καθηγητής ΕΜΠ	Ομότιμος Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ

Αθήνα, Μάρτιος 2015

.....

Διονύσης Ρεβήσιος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

το παρόν εκπόνημα αφιερώνεται
στους γονείς μου Κώστα και Δήμητρα
και στην αδερφή μου Μαρία
ως ελάχιστη ανταπόδοση της στήριξης
και της έμπνευσης
που μου έχουν προσφέρει μέχρι σήμερα

επίσης αφιερώνεται
στους φίλους και τις φίλες μου,
που ήταν η καλύτερη συντροφιά
καθώς βαδίζαμε δίπλα
αυτούς τους ενδιαφέροντες καιρούς

Περίληψη

Η οπτική αναγνώριση μουσικής είναι ο κλάδος που ασχολείται με την εξαγωγή της μουσικής πληροφορίες που υπάρχει σε ένα έγγραφο μουσικής. Εντάσσεται στην κατηγορία «Ανάλυση δομημένων εγγράφων» και έχει αναπτυχθεί κυρίως στον χώρο της δυτικής μουσικής αλλά και σε άλλες μορφές μουσικής γραφής. Στόχος της οπτικής αναγνώρισης μουσικής είναι η παραγωγή ψηφιακών μορφών που αντιπροσωπεύουν μουσικές ακολουθίες όπως είναι το πρωτόκολλο MIDI και το MusicXML. Στην ιστορία εξέλιξης του αντικειμένου έχει διατυπωθεί μεγάλος αριθμός τεχνικών αναγνώρισης, περισσότερες εκ των οποίων εστιάζουν τη διαδικασία γύρω από την επιτυχή κατάτμηση της παρτιτούρας στα επιμέρους σύμβολα. Τα επιστημονικά εργαλεία που χρησιμοποιούνται για την υλοποίηση αυτών των τεχνικών υπάγονται στο πεδίο της «Ανάλυσης Εικόνας» και εκτείνονται από βασικές δομικές και μορφολογικές διαδικασίες μέχρι και τεχνικές αντιστοίχισης προτύπου και εξαγωγής χαρακτηριστικών γνωρισμάτων. Αν και το πρόβλημα της οπτικής αναγνώρισης μουσικής έχει τεθεί εδώ και 50 χρόνια, δεν έχει καταφέρει να επιλυθεί με τέτοια ακρίβεια ώστε να αποτελέσει αξιόπιστο εργαλείο στη διάθεση του μουσικού.

Στην παρούσα διπλωματική εργασία κατατίθενται και αποδομούνται το σύνολο των βασικών όψεων και προβλημάτων της οπτικής αναγνώρισης μουσικής. Μελετάται η συνεισφορά προτάσεων και τεχνικών που κατά την άποψή μας συνέβαλλαν αποφασιστικά στην επίλυση των προβλημάτων που ανακύπτουν και γίνεται επιλογή μεθόδων που συναρθρώνουν ένα στιβαρό, μεγάλης ακρίβειας σύστημα οπτικής αναγνώρισης.

Επιπλέον εισάγουμε δύο μεθόδους αποτελεσματικού εντοπισμού του πενταγράμμου και σταδιακής κατάτμησης και ταξινόμησης της εικόνας. Οι δύο αυτές μέθοδοι αποτελούν και τα βασικά συστατικά αποτελεσματικότητας του συστήματος αφού επιτρέπουν την εξαγωγή θετικών αποτελεσμάτων σε μεγάλο εύρος δειγμάτων εισόδου. Στην εργασία περιλαμβάνονται απεικονίσεις των αποτελεσμάτων κάθε τμήματος του συστήματος OMR, τα πειραματικά αποτελέσματα τεσσάρων σκαναρισμένων παρτιτούρων με σχολιασμό των αποτελεσμάτων τους καθώς και ο πλήρης κώδικας των ενοτήτων εντοπισμού του πενταγράμμου, των νοτών, της κατάτμησης και της ταξινόμησης στο Παράρτημα. Ελπίζουμε με αυτόν τον τρόπο να συμβάλλουμε στο αναπτυσσόμενο πεδίο της χρήσης των υπολογιστών με σκοπό την ανάκτηση, διακίνηση και αναπαράσταση μουσικής, και όχι μόνο, πληροφορίας.

Λέξεις κλειδιά: Οπτική Αναγνώριση Μουσικής, Οπτική Αναγνώριση Παρτιτούρας, Εντοπισμός Πενταγράμμου, Κατάτμηση τυπωμένης μουσικής, Ταξινόμηση μουσικών συμβόλων

Abstract

Optical music recognition is the field that deals with the extraction of music information that is present in a musical document. It falls in the domain of "Structured Document Analysis" and has been developed mainly in the area of western music but also in other forms of musical notation. The main goal of OMR is the production of digital forms that represent musical sequences such as the MIDI protocol and MusicXML. In the field's history there has been a large number of recognition techniques, most of which rely on the successful segmentation of the score to the individual symbols. The scientific tools used to implement these techniques are mostly part of "Image Analysis" domain and range from basic structural and morphological processes to pattern recognition and feature extraction techniques. Although the problem of optical music recognition has been posed for 50 years, still there isn't a precise solution to provide musicians with a reliable tool.

This diploma thesis deposits and decomposes all basic aspects and problems of optical music recognition. It studies the contribution of proposals and techniques that, in our opinion, decisively contributed to solving the problems encountered and selects methods that form a robust, highly accurate OMR system.

Furthermore, we introduce two methods of effective staff detection and progressive segmentation and classification of the image. These two methods are the key components of our system's efficiency allowing the production of positive results over a wide range of input samples. We have included illustrations of the results of each section of the OMR system, the experimental results of four scanned scores with commentary of their results and the full code of the modules staff detection, note detection, segmentation and classification in the Appendix. We hope in this way to contribute to the growing field of using computers for retrieval, distribution and representation of musical, but not only, information.

Keywords: Optical Music Recognition, OMR, Optical Sheet Music Recognition, Staff Detection, Sheet Music Segmentation, Music Symbol Classification

Ευχαριστίες

Με αφορμή την ολοκλήρωση της διπλωματικής εργασίας θα ήθελα να ευχαριστήσω όλους εκείνους που συνέβαλλαν με τον τρόπο τους κατά τη συγγραφής της αλλά και καθ' όλης της διάρκειας ολοκλήρωσης των προπτυχιακών σπουδών μου.

Αρχικά θα ήθελα να ευχαριστήσω τον κ. Γεώργιο Καμπουράκη, επιβλέποντα της εργασίας μου, όχι μόνο για τις συμβουλές και την εύστοχη καθοδήγησή του αλλά και για τα ερεθίσματα και τη δυνατότητα που δίνει σε εμάς τους φοιτητές να συνδυάσουμε τα τεχνικά εφόδια που αποκτούμε μέσα στο πανεπιστήμιο με το χώρο της τέχνης, της αισθητικής και στη συγκεκριμένη περίπτωση με τη μουσική.

Επιπλέον, για την ολοκλήρωση της διπλωματικής οφείλω πολλά στη διαρκή και κοντινή συνεργασία που είχα με τον υπ. διδάκτορα Κωνσταντίνο Μπακογιάννη τον οποίο ευχαριστώ για τη σημαντική συνεισφορά του, την τεχνική και ψυχολογική υποστήριξη και επίσης την προθυμία του να βοηθήσει κάθε στιγμή.

Τέλος, ευχαριστώ το θείο μου Θωμά Ρεβήσιο για την εκτύπωση και το δέσιμο του παρόντος βιβλίου αλλά κυρίως γιατί ίσως ο ίδιος υπήρξε το πρώτο ερέθισμά μου για να εισέλθω στον χώρο της ψηφιακής τέχνης.

Περιεχόμενα

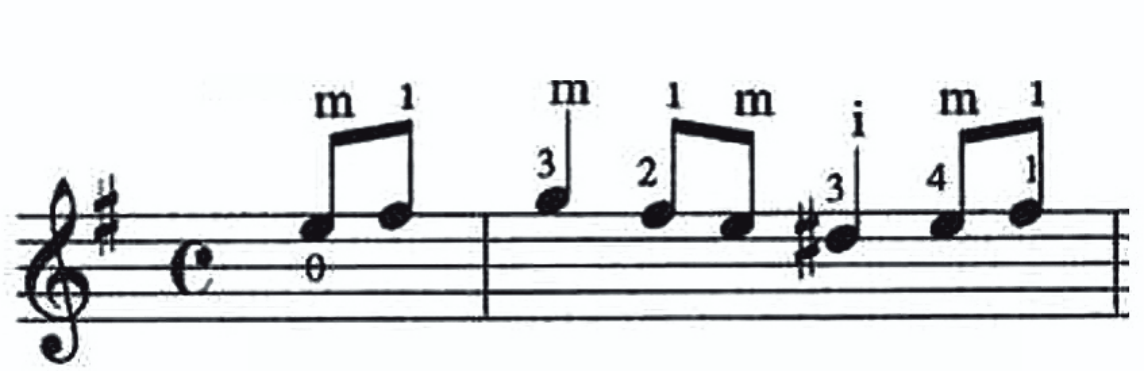
1

Εισαγωγή

1.1. Ορισμός

Ως οπτική αναγνώριση της μουσικής (Optical Music Recognition [OMR]) περιγράφουμε τη διαδικασία εξαγωγής της μουσικής πληροφορίας που περιγράφεται σε μια σελίδα παρτιτούρας. Συχνά αποκαλείται και μουσικό OCR (music OCR) μιας και συγγενεύει με τη διαδικασία οπτικής αναγνώρισης χαρακτήρων. Ωστόσο περιλαμβάνει δυο βασικές διαφορές που την καθιστούν αρκετά πιο δύσκολη:

Αρχικά το βασικό ζήτημα της υπέρθεσης (superimposition) (1.1.1) συμβόλων πάνω σε άλλα. Αποτελεί βάση της μουσικής γραφής, ο ορισμός συμβόλων σύμφωνα με τη σχετική τους θέση στο πεντάγραμμο. Συνεπώς τα σύμβολα δεν συναντώνται όπως σε ένα κείμενο σαν αυτοτελείς χαρακτήρες αλλά σαν μια σειρά συνδεδεμένων στοιχείων μέσω των γραμμών του πενταγράμμου.



Εικόνα 1.1.1

Δεύτερη διαφορά είναι η απουσία πεπερασμένης γραμματικής μεταξύ των συμβόλων. Σε αντίθεση με τους αλγόριθμους OCR που είναι σε θέση μετά το πέρας της αναγνώρισης να πιστοποιήσουν τα αποτελέσματά τους με βάση το λεξικό, το συντακτικό και τη γραμματική της γλώσσας, στην περίπτωση της μουσικής, υπάρχουν μεν συγκεκριμένοι γραμματικοί κανόνες αλλά δεν υπάρχει πεπερασμένο λεξιλόγιο. Ένα κομμάτι μπορεί να αποτελείται από οποιαδήποτε ακολουθία νοτών χωρίς να παραβιάζει κανέναν κανόνα.

Συνεπώς του πρόβλημα του OMR αποκτά άλλες διαστάσεις σε σχέση με το OCR καθιστώντας το ένα αρκετά πιο σύνθετο πρόβλημα που όμως υπόκειται και αυτό σε συγκεκριμένους κανόνες κυρίως ως προς τη δομή. Η ανάλυση μιας παρτιτούρας εντάσσεται στη ενότητα ανάλυσης δομημένων εγγράφων (Structured Document Analysis) διότι όλες οι παρτιτούρες, ανεξαρτήτως γραφής, είδους, οργάνου κλπ. ακολουθούν ένα παραπλήσιο τρόπο παράθεσης της μουσικής πληροφορίας. Αυτοί οι κανόνες δομής είναι και εκείνοι που μας οδηγούν στην αποδόμηση της «superimposed» εικόνας και η αναγωγή του προβλήματος σε κάποιας μορφής OCR.

1.2. Ιστορία του OMR

1.2.1 Ιστορική εξέλιξη της βιβλιογραφίας

Πρώτη αναφορά στην αυτοματοποιημένη αναγνώριση τυπωμένης μουσικής γίνεται από τον **Dennis Howard** στο **MIT** το **1966**. Λαμβάνοντας υπόψιν την ανάπτυξη της απόδοσης των υπολογιστών στα τότε χρονικά δεδομένα τίθεται το ερώτημα της δυνατότητας αυτόματης εκτέλεσης κομματιών που δεν είχαν ηχογραφηθεί μέχρι τότε. Αντίστοιχες παρατηρήσεις καταγράφει και ο **David Stewart** τέσσερα χρόνια αργότερα.

Η ανάπτυξη του OMR θα έπρεπε όμως να περιμένει κυρίως την περαιτέρω αύξηση της υπολογιστικής ισχύος και δευτερευόντως την ανάπτυξη τεχνικών οπτικής ανάλυσης μιας και οι τελευταίες είχαν ήδη καταγραφεί από τη δεκαετία του 60.

Έτσι, τα επόμενα δείγματα γραφής έρχονται μια δεκαετία αργότερα από τον K.N.Fischer στο πανεπιστήμιο του Tennessee (1978) μέχρι που η ενότητα OMR αποκτά μια πολύ εκτενή βιβλιογραφία από τα τέλη της δεκαετίας του 80. Καθοριστικές εργασίες εκείνης της περιόδου αποτελούν:

1. **Computer recognition of musical notation** Fujinaga I., B. Pennycook, και B. Alphonce (1989)
2. **Automatic recognition of printed music** Carter N. και R. Bacon, (1992)
3. **A critical survey of music image analysis** Blostein D. και H. S. Baird (1992)
4. **A complete optical music recognition system: Looking to the future** Bainbridge D. (1994)

Κριτήριο επιλογής αυτών των εργασιών είναι η αναφορά τους στις μετέπειτα εργασίες καθώς και η πληρότητα και η καινοτομία με την οποία, κατά την άποψή μας, προσέγγισαν τα νέα για την εποχή προβλήματα που ανέκυπταν γύρω από την οπτική αναγνώριση μουσικής.

1.2.2. Εφαρμογές OMR

Στις αρχές της δεκαετίας του 90 εμφανίστηκαν τα πρώτα εμπορικά προγράμματα OMR. Μεταξύ αυτών ξεχωρίζουν το Sharp Eye, το οποίο αργότερα θα ενσωματωθεί στο PhotoScore και θα συμπεριληφθεί στο Sibelius), το Capella-Scan, το OmeR και το SmartScore. Εξ'αυτών θα ξεχωρίσουν και θα «επιβιώσουν» το PhotoScore και το SmartScore. Το πρώτο λόγω του πολύ δυνατού συστήματος SharpEye και το δεύτερο κυρίως λόγω των μεγάλων δυνατοτήτων που προσέφερε και δευτερευόντως για το σύστημα OMR που υλοποιούσε. Επιπλέον η εφαρμογή Forte ενσωμάτωσε το επιτυχημένο SharpEye παράγοντας αντίστοιχα καλά αποτελέσματα.

Η ανάπτυξη του ανοιχτού λογισμικού τα τελευταία χρόνια επηρέασε και τον κλάδο OMR. Οι δύο πιο γνωστές ανοιχτές εφαρμογές OMR είναι το Audiveris και το OpenOMR. Και τα δύο αναπτύχθηκαν σε περιβάλλον Java και το SDK τους είναι ανοιχτό στους χρήστες. Το OpenOMR σταμάτησε το 2010 ενώ το Audiveris συνεχίζει μέχρι σήμερα (τελευταία έκδοση Νοέμβρης 2013). Αν και οι δυνατότητες των δυο προγραμμάτων είναι περιορισμένες το Audiveris εισάγει την καινοτομία της εκπαίδευσης του Νευρωνικού Δικτύου από τον τελικό χρήστη δίνοντας

δυνατότητα στην εφαρμογή να επεκτείνεται διαρκώς.

Τέλος, από τον Fujinaga I. αναπτύχθηκε το Gamera Framework. Μια βιβλιοθήκη επεξεργασίας εικόνας το οποία δημιουργήθηκε με αφορμή την εργασία του με τίτλο "Staff Detection and Removal". Το συγκεκριμένο framework περιλαμβάνει συναρτήσεις εντοπισμού και αφαίρεσης πενταγράμμου.

1.2.3. Εξέλιξη των τεχνικών

Οι τεχνικές οπτικής αναγνώρισης μουσικής αναπτύχθηκαν παράλληλα με την ανάπτυξη της υπολογιστικής ισχύος.

Αρχικά υλοποιούνταν βασικές μορφολογικές τεχνικές και τεχνικές δομικής ανάλυσης της παρτιτούρας (vertical – horizontal runs, projections). Ο χρόνος εκτέλεσης ήταν ιδιαίτερα μεγάλος μιας και οι εργασίες εστίαζαν στη μαθηματική διάσταση του προβλήματος και δευτερευόντως στην αποδοτικότητα της λύσης.

Στη συνέχεια, τα νέα μεγέθη επεξεργαστών και μνήμης επέτρεψαν μεθόδους όπως template matching, skeletonization και χρήση νευρωνικών δικτύων καθώς επίσης και τη δυνατότητα fuzzy modeling, δηλαδή την προσεγγιστική κατασκευή μιας εικόνας σύμφωνα με τις πληροφορίες που έχουμε συλλέξει και τη σύγκρισή της με την αρχική.

Συνοπτικά, στην εξέλιξη του χρόνου συντελέστηκε μια μετάβαση από top-down σχεδιασμό του συστήματος σε bottom-up. Ενώ αρχικά έπρεπε να αποδομείται βήμα βήμα η εικόνα για την εξαγωγή πληροφορίας τώρα πια υπάρχει δυνατότητα να συνδυάζονται τα επιμέρους ευρήματα για την πιστοποίηση ευρημάτων προηγούμενων σταδίων.

Ωστόσο τα αρχικά ζητήματα που τέθηκαν από τις πρώτες εργασίες παραμένουν ακόμα και σήμερα. Η επιτυχής αφαίρεση του πενταγράμμου είναι μια διαδικασία που δεν έχει επιλυθεί μέχρι σήμερα με πλήρη τρόπο παρά το πλήθος των τεχνικών που αναπτύχθηκαν από την πρώτη διατύπωση του προβλήματος.

Στην εργασία μας περιγράφουμε και υλοποιούμε πολλές από τις τεχνικές που έχουν προταθεί καταγράφοντας τις δυνατότητες και τις αδυναμίες τους. Βέβαια το κριτήριο είναι υποκειμενικό μιας και όλες οι τεχνικές αν αναπτυχθούν με εκτενή τρόπο μπορούν να παράγουν ικανοποιητικά αποτελέσματα. Συνεπώς, κριτήριό μας είναι και η δυνατότητα ενσωμάτωσης της κάθε τεχνικής σε κάθε στάδιο του συστήματος μας αλλά και η χρονική απόδοση αυτών των τεχνικών.

1.3. Αφορμή και στόχοι αυτής της διπλωματικής εργασίας

Η αφορμή για την επιλογή του συγκεκριμένου θέματος έγκειται σε δύο λόγους, ο ένας είναι μουσικού ενδιαφέροντος και ο άλλος τεχνικού.

Από μουσικής άποψης, μέσω της οπτικής αναγνώρισης μουσικής γεννιέται το ενδιαφέρον της μετάφρασης ενός τεράστιου αρχείου μουσικής που έχει καταγραφεί στην ιστορία της μουσικής γραφής σε ήχο. Υπάρχει δηλαδή η δυνατότητα κομμάτια αιώνων πριν να λάβουν σήμερα την υπόσταση για την οποία προορίζονταν από τους συγγραφείς τους. Για να μην παρερμηνευτούμε, προφανώς ο τελικός προορισμός κάθε μουσικού έργου είναι η εκτέλεση του από τον άνθρωπο που

σημαίνει πολλά περισσότερα από τον καλύτερο αλγόριθμο που μπορεί να υπάρξει. Ωστόσο ακόμα και αυτή, η αυτοματοποιημένη και κβαντισμένη διαδικασία τελικά μπορεί να παράξει ένα ηχητικό αποτέλεσμα που θα προϊδεάσει και θα φέρει τον χρήστη πιο κοντά στην παρτιτούρα που κρατά στα χέρια του και ενδεχομένως θα προσπαθήσει να εκτελέσει.

Από μια πιο τεχνική σκοπιά, η οπτική αναγνώριση μουσικής και κυρίως η διαδικασία υλοποίησης ενός πλήρους συστήματος θέτει ένα θεμελιώδες πρόβλημα στο σχεδιαστή του. Καλούμαστε να «εξηγήσουμε» στον υπολογιστή γιατί αυτό που βλέπουμε εμείς έχει συγκεκριμένο νόημα. Γιατί ένας πίνακας τιμών 0 και 1 συμπεριλαμβάνει συγκεκριμένη και μοναδική μουσική πληροφορία. Το θεμελιώδες αυτό πρόβλημα είναι συστατικό όλων των προβλημάτων οπτικής ανάλυσης και ενώ σε άλλες περιπτώσεις η αυτοπεποίθηση του αποτελέσματος έγκειται στον υπολογιστή, στο OMR έγκειται στον χρήστη. Για παράδειγμα, σε ένα φθαρμένο κείμενο που δεν είναι ευδιάκριτο στον χρήστη καλείται ο αλγόριθμος OCR να βρει τη λύση και πολλές φορές μπορεί να το καταφέρει. Επίσης σε μια ιατρική εικόνα για τη μέτρηση αριθμού κυττάρων ο υπολογιστής έχει τον πρώτο λόγο για την φερεγγυότητα του αποτελέσματος. Στην περίπτωση του OMR όμως, θεωρείται ότι ο σχεδιαστής μπορεί να διαβάσει την παρτιτούρα και άρα κατέχει όλη την απαραίτητη πληροφορία την οποία πρέπει να υπαγορεύσει στον υπολογιστή.

Συνοπτικά, από τεχνικής πλευράς, η μεγαλύτερη πρόκληση κατά την υλοποίηση του συστήματος είναι η αποδόμηση όλων εκείνων των κριτηρίων με τα οποία εμείς αξιολογούμε μια νότα ή ένα σύμβολο σε ένα σύνολο δυαδικών ερωτημάτων στον υπολογιστή. Όσο πιο «σωστά» προσδιοριστούν αυτά τα δυαδικά ερωτήματα τόσο πιο πολύ προσεγγίζουν τελικά και τα κριτήρια που κάποιος εμπειρικά κατανοεί τη μουσική γραφή.

Στόχος αυτής της εργασίας είναι η ανάπτυξη μιας πλήρους εφαρμογής OMR με αποτελέσματα αντίστοιχα του πιο επιτυχημένου συστήματος μέχρι στιγμής του SharpEye. Η δυνατότητες μιας τέτοιας εφαρμογής μπορούν να χρησιμοποιηθούν σε μια σειρά από πεδία:

Ψηφιοποίηση αρχείου: Υπάρχει η δυνατότητα αποθήκευσης, ταξινόμησης και αποκατάστασης μουσικού αρχείου. Η μετατροπή σε MusicXML δίνει τη δυνατότητα αποθήκευσης παρτιτούρων καταλαμβάνοντας ελάχιστο χώρο, εύκολη διακίνηση και διαδικτυακή προσβασιμότητα. Δεδομένου επιπλέον ότι τα βιβλία παρτιτούρων είναι ακριβά δίνεται η δυνατότητα δημιουργίας κοινοτήτων που θα μοιράζονται σε ελάχιστο χρόνο μεγάλο όγκο μουσικών κομματιών.

Ανάκτηση μουσικής πληροφορίας: Σε κάθε κομμάτι υπάρχει πια η δυνατότητα εντοπισμού κλίμακας, τέμπο, θέματος δημιουργώντας μια νέα παράμετρο ταξινόμησης του αρχείου αλλά και διευκολύνοντας τη διαδικασία ανάλυσης του τρόπου γραφής μουσικής από τον συνθέτη μέσω αναζήτησης ομοιοτήτων, διαφορών, επαναλήψεων κλπ.

Εκπαίδευση: Η χρήση μιας αναγνωρισμένης παρτιτούρας μπορεί να γίνει κατά την εκμάθηση μουσικής. Βέβαια αυτό απαιτεί μια αρκετά προσεκτική διαδικασία διότι κατά την εκμάθηση δεν είναι ζητούμενο πχ το πρόγραμμα να δείχνει στο μαθητή το όνομα της κάθε νότας αλλά ζητούμενη είναι η μνημόνευση των διαστημάτων προκειμένου η ερμηνεία σταδιακά να γίνεται φυσικά. Εκεί που μπορεί όμως να βοηθήσει μια εφαρμογή OMR έναν μαθητή είναι στην αναπαραγωγή του κομματιού ή πχ δύσκολων ρυθμικά τμημάτων του προκειμένου να τον προϊδεάσει και να τον διευκολύνει κατά την εκμάθηση. Παράλληλα η επέκταση της εφαρμογής σε χειρόγραφες

παρτιτούρες θα μπορούσε να χρησιμοποιηθεί στη διόρθωση ασκήσεων αρμονίας δεδομένου ότι έχουν παρασχεθεί στο πρόγραμμα οι κατάλληλοι κανόνες αρμονίας.

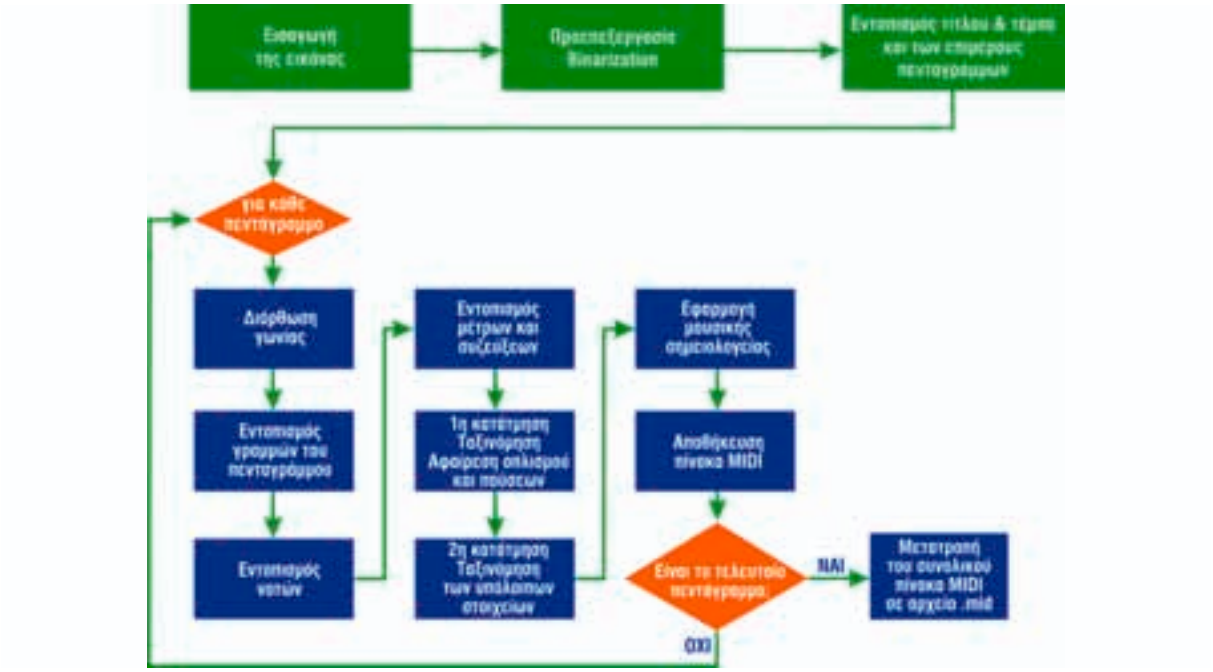
Αναπαραγωγή: Η χρήση αρχείων MIDI σε προγράμματα δημιουργίας μουσικής ήταν ανέκαθεν διευρυμένη. Με κατάλληλη παραμετροποίηση και με τη μεγάλη πληθώρα βιβλιοθηκών ρεαλιστικών ηχητικών δειγμάτων, μπορεί κάποιος να προσθέσει στο κομμάτι του αυτούσια, ή αλλοιωμένα τμήματα κλασσικών και όχι μόνο κομματιών. Επιπλέον μια πλήρως αναγνωρισμένη παρτιτούρα μαζί με παραμετροποιήσεις «εξανθρωπισμού» (humanization, δηλαδή εισαγωγή τυχαίων σφαλμάτων-αποκλίσεων χρόνου και δυναμικής στην εκτέλεση ώστε να προσιδιάζει στην ανθρώπινη εκτέλεση) και με καλή βιβλιοθήκη ήχων μπορεί να παράγει ένα αρκετά εντυπωσιακό αποτέλεσμα.

Δε θα λέγαμε όμως ότι στην ψηφιακή εποχή όλα προορίζονται να υπάρχουν στην ψηφιακή τους μορφή. Όπως η ψηφιακή μουσική, που υποκατέστησε τους δίσκους και έπειτα τα CD, δεν οδήγησε απαραίτητα στη «ζητούμενη» σχέση του ακροατή με το μουσικό κομμάτι έτσι και στη δική μας περίπτωση δεν πιστεύουμε ότι αποτελεί στόχο η αντικατάσταση των σελίδων στα αναλόγια με tablets.

Οι ίδιες οι παρτιτούρες γράφτηκαν από ανθρώπους και συμπεριλαμβάνουν τον ανθρώπινο παράγοντα στη σημειογραφία. Περιγραφές όπως Affettuoso (με συναίσθημα), Appassionato (με πάθος), Animato (ζωηρά) δεν μπορούν και δεν θα μπορέσουν ποτέ να ερμηνευθούν από τον υπολογιστή και αυτό γιατί η αξία τους ίσως βρίσκεται και στην υφή, στο χρώμα του χαρτιού, στο μελάνι και στη νότα που δεν τυπώθηκε σωστά.

2
Περιγραφή του συστήματος OMR

Το σύστημα OMR που υλοποιήσαμε ακολουθεί τα βήματα που φαίνονται στην εικόνα 2.1.1. Αρχικά επεξεργαζόμαστε τη σελίδα στο σύνολό της και εφαρμόζουμε τα απαραίτητα στάδια προεπεξεργασίας προκειμένου η εικόνα να δυαδικοποιηθεί αποτελεσματικά. Επιπλέον εξάγουμε και επιπλέον πληροφορίες από τη σελίδα όπως είναι ο τίτλος και το τέμπο.



Εικόνα 2.1.1

Στη συνέχεια χωρίζουμε τη σελίδα στα επιμέρους τμήματα που αποτελείται. Εννοούμε τα συνδεδεμένα πεντάγραμμα τα οποία εκτελούνται ταυτόχρονα. Τα ταξινομούμε σύμφωνα με την κάθετη θέση τους και έπειτα τα επεξεργαζόμαστε ένα ένα ξεχωριστά.

Για κάθε ένα τμήμα της παρτιτούρας εφαρμόζουμε τις διαδικασίες που περιγράφουμε στο βρόγχο του γραφήματος.

Οι διαδικασίες αυτές περιγράφονται αναλυτικά στην εργασία καθώς και τα αποτελέσματα κάθε μιας από αυτές. Στην εργασία παραθέτουμε πολλές φορές τα αποτελέσματα σε όλη τη σελίδα, ωστόσο αυτές οι διαδικασίες εφαρμόζονται ξεχωριστά σε κάθε τμήμα της σελίδας.

Το διάγραμμα διαδικασιών της εικόνας επαναλαμβάνεται σύμφωνα με τον αριθμό των σελίδων που επιλέγουμε κάθε φορά για αναγνώριση με ίδιο τρόπο με μόνη διαφορά, ότι στην πρώτη σελίδα εντοπίζεται το τέμπο και ο τίτλος.

Επιπλέον κατά την εκτέλεση της πρώτης σελίδας δημιουργείται ο πίνακας MIDI ο οποίος

επεκτείνεται κάθε φορά με τα στοιχεία που προστίθενται ώσπου μετά το πέρας της τελευταίας σελίδας μετατρέπεται στο αντίστοιχο ηχητικό αρχείο (.mid).

Το σύστημα που υλοποιήσαμε παράγει πολύ καλά αποτελέσματα, τα οποία παρατίθενται στο αντίστοιχο κεφάλαιο της εργασίας. Ο χρόνος εκτέλεσης κυμαίνεται μεταξύ 15s για σελίδες ανάλυσης 150dpi έως και 40s για σελίδες ανάλυσης 300dpi.

Η τελική παραγωγή του αρχείου midi γίνεται με χρήση της συνάρτησης matrix2midi του Ken Schutte (http://kenschutte.com/midi) η οποία μετατρέπει έναν πίνακα με πληροφορίες Pitch (τόνος), Velocity (δυναμική), Note_On, Note_Off (χρονική στιγμή που ξεκινά και σταματά μια νότα) στο αντίστοιχο αρχείο midi.

Στην εργασία που ακολουθεί περιγράφουμε και πολλές μεθόδους που δεν χρησιμοποιούμε διότι δοκιμάσαμε ένα μεγάλο εύρος τεχνικών που χρησιμοποιούνται οπότε καταθέτουμε και τις παρατηρήσεις μας πάνω σε αυτές. Επιπλέον περιγράφουμε και έναν αλγόριθμο για διόρθωση της καμπυλότητας της σελίδα (page dewarp) τον οποίο δεν χρησιμοποιούμε στο σύστημα αλλά τον δοκιμάσαμε σε παρτιτούρες φωνογραφημένες από κινητή συσκευή.

Τέλος να σημειώσουμε ότι οι αλγόριθμοι που χρησιμοποιήσαμε περιγράφονται στο αντίστοιχο κεφάλαιο αλλά προφανώς κατά τη δοκιμή του προγράμματος προστέθηκαν σε αυτούς επιπλέον κομμάτια για τη βελτίωση του αποτελέσματος, πολλά εκ των οποίων δεν αναφέρονται. Ο άξονας της λογικής των αλγορίθμων είναι αυτός ο οποίος περιγράφεται και τα αποτελέσματα είναι αυτά που παρατίθενται σε κάθε ενότητα του συστήματος μας.

Συγκεκριμένα:

Στο Κεφάλαιο 3 αναλύονται τρία στάδια προεπεξεργασίας της εικόνας: η δυαδικοποίηση, η διόρθωση του προσανατολισμού, και η διόρθωση της καμπύλωσης ενώ περιγράφονται και οι βασικές παράμετροι staff_thickness, staff_distance.

Στο Κεφάλαιο 4 παρουσιάζουμε και δοκιμάζουμε διάφορες τεχνικές για τον εντοπισμό του πενταγράμμου.

Στο Κεφάλαιο 5 εξετάζουμε μεθόδους κατάτμησης της εικόνας και τεχνικές που έχουν προταθεί αλλά παρουσιάζουμε και διαδικασίες χωρίς κατάτμηση.

Στο Κεφάλαιο 6 αναλύεται η διαδικασία εντοπισμού των νοτών μιας παρτιτούρας, αρχικά εκείνων αξίας τετάρτου η μικρότερης και στη συνέχεια των υπολοίπων.

Στο Κεφάλαιο 7 περιγράφεται συνοπτικά ο εντοπισμός συμπληρωματικών στοιχείων της παρτιτούρας όπως γραμμές μέτρου, συζεύξεις, τίτλος και τέμπο.

Στο Κεφάλαιο 8 περιγράφεται η μεθοδολογία κατάτμηση και ταξινόμηση που εφαρμόσαμε στο σύστημα καθώς και η χρήση νευρωνικού δικτύου για την ταξινόμηση.

Στο Κεφάλαιο 9 εφαρμόζονται οι κανόνες της μουσικής πάνω στα σύμβολα που έχουμε εντοπίσει στα προηγούμενα κεφάλαια.

Στο Κεφάλαιο 10 δοκιμάζουμε και καταγράφουμε τις επιδόσεις του συστήματος σε τέσσερις σελίδες παρτιτούρων.

Στο Κεφάλαιο 11 περιγράφουμε προοπτικές εξέλιξης του συστήματος που σχεδιάσαμε

Στο Παράρτημα καταθέτουμε πλήρως τον κώδικα των βασικών ενοτήτων staff detection, note detection, segmentation & classification.

3
Προεπεξεργασία της εικόνας

3.1 Binarization

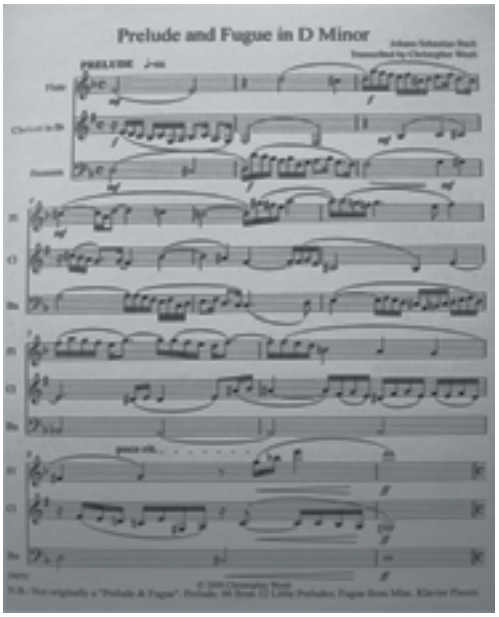
3.1.1. Εισαγωγή

Κάθε διαδικασία OMR αρχικά περνάει από το στάδιο δυαδικοποίησης (Binarization) όπου η εικόνα που έχουμε σαν είσοδο μετατρέπεται από την κλίμακα του γκρι (τιμές 0 έως 255) σε δυαδική (τιμές 0 ή 1).

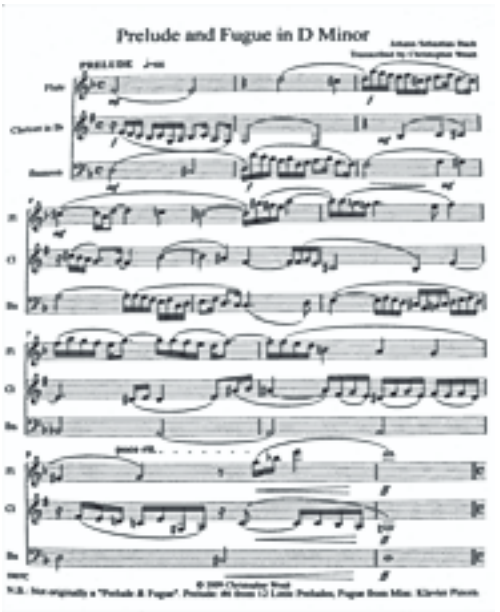
Η δυαδικοποίηση διαδραματίζει σημαντικό ρόλο στη μετέπειτα επεξεργασία της εικόνας, ειδικά αν πρόκειται για εικόνα «μικρής» ανάλυσης, τόσο στην αποφυγή αλλοίωσης της παρτιτούρας όσο και στην περαιτέρω διευκόλυνση των επόμενων σταδίων.

Προφανώς η μετάβαση από τον τρισδιάστατο χώρο της έγχρωμης εικόνας στην κλίμακα του γκρι (rgb σε grayscale) δεν απαιτεί ιδιαίτερη επεξεργασία μιας και δεν υπάρχει επιπλέον χρωματική πληροφορία πέρα από τις αποχρώσεις του γκρι.

Η μετάβαση όμως από grayscale σε binary δεν είναι μοναδική και υπάρχουν διαφορετικοί τρόποι με διαφορετικά αποτελέσματα και διαφορετική χρησιμότητα. Συγκεκριμένα θα μελετήσουμε την απλή χρήση κατωφλίου (threshold) και την χρήση προσαρμοστικού κατωφλίου (adaptive threshold). Πρώτα όμως θα ασχοληθούμε με μια προεπεξεργασία της εικόνας για την διόρθωση πιθανού ανομοιόμορφου φωτισμού στην εικόνα (non-uniform illumination). Το τελευταίο φαινόμενο είναι πολύ συχνό και προέρχεται είτε από την καμπύλωση μιας σελίδας όταν σκανάρεται ή από μια σελίδα που δεν έχει σκαναριστεί αλλά έχει φωτογραφηθεί.



Εικόνα 3.1.1



Εικόνα 3.1.2

3.1.2. Ανομοιόμορφος φωτισμός (Non-uniform Illumination)

Παρακάτω βλέπουμε μία πολύ συνήθη περίπτωση όπου η εικόνα έχει ανομοιόμορφο φωτισμό (3.1.1). Τα προβλήματα που μπορεί να προκύψουν φαίνονται στην παρακάτω εικόνα όπου έχει χρησιμοποιηθεί ένα κατώφλι για την δυαδικοποίηση της εικόνας (3.1.2).

Εικόνα 3.1.3

Συγκεκριμένα αν εστιάσουμε σε ένα κομμάτι της εικόνας (3.1.3) παρατηρούμε ότι η εικόνα αρχίζει και «σπάει» σε κάποια σημεία καθώς γίνεται λάθος αξιολόγηση των foreground και background pixels. Δηλαδή των στοιχείων που θα πάρουν την τιμή 1 ή 0 αντίστοιχα. Το συγκεκριμένο φαινόμενο συμβαίνει διότι ο διαφορετικός φωτισμός της εικόνας μπορεί να οδηγήσει μία περιοχή που είναι τυπωμένη (foreground) να έχει ίδια ή και λιγότερη ένταση (intensity) από ένα στοιχείο που είναι χαρτί (background). Το συγκεκριμένο σπασμένο πεντάγραμμο ή κλειδί του σολ αν μείνει έτσι μπορεί να οδηγήσει σε σφάλματα στα επόμενα βήματα.

Το πρόβλημα αυτό επιλύεται σε ένα βαθμό με τη χρήση προσαρμοστικού κατωφλίου (το οποίο θα εξηγηθεί παρακάτω) αλλά βοηθάει και είναι εφικτή η διόρθωση του ανομοιόμορφου φωτισμού. Η συγκεκριμένη διαδικασία χρησιμοποιείται σε πολλές εφαρμογές και προηγείται της δυαδικοποίησης.

Ακολουθούνε συνοπτικά δύο μέθοδοι διόρθωσης του φωτισμού μιας εικόνας. Και οι δύο βασίζονται στον υπολογισμό του background, δηλαδή το πώς θα ήταν φωτισμένη η εικόνα αν ήταν λευκό χαρτί.

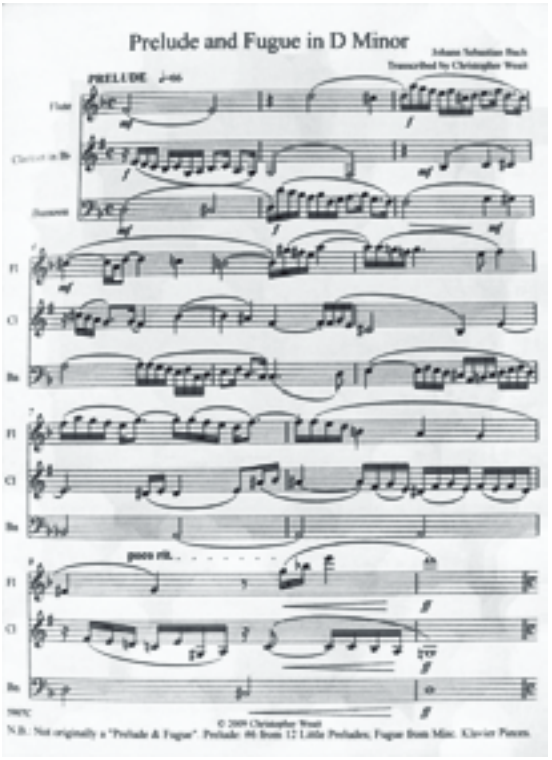
3.1.3. Διόρθωση ανομοιόμορφου φωτισμού

Η πρώτη μέθοδος υπολογίζει το background διεξάγοντας μορφολογικό άνοιγμα της εικόνας (morphological opening, θα ασχοληθούμε ειδικότερα με τις μορφολογικές διαδικασίες παρακάτω). Συγκεκριμένα αν «ανοίξουμε» την εικόνα χρησιμοποιώντας ως δομικό στοιχείο έναν δίσκο μεγάλης διαμέτρου (πχ (πλάτος της εικόνας)/20) παίρνουμε το παρακάτω αποτέλεσμα (3.1.4) το οποίο είναι μια προσέγγιση του φωτισμού της σελίδας χωρίς το foreground. Αν τώρα αφαιρέσουμε από την αρχική εικόνα τον φωτισμό παίρνουμε σαν αποτέλεσμα την αρχική εικόνα με διορθωμένο φωτισμό (3.1.5).

Συνοπτικά ο αλγόριθμος είναι:

```
Background = Imopen(I, strel('disk', width/20))
Icorrected = I – Background
```

Εικόνα 3.1.4



Εικόνα 3.1.5

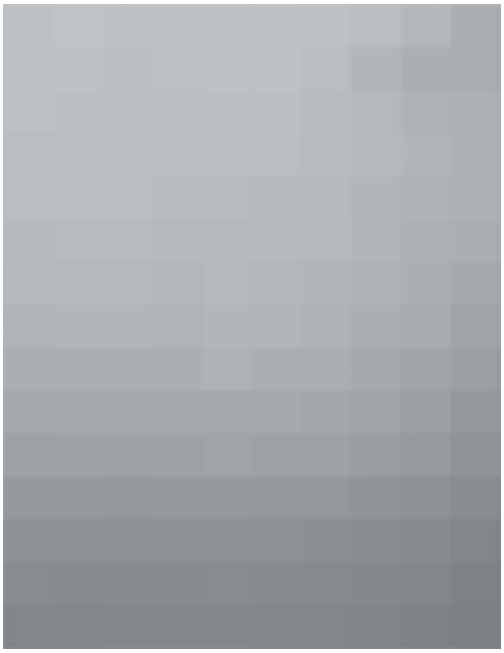
Η δεύτερη μέθοδος αρχικά διεξάγει μια «πρόχειρη» κατωφλίωση (thresholding) χρησιμοποιώντας ένα καθολικό κατώφλι (με τις μεθόδους που περιγράφονται παρακάτω). Έπειτα χωρίζει την εικόνα σε ένα πλέγμα παραθύρων (στο παράδειγμα μας χρησιμοποιούμε πλέγμα 15x10).

Για κάθε παράθυρο υπολογίζεται ο μέσος όρος των στοιχείων που δεν επιλέχθηκαν ως foreground από την πρώτη κατωφλίωση , δηλαδή υπολογίζεται το μέσο χρώμα (intensity) των στοιχείων που εκτιμούμε αρχικά ότι είναι χαρτί (background). Το αποτέλεσμα είναι το παρακάτω (3.1.6).

Εφαρμόζουμε στη συνέχεια ένα φίλτρο τύπου Gauss (3.1.7) για να εξομαλύνουμε την εικόνα μιας και οι μεταβολές στον φωτισμό είναι ομαλές και τέλος αφαιρούμε την εικόνα από την αρχική για να πάρουμε το τελικό αποτέλεσμα (3.1.8).

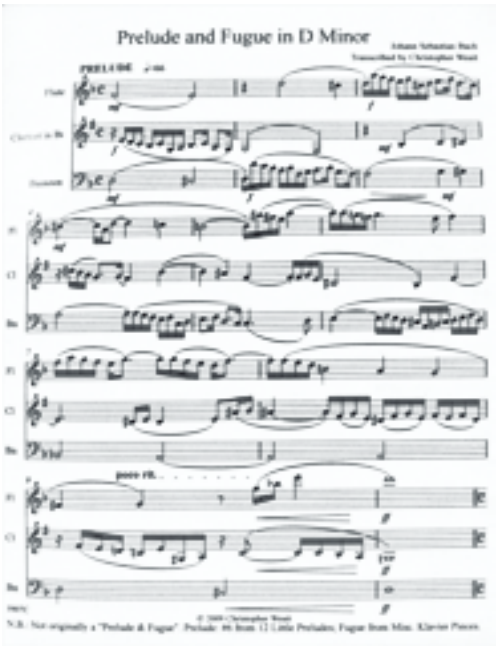
Συνοπτικά ο αλγόριθμος είναι:

```
Ib = bw_threshold(I)
για κάθε παράθυρο Iw του πλέγματος (1:h,1:v)
   Il_w = median(I(΄Ib))
τέλος
Illum_filtered = imfilter(Il,'gaussian',...)
Icorrected = I – Illum_filtered
```



Εικόνα 3.1.6

Εικόνα 3.1.7



Εικόνα 3.1.8

Και τα δύο αποτελέσματα, και ειδικά το δεύτερο, μας διευκολύνουν να προχωρήσουμε στο επόμενο στάδιο της δυαδικοποίησης έχοντας προετοιμάσει την αρχική εικόνα για αυτό.

3.1.4. Δυαδικοποίηση (Binarization)

Η δυαδικοποίηση είναι μια διαδικασία κατάτμησης (segmentation) καθώς επιμερίζει κάτι ενιαίο σε επιμέρους ομάδες σύμφωνα με ένα κριτήριο, στη συγκεκριμένη περίπτωση σε δύο ομάδες (foreground και background) και με κριτήριο το κατώφλι (threshold).

Οι μέθοδοι κατωφλίωσης χωρίζονται αρχικά σε δύο κατηγορίες

1. Στην καθολική κατωφλίωση (global thresholding)
2. Στην προσαρμοστική κατωφλίωση (adaptive thresholding)

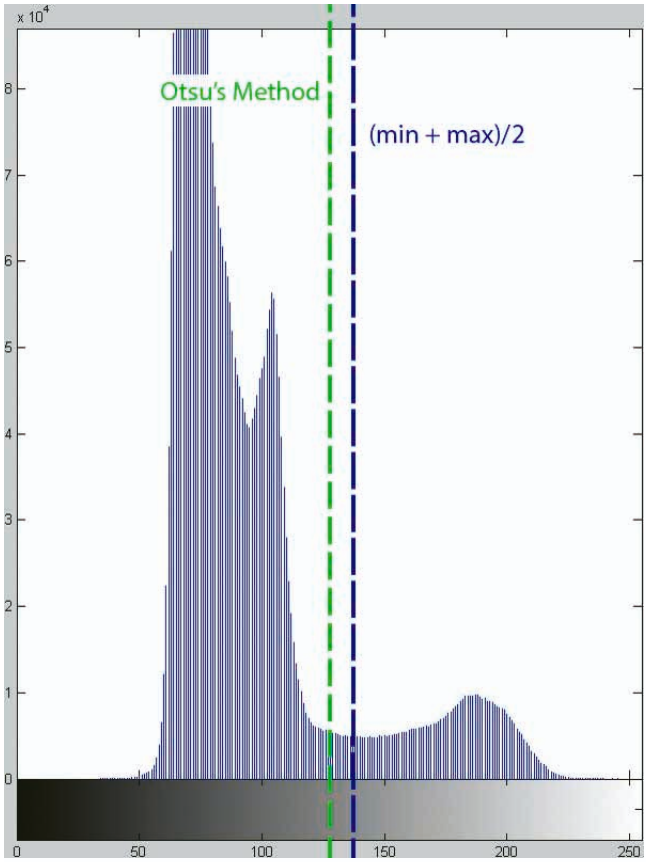
3.1.5. Καθολική κατωφλίωση (Global Thresholding)

Επιλέγεται ένα κατώφλι για όλα τα στοιχεία της εικόνας. Πιο συνήθης τρόπος είναι είτε

$$threshold = \frac{max + min}{2}$$

είτε η μέθοδος του Otsu (Nobuyuki Otsu) η οποία ανατρέχοντας στο ιστόγραμμα της εικόνας προσπαθεί να μειώσει τη διακύμανση μεταξύ των δύο κλάσεων υπολογίζοντας το βέλτιστο κατώφλι ώστε να υπάρχει ελάχιστο inter-class variance. Η συγκεκριμένη μέθοδος είναι ευρέως διαδεδομένη και χρησιμοποιείται και για περισσότερες από δύο κλάσεις.

Στην εικόνα (3.1.9) βλέπουμε τα αποτελέσματα στο ιστόγραμμα της παρτιτούρας των δύο μεθόδων όπου οι δύο γραμμές δείχνουν το global threshold που εντόπισαν οι δύο μέθοδοι για την εικόνα.



Εικόνα 3.1.9

3.1.6. Προσαρμοστική κατωφλίωση (Adaptive Thresholding)

Σε αντίθεση με την πρώτη μέθοδο, τώρα υπολογίζεται για κάθε pixel ένα συγκεκριμένο threshold σύμφωνα με στοιχεία του περιβάλλοντος του. Προφανώς η τοπική κατωφλίωση είναι πιο δαπανηρή σε χρόνο και πόρους από την καθολική αλλά σε συγκεκριμένες περιπτώσεις είναι απολύτως απαραίτητη.

Chow and Kaneko

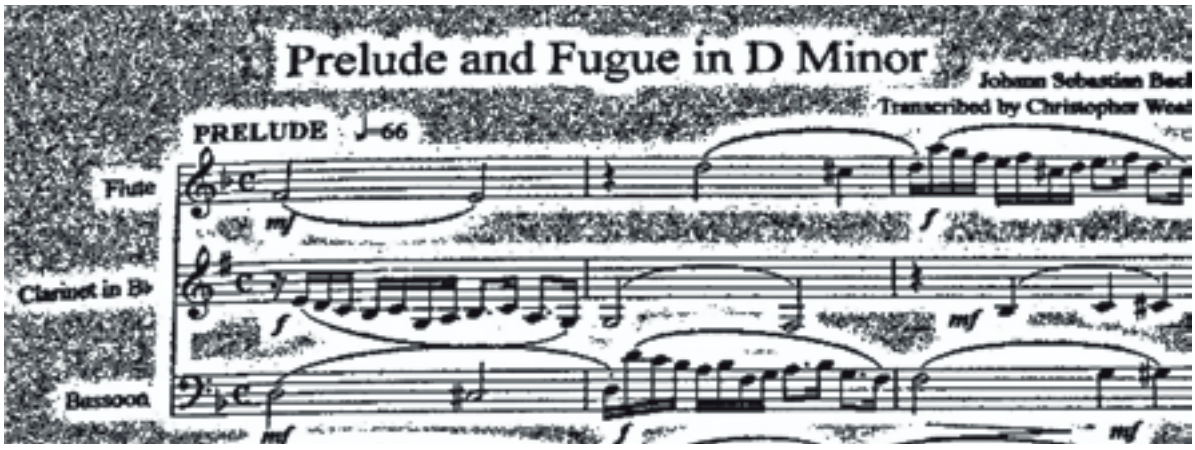
Υπάρχουν δύο βασικοί τρόποι για τον υπολογισμό του κατωφλίου. Ο πρώτος που είναι και ο πιο δαπανηρός είναι η μέθοδος Chow and Kaneko. Αυτή η μέθοδος χωρίζει την εικόνα σε πολλές επί μέρους αλληλεπικαλυπτόμενες εικόνες, υπολογίζει κατώφλι για κάθε μία από αυτές (σαν global δηλαδή) και χρησιμοποιεί interpolation στο τέλος για να σταθμιστούν τα επιμέρους threshold.

Local Thresholding

Ο δεύτερος τρόπος ο οποίος χρησιμοποιείται και πιο συχνά είναι το local thresholding. Η φιλοσοφία του δεν διαφέρει πολύ από την πρώτη μέθοδο αλλά είναι πολύ πιο γρήγορος και αρκετά αποτελεσματικός.

Όπως και στη μέθοδο Chow and Kaneko το threshold για κάθε pixel επιλέγεται σύμφωνα με τις πληροφορίες της γειτονιάς του. Συγκεκριμένα ορίζουμε μέγεθος παραθύρου m x n και το κατώφλι για κάθε pixel ορίζεται συνήθως ως η μέση τιμή της γειτονιάς (mean) ή η διάμεσος (median) με τον υπολογισμό του μέσου να είναι προφανώς πολύ πιο γρήγορος.

Αν όμως εφαρμόσουμε τον αλγόριθμο με τα μέχρι τώρα δεδομένα σε μια γειτονιά 50 x 50 παίρνουμε το παρακάτω αποτέλεσμα (3.1.10).



Εικόνα 3.1.10

Αυτό που συμβαίνει είναι ότι ενώ έχουν εντοπιστεί σωστά τα στοιχεία του foreground, οι περιοχές που είναι αποκλειστικά background έχουν λάθος αποτέλεσμα. Αυτό εξηγείται αφού η μέση τιμή αυτών των «λευκών» περιοχών είναι πολύ κοντά στις τιμές τους μιας και είναι ομοιόμορφες περιοχές. Το πρόβλημα επιλύεται εύκολα αν για κάθε περιοχή το threshold οριστεί όχι στη μέση τιμή (mean) αλλά στην τιμή:

$$Threshold = mean - C$$

όπου C μια σταθερά που παραμετροποιείται σύμφωνα με τις ανάγκες κάθε εφαρμογής.

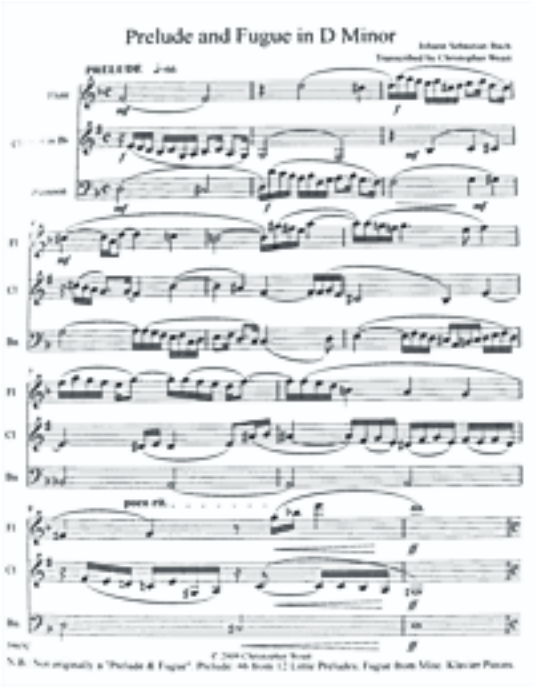
Με αυτόν τον τρόπο ομοιογενείς περιοχές, οι οποίες στην προκειμένη περίπτωση ανήκουν στο background (δεν υπάρχει στην παρτιτούρα περιοχή 50 x 50 όλη μαύρη), αυτόματα περνούν στο background. Ωστόσο πρέπει να επισημάνουμε ότι η επιλογή του C παίζει σημαντικό ρόλο στο αποτέλεσμα και για τα δεδομένα των μουσικών παρτιτούρων μπορεί να σταθεροποιηθεί σε μία τιμή. Επιπλέον σε εφαρμογές OCR χρησιμοποιείται εκτενώς το local thresholding μιας και

τα μεγέθη των χαρακτήρων είναι κοντινά.

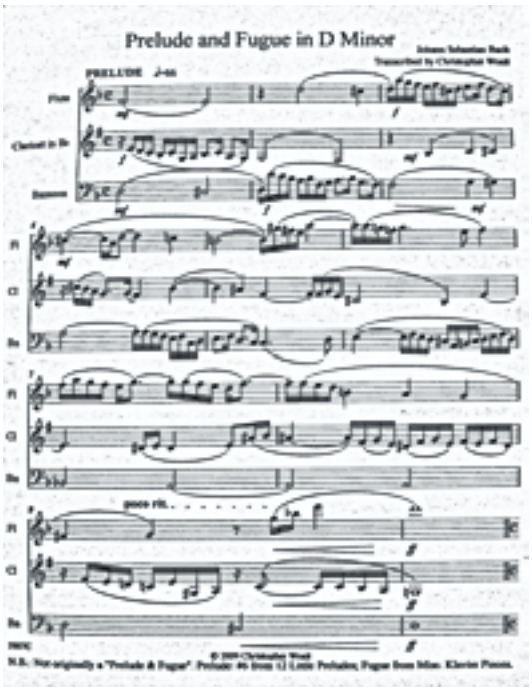
Σημ1: *Το αρνητικό πρόσημο σημαίνει αύξηση του threshold μιας και πάντα μιλάμε στην αρνητική λογική*

Παρακάτω βλέπουμε τα διαφορετικά αποτελέσματα (3.1.11-13) για τις τιμές του C με προφανώς καλύτερη επιλογή την C=0.05 κάτι το οποίο επαληθεύεται για πολλά επιπλέον δείγματα.

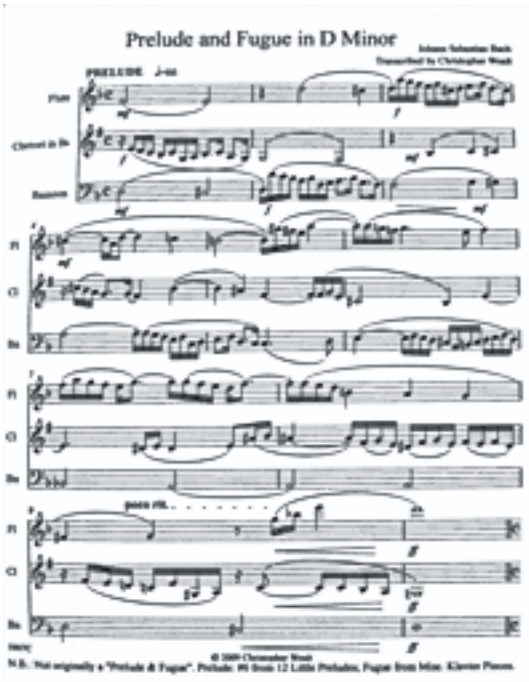
Σημ2: *Το C είναι κανονικοποιημένο οπότε το C=0.05 αντιστοιχεί στην τιμή 0.05 x 50 x 50 = 12.75*



Εικόνα 3.1.11



Εικόνα 3.1.12



Εικόνα 3.1.13

Εστιάζοντας και στο σημείο που εμφανίστηκε ο φθορά του πενταγράμμου παρατηρούμε ότι το πρόβλημα έχει λυθεί ικανοποιητικά (3.1.14)

Εικόνα 3.1.14

Συνοπτικά ο αλγόριθμος για adaptive thresholding μπορεί να γραφεί πολύ γρήγορα με τη χρήση φίλτρων

Δηλαδή:

```
If = φιλτράρουμε την εικόνα με φίλτρο τύπου
'average'
και διαστάσεις φίλτρου [h w]
Itmp = Iαρχική – If – C
Iout = threshold(Itmp,threshold=0)
```

Κλείνοντας αυτό το κεφάλαιο πρέπει να σημειώσουμε ότι πολλές μέθοδοι OMR δεν επιλέγουν σαν πρώτο στάδιο την δυαδικοποίηση ή ακόμα και αν το κάνουν λειτουργούν παράλληλα και με την grayscale εικόνα. Αυτές περιλαμβάνουν είτε template matching που δεν θα μελετήσουμε τόσο εκτενώς σε αυτή την εργασία η ακόμα και εντοπισμό του πενταγράμμου στον grayscale χώρο. Είναι προφανές ότι η μετάβαση από τον grayscale στον δυαδικό χώρο οδηγεί σε απώλεια πληροφορίας και όταν έχουμε φθαρμένες παρτιτούρες αυτή η πληροφορία μπορεί να είναι πολύ χρήσιμη, ωστόσο καμία μέθοδος δεν μπορεί τελικά να παρακάμψει την δυαδικοποίηση αφού διευκολύνει και επιταχύνει καθοριστικά όλα τα επόμενα βήματα.

3.2. Διόρθωση του προσανατολισμού της σελίδας (Angle Correction)

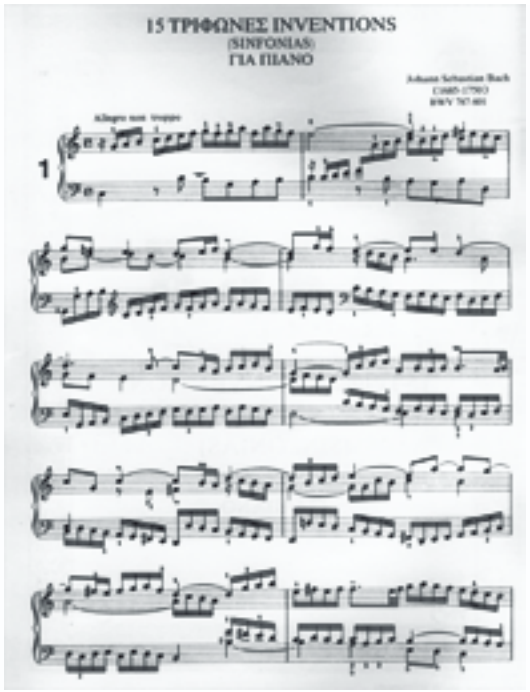
Επόμενο στάδιο προετοιμασίας μιας παρτιτούρας για αναγνώριση είναι η διόρθωση του προσανατολισμού της. Δεν είναι απαραίτητο αυτό το στάδιο να έπεται της δυαδικοποίησης. Σε συγκεκριμένες περιπτώσεις μπορεί και να προηγείται ή στην ιδανική περίπτωση να υπολογίζεται η γωνία με μια πρώτη δυαδικοποίηση, να περιστρέφεται έπειτα η grayscale εικόνα και στη συνέ-

χεια να ξανακάνουμε Binarization. Μπορεί να φαίνεται περιττή αυτή η διαδικασία αλλά όταν έχουμε λεπτές γραμμές όπως είναι οι γραμμές του πενταγράμμου τα αποτελέσματα μπορεί να παρουσιάσουν απόκλιση.

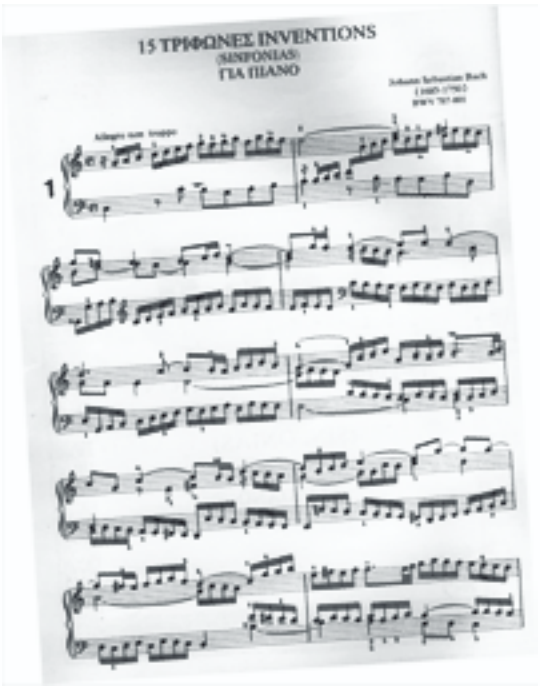
Σε αυτό το στάδιο σκοπός μας είναι να υπολογίσουμε τις μοίρες κατά τις οποίες η αρχική εικόνα απέχει από τον κατακόρυφο προσανατολισμό και τελικά να επαναφέρουμε την εικόνα σε αυτόν.

Θα χρησιμοποιήσουμε μια νέα εικόνα σαν παράδειγμα για να έχουμε πιο ενδεικτικά αποτελέσματα. Πρέπει να σημειωθεί ότι η διαδικασία διόρθωσης της περιστροφής δεν δίνει πάντα πλήρες αποτέλεσμα για μια σελίδα καθώς υπάρχει το ενδεχόμενο είτε τα επιμέρους πεντάγραμμα να έχουν διαφορετικές γωνίες ή να παρουσιάζουν καμπυλότητα (πχ στην άκρη της σελίδας) οπότε το πρόβλημα επεκτείνεται σε επόμενα επίπεδα.

Ωστόσο, θα μελετήσουμε το πρωτογενές πρόβλημα σε μια σελίδα και θα εξετάσουμε τα αποτελέσματα. Ξεκινώντας με την αρχική εικόνα (3.2.1) την περιστρέφουμε κατά γωνία 3.5 μοιρών (3.2.2).



Εικόνα 3.2.1



Εικόνα 3.2.2

Θα χρησιμοποιήσουμε δύο μεθόδους για την εύρεση της γωνίας χρησιμοποιώντας τώρα μόνο την περιεστρεμμένη εικόνα.

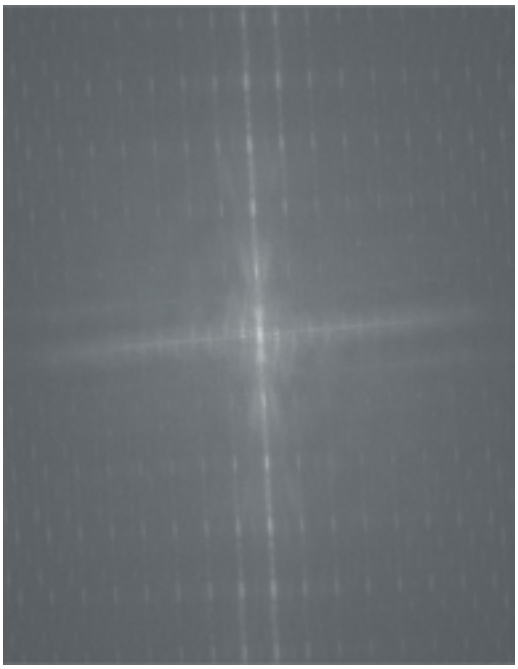
3.2.1. Διόρθωση γωνίας με μετασχηματισμό Fourier

Εφαρμόζοντας τον μετασχηματισμό Fourier στην εικόνα παίρνουμε το παρακάτω αποτέλεσμα (3.2.3). Στην εικόνα φαίνεται ότι η αλληλουχία των γραμμών του πενταγράμμου με συγκεκριμένη απόσταση τονίζουν κάποιες συχνότητες οι οποίες διατάσσονται σε δύο κάθετους άξονες. Στη συγκεκριμένη περίπτωση μάλιστα αυτό το σύστημα των αξόνων είναι περιεστρεμμένο παρόμοια με την περιστροφή που εφαρμόσαμε στην αρχική εικόνα. Για να βρούμε την περιστροφή των αξόνων εντοπίζουμε μια κορυφή P=(Px,Py) (μακρινή από το κέντρο για να μειώσουμε το

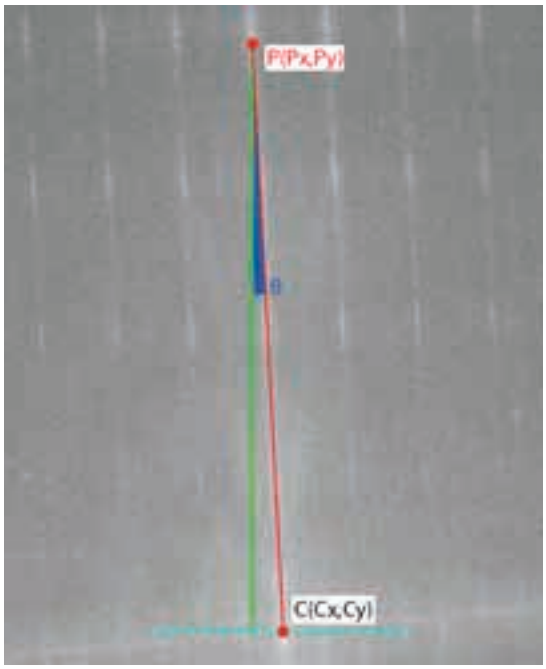
σφάλμα λόγω ανάλυσης) και δεδομένου ότι το κέντρο της εικόνας είναι το C=(Cx,Cy) η κλίση δίνεται από τον τύπο (3.2.4):

$$\theta = arctan\left(\frac{P_x - C_x}{P_y - C_y}\right)$$

Παίρνοντας το σημείο P=(634,182) και το κέντρο C=(669,853) έχουμε θ=2.9851. Η απόκλιση από τις 3.5 μοίρες που εφαρμόσαμε δεν δείχνει απαραίτητα σφάλμα καθώς η γωνία που υπολογίσαμε είναι η απόκλιση από τον ιδανικό κατακόρυφο προσανατολισμό και όχι από την αρχική εικόνα. Θα εξετάσουμε αργότερα τα πιθανά σφάλματα.



Εικόνα 3.2.3



Εικόνα 3.2.4

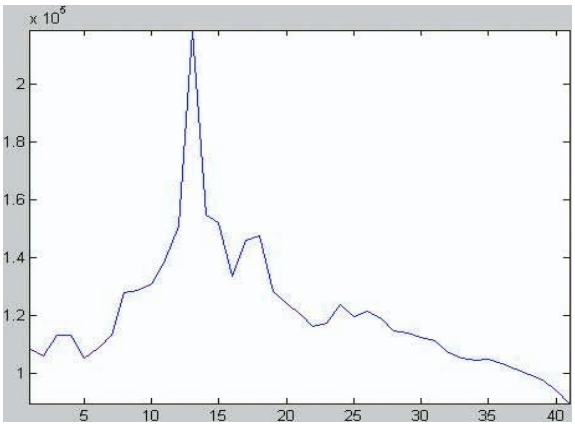
Η συγκεκριμένη μέθοδος είναι αρκετά αξιόπιστη αλλά η εύρεση του κατάλληλου σημείου για να βρεθεί η γωνία δεν είναι πάντα εύκολη. Επίσης εξαρτάται σημαντικά από την ανάλυση της εικόνας και σε μικρές εικόνες, ειδικά με μικρό ύψος δυσχεραίνει τη διαδικασία.

Για αυτό το λόγο μελετάμε ακόμα μία μέθοδο η οποία σε αντίστοιχο χρόνο μπορεί να μας δώσει καλύτερα αποτελέσματα.

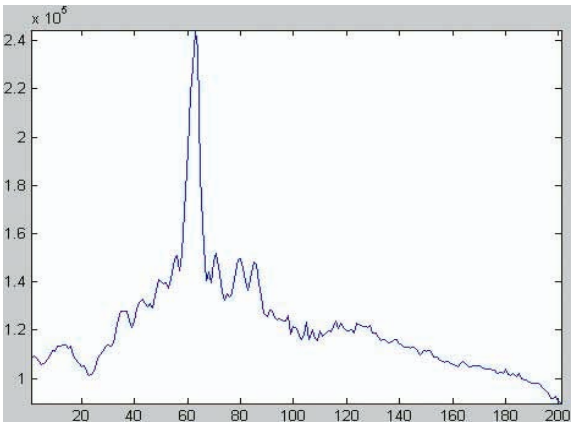
3.2.2. Διόρθωση γωνίας με χρήση οριζοντίων προβολών (Horizontal Projections)

Σύμφωνα με αυτή τη μέθοδο, η γωνία κατά την οποία μια παρτιτούρα έχει τον βέλτιστο προσανατολισμό είναι η γωνία κατά την οποία παρουσιάζει μέγιστη τιμή η οριζόντια προβολή της εικόνας. Αυτό βασίζεται στην υπόθεση ότι κατά τον βέλτιστο προσανατολισμό ευθυγραμμίζονται οι γραμμές το πενταγράμμου και προκαλούν αύξηση στα αποτελέσματα των αντίστοιχων οριζοντίων προβολών.

Μια πρώτη υλοποίηση της μεθόδου είναι η σταδιακή περιστροφή της εικόνας μεταξύ ενός εύρους τιμών και με ένα συγκεκριμένο βήμα και η καταγραφή των μέγιστων τιμών των προβολών. Για να βελτιώσουμε το χρόνο θα περιορίσουμε τις τιμές μεταξύ των 10 και -10 μοιρών και θα θέσουμε το βήμα στο 0.5. Έτσι παίρνουμε το παρακάτω αποτέλεσμα (3.2.5)



Εικόνα 3.2.5

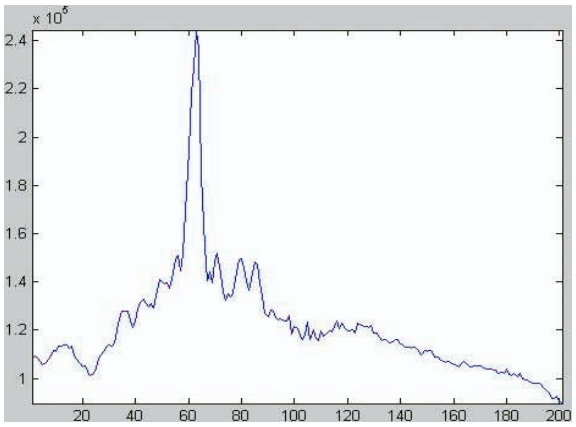


Εικόνα 3.2.6

Μειώνοντας το βήμα σε 0.05 (3.2.6) προκειμένου να αυξήσουμε την ανάλυση της προσέγγισης παίρνουμε αποτέλεσμα θ=-3.8 ενώ με βήμα 0.5 παίρνουμε αποτέλεσμα θ=-4. Ποιοτικά βλέπουμε ότι το διάγραμμα έχει πολύ ενδεικτική συμπεριφορά γύρω από τις 3.8 μοίρες αποδεικνύοντας την αξιοπιστία του αποτελέσματος.

Πριν εξετάσουμε τα αποτελέσματα των δύο μεθόδων οπτικά, θα προτείνουμε μια πολύ πιο γρήγορη και πολλές φορές πιο ακριβή παραλλαγή της δεύτερης μεθόδου. Παρατηρούμε ότι η συμπεριφορά των προβολών γύρω από το ζητούμενο σημείο είναι χαρακτηριστική, δηλαδή πέρα από μικρές διακυμάνσεις, η συνάρτηση πριν το σημείο είναι αύξουσα και μετά το σημείο φθίνουσα. Αν είχαμε αποδεδειγμένα γνήσια μονοτονία γύρω από το σημείο θα μπορούσαμε να εφαρμόσουμε μια μορφή δυαδικής αναζήτησης. Δηλαδή, δοσμένου ενός αρχικού εύρους τιμών (πχ 20 μοίρες) και ενός κέντρου (πχ 0 μοίρες) θα συγκρίνονταν διαδοχικά οι τιμές των προβολών στο μέσο των δύο διαστημάτων (-5,5 μοίρες) και θα ορίζεται νέο κέντρο η γωνία με τη μεγαλύτερη προβολή και νέο εύρος το μισό του προηγούμενου (20/2=10 μοίρες). Αν καμία από τις δύο γωνίες δεν είναι μεγαλύτερη από την τιμή του κέντρου, τότε το κέντρο παραμένει ίδιο και απλά μειώνεται το εύρος.

Στο παρακάτω διάγραμμα (3.2.7) φαίνεται πως σε 4 επαναλήψεις έχει προσεγγιστεί η θέση του μεγίστου. Οι διαφορετικές αποχρώσεις των διακεκομμένων γραμμών δείχνουν τη θέση του κέντρου σε κάθε βήμα. Προφανώς μετά την τέταρτη επανάληψη απλά μειώνεται το εύρος καθώς έχει ήδη εντοπιστεί η κορυφή.

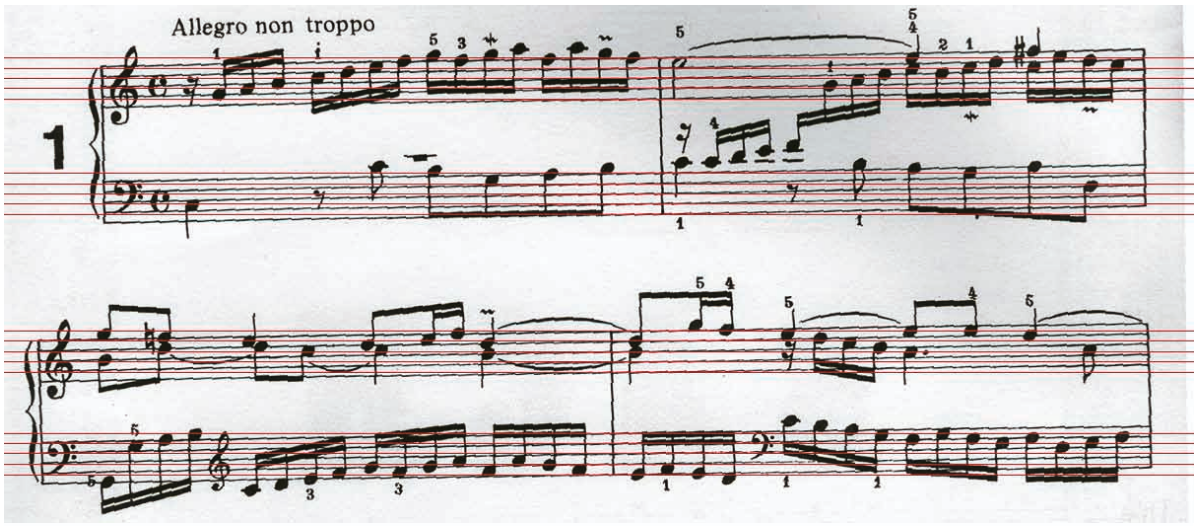


Εικόνα 3.2.7

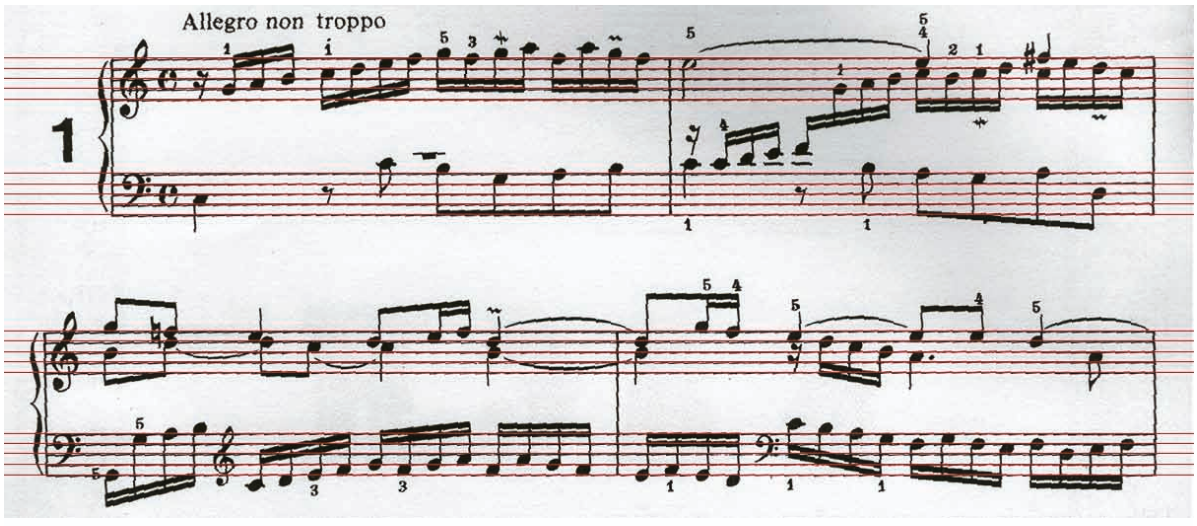
Η τελευταία μέθοδος μπορεί επιπλέον να βελτιωθεί βάζοντας ένα όριο διαφοράς μεταξύ των συγκρίσεων. Αν δηλαδή οι διαφορά μεταξύ της τιμής της προβολής σε δύο γωνίες δεν είναι μεγάλη να κρατούνται και οι δυο γωνίες σαν κέντρα μέχρι να πληρείται μια ελάχιστη διαφορά. Αυτό επιλύει και το πρόβλημα απουσίας γνήσιας μονοτονίας γύρω από το ζητούμενο σημείο.

Εξέταση των αποτελεσμάτων

Για να εξετάσουμε ποιοτικά και οπτικά τα αποτελέσματα θα περιστρέψουμε την εικόνα σύμφωνα με τις μοίρες που υπέδειξαν οι δύο μέθοδοι. Έπειτα από το κέντρο κάθε γραμμής του πενταγράμμου θα «τραβήξουμε» παράλληλες σε όλο το μήκος της σελίδας και θα μετρήσουμε ενδεχόμενες αποκλίσεις στα άκρα (3.2.8-9).



Εικόνα 3.2.8



Εικόνα 3.2.9

Είναι εμφανές ότι η δεύτερη μέθοδος μας δίνει πολύ καλύτερο αποτέλεσμα δεδομένου ότι υπάρχει ελάχιστη απόκλιση των γραμμών των πενταγράμμου από αυτές που ορίσαμε, γεγονός που δείχνει ότι ο προσανατολισμός της σελίδας είναι βέλτιστος.

Είναι πολύ σπάνιο να πετύχουμε τον ιδανικό προσανατολισμό μιας σελίδας μιας και δύσκολα υπάρχει πλήρης παραλληλισμός των πενταγράμμων. Αυτό γιατί σύνηθες φαινόμενο είναι η καμπύλωση της σελίδας κοντά στο σημείο που δένεται το βιβλίο. Την αντιμετώπιση αυτού του φαινομένου θα τη μελετήσουμε παρακάτω, ωστόσο οι αλγόριθμοι που χρησιμοποιούνται στα επόμενα βήματα του OMR δεν πρέπει να είναι εξαρτημένοι από μια ιδανικά προσανατολισμένη και παραλληλισμένη σελίδα αλλά πρέπει να μπορούν να ενσωματώσουν τέτοια σφάλματα. Επιπλέον να σημειώσουμε ότι ακόμα καλύτερα αποτελέσματα έχουμε αν διορθώσουμε τον προσανατολισμό όχι συνολικά μιας σελίδας αλλά κάθε «παραθύρου» (που περιέχει το πεντάγραμμο και τα σύμβολα που αναφέρονται σε αυτό) ξεχωριστά.

3.3. Image Dewarping

3.3.1. Το πρόβλημα της καμπύλωσης (warping)

Ένα βήμα, το οποίο δεν είναι πάντα προϋπόθεση της προεπεξεργασίας μιας παρτιτούρας, είναι η επαναφορά μιας «σκεβρωμένης» ή καμπυλωμένης εικόνας το οποίο θα αναφέρουμε από δω και πέρα ως dewarping. Η συγκεκριμένη λειτουργία δεν είναι παρούσα σε όλες τις εφαρμογές OMR μιας και δεν μπορεί πάντα να ολοκληρωθεί με πλήρη επιτυχία. Αν όμως εντοπιστεί σωστά η παραμόρφωση και επαναφερθεί η εικόνα, η μετέπειτα διαδικασία γίνεται πολύ πιο εύκολη.

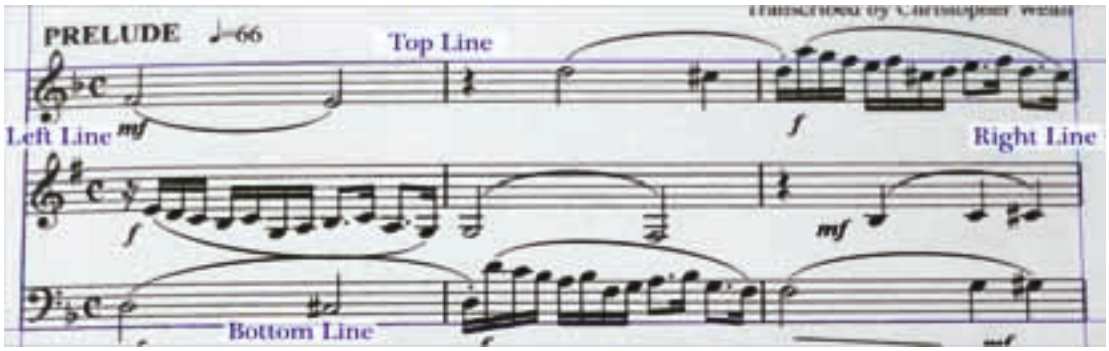
Το βασικό πρόβλημα του Image Dewarping είναι ουσιαστικά ένα πρόβλημα προβολής από ένα σύστημα συντεταγμένων σε ένα άλλο. Το δεύτερο σύστημα πάνω στο οποίο θέλουμε να προβάλουμε την αρχική εικόνα είναι γνωστό και είναι το καρτεσιανό επίπεδο. Ωστόσο το αρχικό σύστημα δεν μας είναι γνωστό διότι, για να το γνωρίζαμε, θα έπρεπε να ξέρουμε πλήρως την γεωμετρία (το ανάγλυφο δηλαδή) της σελίδας κάτι το οποίο είναι αδύνατο. Συνήθως η καμπυλότητα μιας σελίδας προσεγγίζει μια κυλινδρική επιφάνεια και αυτό χρησιμοποιείται συνήθως σαν υπόθεση για την υλοποίηση της διαδικασίας.

Βασιζόμενοι και εμείς σε αυτή την υπόθεση θα υλοποιήσουμε μια μέθοδο που αποκαθιστά με πολύ καλά αποτελέσματα την καμπυλωμένη σελίδα. Η παρτιτούρα που χρησιμοποιούμε σε αυτή την εργασία είναι ενδεικτική για δοκιμή μιας και έχει μια «ρεαλιστική» παραμόρφωση που συναντάται συχνά σε τέτοια έγγραφα.

3.3.2. Περιγραφή της διαδικασίας διόρθωσης της καμπύλωσης

Η μέθοδος που χρησιμοποιούμε απαιτεί αρχικά τον εντοπισμό των περιβαλλουσών της εικόνας που θέλουμε να επαναφέρουμε. Αυτό γιατί οι περιβάλλουσες μας δίνουν μια προσέγγιση του χώρου πάνω στον οποίο είναι προβεβλημένη η παρτιτούρα, της γεωμετρίας δηλαδή της σελίδας. Αυτές οι πληροφορίες είναι απλά συμβατική προσέγγιση, μιας και μοναδικότητα της λύσης μπορούμε να έχουμε μόνο αν γνωρίζουμε τις συντεταγμένες του χώρου προέλευσης. Οι συμβάσεις αυτές όμως αρκούν για να έχουμε τα επιθυμητά αποτελέσματα.

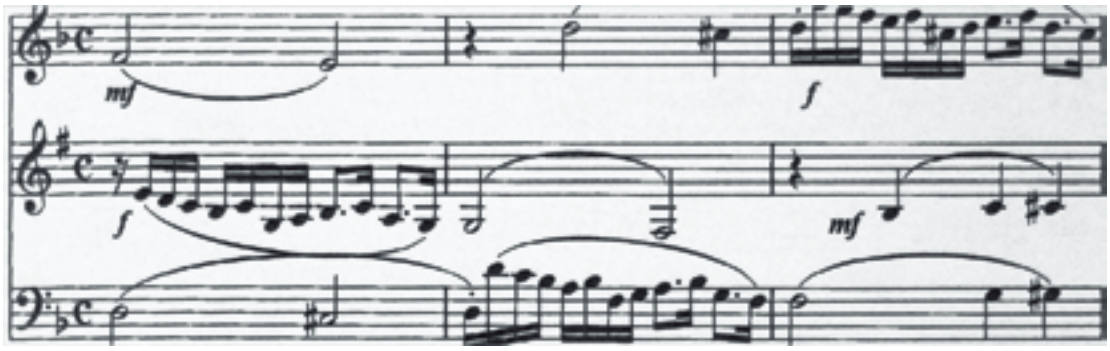
Οι περιβάλλουσες που χρειαζόμαστε είναι οι: άνω περιβάλλουσα (top line), η κάτω περιβάλλουσα (bottom line), η αριστερή (left line) και η δεξιά (right line) όπως φαίνονται στο σχήμα παρακάτω (3.3.1).



Εικόνα 3.3.1

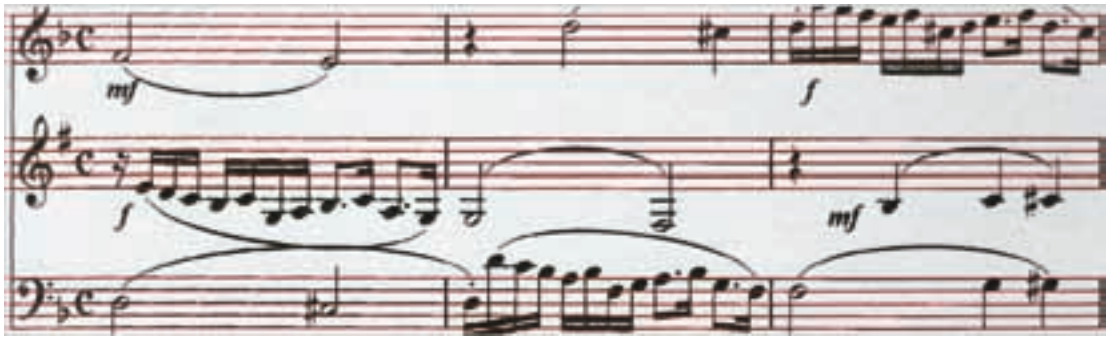
Η συγκεκριμένη μέθοδος λειτουργεί στην υπόθεση της κυλινδρικής παραμόρφωσης της σελίδας και γι' αυτό το λόγο δέχεται σαν είσοδο τις άνω και κάτω περιβάλλουσες με μορφή πολυωνύμου οποιουδήποτε βαθμού αλλά την αριστερή και τη δεξιά μόνο με τη μορφή ευθείας γραμμής, δηλαδή πολυωνύμου 1ου βαθμού.

Με αυτά τα δεδομένα και μόνο μπορούμε να εξάγουμε το τελικό αποτέλεσμα και μάλιστα σε οποιεσδήποτε διαστάσεις θέλουμε. Συνήθως η τελική εικόνα έχει σαν πλάτος το ελάχιστο του μήκους μεταξύ των top line και bottom line και σαν ύψος το ελάχιστο μεταξύ left line και right line. Εφαρμόζοντας τη μέθοδο στην παραπάνω εικόνα παίρνουμε το παρακάτω αποτέλεσμα (3.3.2)

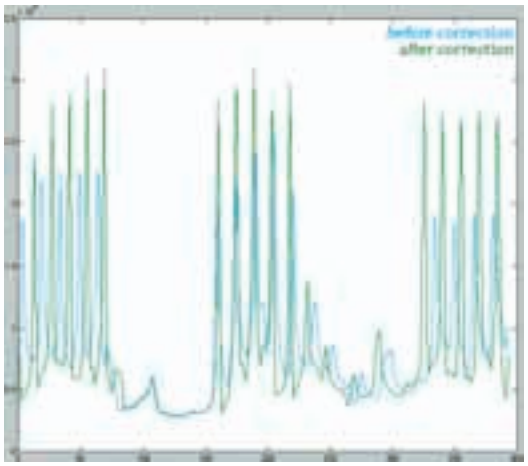


Εικόνα 3.3.2

Η ποιότητα του αποτελέσματος φαίνεται όχι μόνο οπτικά. Αν φέρουμε πάλι οριζόντιες ευθείες από το μέσο των γραμμών του πενταγράμμου θα δούμε ότι ταυτίζονται με τις γραμμές σε όλο το πλάτος της εικόνας (3.3.3). Επιπλέον αν δούμε την οριζόντια προβολή θα δούμε μεγαλύτερες κορυφές στα σημεία του πενταγράμμου γεγονός που μαρτυρά καλύτερο οριζόντιο προσανατολισμό (3.3.4).

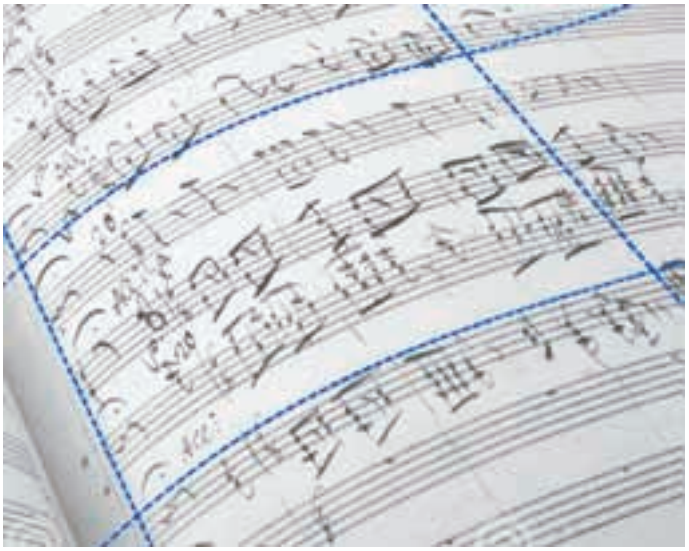


Εικόνα 3.3.3

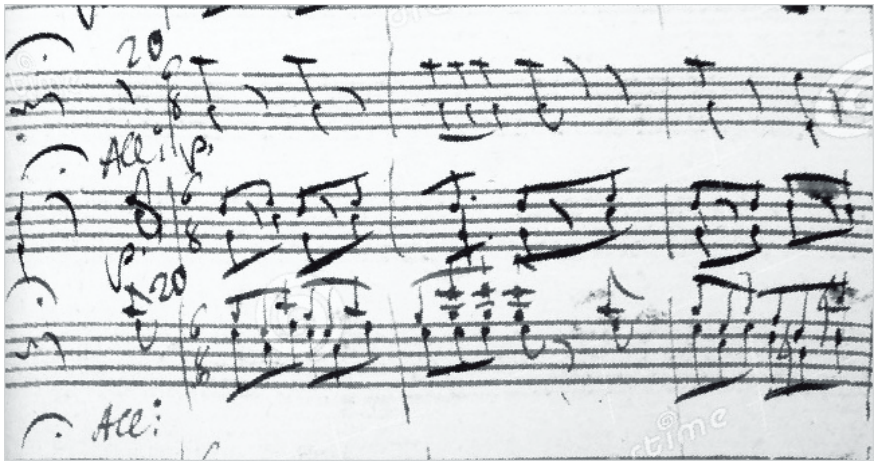


Εικόνα 3.3.4

Σαν τελευταίο παράδειγμα θα χρησιμοποιήσουμε μια πολύ παραμορφωμένη παρτιτούρα (3.3.5) για να εξετάσουμε τη συμπεριφορά του αλγορίθμου. Το αποτέλεσμα είναι αρκετά ικανοποιητικό ακόμα και σε αυτόν τον βαθμό παραμόρφωσης (3.3.6).

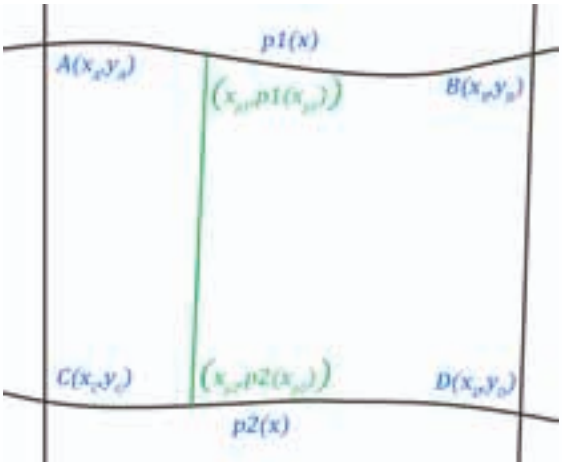


Εικόνα 3.3.5



Εικόνα 3.3.6

3.3.3. Αναλυτική περιγραφή του αλγορίθμου



Εικόνα 3.3.7

Υπολογίζουμε αρχικά το μήκος του πολυώνυμου p1 μεταξύ των σημείων A και B

$$l_{p1} = \int_{x_A}^{x_B} \sqrt{1 + \left(\frac{dp_1(x)}{dx}\right)^2}\, dx$$

Ορίζουμε τις διαστάσεις της εικόνας εξόδου πχ:

$$height = |AC|$$
$$width = l_{p1}$$

Καθώς και τις παρακάτω τιμές

$$\delta x_{p1} = \frac{(x_B - x_A)}{l_{p1}}$$
$$\delta x_{p2} = \frac{(x_C - x_D)}{l_{p1}}$$

Οπότε για κάθε σημείο της τελικής εικόνας **V(x,y)** έχουμε:

$$V(x,y) = I(x',y')$$

$$0 \le x \le width$$
$$0 \le y \le height$$

Όπου για τα $\mathbf{x', y'}$ ισχύει

$$x' = w_t x_{p1} + (1 - w_t) x_{p2}$$
$$w_t = \frac{|AC| - y}{|AC|}$$

Και τα σημεία $\mathbf{x_{p1}}$και $\mathbf{x_{p2}}$ υπολογίζονται από τους τύπους

$$x_{p1} = x\delta x_{p1} + x_A$$
$$x_{p2} = x\delta x_{p2} + x_C$$

Και για το $\mathbf{y'}$ισχύει:

$$y' = w_t p_1(x_{p1}) + (1 - w_t) p_2(x_{p2})$$

Στον διακριτό χώρο της εικόνας μπορούμε να εφαρμόσουμε και κάποιες μεθόδους παρεμβολής (interpolation) μιας και οι μεταβλητές x^',y' δεν είναι ακέραιες και άρα δεν αντιστοιχούν σε συγκεκριμένο pixel της αρχικής εικόνας. Έτσι για πιο ομαλά αποτελέσματα μπορούμε να εφαρμόσουμε είτε bilinear είτε bicubic interpolation για πιο ακριβές αποτέλεσμα.

Να σημειώσουμε ότι το δύσκολο στάδιο και το πιο κρίσιμο αυτής της διαδικασίας είναι ο ακριβής προσδιορισμός των περιβαλλουσών. Όσο πιο κοντά είναι οι περιβάλλουσες στη συμπεριφορά του πενταγράμμου τόσο πιο ακριβές είναι και το αποτέλεσμα. Για να γίνει όμως αυτό χρειάζεται αρχικά να εντοπιστούν οι γραμμές του πενταγράμμου κάτι το οποίο δεν έχουμε περιγράψει ακόμα. Έτσι η διαδικασία του dewarping τοποθετείται συχνά μετά τον εντοπισμό του πενταγράμμου μιας και είναι απαραίτητο βήμα για να πάρουμε τα πολυώνυμα των περιβαλλουσών. Μπορούμε έτσι να φτιάξουμε ένα πλέγμα στο σύνολο της παρτιτούρας και να αντιμετωπίσουμε κάθε κομμάτι του σαν αυτοτελές κατασκευάζοντας πλήρως την αρχική εικόνα.

Στα επόμενα βήματα θα χρησιμοποιήσουμε τόσο την διορθωμένη εικόνα (κυρίως σε μεθόδους που έχουν σαν προϋπόθεση την διόρθωση της καμπύλωσης) αλλά και την αρχική για να δείξουμε ότι τελικά το πρόβλημα της καμπύλωσης μπορεί και να παρακαμφθεί αν προφανώς η παραμόρφωση είναι εντός κάποιων ορίων.

3.4. Τα βασικά μεγέθη Staff Thickness και Staff Distance

3.4.1. Η χρησιμότητα των δύο παραμέτρων

Τελευταίο στάδιο προεπεξεργασίας μιας εικόνας παρτιτούρας και προετοιμασίας για το βασικό κομμάτι του OMR είναι ο υπολογισμός δύο παραμέτρων: του staff thickness, δηλαδή του πάχους του πενταγράμμου και του staff distance, δηλαδή της απόστασης μεταξύ δύο γραμμών του πενταγράμμου.

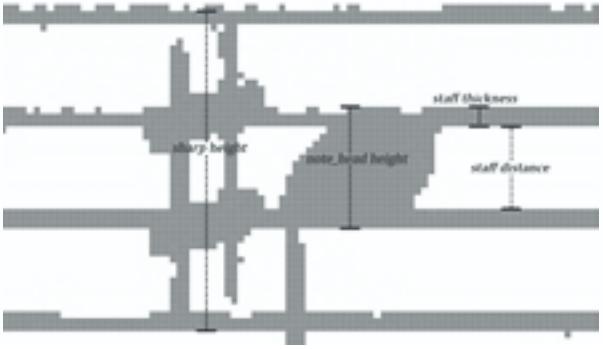
Ο υπολογισμών αυτών των παραμέτρων είναι απαραίτητος αφού η αυτοματοποίηση όλων των λειτουργιών και συναρτήσεων στα επόμενα βήματα του OMR γίνεται με σημείο αναφοράς αυτές τις διαστάσεις της παρτιτούρας. Έτσι μπορούμε να γνωρίζουμε ότι πχ το ύψος της κεφαλής μιας νότας είναι περίπου

$$height_{notehead} \cong 2 * thickness + distance$$

ή ότι το ύψος της δίεσης είναι περίπου

$$height_{sharp} \cong 4 * thickness + 3 * distance$$

όπως φαίνεται και στην εικόνα παρακάτω (3.4.1).

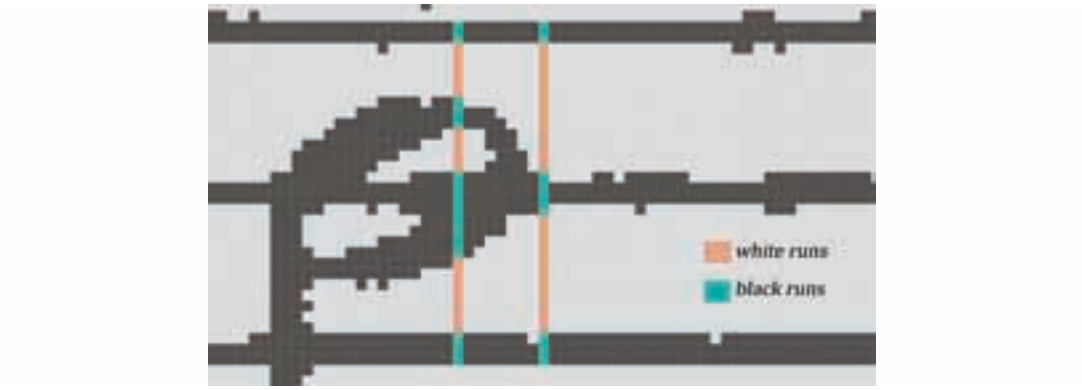


Εικόνα 3.4.1

Προφανώς, όπως φαίνεται και στην εικόνα, και το πάχος του πενταγράμμου αλλά και οι μεταξύ τους αποστάσεις δεν είναι σταθερές σε όλη την παρτιτούρα κάτι το οποίο οφείλεται είτε στην διαδικασία της δυαδικοποίησης, είτε στην καμπύλωση της σελίδας ή στην ίδια την εκτύπωση. Αρκεί όμως να βρούμε τις πιο συχνές τιμές που παίρνουν αυτές οι παράμετροι για έχουμε έστω και προσεγγιστικά ένα σημείο αναφοράς.

3.4.2 Τρόπος Υπολογισμού του Staff Thickness και Staff Distance

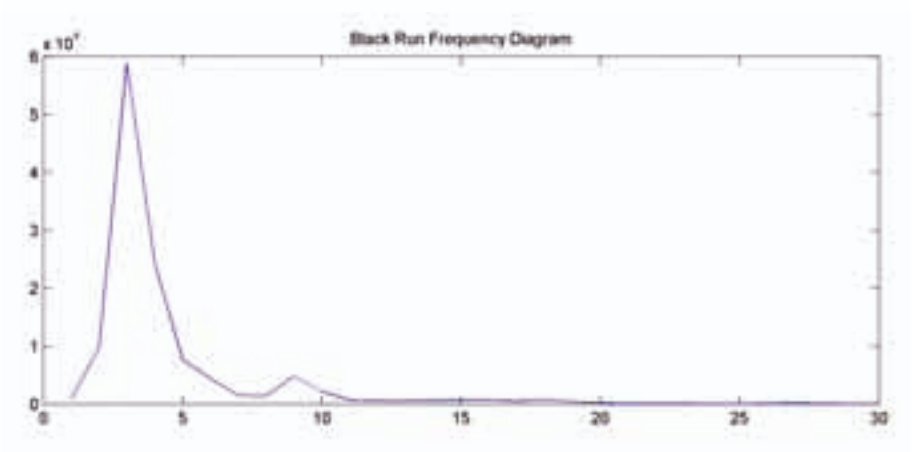
Ο υπολογισμός του staff thickness και staff distance γίνεται σε όλες της μεθόδους OMR με παρόμοιο τρόπο. Αρχικά η εικόνα χωρίζεται σε κάθετες διαδοχές μαύρων και άσπρων στοιχείων (vertical black runs, vertical white runs) και αποθηκεύονται σε δύο πίνακες τα μήκη αυτών των κομματιών (3.4.2).



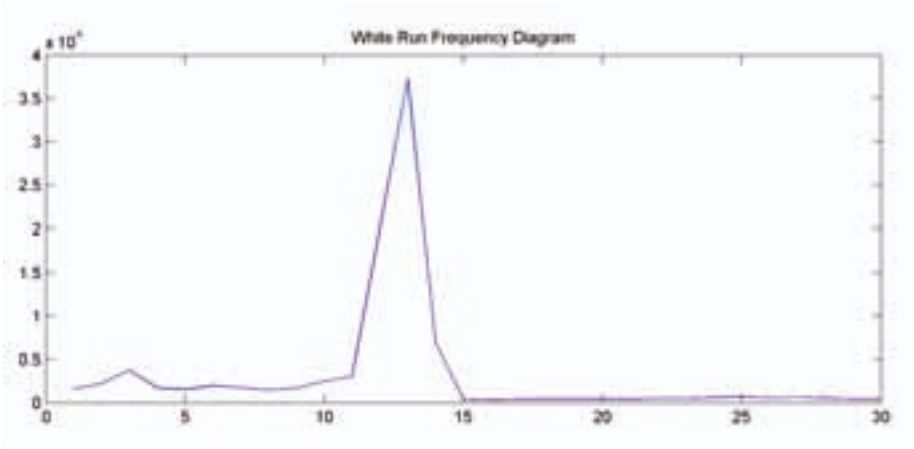
Εικόνα 3.4.2

Αν εξετάσουμε τα ιστογράμματα των δύο πινάκων θα δούμε ότι υπάρχει μία πολύ εμφανής κορυφή για κάθε πίνακα. Αυτό συμβαίνει προφανώς γιατί κατά μήκος της παρτιτούρας αυτές οι δύο τιμές επαναλαμβάνονται πάρα πολύ συχνά σε σχέση με τις υπόλοιπες. Έτσι έχουμε τα παρακάτω διαγράμματα συχνότητας των μηκών των black runs (3.4.3) και των white runs (3.4.4). Οι κορυφές αυτών των δύο διαγραμμάτων μας δίνουν τελικά τις ζητούμενες τιμές staff thickness και staff height αντίστοιχα. Στην προκειμένη περίπτωση

staff thickness=3
staff distance=13



Εικόνα 3.4.3



Εικόνα 3.4.3

Τέλος, κάθε φορά που χρησιμοποιούμε αυτές τις δυο παραμέτρους, συνήθως, χρησιμοποιούμε ένα εύρος τιμών γύρω από αυτές (πχ $(0.5 * thickness, 1.5 * thickness)$) προκειμένου να συμπεριλάβουμε το σύνολο των τιμών που παίρνει τα black runs του πενταγράμμου. Εναλλακτικά μπορούμε με γραμμική παρεμβολή να προσδιορίσουμε το εύρος τιμών θέτοντας ένα threshold πχ σε συχνότητες άνω του $1.5 * 10^4$ και να πάρουμε ένα πιο αντιπροσωπευτικό διάστημα τιμών.

4
Εντοπισμός του Πενταγράμμου

4.1. Εισαγωγή

Έχοντας εισέλθει πλήρως στη διαδικασία ανάλυσης και αναγνώρισης μιας παρτιτούρας και έχοντας ολοκληρώσει τα προηγούμενα στάδια προεπεξεργασίας, ακολουθεί ένα από τα πιο θεμελιώδη βήματα σε κάθε σύστημα OMR, αυτό του εντοπισμού του πενταγράμμου (staff detection).

Η οπτική αναγνώριση μουσικής, ως υποενότητα της ανάλυσης δομημένων εγγράφων, βασίζεται στο γεγονός ότι το έγγραφο το οποίο αναλύει έχει κάποια συγκεκριμένα χαρακτηριστικά, τα οποία τελικά αποτελούν την αφετηρία ανάλυσης του εγγράφου. Στην περίπτωση, λοιπόν, του OMR το βασικό συστατικό που «δένει» όλα τα στοιχεία μαζί και μας δίνει την πληροφορία ότι έχουμε μια παρτιτούρα είναι το πεντάγραμμο.

Έχει αναλυθεί προηγουμένως ο ρόλος του πενταγράμμου στην ερμηνεία της μουσικής από τον άνθρωπο. Από δω και πέρα θα εξετάσουμε τη δυνατότητα του υπολογιστή να εντοπίσει και να αποθηκεύσει της πληροφορίες που μας δίνουν οι γραμμές του πενταγράμμου.

Θα εξετάσουμε παρακάτω μια σειρά από δημοφιλείς μεθόδους και θα τις δοκιμάσουμε πάνω στην ίδια παρτιτούρα που εξετάζουμε από την αρχή της εργασίας για να δούμε τα αποτελέσματα τους. Είναι γεγονός ότι πολλές από αυτές τις μεθόδους έχουν δημιουργηθεί για επίλυση πιο «ακραίων» περιπτώσεων από αυτές που εξετάζουμε σε αυτή την εργασία. Δηλαδή αφορούν περιπτώσεις μεγάλης φθοράς και παραμόρφωσης ενώ εμείς εξετάζουμε το σενάριο μιας σκαναρισμένης παρτιτούρας περιορισμένης παραμόρφωσης. Γι'αυτό το λόγο κάποιοι από τους αλγόριθμους που ακολουθούν δίνουν πολύ μεγαλύτερη έμφαση στην ακρίβεια του αποτελέσματος παρά στον χρόνο εκτέλεσης.

Τέλος να σημειώσουμε ότι αναφερόμενοι στον εντοπισμό του πενταγράμμου εννοούμε κυρίως την μαθηματική έκφραση της νοητής γραμμής του πενταγράμμου και όχι των συγκεκριμένων στοιχείων (pixels) που είναι κομμάτια του πενταγράμμου. Ο επακριβής προσδιορισμού του τελευταίου εντάσσεται στο επόμενο βήμα, δηλαδή στην αφαίρεση ή αλλιώς διαχωρισμός (segmentation) του πενταγράμμου στο οποίο προστίθενται επιπλέον προβλήματα.

4.2. Συνήθεις μέθοδοι εντοπισμού πενταγράμμου

Οι μέθοδοι που θα μελετήσουμε είναι οι παρακάτω. Συχνά χρησιμοποιείται συνδυασμός αυτών των μεθόδων για επιβεβαίωση ή ακόμα και άλλες μέθοδοι αλλά όλες έχουν στον πυρήνα τους τις αφετηρίες των παρακάτω:

1. Εντοπισμός μέσω οριζοντίων προβολών

2. Χρήση προβολών σε κινούμενο παράθυρο (sliding window)

3. Χρήση μικρών κάθετων διαδοχικών στοιχείων (short vertical runs)

4. Μορφολογικές διαδικασίες

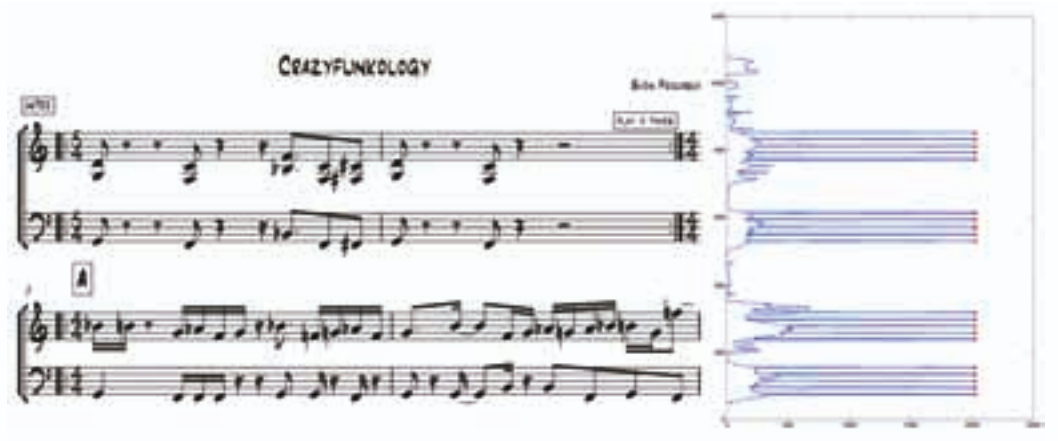
5. Πολυωνυμική παρεμβολή

Η εφαρμογή των μεθόδων θα γίνει με χρήση της εικόνας χωρίς dewarping μιας και όπως αναφέρθηκε νωρίτερα η σωστή ολοκλήρωση του dewarping απαιτεί να γνωρίζουμε τη θέση και τη συμπεριφορά των γραμμών του πενταγράμμου και συγκεκριμένα την πολυωνυμική τους έκφραση.

4.2.1. Εντοπισμός μέσω οριζοντίων προβολών (Horizontal Projection)

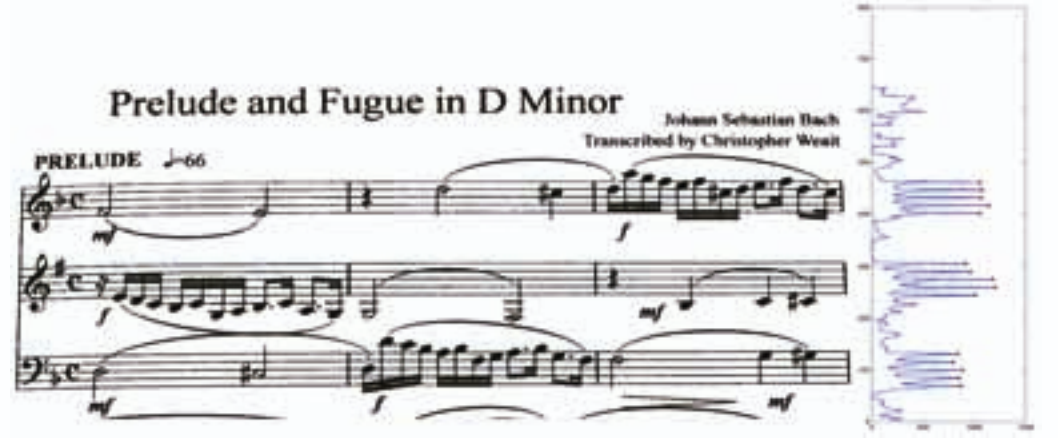
Υπό ιδανικές συνθήκες σε μια παρτιτούρα υπάρχουν ευθείες γραμμές με μήκος περίπου ¾ του πλάτους της σελίδας και γωνίας 0°. Έτσι αν εξετάσουμε το διάγραμμα της οριζόντιας προβολής της εικόνας θα εμφανιστούν εμφανή τοπικά μέγιστα που θα μας δείχνουν τη θέση των γραμμών του πενταγράμμου.

Όντως, εξετάζοντας μια πρότυπη παρτιτούρα παίρνουμε το παρακάτω διάγραμμα (4.2.1).



Εικόνα 4.2.1

Τα σημεία το κορυφών στο διάγραμμα μας δίνουν την ακριβή θέση των ευθειών που αντιπροσωπεύουν τις γραμμές του πενταγράμμου. Ωστόσο η παραπάνω περίπτωση είναι πολύ σπάνια σε σκαναρισμένες παρτιτούρες. Η συγκεκριμένη έχει προέλθει από αρχείο MusicXML και όχι από αναλογικό έγγραφο. Στις περισσότερες περιπτώσεις αναλογικών εγγράφων συναντάμε λιγότερο ομοιόμορφη συμπεριφορά.



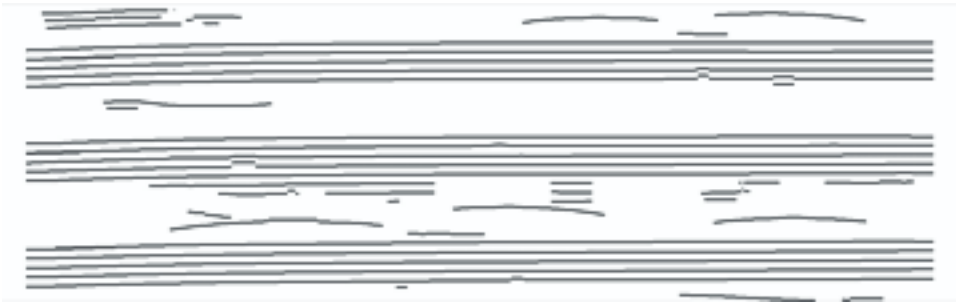
Εικόνα 4.2.2

Το πρόβλημα στη συγκεκριμένη περίπτωση είναι ότι οι γραμμές των πενταγράμμων δεν είναι ευθείες και ενώ έχουμε εντοπίσει κορυφές στην οριζόντια προβολή αν τις προεκτείνουμε θα αποκλίνουν από τις γραμμές που αναζητούμε. Στην περίπτωση βέβαια που επαναφέρουμε τη σελίδα μέσω dewarping και γνωρίζουμε πια ότι οι γραμμές είναι ευθείες μπορούμε να χρησιμοποιήσουμε αυτή τη μέθοδο.

Για την αντιμετώπιση των ελλείψεων της πρώτης μεθόδου οδηγηθήκαμε στην εξελιγμένη μορφή της που είναι η χρήση των προβολών σε όλο το πλάτος της εικόνας.

4.2.2. Χρήση προβολών σε κινούμενο παράθυρο (sliding window)

Γνωρίζουμε ότι υπάρχει το ενδεχόμενο οι γραμμές του πενταγράμμου να μην είναι παράλληλες ευθείες και άρα οι οριζόντιες προβολές να μην μας δίνουν επιθυμητά αποτελέσματα. Συνεπώς για να εξετάσουμε την ανομοιογενή συμπεριφορά του πενταγράμμου στο πλάτος της παρτιτούρας ορίζουμε ένα παράθυρο με ύψος το ύψος της εικόνας και πλάτος περίπου **width_window=(width_page)/20.** Το παράθυρο αυτό θα ολισθαίνει από τα αριστερά προς τα δεξιά και θα αποθηκεύει κάθε φορά την θέση των τοπικών μεγίστων στη θέση **x=i+(width_window)/2=i+(width_page)/40** όπου **i** είναι κάθε φορά η θέση του αριστερού άκρου του παραθύρου.



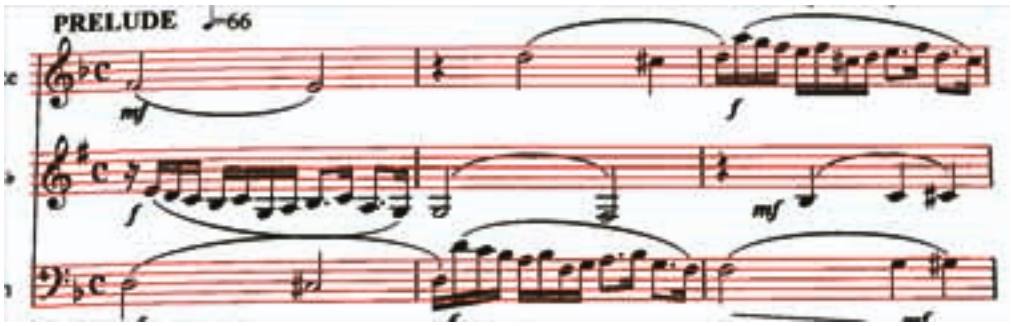
Εικόνα 4.2.3

Εφαρμόζοντας τον αλγόριθμο παίρνουμε το παραπάνω αποτέλεσμα (4.2.3). Η εικόνα δείχνει τα σημεία που βρέθηκαν τα τοπικά μέγιστα των προβολών του κινούμενου παραθύρου και όπως φαίνεται ακολουθούν τη μορφή της παρτιτούρας. Προφανώς εντοπίζονται και άλλα σημεία πέρα από τα ζητούμενα. Για να καταλήξουμε μόνο στις ζητούμενες γραμμές αρκεί να αφαιρέσουμε με κριτήριο το μήκος τα μικρά τμήματα (πχ **width<(width_window)/2**) και να ενώσουμε τις άκρες που είναι κοντά μεταξύ τους. Το τελευταίο μπορεί να γίνει εύκολα αν αποθηκεύσουμε τα άκρα όλων των γραμμών και για κάθε αριστερό άκρο αναζητούμε στην περιοχή του αν υπάρχει ένα δεξί άκρο που να πληροί κάποια κριτήρια απόστασης.

Εικόνα 4.2.4

Με αυτές τις τροποποιήσεις καταλήγουμε σε ένα πολύ καλύτερο αποτέλεσμα (4.2.4). Με κόκκινο απεικονίζονται τα στοιχεία που προστέθηκαν ενώνοντας τις άκρες που ήταν «κοντά» μεταξύ τους.

Τέλος, αναζητούμε στην εικόνα τα συνδεδεμένα στοιχεία (connected components) που έχουν **width_cc≥width_page*0.75** και έχουμε πια αποθηκεύσει όλες τις γραμμές των πενταγράμμων. Για να επιβεβαιώσουμε το τελικό αποτέλεσμα τις εναποθέτουμε πάνω στην αρχική εικόνα (4.2.5).



Εικόνα 4.2.5

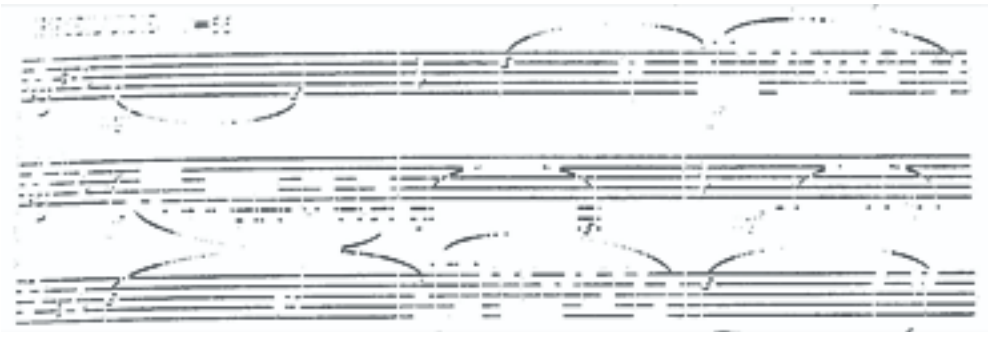
Είναι προφανές ότι έχουμε μεγάλη ακρίβεια στο αποτέλεσμα κάτι που δεν ήταν δυνατό με την προηγούμενη μέθοδο. Τώρα πια μπορούμε να αποθηκεύσουμε σε έναν πίνακα staff_lines διαστάσεων [num of staff_lines, width_page] την τιμή y που έχει κάθε στοιχείο των γραμμών που εντοπίσαμε για κάθε x που κυμαίνεται από 1 έως width_page.

Η ακρίβεια της συγκεκριμένης μεθόδου είναι καλή αλλά γενικά οι μέθοδοι που χρησιμοποιούν sliding window είναι δαπανηρές. Στη συγκεκριμένη περίπτωση βέβαια μπορούμε να ελαττώσουμε επιπλέον τον χρόνο αυξάνοντας το βήμα με το οποίο κινείται το παράθυρο μειώνοντας έτσι όμως τον αριθμό των κορυφών που καταγράφουμε.

4.2.3. Εντοπισμός μέσω των short vertical runs

Ένας εύκολος τρόπος να βρούμε τη θέση των γραμμών του πενταγράμμου είναι να απομονώσουμε τα στοιχεία που ανήκουν σε αυτό. Προφανώς αυτό είναι δύσκολο αλλά μια καλή προσέγγιση είναι να απομονώσουμε όλες τις κάθετες διαδοχές (vertical runs) που έχουν πάχος κοντά στο μέσο πάχος του πενταγράμμου, ότι οποίο έχουμε ήδη εντοπίσει. (staff_thickness).

Αν σαρώσουμε την εικόνα ανά στήλη και σβήσουμε τα runs με πάχος μεγαλύτερο από 1.5*staff_thickness παίρνουμε την παρακάτω εικόνα.(4.2.6).

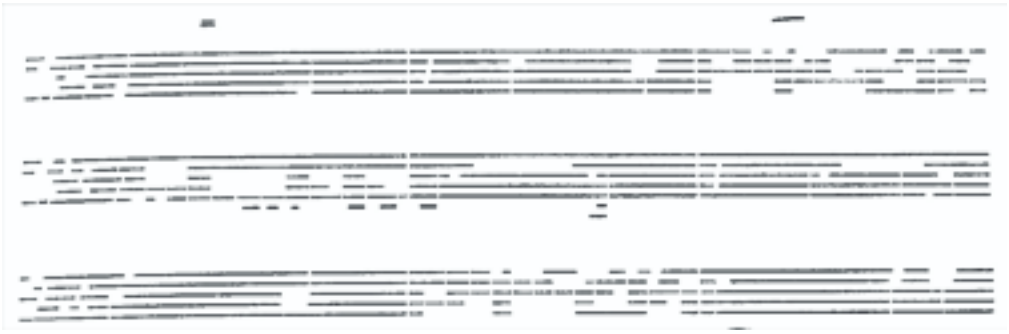


Εικόνα 4.2.6

Για να εντοπίσουμε τώρα ποια από τα παραπάνω τμήματα της εικόνας ανήκουν στο πεντάγραμμο πρέπει να ακολουθήσουμε μια διαδικασία αντίστοιχη με την προηγούμενη. Αρχικά, δηλαδή, να αφαιρέσουμε τα τμήματα που δεν είναι ευθεία καθώς και τα πολύ μικρά και έπειτα να ενώσουμε αυτά που απομένουν.

Θέτουμε ένα ελάχιστο μήκος min_width = staff_distance και επιπλέον το κριτήριο της κλίσης κάθε τμήματος, η οποία υπολογίζεται εύκολα με πολλούς τρόπους. Έτσι μπορούμε να αποκλείσουμε τμήματα από τα slurs (legato και συζεύξεις) τα οποία έχουν διαγώνια κλίση. Εφαρμόζοντας το φιλτράρισμα που περιγράψαμε καταλήγουμε στο παρακάτω αποτέλεσμα (4.2.7).

Αυτό που μένει πια είναι να ενώσουμε με την ίδια λογική που εφαρμόσαμε παραπάνω τα τμήματα που βρίσκονται περίπου στο ίδιο ύψος και απέχουν λίγο μεταξύ τους και τέλος να κρατήσουμε μόνο τις γραμμές που εκτείνονται σε όλο το μήκος της σελίδας. (4.2.8).



Εικόνα 4.2.7

Εικόνα 4.2.8

Το αποτέλεσμα είναι ακριβές αλλά έχουμε απώλεια κάποιων στοιχείων στα αριστερά και δεξιά του πενταγράμμου. Αν μειώσουμε το min_width σε staff_distance/2 παίρνουμε πλήρως το ζητούμενο αποτέλεσμα (4.2.9).

Εικόνα 4.2.9

4.2.4. Εισαγωγικά για τη Μαθηματική Μορφολογία

Πριν περιγράψουμε την επόμενη μέθοδο εντοπισμού με χρήση μορφολογίας θα κάνουμε μια μικρή περιγραφή της θεωρίας των μορφολογικών διαδικασιών μιας και θα τις χρησιμοποιήσουμε και σε επόμενα βήματα.

Η **Μαθηματική Μορφολογία** (ΜΜ) είναι μια θεωρία και τεχνική για την ανάλυση και επεξεργασία γεωμετρικών δομών, με βάση τις θεωρίες συνόλων, δικτύων, την τοπολογία και επιμέρους μαθηματικές διαδικασίες. Η ΜΜ κυρίως χρησιμοποιείται σε ψηφιακές εικόνες και κατ' επέκταση στην Τηλεπισκόπηση και στην Όραση Υπολογιστών.

Η Μαθηματική Μορφολογία πρωτοπαρουσιάστηκε από τους Matheron και Serra το 1964 και αφορούσε αρχικά δυαδικές εικόνες. Αργότερα η θεωρία επεκτάθηκε και σε εικόνες τόνων του γκρί (Grayscale Morphology)

Οι δυο πιο βασικές μορφολογικές διαδικασίες για ψηφιακές εικόνες είναι η **συστολή (erosion)** και η **διαστολή (dilation).**

Η συστολή (erosion) ορίζεται ως:

$$A \ominus B = \{z \in E | B_z \subseteq A\}$$

Όπου **E** ο Ευκλείδειος χώρος ή ένα ακέραιο πλέγμα (πχ ψηφιακή εικόνα) και όπου:

$$B_z = \{b + z | b \in B\}, \forall z \in E$$

Αντίστοιχα η **διαστολή (dilation)** ορίζεται ως:

$$A \oplus B = \{z \in E | B^S{}_z \cap A \neq \emptyset\}$$

Όπου $\mathbf{B}^{\mathbf{S}}$ είναι το συμμετρικό του **B** δηλαδή:

$$B^S = \{x \in E | -x \in B\}$$

Το **B** είναι το δομικό στοιχείο (structural element) που χρησιμοποιούμε το οποίο είναι μια οποιαδήποτε δυαδική εικόνα με μήκος και πλάτος μικρότερα των διαστάσεων του **A**.

Αν συνδυάσουμε τις παραπάνω διαδικασίες παίρνουμε δυο επιπλέον πολύ χρήσιμες: το **άνοιγμα (opening)** και το **κλείσιμο (closing).**

Το **άνοιγμα** είναι η συστολή μιας εικόνας με ένα δομικό στοιχείο ακολουθούμενη από τη διαστολή του αποτελέσματος με το ίδιο δομικό στοιχείο. Δηλαδή:

$$A \circ B = (A \ominus B) \oplus B$$

Αντίστοιχα το **κλείσιμο** είναι η διαστολή μιας εικόνας ακολουθούμενη από συστολή με το ίδιο δομικό στοιχείο.

$$A \bullet B = (A \oplus B) \ominus B$$

Παραδείγματα μορφολογικών διαδικασιών:

Original Image

Structural Element

Erosion

Dilation

Opening

Closing

Βλέπουμε τα αποτελέσματα συστολής, διαστολής, ανοίγματος, κλεισίματος της αρχικής εικόνας με το συγκεκριμένο δομικό στοιχείο. Σαν δομικό στοιχείο μπορούμε να χρησιμοποιήσουμε όποιο δυαδικό σχήμα επιθυμούμε και αυτό μας βοηθάει να αναζητήσουμε συγκεκριμένα χαρακτηριστικά της εικόνας. Το γεγονός ότι μια παρτιτούρα είναι ένα δομημένο έγγραφο με συγκεκριμένες προδιαγραφές, ιδιαίτερα αυστηρές ότι ειδικά μιλάμε για τυπωμένες παρτιτούρες, μας δίνει τη δυνατότητα να γνωρίζουμε συγκεκριμένα χαρακτηριστικά και αντίστοιχα να τα εντοπίσουμε όπως θα δούμε παρακάτω.

Πρακτικά, στην ερμηνεία τους, η **συστολή** είναι η θέση των κέντρων του δομικού στοιχείου όσο αυτό κινείται στο εσωτερικό της εικόνας. Η **διαστολή** είναι η θέση όλων των στοιχείων του δομικού στοιχείου όταν το κέντρο του παίρνει την τιμή κάθε στοιχείου της εικόνας. **Άνοιγμα** είναι κάθε δυνατή τοποθέτηση του στοιχείου στο εσωτερικό της εικόνας και **κλείσιμο** είναι η ένωση δύο μαύρων περιοχών, αν αυτή μπορεί να επιτευχθεί με τοποθέτηση ενός δομικού στοιχείου στο μεταξύ τους κενό.

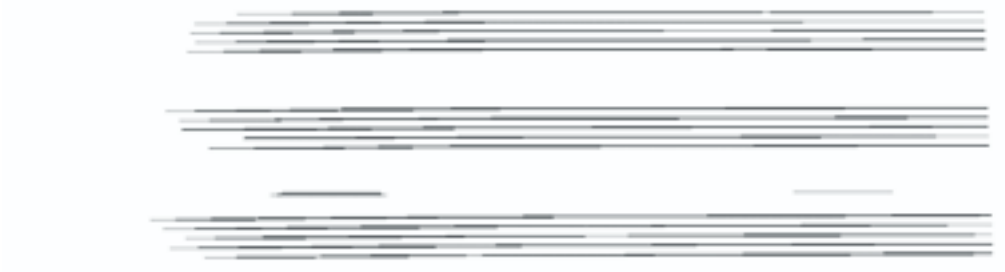
4.2.5. Εντοπισμός με χρήση Μαθηματική Μορφολογίας

Με τα παραπάνω δεδομένα και χρήση κατάλληλων δομικών στοιχείων μπορούμε να εντοπίσουμε εύκολα τις γραμμές των πενταγράμμων σε μια παρτιτούρα. Επιπλέον οι μορφολογικές διαδικασίες κοστίζουν λίγο σε πόρους και είναι πολύ εύκολα παραμετροποιήσιμες αλλά πάλι υπάρχουν περιορισμοί ως προς τον βαθμό που καμπυλώνει το πεντάγραμμο και μάλιστα πιο περιοριστικοί όπως θα δούμε παρακάτω.

Γνωρίζοντας ότι τα στοιχεία του πενταγράμμου σχηματίζουν μεγάλες οριζόντιες γραμμές προφανώς θα χρησιμοποιήσουμε σαν δομικό στοιχείο μια οριζόντια γραμμή μήκους **(width_page)/5** και θα διεξάγουμε άνοιγμα της σελίδας (4.2.10).

Εικόνα 4.2.10

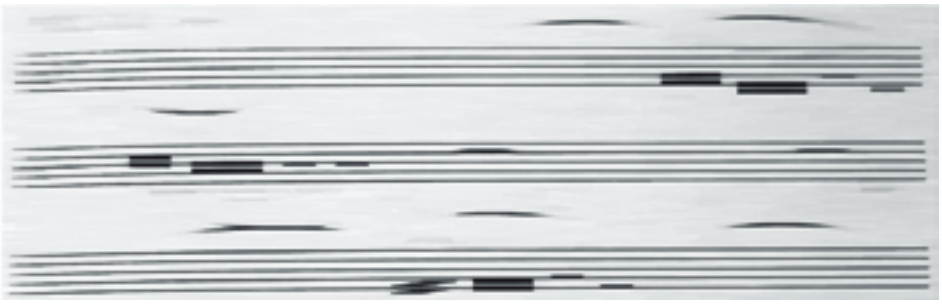
Το άνοιγμα εντοπίζει σωστά τα τμήματα που παρουσιάζουν ελάχιστη, έως και μηδενική, καμπυλότητα αλλά όχι το αριστερό τμήμα της σελίδας όπου εμφανίζεται το πρόβλημα. Αυτό συμβαίνει γιατί το μήκος του δομικό στοιχείου είναι μεγάλο και όταν αυξάνεται η καμπυλότητα δεν υπάρχουν οριζόντιες διαδοχές μαύρων στοιχείων με μήκος **(width_page)/5.** Μικραίνοντας το μήκος σε **(width_page)/10** παίρνουμε το παρακάτω αποτέλεσμα (4.2.11).



Εικόνα 4.2.11

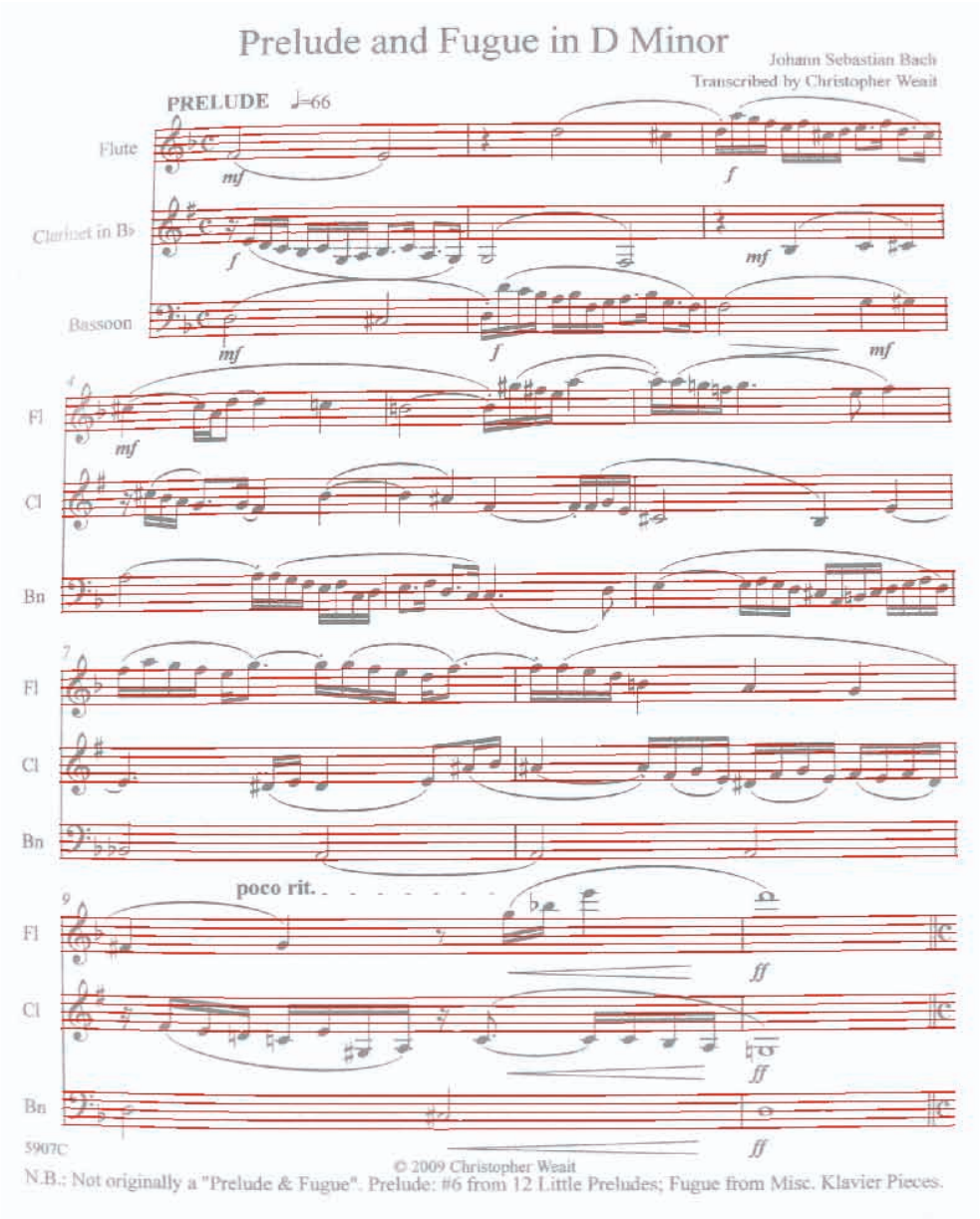
Το αποτέλεσμα αρχίζει και προσεγγίζει τα άκρα αλλά ακόμα υπάρχει σοβαρή απώλεια. Επιπλέον να σημειώσουμε ότι απώλεια μπορεί να εμφανίζεται όχι μόνο λόγω καμπυλότητας αλλά και λόγω σπασμένων τμημάτων του πενταγράμμου ως αποτέλεσμα κακής δυαδικοποίησης ή κακής εκτύπωσης. Άρα για να ξεπεράσουμε αυτά τα προβλήματα μπορούμε να διεξάγουμε μορφολογικές διαδικασίες όχι πια στον δυαδικό αλλά στις αποχρώσεις του γκρί (grayscale).

Εφαρμόζοντας μορφολογικό άνοιγμα στην grayscale εικόνα, με μικρότερο μήκος του δομικού στοιχείου, εντοπίζουμε πια όλα το τμήματα του πενταγράμμου αλλά και κάποια επιπλέον τμήματα (4.2.12). Εφαρμόζουμε πάλι προσαρμοστική κατωφλίωση για να πάρουμε την δυαδική εικόνα χωρίς απώλειες. Για να προχωρήσουμε και να απομονώσουμε τις γραμμές του πενταγράμμου ακολουθούμε ξανά την διαδικασία αφαίρεσης των τμημάτων με μεγάλο πάχος και σύνδεση των ευθείων τμημάτων που είναι κοντά μεταξύ τους. Τέλος φιλτράρουμε και επιλέγουμε μόνο τις γραμμές με μεγάλη έκταση στο πλάτος της εικόνας (4.2.13).



Εικόνα 4.2.12

Εικόνα 4.2.13



Εικόνα 4.2.14

Εφαρμόζουμε τον αλγόριθμο σε όλη την εικόνα και παίρνουμε το τελικό αποτέλεσμα (4.2.14) που έχει μεγάλη ακρίβεια καθώς καλύπτει τις γραμμές του πενταγράμμου πλήρως από τα αριστερά προς τα δεξιά και ακολουθεί τέλεια την καμπυλότητα της σελίδας.

Όλες οι παραπάνω μέθοδοι έχουν μια κοινή λογική. Τον εντοπισμό κάποιων τμημάτων της παρτιτούρας που έχουν κάποιες ιδιότητες (μεγάλος μήκος, μικρό πάχος) και η σύνδεση μεταξύ τους έτσι ώστε αν σχηματίζουν μια μεγάλη ευθεία ορίζονται ως γραμμές του πενταγράμμου. Το πιο κρίσιμο τμήματα αυτής της διαδικασίας δεν είναι ο αρχικός εντοπισμός των σημείων αλλά η διαδικασία της σύνδεσης.

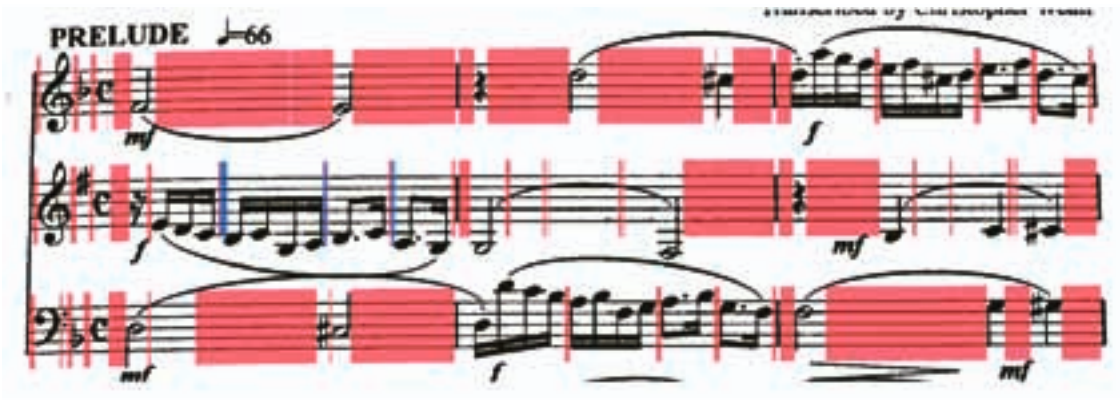
Σε περιπτώσεις φυσιολογικές όπως είναι η παρτιτούρα που εξετάζουμε η σύνδεση ολοκληρώνεται πάντα με επιτυχία, αλλά σε περιπτώσεις μεγάλη φθοράς, μεγάλης καμπυλότητας ή πυκνής γραφής ο αλγόριθμος της σύνδεσης πρέπει να βελτιωθεί περισσότερο από αυτόν που περιγράψαμε. Ωστόσο στο εύρος των περιπτώσεων που εξετάζουμε σε αυτή την εργασία οι παραπάνω μέθοδοι είναι πλήρως λειτουργικές με γρηγορότερη και πιο αποτελεσματική από αυτές την χρήση μορφολογίας.

4.2.6. Εντοπισμός με τη χρήση πολυωνυμικής παρεμβολής (polynomial interpolation)

Εκτός από τον εντοπισμό όλων των στοιχείων που βρίσκονται πάνω στο πεντάγραμμο, όπως κάναμε στις προηγούμενες μεθόδους, είναι δυνατόν να εντοπίσουμε μικρό αριθμό στοιχείων που ανήκουν στο πεντάγραμμο και να τα ενώσουμε μεταξύ τους με χρήση πολυωνυμικής παρεμβολής. Δεδομένου ότι η καμπυλότητα είναι εντός φυσιολογικών ορίων, ο βαθμός του πολυωνύμου που χρειάζεται για να ακολουθήσει την μορφή της σελίδας είναι το πολύ 3ος αν και μπορεί να επεκταθεί σε ότι τιμές θέλουμε.

Για να υλοποιήσουμε λοιπόν τη συγκεκριμένη μέθοδο χρειαζόμαστε την ακριβή θέση σημείων του πενταγράμμου σε όλο το πλάτος της εικόνας. Για να είναι καλύτερο το αποτέλεσμα τα σημεία πρέπει να είναι περίπου 10 και να είναι κάπως ομοιόμορφα κατανεμημένα κατά μήκος της εικόνας. Αν εντοπίσουμε τα σημεία μπορούμε εύκολα να εξάγουμε τα ζητούμενα πολυώνυμα.

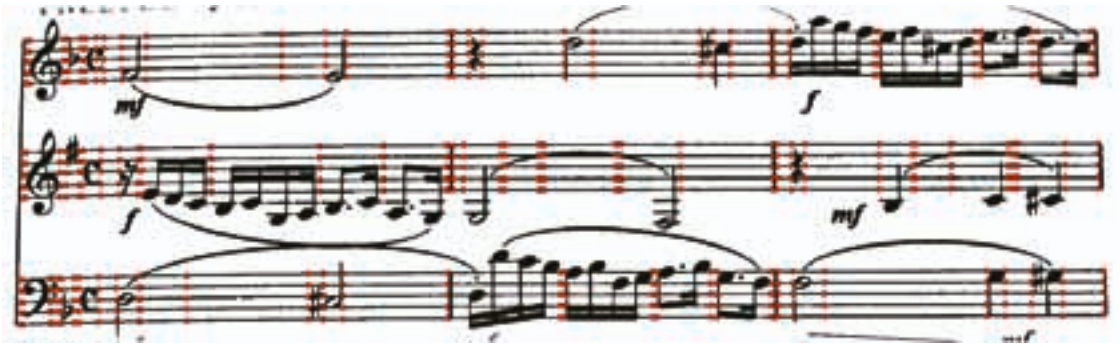
Ο εντοπισμός των σημείων της εικόνας που αποτελούνται μόνο από τις γραμμές του πενταγράμμου γίνεται με την εξής διαδικασία. Σαρώνουμε όλη την εικόνα με χρήση μιας στήλης μήκους 8.5*staff_distance. Προϋπόθεση για να περιέχει αυτή η στήλη μόνο τμήματα του πενταγράμμου είναι περιέχει στο εσωτερικό της ακριβώς πέντε λευκές περιοχές με μήκος staff_distance και δύο λευκές περιοχές στα άκρα της με μήκος μεγαλύτερο από 1.5*staff_distance. Ο λόγος ύπαρξης του τελευταίου κριτηρίου είναι για να μην συμπεριληφθούν περιοχές όπου οι νότες γράφονται εκτός πενταγράμμου. Το αποτέλεσμα φαίνεται καλύτερα στην παρακάτω εικόνα (4.2.15).



Εικόνα 4.2.15

Οι κόκκινες περιοχές είναι οι περιοχές που θα μας δώσουν τα σημεία που σίγουρα ανήκουν στο πεντάγραμμο. Αν δεν συμπεριλαμβάναμε το τελευταίο κριτήριο θα παίρναμε σαν αποτέλεσμα και τις μπλε περιοχές γιατί τηρούν το κριτήριο των αποστάσεων. Ωστόσο αυτές οι περιοχές είναι ανεπιθύμητες γιατί περιλαμβάνουν γραμμές πέρα από αυτές του πενταγράμμου.

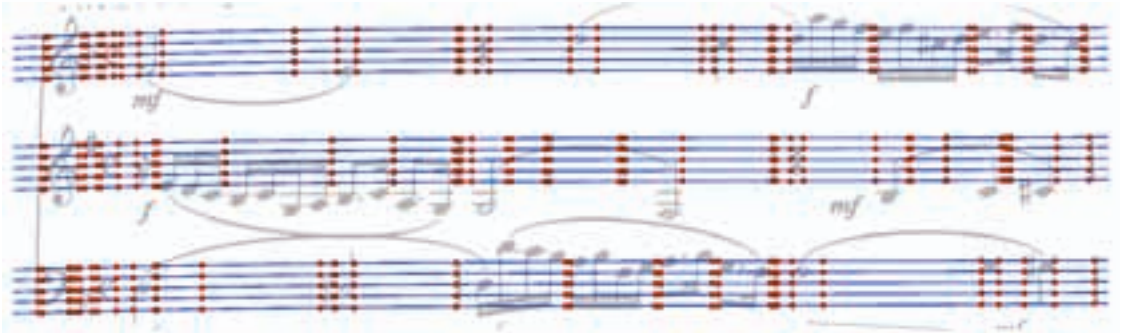
Μπορούμε πια να εξάγουμε όσα σημεία θέλουμε από τις συγκεκριμένες περιοχές μιας και είναι δεδομένο ότι όλα ανήκουν στο πεντάγραμμο. Επιλέξαμε σημεία από τα άκρα αυτών των περιοχών προκειμένου να καλύψουμε όλο το εύρος αλλά και τα άκρα του πενταγράμμου μιας και εκεί εμφανίζεται η καμπυλότητα. Έτσι κρατάμε τα παρακάτω σημεία (4.2.16).



Εικόνα 4.2.16

Τώρα πια το μόνο που μένει είναι να υπολογίσουμε τα πολυώνυμα που περιλαμβάνουν όσο το δυνατόν πιο κοντά κάθε σειρά σημείων με τη μέθοδο της πολυωνυμικής παρεμβολής. Το πλήθος των σημείων είναι αρκετό και αντιπροσωπευτικό ενώ ο βαθμός του πολυωνύμου συνήθως δεν χρειάζεται να είναι μεγαλύτερος από 3ος.

Εφαρμόζοντας την πολυωνυμική παρεμβολή παίρνουμε το παρακάτω αποτέλεσμα (4.2.17).



Εικόνα 4.2.17

Το αποτέλεσμα είναι αλάνθαστο και αυτή η μέθοδος είναι από τις πιο αποδοτικές. Το μοναδικό κριτήριο για την αποτελεσματικότητά της είναι η αντιπροσωπευτικότητα των σημείων που εντοπίζουμε και κυρίως η ύπαρξη σημείων στα άκρα του πενταγράμμου. Σε πιο ακραίες περιπτώσεις είναι πιθανό να «χαθούν» τα σημεία που βρίσκονται αριστερά των κλειδιών αν υπάρχει μεγάλη παραμόρφωση, γεγονός που μπορεί να οδηγήσει σε λαθεμένο υπολογισμό των πολυωνύμων.

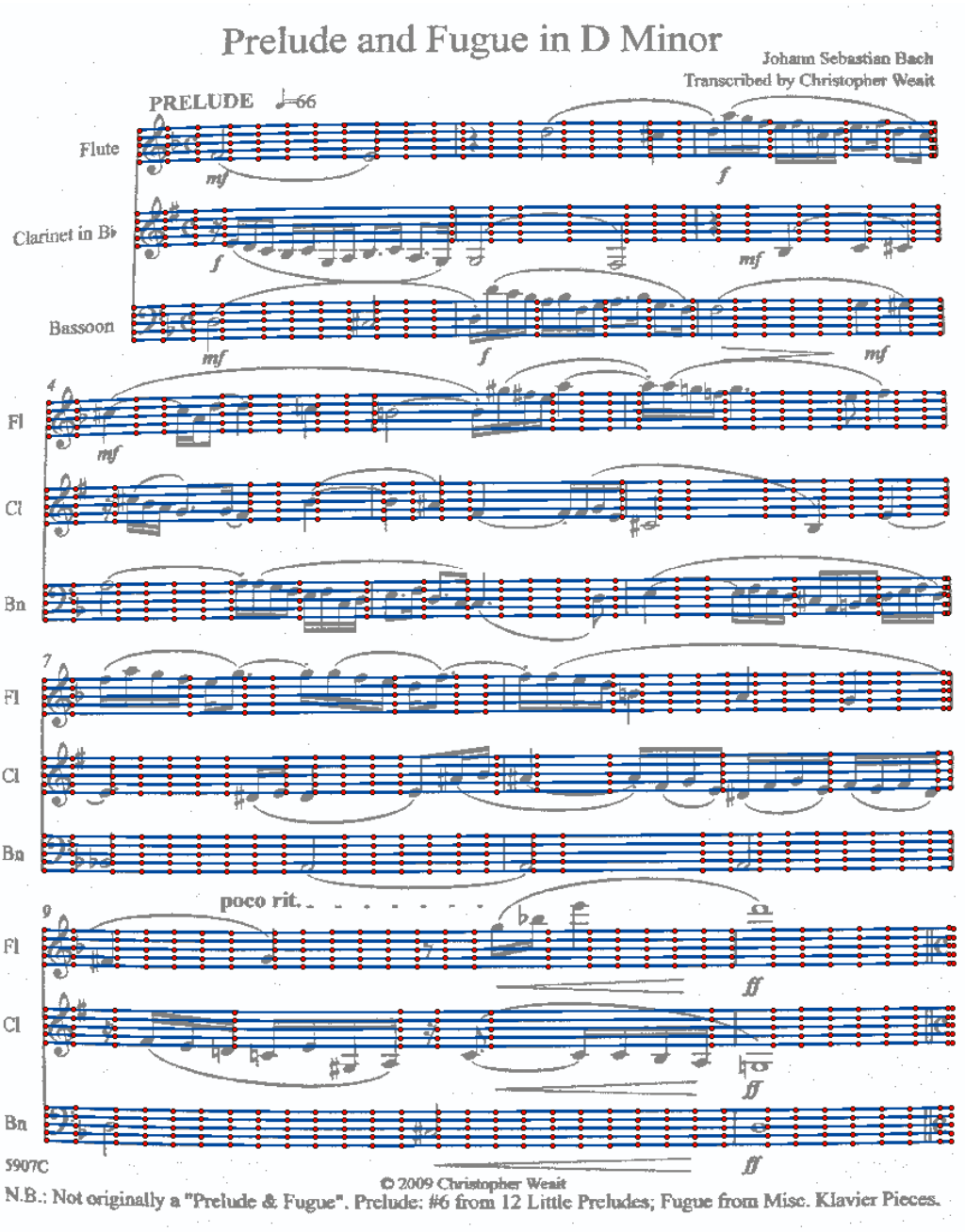
Επιπλέον, η συγκεκριμένη μέθοδος απαιτεί πόρους στο κομμάτι του σαρώματος για την εύρεση των ζητούμενων περιοχών. Ωστόσο, αν την εφαρμόσουμε σε μια μικρότερη εκδοχή της εικόνας μπορούμε να ρίξουμε σημαντικά το χρόνο εκτέλεσης χωρίς απώλεια ακρίβειας.

Κλείνοντας το κεφάλαιο εντοπισμού του πενταγράμμου βλέπουμε ότι υπάρχουν αρκετές

μέθοδοι εντοπισμού, η κάθε μία με περιορισμούς και δυνατότητες. Όλες περιορίζονται, σε διαφορετικό βαθμό βέβαια, από τα χαρακτηριστικά της εικόνας και το βαθμό της παραμόρφωσης είτε σε επίπεδο καμπυλότητας, ή σε επίπεδο φθοράς.

Η τελευταία μέθοδος είναι ιδιαίτερα αποτελεσματική και αυτή θα χρησιμοποιήσουμε στα επόμενα βήματα αλλά σε μια εμπορική εφαρμογή μπορεί να μην ήταν προτιμητέα λόγω απαιτήσεων σε πόρους. Για τη συγκεκριμένη εργασία η ακρίβειά της θα μας βοηθήσει να πετύχουμε περαιτέρω ακρίβεια στα επόμενα βήματα της αναγνώρισης.

Τέλος, παραθέτουμε την εφαρμογή της μεθόδου σε όλη την εικόνα (4.2.18).



Εικόνα 4.2.18

5
Μέθοδοι κατάτμησης

Προχωράμε στο επόμενο κρίσιμο στάδιο του συστήματος OMR το οποίο είναι η κατάτμηση της εικόνας. Από αυτό το σημείο το προχώρημα της διαδικασίας χρειάζεται αυτή τη στιγμή μια διαδικασία κατάτμησης (segmentation). Χρειάζεται όλα τα σύμβολα της εικόνας να γίνουν διακριτά μεταξύ τους προκειμένου να αξιολογηθούν. Η πιο συχνή τεχνική για να ξεχωρίσουν τα σύμβολα είναι η αφαίρεση του πενταγράμμου που θα μελετήσουμε εκτενώς σε αυτό το κεφάλαιο.

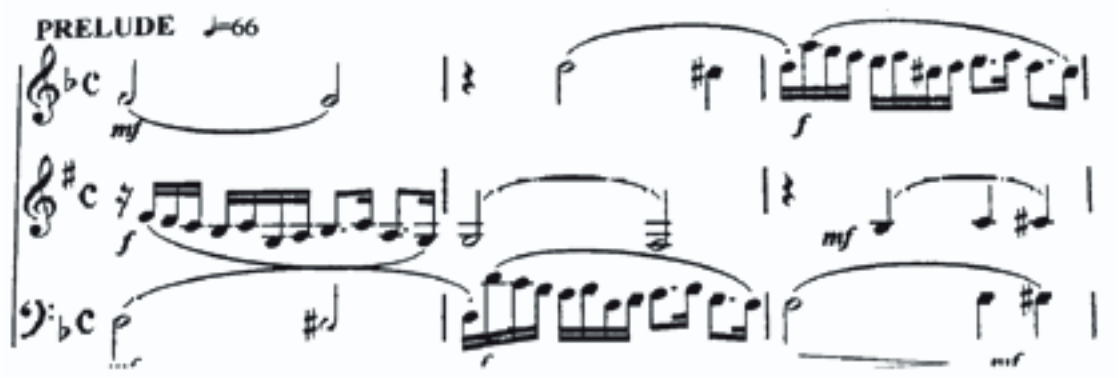
Επειδή η αφαίρεση εμπεριέχει σφάλματα και απώλειες όπως θα δούμε παρακάτω, εναλλακτικά υπάρχει η δυνατότητα κατάτμησης με βάση τις προβολές, κάτι το οποίο θα μελετήσουμε περιληπτικά για να εντοπίσουμε τα πλεονεκτήματα και της δυσκολίες που δημιουργεί.

5.1. Κατάτμηση με αφαίρεση του πενταγράμμου

Είναι η πιο συνηθισμένη μέθοδος κατάτμησης και χρησιμοποιείται ευρέως στα εμπορικά προγράμματα και άλλες εφαρμογές και αυτό γιατί αν υλοποιηθεί με ακρίβεια ανάγει μετά το πρόβλημα της αναγνώρισης σε πεπερασμένο πρόβλημα ταξινόμησης, διαδικασία που έχει αναπτυχθεί πολύ στο κομμάτι του Image Analysis και ιδιαίτερα στην οπτική αναγνώριση χαρακτήρων (OCR).

Αν και αρχικά φαίνεται απλό πρόβλημα, τελικά είναι το πιο κρίσιμο και πιο πιθανό να δημιουργήσει σφάλματα στα επόμενα βήματα. Αυτό συμβαίνει γιατί υπάρχει μια ιδιαιτερότητα σε αυτή τη διαδικασία που δεν έχει ξεπεραστεί με μεγάλη ακρίβεια ακόμη.

Αρχική υπόθεση είναι ότι το πεντάγραμμο αποτελείται από όλα τα μικρά κάθετα black runs τα οποία είναι «πολύ κοντά» ή απλά «πάνω» στις γραμμές που έχουμε εντοπίσει. Εφαρμόζοντας αυτή την υπόθεση θα συνειδητοποιήσουμε ότι αφαιρούνται τμήματα τα οποία αν και πληρούν αυτό το κριτήριο δεν ανήκουν στο πεντάγραμμο αλλά σε κάποιο σύμβολο. Η παρακάτω εικόνα είναι ενδεικτική το προβλήματος που μελετάμε (5.1.1).



Εικόνα 5.1.1

Ήδη φαίνεται το πρόβλημα των σπασμένων συμβόλων αλλά για να γίνει πιο σαφές θα εστιάσουμε στα πιο συνήθη και κρίσιμα σημεία που συναντάται το συγκεκριμένο πρόβλημα.

Εικόνα 5.1.2

Εικόνα 5.1.3

Εικόνα 5.1.4

Στις εικόνες (5.1.2-4) φαίνονται τα σφάλματα που προκαλούνται κατά την αφαίρεση του πενταγράμμου. Συγκεκριμένα όσα σύμβολα έχουν τμήματα τα οποία δεν τέμνουν το πενταγράμμου αλλά το ακουμπάνε εφαπτομενικά καταλήγουν να σπάνε διότι ο αλγόριθμος που χρησιμοποιήσαμε δεν μπορεί να τα ξεχωρίσει. Δεν γνωρίζει τα συμφραζόμενα που γνωρίζουμε εμείς για να το θεωρήσει τμήμα κάποιου άλλου συμβόλου μιας και αντιμετωπίζει κάθε τμήμα ξεχωριστά.

Τα σφάλματα προκύπτουν συνήθως στα κλειδιά του σολ και του φα, στο σύμβολο χρόνου c (common time), στις μισές και ολόκληρες νότες και στις συζεύξεις. Έτσι όταν φτάνουμε στο στάδιο να αξιολογήσουμε τα σύμβολα ο υπολογιστής βλέπει το κλειδί του φα σαν δύο σύμβολα κανένα εκ των οποίο δεν πληροί τις ιδιότητες των συμβόλων αυτής της κλάσης.

Ανακύπτει λοιπόν το πρόβλημα των κριτηρίων σύμφωνα με τα οποία θα ορίζεται αν ένα από τα τμήματα που εντοπίσαμε είναι κομμάτι του πενταγράμμου.

Το πρόβλημα δεν έχει λυθεί αποτελεσματικά μέχρι σήμερα γι'αυτό και πολλά συστήματα εστιάζουν στην ανακατασκευή των σπασμένων συμβόλων λαμβάνοντας υπόψη το δεδομένο σφάλμα. Θα εξετάσουμε και τις δύο μεθόδους ξεκινώντας από την πιο αποτελεσματική κατά την άποψη μας μέχρι τώρα η οποία είναι η μέθοδος των Nicholas P. Carter και Richard A. Bacon.

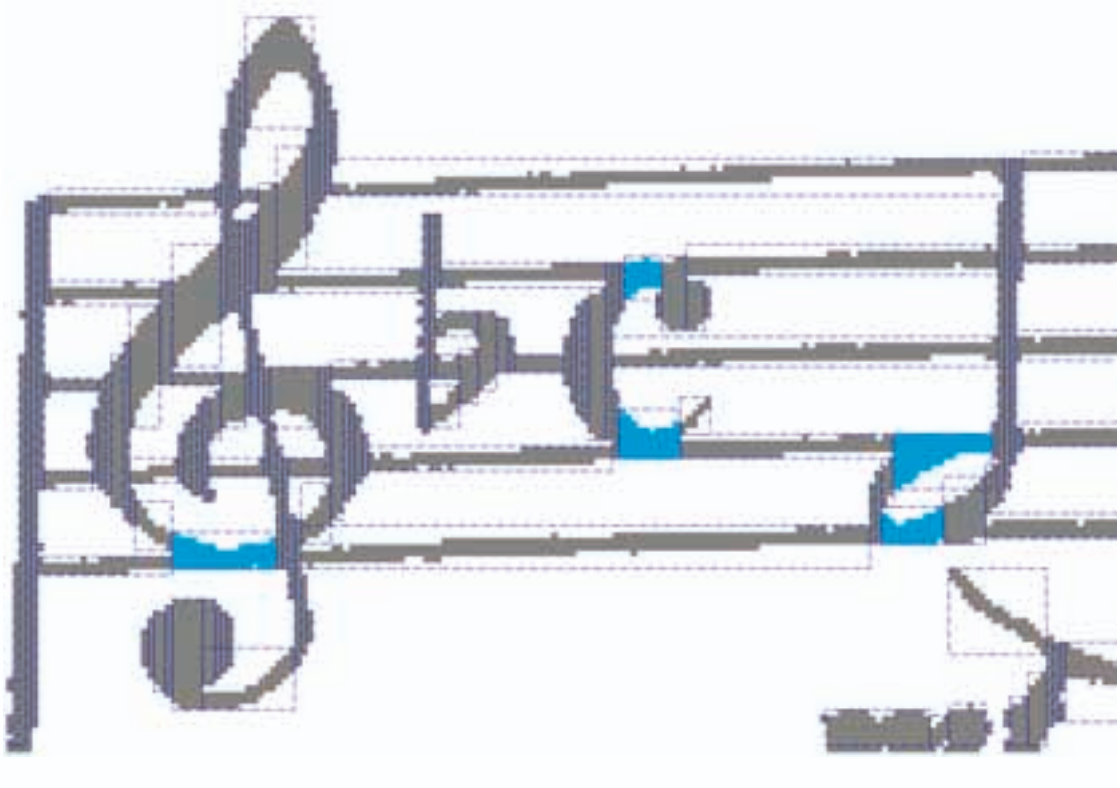
5.1.1. Η Μέθοδος των Carter και Bacon

Στη δημοσίευση με τίτλο "Automatic Recognition of Printed Music" οι Carter και Bacon αν και ασχολήθηκαν με ειδικές περιπτώσεις παρτιτούρας (χωρίς μισές ή ολόκληρες νότες) προσέγγισαν πολύ αποτελεσματικά τον ακριβή προσδιορισμό του πενταγράμμου και μάλιστα χωρίς να έχουν εντοπίσει από πριν τη θέση του. Αντίθετα χρησιμοποίησαν τα στοιχεία που εντόπισαν ως στοιχεία του πενταγράμμου και από εκεί εξήγαγαν τη θέση των γραμμών.

Εμείς έχουμε πια την πολυτέλεια να γνωρίζουμε τη θέση του πενταγράμμου αλλά μένει να προσδιορίσουμε επακριβώς ποια στοιχεία (pixels) ανήκουν σε αυτό.

Πρέπει δηλαδή να τροφοδοτήσουμε τον αλγόριθμο με τα συμφραζόμενα της εικόνας. Άλλος τρόπος που έχει επιχειρηθεί, παρόμοιος στη λογική του, είναι η μέθοδος του μήκους χορδής η οποίο είναι ιδιαίτερα δαπανηρή. Η συγκεκριμένη μέθοδος περιστρέφει με κέντρο κάθε υποψήφιο στοιχείο μια ευθεία και αποθηκεύει τα μήκη. Αν η ευθεία σημειώνει μέγιστο μόνο κοντά στις 0° τότε ανήκει στο πεντάγραμμο. Η συγκεκριμένη μέθοδος είναι αποτελεσματική μόνο σε μεγάλες αναλύσεις αλλά παράλληλα είναι και πολύ δαπανηρή γι'αυτό και δεν χρησιμοποιείται συχνά.

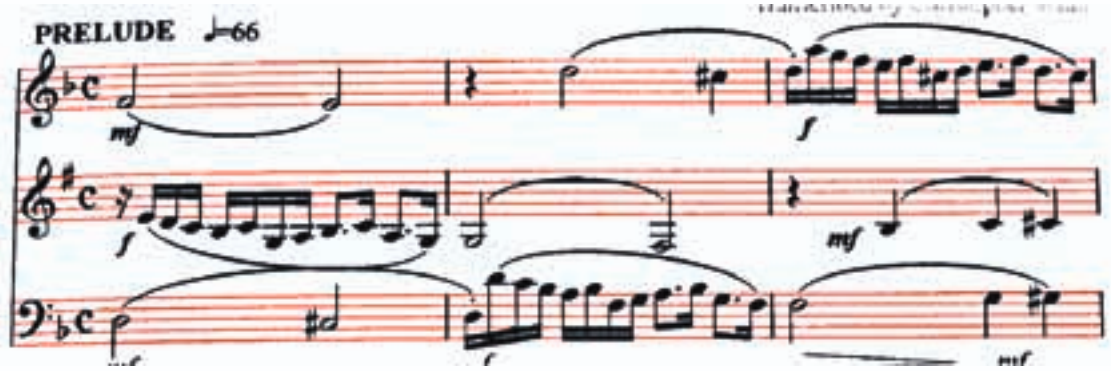
Ο τρόπος με τον οποίο προσέθεσαν ένα κρίσιμο κριτήριο για τον προσδιορισμό του πενταγράμμου οι Carter και Bacon έγινε λαμβάνοντας υπόψη και τα γύρω στοιχεία της εικόνας. Συγκεκριμένα δεν απομονώνουμε όλα τα μικρά κάθετα black runs αλλά ομαδοποιούμε όλα τα vertical runs με κριτήριο το μήκος τους. Με αυτόν τον τρόπο αν υπάρχουν γειτονικά vertical runs με μικρή διαφορά μήκους (όπως ένα εφαπτόμενο στοιχείο) ομαδοποιούνται και αξιολογούνται ως τέτοια. Η λογική του αλγορίθμου φαίνεται καθαρά στην παρακάτω εικόνα (5.1.5).



Εικόνα 5.1.5

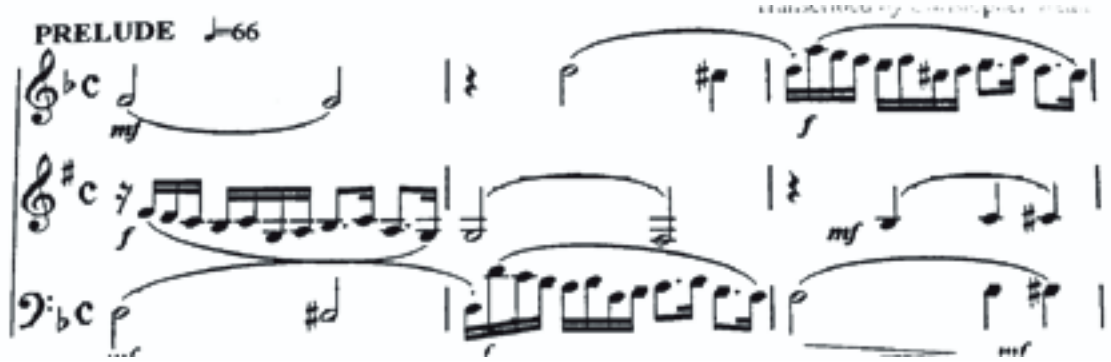
Στην εικόνα φαίνονται οι ομαδοποιήσεις που προέκυψαν από αυτή τη μέθοδο. Όλα τα vertical black runs έχουν ομαδοποιηθεί με κριτήριο τα μήκη τους. Έτσι όσα έχουν παρόμοια μήκη και γειτονεύουν εντάσσονται στην ίδια ομάδα. Τα τμήματα με γαλάζιο χρώμα είναι τα κρίσιμα στοιχεία που χάνονται αν δεν εφαρμόσουν τη συγκεκριμένη μέθοδο. Αυτή τη φορά όμως τα runs που αφαιρέθηκαν προηγουμένων έχουν ομαδοποιηθεί με άλλα μεγαλύτερα και με αυτόν τον τρόπο μπορούμε να τα κρατήσουμε εφαρμόζοντας κάποια κριτήρια όπως πχ μέσο κάθετο μήκος κάθε τμήματος. Έτσι τα τμήματα που ανήκουν στο πεντάγραμμο παρουσιάζουν μικρό μέσο μήκος ενώ για παράδειγμα τα χρωματισμένα μεγαλύτερο.

Εφαρμόζουμε λοιπόν τον αλγόριθμο στην εικόνα μας και παίρνουμε το παρακάτω αποτέλεσμα (5.1.6).



Εικόνα 5.1.6

Με κόκκινο φαίνονται τα στοιχεία που επιλέχθηκαν ως στοιχεία του πενταγράμμου. Βλέπουμε ότι τώρα πια τα κλειδιά του σολ και του φα είναι ανέπαφα, όπως επίσης και οι μισές νότες και οι συζεύξεις. Αφαιρώντας τώρα το πεντάγραμμο το αποτέλεσμα είναι πιο εμφανές (5.1.7).

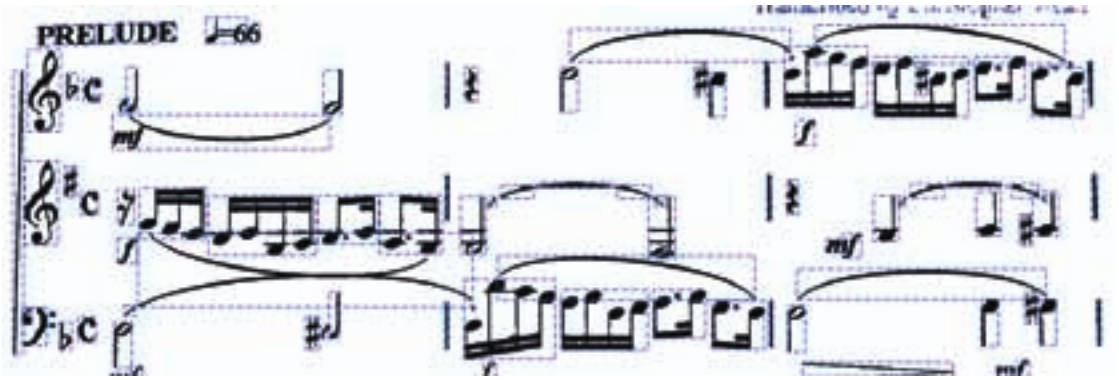


Εικόνα 5.1.7

Στο παράδειγμα μας ο αλγόριθμος λειτουργεί χωρίς σφάλματα και επιτυγχάνει την καλύτερη δυνατή αφαίρεση του πενταγράμμου. Βέβαια υπάρχουν περιπτώσεις όπου γίνεται λάθος αξιολόγηση κάποιων τμημάτων και δεν έχουμε την επιθυμητή κατάτμηση (segmentation). Πρέπει δηλαδή να υπάρχουν επιπλέον φίλτρα στην μετέπειτα διαδικασία προκειμένου να απορροφήσουν ενδεχόμενα σφάλματα. Ωστόσο η εφαρμογή αυτού του αλγορίθμου μας εξασφαλίζει μια πρώτη πολύ αποτελεσματική κατάτμηση που αφήνει τα σύμβολα ανέπαφα προκειμένου να αξιολογηθούν με μεγαλύτερη ακρίβεια μετέπειτα.

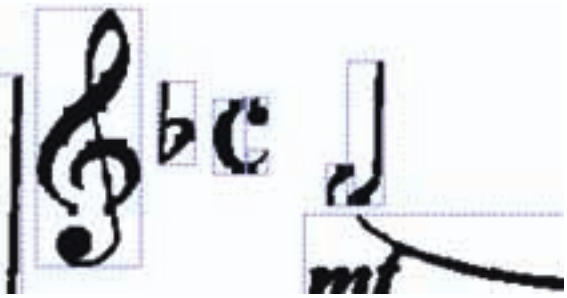
5.1.2. Αφαίρεση πενταγράμμου και ανακατασκευή συμβόλων

Η δεύτερη συνηθισμένη μέθοδος αφαίρεσης του πενταγράμμου βασίζεται στην ανακατασκευή των συμβόλων. Θεωρεί δεδομένο ότι η αφαίρεση του πενταγράμμου θα οδηγήσει σε σπασμένα σύμβολα και εφαρμόζει κριτήρια γειτνίασης για να ελέγξει αν δυο σύμβολα είναι αυτόνομα ή είναι σπασμένα τμήματα του ίδιου συμβόλου. Ο τρόπος που ελέγχεται η γειτνίαση γίνεται συνήθως με χρήση των συνοριακών κουτιών (bounding boxes), δηλαδή των ορθογώνιων παραλληλογράμμων που περιβάλλουν κάθε στοιχείο. Στο παράδειγμα μας αν εφαρμόσουμε την αρχική αφαίρεση του πενταγράμμου βλέπουμε τα παρακάτω bounding boxes (5.1.8).

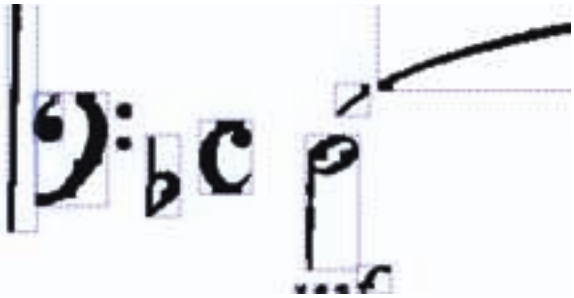


Εικόνα 5.1.8

Βλέπουμε τα παραλληλόγραμμα που περιβάλλουν κάθε σύμβολο. Προφανώς τα σύμβολα που έχουν διασπαστεί από την αφαίρεση του πενταγράμμου έχουν αντίστοιχα παραλληλόγραμμα με πλευρές είτε πολύ κοντά μεταξύ τους, ή ακόμα και το ένα σύμβολο εντάσσεται πλήρως μέσα στο bounding box του άλλου όπως θα βλέπουμε παρακάτω.(5.1.9).



Εικόνα 5.1.9



Εικόνα 5.1.10

Βλέπουμε ότι τα σύμβολα που έχουν σπάσει, έχουν τις γωνίες των bounding boxes τους σε μικρή απόσταση μεταξύ τους. Μπορεί να εφαρμοστεί ένα κριτήριο αναζήτησης κοντινών γωνιών και να ενωθούν τελικά τα στοιχεία που πληρούν αυτό το κριτήριο. Να σημειώσουμε ότι εδώ οι περιπτώσεις είναι πάρα πολλές και είναι δύσκολο ο αλγόριθμος να δουλέψει με πλήρη επιτυχία. Η βελτιστοποίησή του θα απαιτούσε καταγραφή όλων των περιπτώσεων των σπασμένων συμβόλων και η εφαρμογή διαφορετικών κριτηρίων σχετικά με την απόσταση μεταξύ των bounding boxes. Στην εργασία δεν χρησιμοποιούμε καμία από τις δύο μεθόδους αν και η μέθοδος Carter δίνει συχνά καλά αποτελέσματα. Ωστόσο για τη μέγιστη αποτελεσματικότητα θα εφαρμόσουμε αρχικά μια απλή αφαίρεση πενταγράμμου, θα αφαιρέσουμε πρώτα στοιχεία που εντοπίζονται με μεγάλη βεβαιότητα και στη συνέχεια θα ενώσουμε τα υπόλοιπα εφαρμόζοντας κάποια κριτήρια που θα δούμε στο στάδιο της ταξινόμησης.

Υπάρχουν επίσης εφαρμογές που κατά την ταξινόμηση (classification) χρησιμοποιούν και κλάσεις σπασμένων συμβόλων προκειμένου να αναγνωρίζουν και τα επιμέρους τμήματα και όχι μόνο τα σύμβολα σαν σύνολο.

Κλείνοντας το κεφάλαιο αφαίρεσης του πενταγράμμου, βλέπουμε ότι η αφαίρεση δεν μπορεί να ολοκληρωθεί εγγυημένα με πλήρη αποτελεσματικότητα γι' αυτό αναπτύχθηκαν και μέθοδοι κατάτμησης χωρίς αφαίρεση του πενταγράμμου που θα δούμε στη συνέχεια.

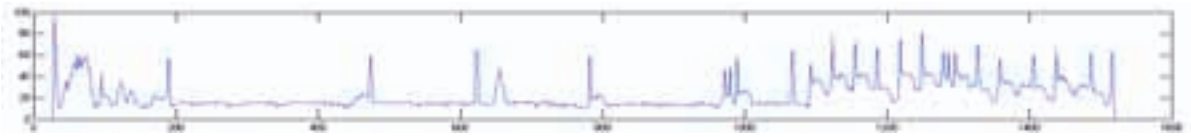
5.2. Κατάτμηση με χρήση προβολών

Εναλλακτικός τρόπος για την κατάτμηση της εικόνας μιας παρτιτούρας και την ταξινόμηση των επιμέρους συμβόλων είναι η χρήση των κάθετων προβολών. Παρατηρώντας τις διακυμάνσεις των προβολών μπορούμε με χρήση κατωφλίων να ορίσουμε τις περιοχές που αποτελούνται από σύμβολα και αυτές που αποτελούνται από γραμμές του πενταγράμμου. Με αυτόν τον τρόπο δεν αφαιρούμε κανένα στοιχείο και άρα χρησιμοποιούμε την εικόνα ανέπαφη χωρίς κίνδυνο σφάλματος. Η διαδικασία αυτή ωστόσο δεν είναι τόσο εύκολη σε σύνθετες παρτιτούρες μιας και το πλήθος των στοιχείων, πέρα από τα βασικά, μπορούν να αλλοιώσουν σημαντικά το αποτέλεσμα. Αν και δεν θα χρησιμοποιήσουμε αυτή τη μέθοδο θα παραθέσουμε τον τρόπο λειτουργίας της.

Αρχικά, η μέθοδος εφαρμόζεται σε κάθε ένα πεντάγραμμο ξεχωριστά προκειμένου να μην συναθροιστούν οι προβολές των άλλων πενταγράμμων. Δεδομένου ότι γνωρίζουμε με ακρίβεια τις θέσεις των πενταγράμμου ο διαχωρισμός είναι εύκολος. Έπειτα αφαιρούμε όλα τα στοιχεία που δεν είναι συνδεδεμένα στο πεντάγραμμο και έχουμε έτοιμη την εικόνα για την επεξεργασία των προβολών (5.2.1).

Εικόνα 5.2.1

Έπειτα υπολογίζουμε την κάθετη προβολή της εικόνας (5.2.2).

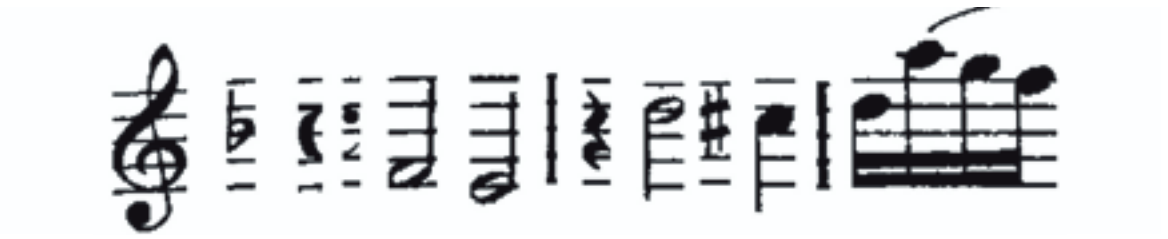


Εικόνα 5.2.2

Στη συνέχεια αν θέσουμε ένα κατώφλι με τιμή περίπου 5*staff_thickness κρατάμε μόνο τις περιοχές που έχουν πληροφορία πέρα από τις γραμμές του πενταγράμμου (5.2.3).

Εικόνα 5.2.3

Βλέπουμε ότι το σύμβολο c ακόμα και με αυτή τη μέθοδο σπάει μιας και στο συγκεκριμένο σημείο τα δύο τμήματα ταυτίζονται με τις γραμμές του πενταγράμμου. Άρα σε αυτή την περίπτωση έχουμε σφάλμα. Ωστόσο τα υπόλοιπα σύμβολα έχουν χωριστεί σωστά.



Παραπάνω βλέπουμε το αποτέλεσμα της κατάτμησης. Κάθε εικόνα στη συνέχεια αξιολογείται συνήθως με κριτήριο τα χαρακτηριστικά των κάθετων και οριζόντιων προβολών. Αυτά αποθηκεύονται και έπειτα ταξινομούνται με χρήση διαφόρων μεθόδων που θα δούμε παρακάτω.

Σαν μέθοδος κατάτμησης μπορούμε να παρατηρήσουμε ότι επιλύεται το πρόβλημα των σπασμένων μισών νοτών, ωστόσο παραμένουν άλλα προβλήματα όπως είδαμε με το σύμβολο c. Επίσης τα σύμβολα πια έχουν μαζί τους και τα τμήματα του πενταγράμμου που βρίσκονται στην ίδια περιοχή γεγονός που δυσκολεύει τη διαδικασία ταξινόμησης.

Επιπλέον, στοιχεία όπως οι συζεύξεις είναι σχεδόν αδύνατο τα εντοπιστούν μιας και δεν έχουν συγκεκριμένη συμπεριφορά στις προβολές τους όπως έχουν τα άλλα σύμβολα. Η μέθοδος αυτή λοιπόν μπορεί να είναι αποτελεσματική μόνο σε απλές παρτιτούρες χωρίς πυκνή παρουσία συμβόλων. Στο παράδειγμα μας δεν μπορεί να χρησιμοποιηθεί μιας και υστερεί σημαντικά σε σχέση με την αφαίρεση του πενταγράμμου.

Συνηθίζεται, επίσης, η χρήση των προβολών για τον εντοπισμό των νοτών και των αξιών τους αφού έχουν πρώτα διαχωριστεί με κάποια μέθοδο. Συχνά, μάλιστα επιτυγχάνει καλά αποτελέσματα σε περιπτώσεις φθαρμένης παρτιτούρας μιας και δεν βασίζεται στην πλήρη ακεραιότητα της δυαδικής εικόνας όπως βασίζεται η μέθοδος που θα χρησιμοποιήσουμε αργότερα για τον εντοπισμό των νοτών.

5.3. Μέθοδοι χωρίς κατάτμηση

Είδαμε στις προηγούμενες ενότητες ότι η διαδικασία της κατάτμησης (segmentation) είναι σημαντική για τον επιμερισμό της εικόνας σε επιμέρους εικόνες προκειμένου αυτές να αξιολογηθούν ως προς το τι αντιπροσωπεύουν. Η λογική της κατάτμησης είναι ότι χωρίζουμε την εικόνα σε υποσύνολα και έπειτα τα αξιολογούμε ανάλογα με τις ιδιότητές τους.

Υπάρχει όμως και η αντίστροφη λογική. Δηλαδή, η αποθήκευση από πριν βασικών υποσυνόλων που θεωρούμε ότι υπάρχουν μέσα στην παρτιτούρα και η αναζήτηση μετέπειτα μέσα στην εικόνα για να εντοπίσουμε αν υπάρχουν και πια είναι η θέση τους.

Η τεχνική που προβλέπεται για την υλοποίηση αυτής της μεθόδου είναι το ταίριασμα προτύπως που είναι πιο κοινώς γνωστό ως template matching.

5.3.1. Template Matching

Η τεχνική αυτή ακολουθεί την παρακάτω διαδικασία. Δέχεται ως είσοδο μια εικόνα πρότυπο και μια εικόνα μέσα στην οποία θα διεξάγει την αναζήτηση. Έπειτα σαρώνει όλη την εικόνα με την εικόνα πρότυπο και αποθηκεύει τις διαφορές μεταξύ των δυο εικόνα. Το τελικό αποτέλεσμα είναι οι διαφορές των δύο εικόνων, συνήθως αντεστραμμένο ή συμπληρωμένο, προκειμένου οι μεγάλες τιμές να αντιστοιχούν σε μικρές διαφορές και άρα σε μεγαλύτερο ταίριασμα.

Το βασικό μειονέκτημα αυτής της μεθόδου είναι το κόστος της σε πόρους. Όλες οι διαδικασίες που απαιτούν σάρωση της εικόνας με ένα παράθυρο δεδομένων απαιτούν μεγάλο αριθμό προσπελάσεων.

Ωστόσο μπορούμε να περιορίσουμε αυτούς τους χρόνους δεδομένου ότι «υποψιαζόμαστε» σε ποιες περιοχές είναι δυνατόν να βρίσκονται τα στοιχεία που αναζητούμε και άρα να μειώσουμε σημαντικά την επιφάνεια που καλύπτει κάθε σάρωση. Ωστόσο παραμένει μια βαριά μέθο-

δος που όμως το δυνατό της πλεονέκτημα είναι ότι επηρεάζεται ελάχιστα από ενδεχόμενη φθορά της εικόνας αλλά και από τις γραμμές του πενταγράμμου.

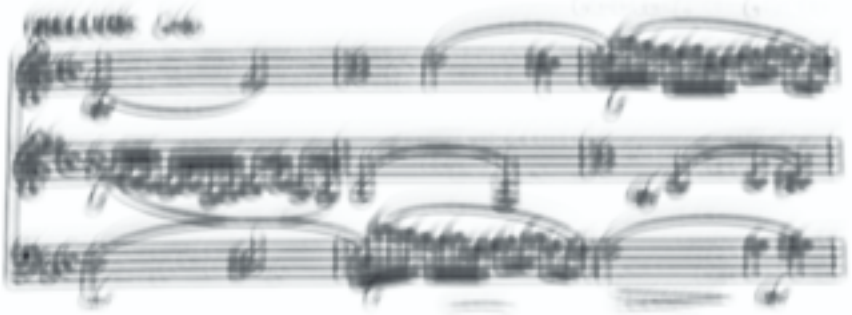
Χρησιμοποιώντας, έτσι, ως template μια εικόνα του κλειδιού του σολ (5.3.1), μεταβάλλουμε το μέγεθος της ώστε το ύψος να είναι 9.4*staff_distance κάτι που το έχουμε υπολογίσει από πριν και εφαρμόζοντας συνάρτηση δυσδιάστατης συσχέτισης των δύο σημάτων παίρνουμε το παρακάτω αποτέλεσμα (5.3.2).

Εικόνα 5.3.1

Εικόνα 5.3.2

Στην εικόνα βλέπουμε δύο εμφανείς μαύρες κουκίδες που αντιστοιχούν στα σημεία που εντοπίστηκαν οι ελάχιστες διαφορές και άρα οι μέγιστες τιμές. Οι τιμές σε αυτά τα σημεία είναι περίπου 1100 και δεδομένου ότι το άθροισμα όλων των στοιχείων του προτύπου είναι 1261 έχουμε ταίριασμα με ποσοστό 87%, το οποίο είναι μεγάλο και άρα έγκυρο. Αντίστοιχα με το κλειδί του φα παίρνουμε το παρακάτω αποτέλεσμα.

Εικόνα 5.3.3



Εικόνα 5.3.4

Στην δεύτερη εικόνα (5.3.4) παρατηρούμε ότι αφενός έχει εντοπιστεί πάλι στο σημείο που αναμένουμε η ύπαρξη του συμβόλου που αναζητούμε, αφετέρου η διαφορά της έντασης και άρα της εγκυρότητας σε σχέση με άλλα σημεία δεν είναι τόσο μεγάλη όσο ήταν στην αναζήτηση του κλειδιού του σολ. Αυτό συμβαίνει διότι όσο μεγαλύτερο είναι το σύμβολο που αναζητούμε τόσο μεγαλύτερη είναι η διαφορά μεταξύ των αποτελεσμάτων στη σωστή θέση και σε άλλες θέσεις ενώ όσο μικρότερα είναι τα σύμβολα τόσο μεγαλύτερη είναι η πιθανότητα σφάλματος.

Για τους λόγους που αναφέραμε παραπάνω η μέθοδος του template matching δεν ωφελεί να χρησιμοποιηθεί για την εύρεση όλων των συμβόλων αλλά μπορεί να συνδυαστεί με επιπλέον μεθόδους με άλλες μεθόδους για να διευκολύνει το τελικό αποτέλεσμα .Τέλος αυτή η μέθοδος μπορεί να βοηθήσει στο ενδεχόμενο που δεν έχει γίνει σωστό segmentation και κάποια σύμβολα έχουν μείνει συνδεδεμένα ενώ δεν θα έπρεπε.

6
Εντοπισμός των νοτών

6.1. Εισαγωγή

Προχωράμε στο πιο σημαντικό στάδιο του συστήματος OMR που είναι ο εντοπισμός των νοτών. Σε αυτό το κεφάλαιο θα ασχοληθούμε με τον εντοπισμό όλων των νοτών στην παρτιτούρα που σημαίνει ότι θα εντοπίσουμε τη σχετική τους θέση στο πεντάγραμμο και τη διάρκειά τους.

Σε αυτό το στάδιο δεν μπορούμε να εξάγουμε την τονικότητα μιας και αυτή εξαρτάται από το κλειδί του πενταγράμμου, τον οπλισμό και όλα τα πιθανά σημεία στίξεως που μπορεί να υπάρχουν τα οποία θα εντοπίσουμε αργότερα. Η θέση και η διάρκεια είναι ότι ακριβώς χρειαζόμαστε αυτή τη στιγμή και έπειτα θα εφαρμόσουμε τις υπόλοιπες πληροφορίες προκειμένου να εξάγουμε την ακριβή τονικότητα και την ακριβή διάρκεια.

Η διαδικασία εντοπισμού των νοτών δεν είναι ίδια για όλες τις νότες. Διαφοροποιείται για όλες τις νότες με αξία τετάρτου η μικρότερη και για τις μισές και ολόκληρες που έχουν άλλο σχήμα. Ειδικά οι τελευταίες παρουσιάζουν επιπλέον δυσκολία λόγω των φθορών που υφίστανται από την αφαίρεση του πενταγράμμου. Γι' αυτό το λόγο θα περιγράψουμε δυο διαφορετικές μεθοδολογίες για να πετύχουμε όσο το δυνατόν πιο ακριβή αποτελέσματα σε αυτό το στάδιο.

6.2. Εντοπισμός των νοτών με αξία τετάρτου ή μικρότερη

Ξεκινάμε με τον εντοπισμό των νοτών των οποίων η κεφαλή είναι «γεμάτη». Αναφέρουμε αυτό το χαρακτηριστικό γιατί είναι το βασικό μας εργαλείο για να εντοπίσουμε τις νότες με χρήση μαθηματικής μορφολογίας.

Αν εφαρμόσουμε άνοιγμα της εικόνας με κατάλληλο δομικό στοιχείο μπορούμε να εντοπίσουμε αμέσως την θέση όλων των μαύρων κεφαλών στην παρτιτούρα. Στη συνηθισμένη τυπογραφία των παρτιτούρων οι νότες έχουν όλες με πολύ μικρή απόκλιση το ίδιο σχήμα το οποίο μπορούμε να προσεγγίσουμε μεγεθύνοντας ένα πρότυπο δομικό στοιχείο ώστε να έχει ύψος ίσο με staff_distance. Προφανώς το μέγεθος μιας νότας έχει στην πραγματικότητα ύψος ίσο με staff_distance + 2*staff_thickness αλλά εφόσον θα διεξάγουμε μορφολογικό άνοιγμα για να είμαστε σίγουροι ότι το δομικό στοιχείο θα είναι πλήρες υποσύνολο της κάθε νότας, πρέπει οι διαστάσεις του να είναι αρκετά μικρές ώστε να «χωράει» σε όλες τις νότες αλλά και αρκετά μεγάλες ώστε να μην «χωράει» σε πολλά άλλα σημεία.

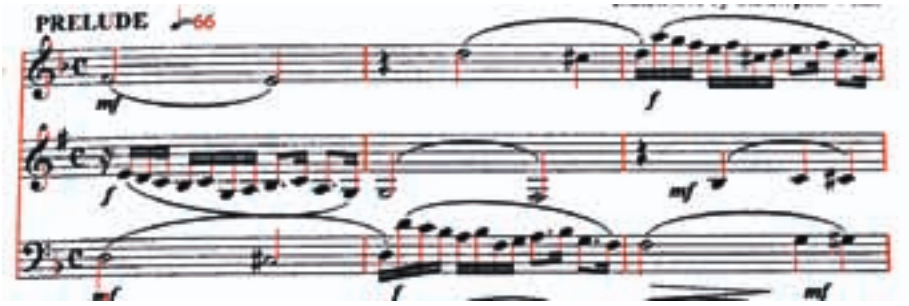
Αναφέρουμε «σε πολλά άλλα» γιατί είναι δεδομένο ότι το μορφολογικό άνοιγμα θα μας δώσει και σφάλματα για αυτό πρέπει πέρα από τις οπτικές επεξεργασίες να προσθέσουμε και μιας μορφής γραμματική σε αυτό το σημείο προκειμένου να εξάγουμε τα σωστά αποτελέσματα.

Υπενθυμίζουμε ότι κάθε νότα αποτελείται από την κεφαλή της (note head), το stem και πιθανώς από το beam (6.2.1). Με κόκκινο απεικονίζονται οι κεφαλές, με γκρι τα stems και με μπλε το beam. Χρησιμοποιούμε την αγγλική ορολογία μιας και είναι πιο δόκιμη σε αυτή την περίπτωση.

Εικόνα 6.2.1

Έτσι σαρώνουμε την εικόνα τρείς φορές αναζητώντας τα υποψήφια στοιχεία κάθε κατηγορίας. Αρχικά τα stems, έπειτα τα beams και τέλος τις κεφαλές. Η σειρά είναι τέτοια διότι για να απομονώσουμε τα beams από τις κεφαλές πρέπει να αφαιρέσουμε πρώτα τα stems. Επίσης για να αποκλείσουμε το ενδεχόμενο να εντοπιστούν κεφαλές μέσα στα beams τα αφαιρούμε πριν τις κεφαλές.

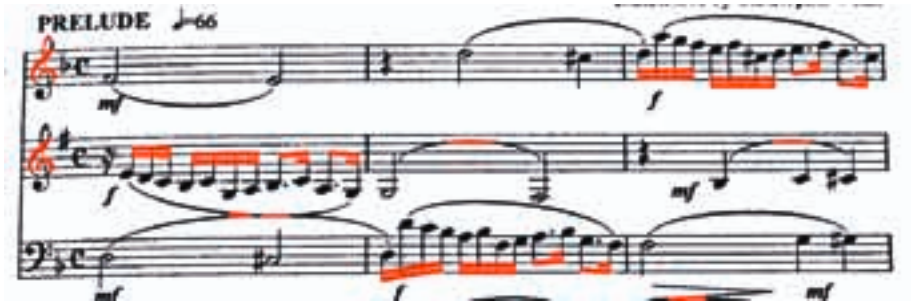
Έτσι αρχικά υπολογίζουμε τα υποψήφια stem με κριτήριο το μικρό οριζόντιο μήκος (≤2*staff_thickness) και το σχετικά μεγάλο κάθετο μήκος (≥3*staff_thickness) και παίρνουμε το παρακάτω αποτέλεσμα. (6.2.2).



Εικόνα 6.2.2

Εντοπίστηκαν όλα τα stems αλλά επιπλέον και άλλα στοιχεία όπως οι γραμμές των μέτρων και άλλα. Είναι λογικό με βάση τα κριτήρια που βάλαμε αλλά αυτό που μας ενδιαφέρει αυτή τη στιγμή είναι τουλάχιστον να έχουν εντοπιστεί όλα τα stems μαζί με τα επιπλέον στοιχεία.

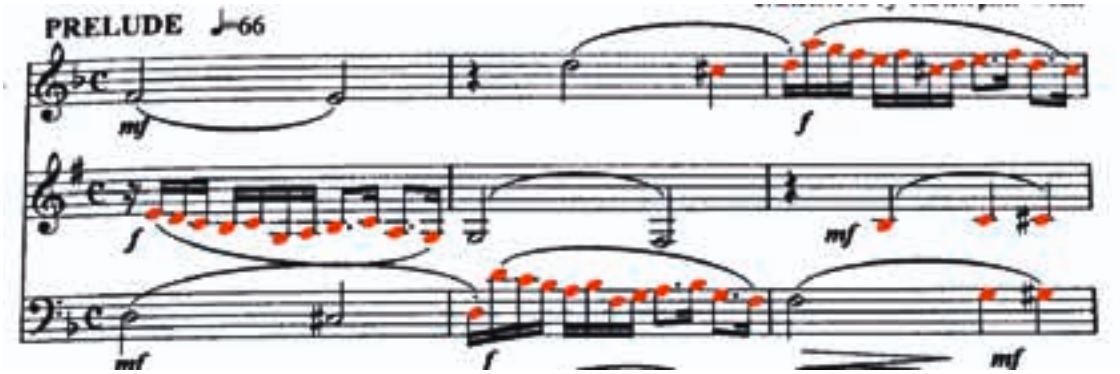
Στη συνέχεια εντοπίζουμε τα υποψήφια beams. Κριτήριο είναι το ελάχιστο πάχος αυτών των στοιχείων αλλά και το ελάχιστο μήκος (περίπου 2.5*staff_distance). Τα υποψήφια beams απεικονίζονται παρακάτω (6.2.3).



Εικόνα 6.2.3

Αντίστοιχα εντοπίσαμε όλα τα υποψήφια beams μαζί προφανώς και με επιπλέον στοιχεία που θα απομακρυνθούν στη συνέχεια.

Τέλος υπολογίζουμε τα υποψήφια στοιχεία για κεφαλές νοτών (6.2.4).

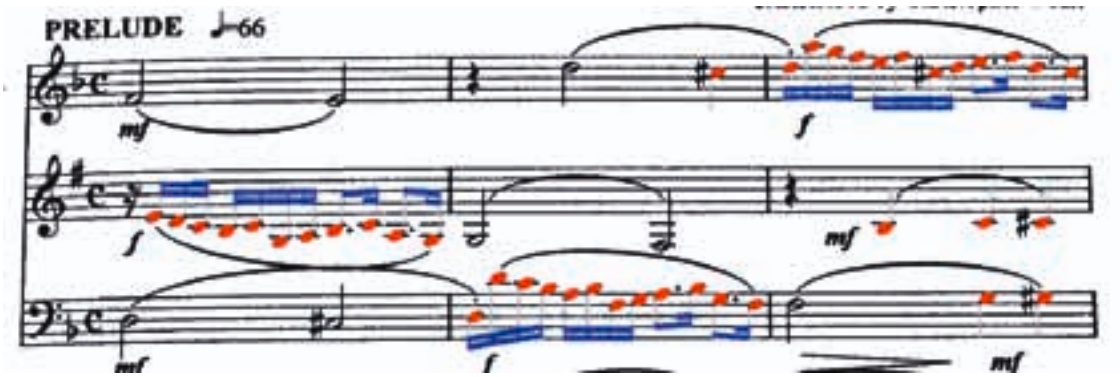


Εικόνα 6.2.4

Αυτή τη φορά εντοπίζονται μόνο τα ζητούμενα στοιχεία και τίποτα επιπλέον. Ωστόσο αυτό δεν συμβαίνει πάντα. Γι' αυτό το λόγο εφαρμόζουμε τους παρακάτω κανόνες προκειμένου να κρατήσουμε μόνο τα ζητούμενα στοιχεία.

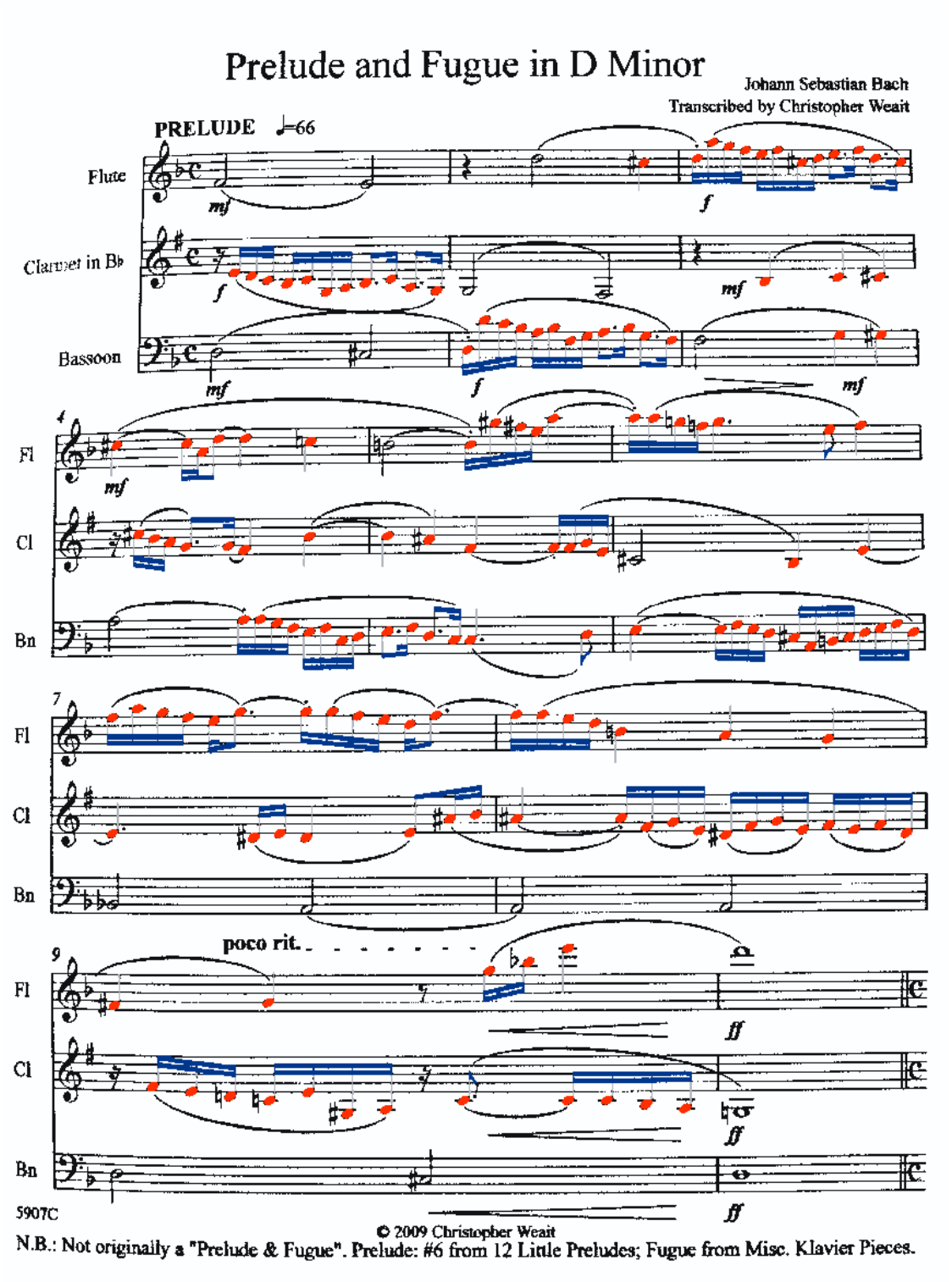
1. Για κάθε κεφαλή αναζητούμε την ύπαρξη stem σε κατάλληλη απόσταση. Αν υπάρχει κρατάμε και τα δυο στοιχεία ως έγκυρα. Αν όχι απορρίπτεται η κεφαλή
2. Για κάθε stem που είναι έγκυρο αναζητούμε στα άκρα του αν έρχεται σε επαφή με κάποιο beam. Αν ναι κρατάμε το συγκεκριμένο beam, αν όχι κρατάμε μόνο το stem
3. Τα stems που δεν ακουμπάνε σε κάποιο beam είτε έχουν αξία τετάρτου ή ακουμπάνε σε κάποιο άλλο στοιχείο από αυτά που έχουν απομείνει στην εικόνα το οποίο το αποθηκεύουμε για να μετρήσουμε τις ιδιότητές του.
4. Ξανά σαρώνουμε τις κεφαλές που έχουν απομείνει και αν βρίσκονται σε μικρή κάθετη απόσταση από κάποια κεφαλή που έχει κρατηθεί, παίρνει και αυτή την αξία της κεφαλής που βρήκαμε.

Αν εφαρμόσουμε τα παραπάνω κριτήρια, τα στοιχεία που μένουν είναι τα παρακάτω (6.2.5).



Εικόνα 6.2.5

Τέλος, βλέπουμε το αποτέλεσμα για όλη την εικόνα (6.2.6).



Εικόνα 6.2.6

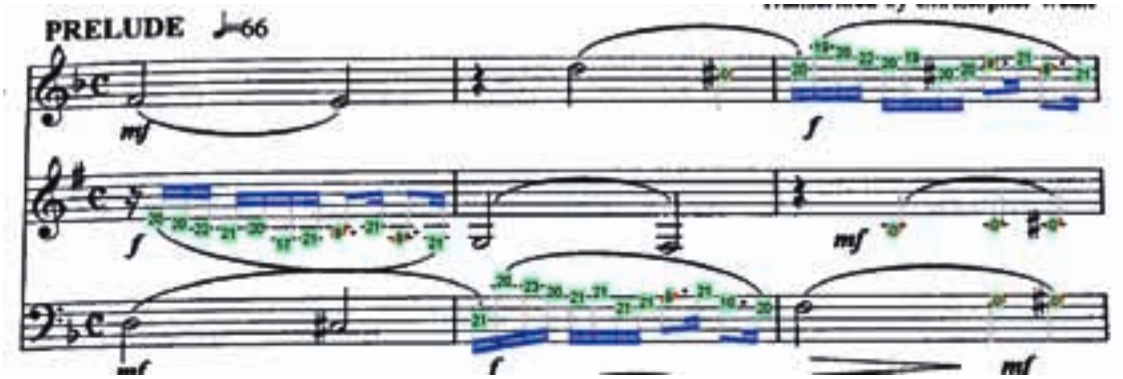
Έχουμε εντοπίσει όλα τα στοιχεία που χρειαζόμαστε για να αξιολογήσουμε την τονικότητα και την χρονική αξία κάθε νότας μιας και γνωρίζουμε και έχουμε αποθηκεύσει κατάλληλα τον τρόπο με τον οποίο συνδέονται τα παραπάνω στοιχεία μεταξύ τους άρα αυτό που μένει είναι ο προσδιορισμό της τονικότητας και της χρονικής διάρκειας.

6.2.1. Εντοπισμός της χρονικής αξίας

Έχοντας συλλέξει τις παραπάνω πληροφορίες μπορούμε να προσδιορίσουμε τη χρονική διάρκεια κάθε νότας τουλάχιστον ως προς την αξία που απεικονίζεται. Πιθανώς αν κάποια νότα είναι παρεστιγμένη (δηλαδή υπάρχει μία τελεία δίπλα της) τότε ο στην αξία της προστίθεται επιπλέον το μισό αλλά αυτό θα το εφαρμόσουμε παρακάτω.

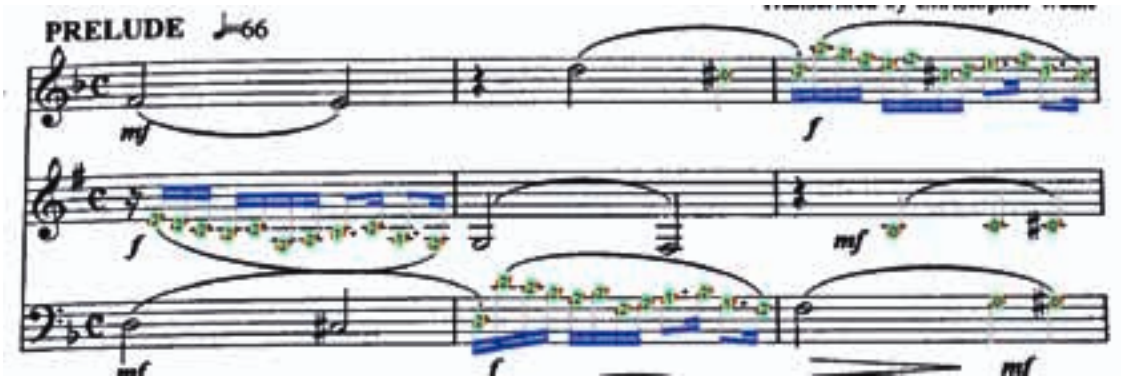
Η αξία κάθε νότας αντιστοιχεί στον τύπο του beam στο οποίο ακουμπάει το αντίστοιχο stem. Οπότε για κάθε νότα αποθηκεύουμε την τιμή της κάθετης προβολής του beam στο σημείο, στο οποίο ακουμπάει το αντίστοιχο stem. Η τιμή της προβολής αντιστοιχεί στο «πάχος» του beam στο σημείο επαφής.

Απεικονίζοντας λοιπόν αυτές τις τιμές παίρνουμε την παρακάτω εικόνα (6.2.7).



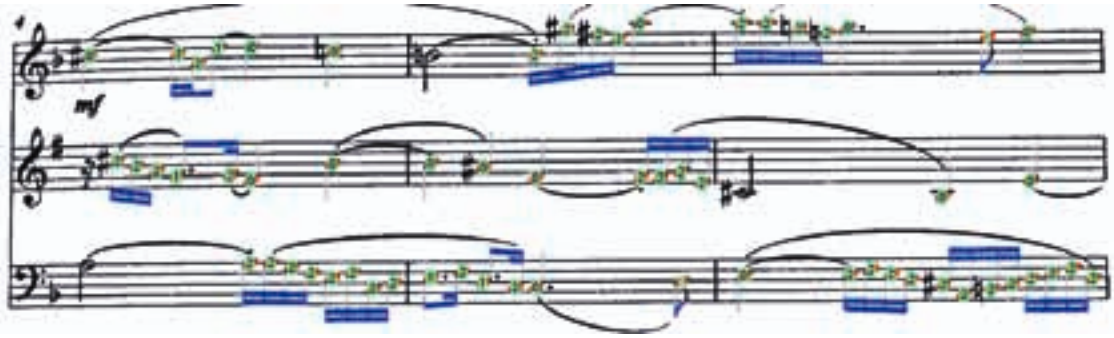
Εικόνα 6.2.7

Βλέπουμε ότι η νότες με αξίες τετάρτου έχουν τον αριθμό 0 που σημαίνει ότι δεν ακουμπάνε σε κάποιο beam. Οι νότες με αξία ογδόου έχουν τιμές από 8 έως 10 και τα δέκατα έκτα από 19 έως 23. Επίσης μπορούμε να υπολογίσουμε το μέσο πάχος των γραμμών των beams το οποίο στη συγκεκριμένη περίπτωση είναι 9. Έτσι αν υπολογίσουμε το στρογγυλοποιημένο πηλίκο των προβολών διά του beam_height+staff_thickness/2 καταφέρνουμε να χωρίσουμε τις νότες σε κλάσεις στις οποίες το 0 αντιστοιχεί σε τέταρτο, το 1 σε όγδοο, το 2 σε δέκατο έκτο κοκ. (6.2.8).



Εικόνα 6.2.8

Υπολογίσαμε τελικά επιτυχημένα την αξία των νοτών και μάλιστα χωρίς σφάλματα στο παράδειγμά μας. Θα εστιάσουμε και σε ένα άλλο κομμάτι της εικόνας (6.2.9) για να δείξουμε ότι για τις μεμονωμένες νότες που δεν είναι τέταρτα χρειάζεται άλλος παράγοντας διαίρεσης ο οποίος πειραματικά υπολογίσαμε ότι είναι 2*beam_height.



Εικόνα 6.2.9

Αν χρησιμοποιούσαμε τον ίδιο παράγοντα διαίρεσης τα μεμονωμένα όγδοα τις εικόνας θα μας δίναν αποτέλεσμα ίσο με 2. Χρειάζεται συνεπώς ειδική μεταχείριση αυτών των στοιχείων.

Έχοντας εντοπίσει τις σχετικές αξίες των νοτών είμαστε έτοιμοι να προχωρήσουμε στον υπολογισμό της τονικότητας τους και συγκεκριμένα της σχετικής τους θέσης μέσα στο πεντάγραμμο.

6.2.2. Εντοπισμός της τονικότητας

Αναφέραμε προηγουμένως ότι η ακριβής τονικότητα δεν μπορεί να εξαχθεί σε αυτό το στάδιο παρά μόνο η σχετική θέση της νότας στο πεντάγραμμο. Αργότερα θα εφαρμοσθούν επιπλέον πληροφορίες που θα καθορίσουν την ακριβή τονικότητα.

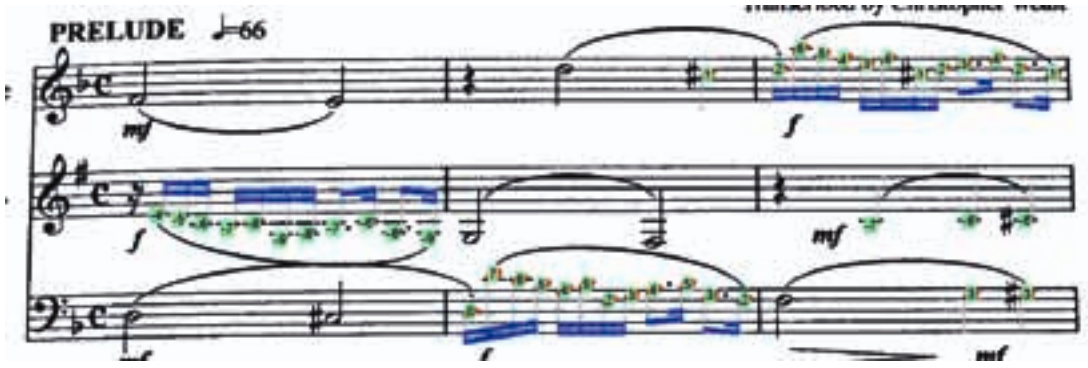
Σε αυτό το στάδιο αυτό που χρειαζόμαστε είναι να υπολογίσουμε:

1. Σε ποιο πεντάγραμμο αντιστοιχεί η κάθε νότα.

2. Πόσα διαστήματα απέχει από τη μεσαία γραμμή του συγκεκριμένου πενταγράμμου.

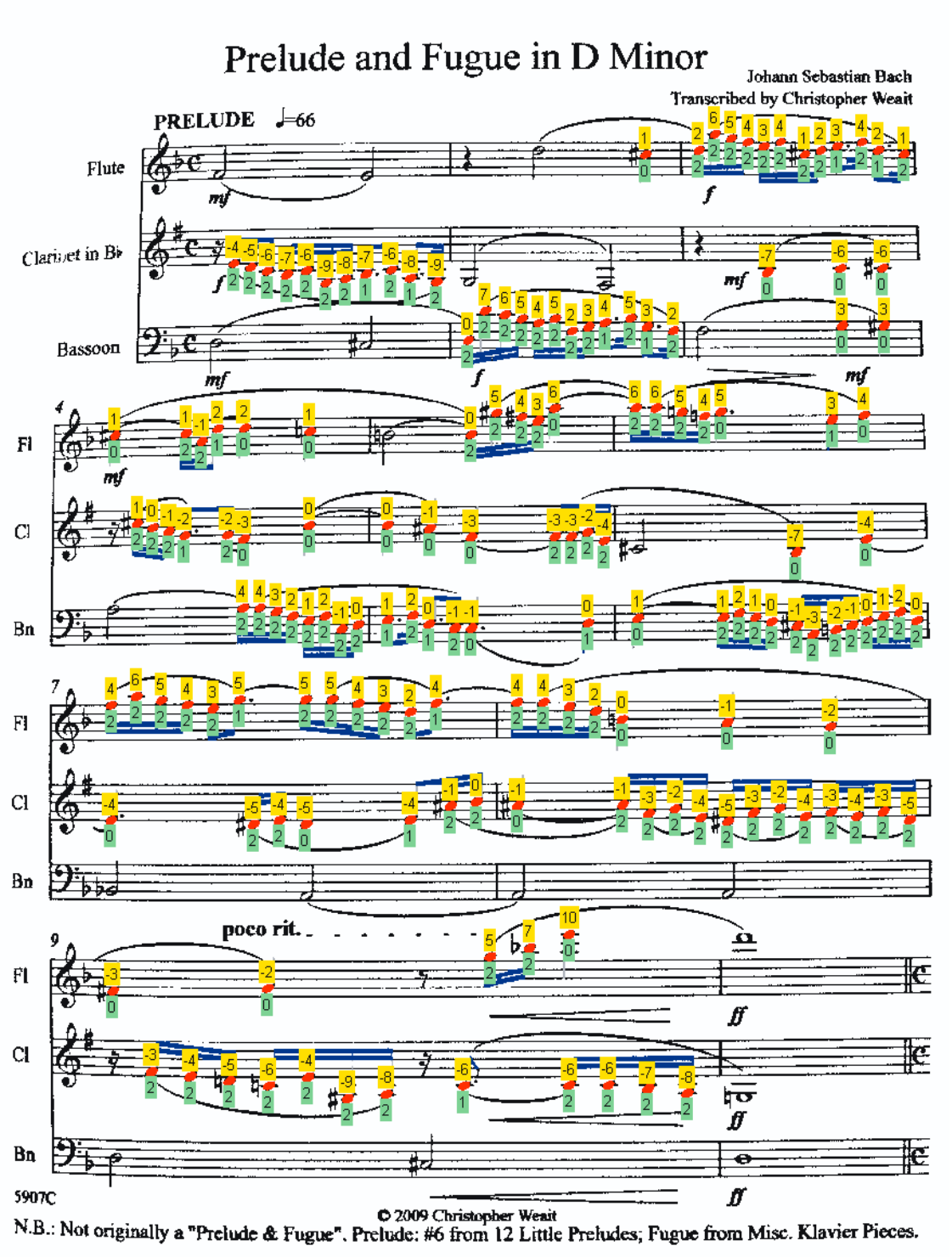
Για τον υπολογισμού του πρώτου αρκεί να μετρήσουμε την απόσταση κάθε νότας από τις μεσαίες γραμμές κάθε πενταγράμμου. Συνήθως αυτός ο κανόνας δεν παραβιάζεται στη συνηθισμένη τυπογραφία. Στη συνέχεια αφού εντοπίσουμε την αντιστοιχία υπολογίζουμε το πηλίκο της απόστασης διά της ποσότητας (staff_distance + staff_thickness)/2 προκειμένου να αποθηκεύσουμε το πλήθος των ημιδιαστημάτων κατά το οποίο η νότα απέχει από τη μεσαία γραμμή.

Να σημειώσουμε ότι εφ' όσον έχουμε αποθηκευμένες τις γραμμές σαν πολυώνυμα αρκεί να θέσουμε την τετμημένη της νότας σε κάθε πολυώνυμο και να συγκρίνουμε τις δυο τεταγμένες για να υπολογίσουμε την κάθετη απόσταση από κάθε γραμμή. Εφαρμόζοντας το συγκεκριμένο αλγόριθμο παίρνουμε το παρακάτω αποτέλεσμα (6.2.10).



Εικόνα 6.2.10

Υπολογίσαμε επιτυχώς τις αποστάσεις από τις μεσαίες γραμμές κάθε πενταγράμμου η οποίες στην ουσία είναι τα βήματα της εκάστοτε κλίμακας. Αργότερα γνωρίζοντας το κλειδί και τον οπλισμό θα εφαρμόσουμε σε αυτά τα βήματα την κλίμακα που θα εντοπίσουμε και την αφετηρία της (ανάλογα με το κλειδί) και θα πάρουμε την ακριβή τονικότητα. Τέλος παραθέτουμε το πλήρες αποτέλεσμα με τις διάρκειες και τις αποστάσεις (6.2.11).



Εικόνα 6.2.11

6.3. Εντοπισμός των νοτών αξίας δευτέρου και ολόκληρου

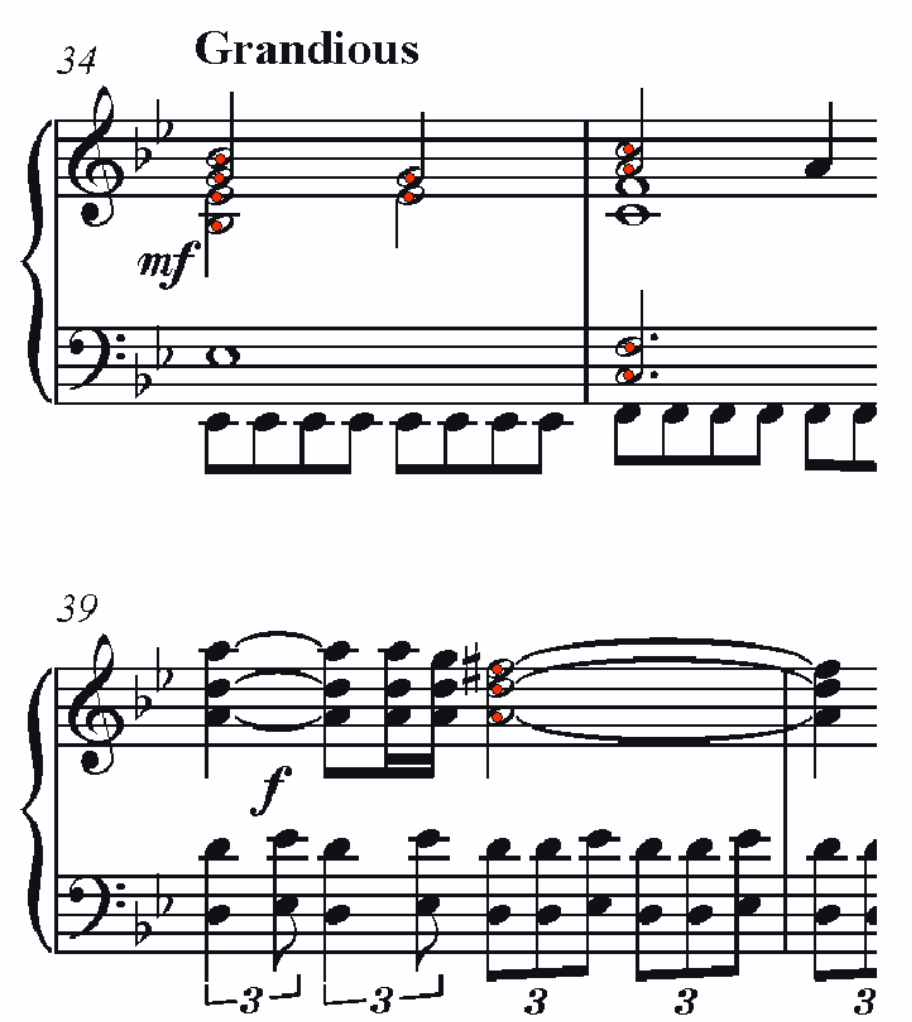
6.3.1. Εντοπισμός των νοτών αξίας δευτέρου

Για το συγκεκριμένο βήμα θα χρησιμοποιήσουμε πάλι τη μέθοδο της συσχέτισης δύο σημάτων αλλά για να επιτύχουμε μεγαλύτερη ακρίβεια και μικρότερο χρόνο εκτέλεσης θα περιορίσουμε τις περιοχές που θα το εφαρμόσουμε. Έτσι έχοντας εντοπίσει τα stems στο προηγούμενο μέρος θα διεξάγουμε συσχέτιση της εικόνας με μια πρότυπη κεφαλή μισής νότας σε περιοχές γύρω από τα stems. Με αυτόν τον τρόπο και κατάλληλο threshold του αποτελέσματος της συσχέτισης μπορούμε να πάρουμε ικανοποιητικό αποτέλεσμα.

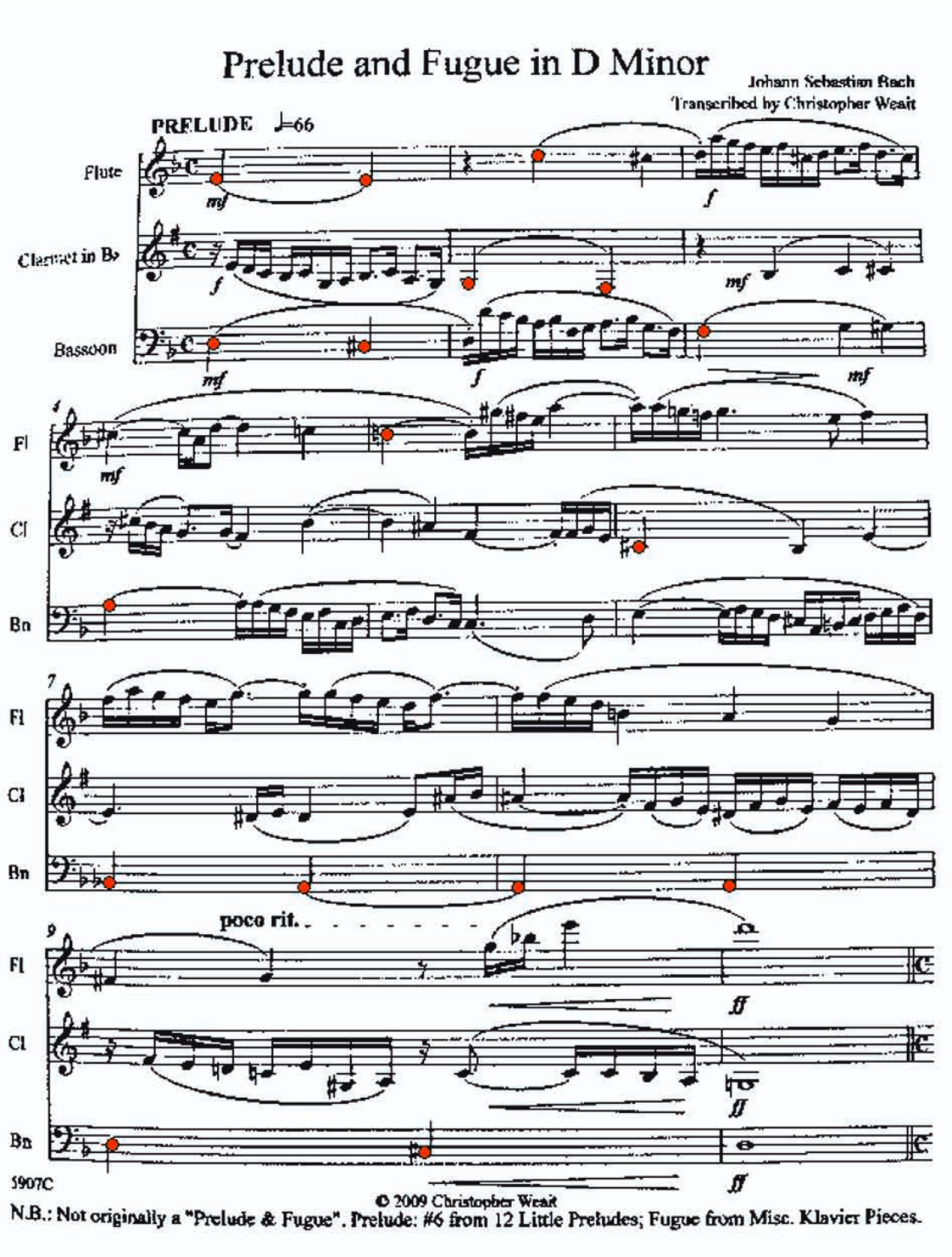
Να σημειωθεί ότι οι μισές και ολόκληρες νότες είναι πάντα δύσκολο ζήτημα να εντοπιστούν με μεγάλη ακρίβεια. Ακόμα και τα πιο λειτουργικά προγράμματα όπως το Photoscore είτε χάνουν είτε βρίσκουν σε λάθος θέσεις μισές και ολόκληρες νότες.

Στην δικής μας εφαρμογή, όσο καλύτερη η ποιότητα της παρτιτούρας, τόσο μεγαλύτερη η πιθανότητα σωστού εντοπισμού.

Παραθέτουμε παρακάτω παραδείγματα εντοπισμού νοτών μισής αξίας με τη μέθοδο μας (6.2.1.2).



Εικόνα 6.3.1


Prelude and Fugue in D Minor
Johann Sebastian Bach
Transcribed by Christopher Weait
PRELUDE ♩=66
Flute
Clarinet in B♭
Bassoon
Fl
Cl
Bn
poco rit.
5907C
© 2009 Christopher Weait
N.B.: Not originally a "Prelude & Fugue". Prelude: #6 from 12 Little Preludes; Fugue from Misc. Klavier Pieces.

Εικόνα 6.3.2

6.3.2. Εντοπισμός νοτών αξίας ολοκλήρου

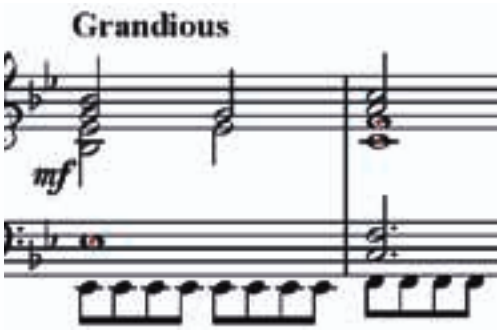
Τέλος, για το κεφάλαιο των νοτών με την εύρεση του πιο δύσκολου στοιχείου που είναι οι νότες αξίας ολοκλήρου. Η δυσκολία του συγκεκριμένου στοιχείου έγκειται στο ότι δεν περιέχει κάποια δομικά στοιχεία που να το διακρίνουν σαφώς από άλλα τμήματα της παρτιτούρας. Οι νότες που είδαμε πριν έχουν είτε το διακριτό εμβαδό της κεφαλής, ή το stem που χρησιμοποιήσαμε για τις μισές νότες.

Επιπλέον στις μεθόδους αφαίρεσης του πενταγράμμου, δεν υπάρχει καμία μέθοδος που να εξασφαλίζει ότι δεν θα σπάσουν τα δυο άκρα μιας νότας κατά την αφαίρεση. Ακόμα και η μέθοδος Carter δεν εξασφαλίζει την «διάσωση» των ολόκληρων νοτών κατά την αφαίρεση του πενταγράμμου.

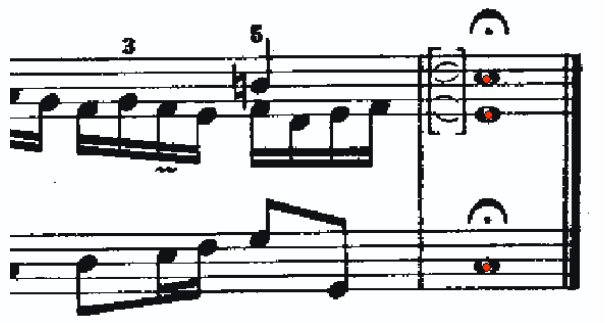
Για αυτούς τους λόγους, χρησιμοποιούμε ένα συνδυασμό κριτηρίων για να πετύχουμε ένα ικανοποιητικό αποτέλεσμα.

Εντοπίσουμε αρχικά τις τρύπες στο εσωτερικό των νοτών (ουσιαστικά μελετάμε τη συμπληρωματική εικόνα τώρα). Κρατάμε όσες πληρούν κριτήρια μεγέθους και εφαρμόζουμε συσχέτιση στην περιοχή. Αν η κορυφή της συσχέτισης είναι πολύ κοντά στο κέντρο της τρύπας τότε κρατάμε το αποτέλεσμα (6.2.3-4).

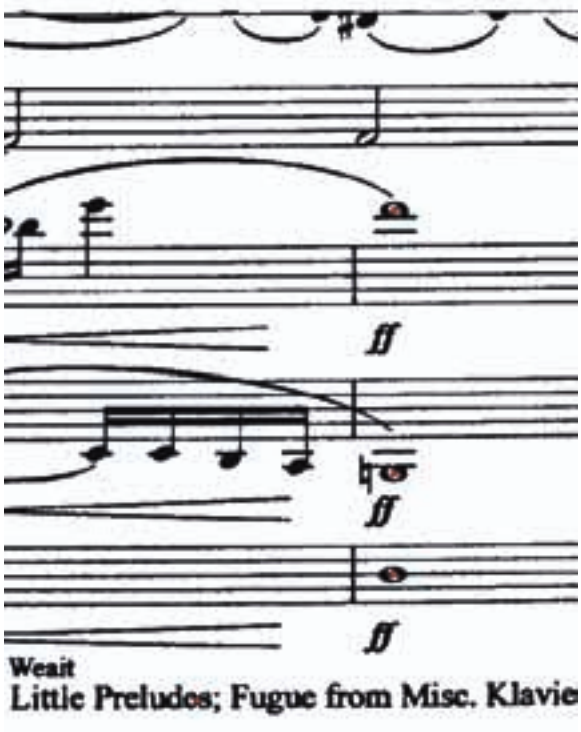
Βέβαια με αυτόν τον τρόπο μπορεί να βρούμε επιπλέον στοιχεία, τα οποία μπορούμε να αποκλείσουμε είτε με λογική, λόγω της θέσης τους ή με επιπλέον κριτήρια στη ταξινόμηση αργότερα.



Εικόνα 6.3.3



Εικόνα 6.3.4



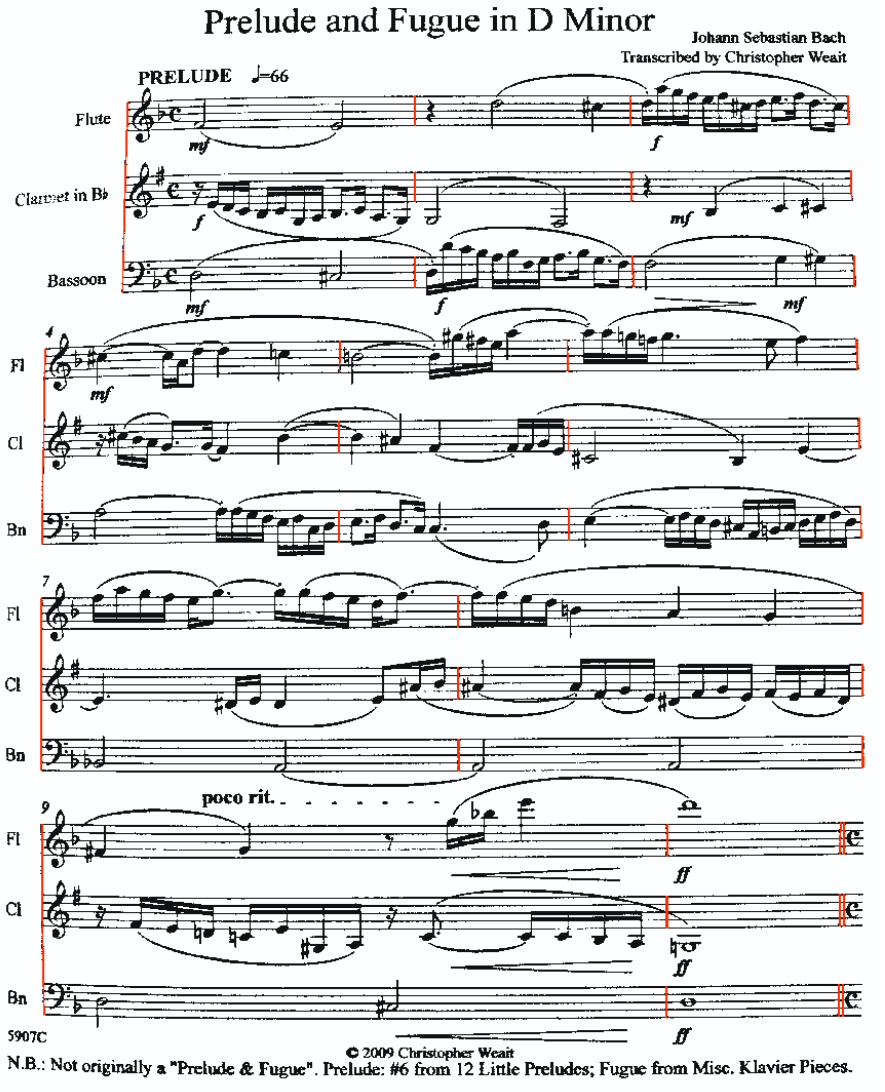
Εικόνα 6.3.5

Στο παράδειγμα μας στην εικόνα 6.3.5 εντοπίστηκε επιπλέον κομμάτι του κειμένου το οποίο λόγω της απόστασής του από το πεντάγραμμο θα απορριφθεί ωστόσο είναι αρκετά συχνό φαινόμενα ειδικά σε πυκνογραμμένες παρτιτούρες.

7
Εντοπισμός Μέτρων και Συζεύξεων

7.1. Εντοπισμός των γραμμών του μέτρου

Ο εντοπισμός των γραμμών του μέτρου είναι ιδιαίτερα εύκολος σε αυτό το σημείο δεδομένου ότι γνωρίζουμε πλήρως τη θέση των γραμμών του πενταγράμμου. Αφαιρούμε όλα μικρά vertical runs και τέλος κρατάμε τα στοιχεία των οποίων τα πάνω άκρα ακουμπούν στην πρώτη γραμμή κάποιου πενταγράμμου και τα κάτω στην τελευταία. Επιπλέον αποθηκεύουμε σε ποιο η ποια πεντάγραμμα αντιστοιχεί το μέτρο. Το αποτέλεσμα είναι το παρακάτω (7.1.1).



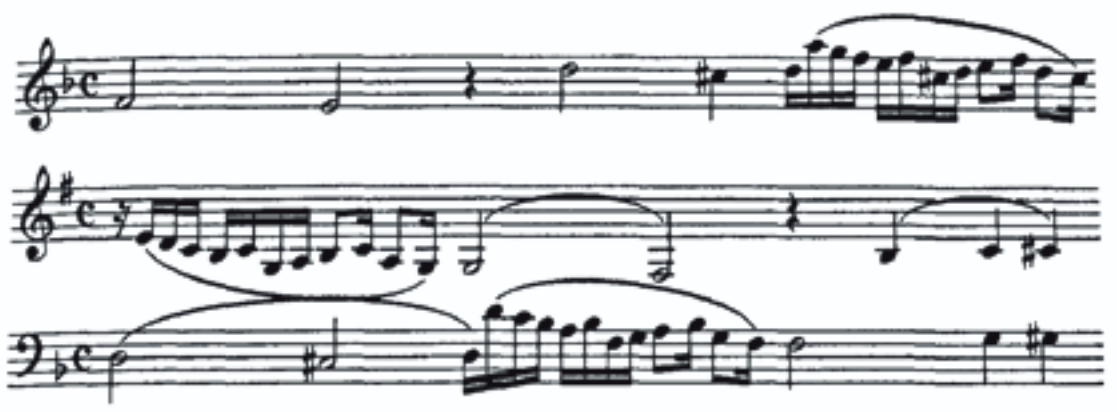
Εικόνα 7.1.1

7.2. Εντοπισμός συζεύξεων

Οι συζεύξεις είναι οι γραμμές με μεγάλο οριζόντιο μήκος και μικρό κάθετο πάχος. Όσες δεν είναι συνδεδεμένες στο πεντάγραμμο εντοπίζονται εύκολα με κριτήρια μήκους και χαμηλού μέσου όρου της κάθετης προβολής.

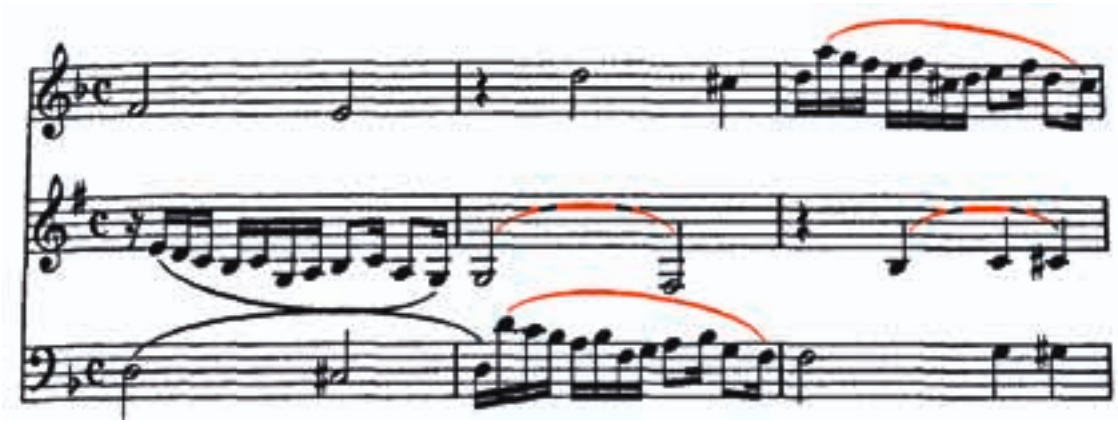
Οι συζεύξεις που είναι συνδεδεμένες με το πεντάγραμμο απαιτούν πρώτα την αφαίρεση του πενταγράμμου. Θα εφαρμόσουμε την απλή αφαίρεση του πενταγράμμου και θα αποκαταστήσουμε στη συνέχεια συζεύξεις που έχουν σπάσει.

Για να έχουμε καλύτερα αποτελέσματα, μιας και έχουμε εντοπίσει τις γραμμές των μέτρων, μπορούμε να τις αφαιρέσουμε χωρίς να σπάσουμε τις γραμμές που ακουμπούν στις γραμμές των μέτρων (7.2.1).



Εικόνα 7.2.1

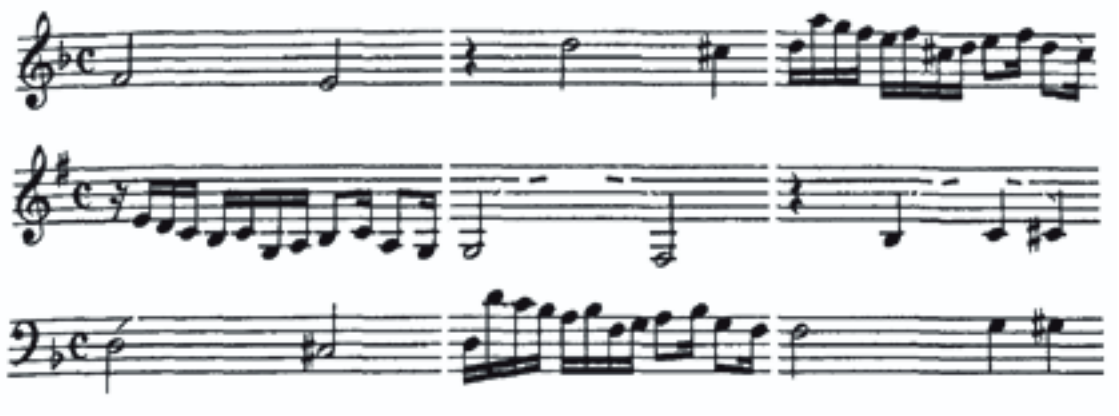
Αφαιρούμε τις γραμμές του πενταγράμμου και ενώνουμε τα άκρα που έχουν σπάσει (7.2.2).



Εικόνα 7.2.2

Οι συζεύξεις που δεν έχουν εντοπιστεί έχουν αποθηκευτεί σε μια νέα εικόνα προκειμένου αφενός να μην επηρεάζουν παρακάτω την ταξινόμηση των στοιχείων, αφετέρου για να αξιολογηθούν εκ νέου μετά. Στη συγκεκριμένη περίπτωση οι δύο συζεύξεις επειδή ακουμπούν μεταξύ τους δεν πληρούν τα κριτήρια που έχουμε θέσει. Ωστόσο υπάρχει τρόπος στη συνέχεια να αναγνωριστούν. Προς το παρόν απλά αφαιρούνται από την εικόνα.

Έτσι, η εικόνα χωρίς τις συζεύξεις και τις γραμμές του μέτρου είναι όπως παρακάτω (7.2.3).

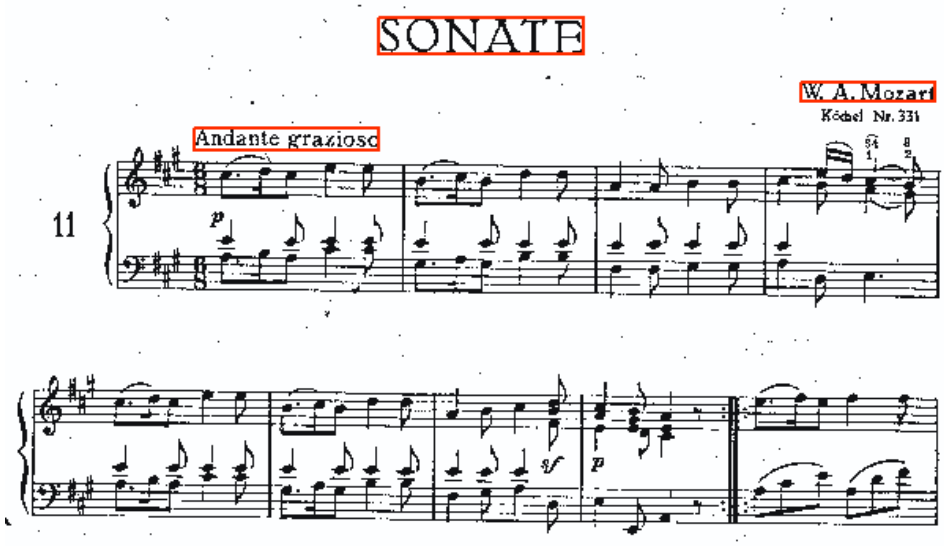


Εικόνα 7.2.3

Σε αυτό το σημείο μπορούμε να προχωρήσουμε στην τελική ταξινόμηση των στοιχείων που έχουν απομείνει. Η αφαίρεση των συζεύξεων είναι απαραίτητη μιας και υπάρχει το ενδεχόμενο να είναι συνδεδεμένες με άλλα στοιχεία του πενταγράμμου και να εμποδίζουν την τελική ταξινόμηση. Επιπλέον, δεδομένου ότι οι συζεύξεις έχουν ακαθόριστο μέγεθος και σχήμα δεν μπορούν να ενταχθούν στη διαδικασία ταξινόμησης καθώς αυτή απαιτεί σύμβολα των οποίων οι παράμετροι είναι κοντινές για όλες τις παρτιτούρες.

7.3. Εντοπισμός του τίτλου και του τέμπο

Συμπληρωματικά εφαρμόσαμε σε κάθε σελίδα έναν αλγόριθμο ομαδοποίησης στοιχείων με παρόμοιο μέγεθος και σε κοντινή απόσταση. Με αυτόν τον τρόπο μπορούμε να εντοπίσουμε τον τίτλο και το τέμπο του κάθε κομματιού που θα το χρειαστούμε για να παράγουμε ένα αντιπροσωπευτικό ηχητικό αποτέλεσμα.



Εικόνα 7.3.1

Κάνοντας χρήση της OCR συνάρτησης του Matlab μπορούμε να εξάγουμε τον τίτλο αλλά και το τέμπο μιας και γνωρίζουμε πάντα τη θέση του στο πάνω αριστερό μέρος της σελίδας. Επιπλέον έχουμε φτιάξει μια λίστα που αντιστοιχεί τις λέξεις που εντοπίζονται στον αντίστοιχο ρυθμό σε bpm.

8
Ταξινόμηση (Classification)

8.1. Εισαγωγή

Πρόκειται για ένα από τα πιο σημαντικά τμήματα του συστήματος, διότι αν δεν υλοποιηθεί σωστά τότε ότι έχουμε εντοπίσει μέχρι τώρα δεν μπορεί να αποκτήσει νόημα.

Στην εικόνα μας έχουν μείνει μια σειρά από σύμβολα τα οποία πρέπει να αναγνωρίσουμε προκειμένου να συνδέσουμε νοηματικά το σύνολο των στοιχείων και να παράξουμε το τελικό αποτέλεσμα.

Μας ενδιαφέρουν συγκεκριμένα τα κλειδιά των πενταγράμμων, ο οπλισμός (δίεση, ύφεση, αναίρεση) και οι παύσεις. Επιπλέον στην εικόνα έχουν απομείνει και οι νότες μισής και ολόκληρης αξίας, όπως επίσης και τα υπόλοιπα σύμβολα όπως οι χρόνοι των μέτρων, οι δυναμικές ενδεχομένως και άλλα.

Τις νότες μισής και ολόκληρης αξίας θα τις συμπεριλάβουμε στον ταξινομητή για να επιβεβαιώσουμε και ενδεχομένως να αποκλείσουμε αποτελέσματα που βρήκαμε μέχρι τώρα μιας και είδαμε ότι τυχαίνει να έχουμε σφάλματα (6.3.5).

Υπάρχουν δύο βασικοί μέθοδοι ταξινόμησης που χρησιμοποιούνται:

1. Η ταξινόμηση σύμφωνα με τον κοντινότερο γείτονα **(k-Nearest Neighbor)**

2. Η ταξινόμηση με χρήση νευρωνικού δικτύου **(Neural Network)**

Η πρώτη μέθοδος, την οποία δεν θα χρησιμοποιήσουμε, δημιουργεί ένα χώρο k διαστάσεων. Κάθε διάσταση είναι μια παράμετρος των δειγμάτων που θέλουμε να αξιολογήσουμε (πχ το ύψος, το εμβαδό κλπ.).

Στη συνέχεια σύμφωνα με τις πρότυπες διαστάσεις που θέτουμε για τα σύμβολα που αναζητούμε υπολογίζονται οι αποστάσεις στο k–διάστατο χώρο και τελικά το δείγμα ταξινομείται στην κλάση από την οποία απέχει λιγότερο.

Ο λόγος που δεν χρησιμοποιούμε τη συγκεκριμένη μέθοδο είναι ότι είναι στατική. Δεν μπορεί να εξελιχθεί με χρήση νέων δειγμάτων και έχει περιορισμένη αποτελεσματικότητα όταν τα σύμβολα αρχίζουν να αποκλίνουν από τα πρότυπα.

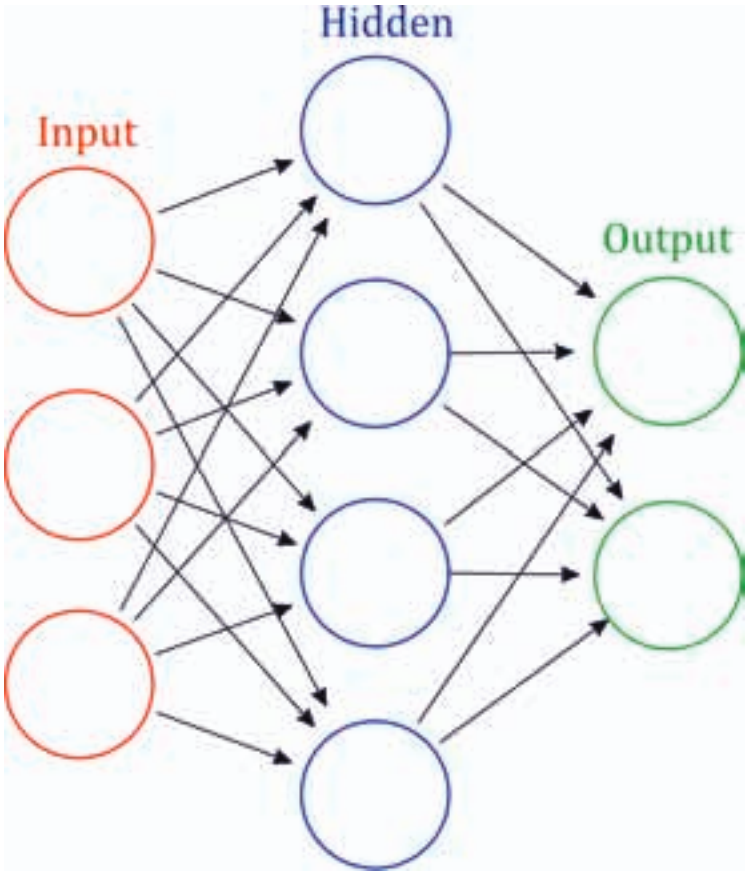
Για αυτούς τους λόγους θα κάνουμε χρήση νευρωνικού δικτύου.

8.2. Νευρωνικό Δίκτυο (Neural Network)

Τα νευρωνικά δίκτυα, συγκεκριμένα τεχνητά νευρωνικά δίκτυα, αποτελούν μια οικογένεια αλγορίθμων οι οποίοι εκπαιδεύονται με στατιστικά δεδομένα. Παίρνουν το όνομα τους από το νευρικό σύστημα των ανθρώπων μιας και μιμούνται τη λειτουργία του. Αντίστοιχα δηλαδή σε κάθε τεχνητό νευρωνικό δίκτυο υπάρχουν μονάδες που ονομάζονται νευρώνες και εκπαιδεύονται προκειμένου να ταξινομούν δεδομένα σε συγκεκριμένες κλάσεις.

Συγκεκριμένα, η εκπαίδευση των νευρωνικών δικτύων είναι ο υπολογισμός πολυωνυμικών συναρτήσεων στο κρυφό στρώμα νευρώνων (8.2.1) (όχι απαραίτητα γραμμικών όπως ισχύει στη μέθοδο των κοντινότερων γειτόνων) που θα ικανοποιούν τα δεδομένα εισόδου και εξόδου του training set, δηλαδή των στοιχείων που έχουμε εισάγει μαζί με την κλάση τους. Επιπλέον υπολογίζονται και τα βάρη (weights) κάθε παραμέτρου που αξιολογείται στο κατά πόσο επηρεάζει το αποτέλεσμα.

Για παράδειγμα στην περίπτωση των κοντινότερων γειτόνων τα βάρη είναι ίδια για κάθε παράμετρο και είναι ίσα με 1/d όπου d η απόσταση του δείγματος και του προτύπου στην διάσταση της εκάστοτε παραμέτρου.



Εικόνα 8.2.1

Με χρήση όμως νευρωνικού δικτύου μπορούμε να αναθέσουμε τον υπολογισμών αυτών των συντελεστών στην διαδικασία της εκπαίδευσης. Εκείνο που απαιτείται όμως προκειμένου να έχουμε καλή παραμετροποίηση του νευρωνικού δικτύου είναι ένα αρκετά μεγάλο δείγμα εκπαίδευσης (training set). Στην εφαρμογή μας μέχρι τώρα έχουμε χρησιμοποιήσει περίπου 2000 στοιχεία τα οποία αν και θεωρούνται λίγα παράγουν ένα ικανοποιητικό αποτέλεσμα.

8.2.1. Επιλογή στατιστικών για την αξιολόγηση κάθε συμβόλου

Ένα μεγάλο training set μπορεί να οδηγήσει σε πολύ κακά αποτελέσματα αν δεν επιλέξουμε προσεκτικά τα δεδομένα που θα τροφοδοτήσουμε στο δίκτυο. Αντίστοιχα καλή επιλογή των δεδομένων μπορεί να οδηγήσει σε μεγάλη ακρίβεια ακόμα και με περιορισμένο training set.

Αναζητούμε δηλαδή μια σειρά από αριθμούς για κάθε εικόνα που να μπορεί να περιγράφει την ομοιότητά της με άλλες της ιδίας κλάσης αλλά και τη διαφορά της με εκείνες άλλων κλάσεων.

Θα χρησιμοποιήσουμε αρχικά τις παραμέτρους ύψος, πλάτος και εμβαδό. Από μόνες τους αυτές παράγουν μια πρώτη διαφοροποίηση των συμβόλων μιας και με πολύ σύντομα το νευρωνικό δίκτυο θα μπορέσει να ξεχωρίσει τις παύσεις και τον οπλισμό από τα κλειδιά του "σολ" και το "φα".

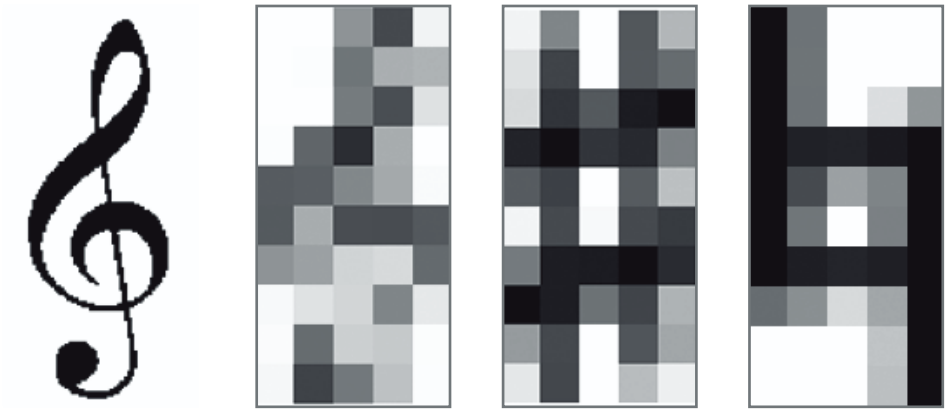
Να σημειώσουμε ότι αυτές οι τρεις παράμετροι εισάγονται ως στατιστικά του δείγματος διαιρεμένες με την παράμετρο staff_distance προκειμένου να είναι κανονικοποιημένες στις διαστάσεις της κάθε σελίδας.

Ωστόσο αυτά τα μεγέθη δεν αρκούν μιας και η δίεση από την αναίρεση δεν απέχουν σχεδόν καθόλου ως προς το ύψος και το πλάτος τους και ελάχιστα ως προς το εμβαδό.

Αποφασίσαμε λοιπόν να εισάγουμε όσο το δυνατόν περισσότερη πληροφορία ως προς την εικόνα κάθε συμβόλου και συγκεκριμένα εισάγαμε την ίδια την εικόνα. Τα στατιστικά κάθε δείγματος είναι λοιπόν το ύψος, το πλάτος, το εμβαδό και ένας πίνακας 50 στοιχείων (διαστάσεων 1x50) ο οποίος είναι η παράθεση των 10 γραμμών που αποτελούνται από 5 στοιχεία η καθεμία της εικόνα **Is.**

Όπου **Is** έχουμε ορίσει τη μεγέθυνση της εικόνας του δείγματος προκειμένου να έχει διαστάσεις 10 x 5. Επιπλέον κατά τη μεγέθυνση της εικόνας, μετατρέπουμε την εικόνα από δυαδική σε πραγματική.

Στην ουσία, με την παραπάνω διαδικασία χωρίζουμε κάθε εικόνα σε ένα πλέγμα 10 x 5 και αποθηκεύουμε το άθροισμα των μαύρων στοιχείων του κάθε τμήματος, διαδικασία που ισοδυναμεί με τη μεγέθυνση της εικόνας στον πραγματικό χώρο.



Βλέπουμε την πληροφορία που εισάγεται ως κλειδί του σολ. Παράλληλα με τις διαστάσεις οδηγεί σε μεγάλη ακρίβεια εντοπισμού.

Ενδεικτικά μπορούμε να δούμε ότι όσο εκπαιδεύουμε το δίκτυο, θα υπολογιστεί ότι η τιμή των pixels πάνω δεξιά και κάτω αριστερά παίζει καθοριστικό ρόλο ως προς την ταξινόμηση μεταξύ δίεσης και αναίρεσης σε αντίθεση με τις άλλες τιμές που είναι περίπου ίδιες στην προκειμένη περίπτωση. Αυτές οι τιμές είναι τα βάρη, οι συντελεστές δηλαδή που έχει η κάθε παράμετρος ως προς τον επικαθορισμό της τελικής επιλογής.

8.3. Κατάτμηση και ταξινόμηση

Όπως αναφέραμε στο κεφάλαιο της κατάτμησης θα εφαρμόσουμε μια συνδυαστική μέθοδο δυο κατατμήσεων. Αρχικά αφαιρούμε τις γραμμές του πενταγράμμου χωρίς κανένα κριτήριο (8.3.1). Να σημειώσουμε ότι από την εικόνα έχουμε αφαιρέσει όλα τα στοιχεία που έχουμε εντοπίσει πριν.

Εικόνα 8.3.1

Στο πρώτο στάδιο της κατάτμησης παρατηρούμε ότι τα σύμβολα του οπλισμού (διέσεις, υφέσεις και αναιρέσεις) όπως και οι παύσεις είναι ανέπαφα. Δίνουμε λοιπόν τα στατιστικά όλων των στοιχείων στο νευρωνικό δίκτυο και κρατάμε μόνο εκείνα τα οποία με μεγάλη βεβαιότητα (μεγαλύτερη του 80%) ανήκουν στις κλάσεις των παύσεων και του οπλισμού (8.3.2).

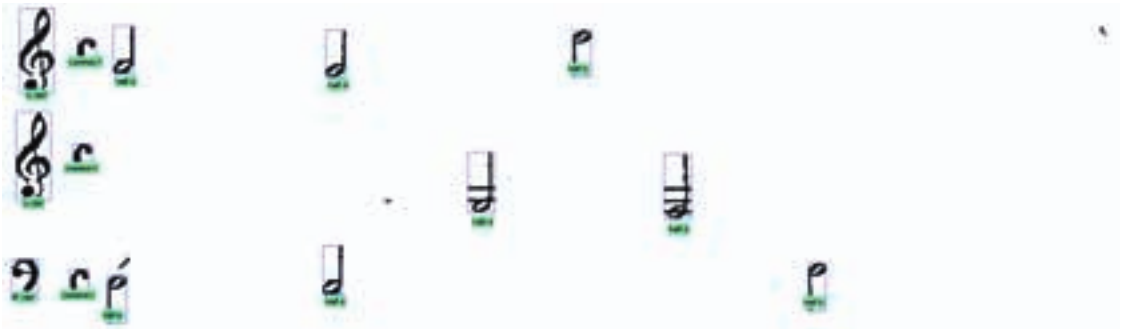
Εικόνα 8.3.2

Στο επόμενο βήμα αφαιρούμε τα στοιχεία που εντοπίσαμε και ενώνουμε αυτά που έχουν απομείνει (αν τα ενώνουν γραμμές του πενταγράμμου εξαρχής) με κριτήριο την αλληλοεπικάλυψη τους, την απόστασή τους και το μέγεθος του τελικού ενωμένου συμβόλου. Εφαρμόζοντας αυτά τα κριτήρια παίρνουμε την παρακάτω εικόνα. (8.3.3).

Εικόνα 8.3.3

Με αυτό τον τρόπο όλα τα στοιχεία είναι συνδεδεμένα και μπορούμε να τα αξιολογήσουμε σωστά. Η κάτω γραμμή των κλειδιών του σολ δεν αποκαταστάθηκε γιατι τα άκρα της ανήκουν στο ίδιο σύμβολο. Επίσης, η επιπλέον γραμμή στο κλειδί του φα δεν επηρεάζει το αποτέλεσμα. Προχωρούμε τώρα και στην τελική ταξινόμηση των στοιχείων αυτής της εικόνας για να πάρουμε

το παρακάτω αποτέλεσμα.(8.3.4).

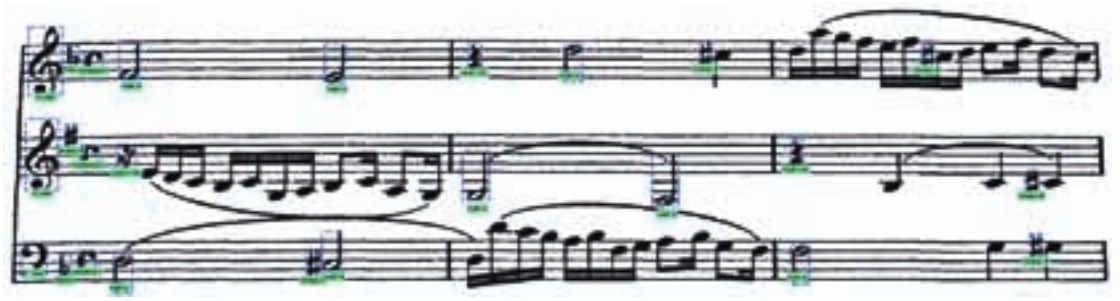


Εικόνα 8.3.4

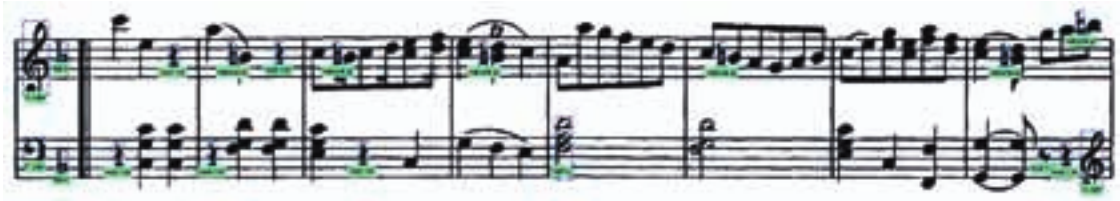
Βλέπουμε το νευρωνικό δίκτυο μας δίνει με μεγάλη σιγουριά και ακρίβεια τα αποτελέσματα της ταξινόμησης. Η διαδικασία της κατάτμησης και ταξινόμησης δύο σταδίων είναι λίγο περισσότερο δαπανηρή ως προς τον χρόνο αλλά μας εξασφαλίζει το γεγονός ότι ο ταξινομητής δεν θα πάρει σπασμένα σύμβολα και άρα θα δίνει καλύτερα αποτελέσματα.

Να σημειώσουμε σε αυτό το σημείο ότι για τις μισές νότες χρειάστηκε να εκπαιδεύσουμε το νευρωνικό δίκτυο με πολλές περιπτώσεις κατά τις οποίες συναντώνται. Παρατηρήσαμε επίσης σφάλματα σε κάποιες περιπτώσεις μισών νοτών αλλά εκτιμούμε ότι οφείλονται στο μικρό εύρος των δειγμάτων που δόθηκαν κατά την εκπαίδευση.

Συνοψίζουμε λοιπόν το αποτέλεσμα της ταξινόμησης και παραθέτουμε μερικά ακόμα δείγματα εφαρμογής της (8.3.5-7).



Εικόνα 8.3.5



Εικόνα 8.3.6

Εικόνα 8.3.7

9
Μουσική Σημειολογία

9.1. Εισαγωγή

Το τελευταίο στάδιο αναγνώρισης μιας παρτιτούρας είναι η εφαρμογή της μουσικής σημειολογίας (music semantics). Σε αυτό το στάδιο αποδίδουμε νόημα στα στοιχεία που έχουμε εντοπίσει αναγνωρίζοντας τη νοηματική σχέση που υπάρχει μεταξύ τους.

Ένα από τα στάδια αυτής της διαδικασίας το έχουμε υλοποιήσει ήδη αφού έχουμε εντοπίσει τις αξίες που αποδίδουν τα beams στις νότες. Βέβαια μένει τώρα να αντιστοιχήσουμε τις μουσικές αξίες (μισό, τέταρτο, ολόκληρο κλπ.) σε πραγματικές χρονικές διάρκειες συμπεριλαμβάνοντας και το τέμπο που έχουμε εντοπίσει.

Το πιο κρίσιμο στάδιο ωστόσο είναι ο εντοπισμός της ακριβής τονικότητας. Να θυμίσουμε ότι μέχρι τώρα απλά γνωρίζουμε τον αριθμό των ημιδιαστημάτων κατά τον οποίο απέχει μια νότα από τη μεσαία γραμμή του πενταγράμμου στο οποίο ανήκει αυτή η νότα. Σε αυτό το κεφάλαιο θα περιγράψουμε πως εξάγεται εξολοκλήρου ο τόνος μιας νότας συμπεριλαμβάνοντας το κλειδί, τον οπλισμό του μέτρου και τον οπλισμό κάθε νότας ξεχωριστά.

Η διαδικασία αυτή γίνεται ξεχωριστά για κάθε ομάδα πενταγράμμων που εκτελούνται παράλληλα. Για τα επόμενα βήματα χρησιμοποιούμε μια συγκεκριμένη δομή που φτιάξαμε η οποία είναι ένας πίνακας (με το όνομα element) που περιέχει όλες τις νότες και τις παύσεις (τα στοιχεία δηλαδή που έχουν χρονική αξία) διατεταγμένα σύμφωνα με την οριζόντια θέση τους. Επιπλέον σε αυτόν τον πίνακα περιλαμβάνονται όλα τα στοιχεία που θα χρειαστούμε στη συνέχεια όπως το πεντάγραμμο στο οποίο αντιστοιχούν, η σχετική θέση τους σε αυτό κλπ.

X Position	Y Position	Type(1=Note,2=Rest)	Staff	Staff_space
***	***	***	***	***
***	***	***	***	***

Η μεταβλητή Element

Παράλληλα κάνουμε χρήση μιας δεύτερης μεταβλητής (time_steps) με μήκος ίδιο με εκείνο του πίνακα element η οποία με κριτήριο ένα όριο οριζόντιας απόστασης ομαδοποιεί στα στοιχεία του πίνακα element σε ομάδες που εκείνων των στοιχείων που εκτελούνται ταυτόχρονα.

Τέλος έχουμε ακόμα τη μεταβλητή bar_changes που αντιστοιχεί σε κάθε στοιχείο τον αριθμό του μέτρου στο οποίο ανήκει αυτό το στοιχείο.

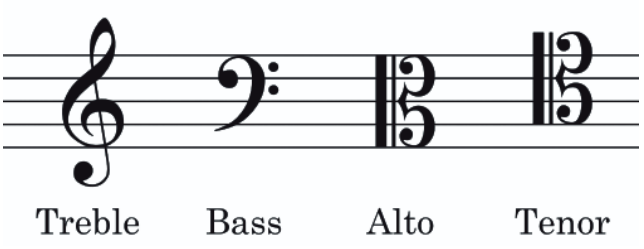
Time Steps
1
1
2
3
...

Bar Changes
1
1
1
2
...

9.2. Εντοπισμός του κλειδιού

Τα κλειδιά είναι σύμβολα τα οποία τοποθετούνται σίγουρα στην αρχή κάθε πενταγράμμου και ενδεχομένως σε σημεία στο εσωτερικό του. Όπως περιγράψαμε παραπάνω καθορίζουν την τονικότητα που αντιστοιχεί σε κάθε γραμμή ή διάστημα του πενταγράμμου και άρα στις νότες που βρίσκονται σε αυτό.

Υπάρχουν τρία είδη κλειδιών, το κλειδί του σολ, το φα και του ντο με το τελευταίο να χρησιμοποιείται πιο σπάνια. Συνήθως η κάθετη θέση των κλειδιών είναι προκαθορισμένη με το κλειδί του σολ να ακουμπάει η άκρη του στην 4 γραμμή (από πάνω), το κλειδί του φα να ξεκινάει από την πρώτη γραμμή και οι δυο τελείες να είναι εκατέρωθεν της δεύτερης γραμμής και το κλειδί του ντο να καλύπτει όλες τις γραμμές. Ωστόσο υπάρχουν περιπτώσεις, αρκετά σπάνιες, που τα κλειδιά έχουν άλλες θέσεις σε σχέση με το πεντάγραμμο και άρα ορίζουν την τονικότητα των γραμμών με άλλο τρόπο.



Θα ασχοληθούμε μόνο με τις πρώτες δυο περιπτώσεις των κλειδιών που είναι και οι πιο συχνές. Για κάθε μέτρο αναζητούμε τα κλειδιά τα οποία αντιστοιχούν σε αυτό το πεντάγραμμο και τα διατάσσουμε σε αύξουσα σειρά με κριτήριο την τετμημένη τους (την οριζόντια θέση τους).

Δημιουργούμε ένα πίνακα (clef change) με ύψος ίσο με εκείνο του element και με πλάτος τον αριθμό των πενταγράμμων. Σύμφωνα με τη διάταξη των κλειδιών γεμίζουμε τον πίνακα με τέτοιο τρόπο προκειμένου κάθε στοιχείο element να αντιστοιχεί ή στον αριθμό 1 (κλειδί του σολ), ή στον αριθμό 2 (κλειδί του φα).(9.2.1)

Εικόνα 9.2.1

St.1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
St.2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1

ο πίνακας ***clef change*** *της 9.2.1*

St.1-2	1	1	1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3

ο πίνακας ***bar change*** *της 9.2.1*

Μέχρι στιγμής έχουμε όλη την πληροφορία που χρειαζόμαστε σχετικά με τα κλειδιά της παρτιτούρας. Συνεχίζουμε με τον εντοπισμό του οπλισμού του κομματιού.

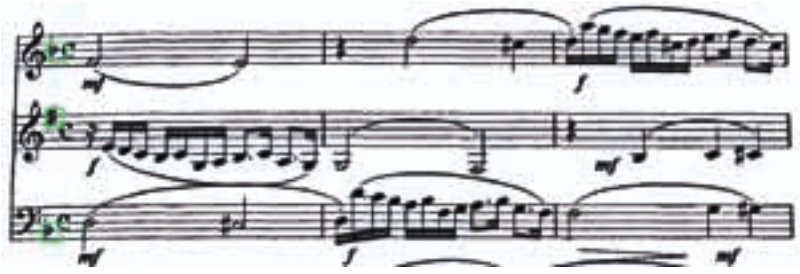
9.3. Εντοπισμός της κλίμακας

Η κλίμακα κάθε κομματιού ή τμημάτων αυτού περιγράφεται από τον οπλισμό που υπάρχει στην αρχή κάθε πενταγράμμου ή και στο εσωτερικό του. Η διάταξη του οπλισμού είναι προκαθορισμένη γεγονός που μας βοηθάει να εντοπίσουμε με μεγάλη σιγουριά τις κλίμακες μέσα στο κομμάτι.

Θα δημιουργήσουμε ένα πίνακα **key change** αντίστοιχο με τον **clef change** μόνο που τώρα οι τιμές που θα λαμβάνει θα εκτείνονται από -7 εώς 7. Το μηδέν συμβολίζει την απουσία οπλισμού και άρα την κλίμακα Ντο ματζόρε όπου όλες η νότες έχουν τη φυσική τονικότητά τους. Οι θετικές τιμές αντιστοιχούν στο αριθμό διέσεων (πχ 1=Σολ ματζόρε, 2=Ρε ματζόρε κοκ.) ενώ οι αρνητικές στο αριθμό υφέσεων (-1=Φα ματζόρε, -2=Σι ύφεση ματζόρε κοκ.).

Για να γεμίσουμε λοιπόν αυτόν τον πίνακα αρχικά απομονώνουμε όλες τις διέσεις και υφέσεις από αυτές που έχουμε εντοπίσουμε οι οποίες δεν αντιστοιχούν σε κάποια νότα. Στη συνέχεια τις ομαδοποιούμε ανάλογα με τις αποστάσεις τους έτσι ώστε σε κάθε ομάδα να έχουμε αυτές που βρίσκονται κοντά μεταξύ τους. Αφού τις ομαδοποιήσουμε και τις διατάξουμε σύμφωνα με την οριζόντια θέση τους ελέγχουμε τις κάθετες θέσεις τους.

Αν είναι διέσεις πρέπει να ακολουθούν το μοτίβο **[4,1,5,2,-1,3,0]** όπου κάθε αριθμός αντιστοιχεί στον αριθμό ημιδιαστημάτων κατά τον οποίο απέχει κάθε δίεση από τη μεσαία γραμμή. Τα διαστήματα αυτά αντιστοιχούν στις νότες **[φα, ντο, σολ, ρε, λα, μι, σι].** Στη περίπτωση που έχουμε υφέσεις ο αντίστοιχος πίνακας είναι ο **[0,3,-1,2,-2,1,4]** με την ίδια λογική ακριβώς. Επιπλέον από αυτές τις μεταβλητές αφαιρούμε την τιμή **(curr_cl-1)*2** όπου curr_cl είναι η τιμή του πίνακα clef_change στο σημείο που αναζητούμε τον οπλισμό. Αυτό το κάνουμε διότι αν έχουμε κλειδί του φα και άρα curr_cl=2 τα διαστήματα μετατοπίζονται κατά -2. Με αυτόν τον τρόπο γεμίζουμε τον πίνακα **key_change** που μας δίνει την κλίμακα κάθε πενταγράμμου σε κάθε σημείο (9.3.1).



Εικόνα 9.3.1

St.1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
St.2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
St.3	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

*ο πίνακας **key change** της 9.3.1*

***Σημ:** Συνήθως όλα τα μέτρα έχουν την ίδια κλίμακα. Στο προηγούμενο παράδειγμα βλέπουμε μια αρμονική αναντιστοιχία. Ο αλγόριθμος έχει εντοπίσει σωστά αποτελέσματα μόνο που στην ουσία το δεύτερο πεντάγραμμο αντιστοιχεί σε κλαρινέτο σε σι ύφεση ματζόρε το οποίο όταν παίζει σε σολ ματζόρε στην ουσία παίζει σε φα ματζόρε όπως και οι άλλες φωνές. Δυστυχώς δεν μπορούμε να ενσωματώσουμε αυτή την πληροφορία στο πρόγραμμα οπότε θα παραχθεί λάθος ηχητικό. Ωστόσο η μέθοδος λειτουργεί σωστά.*

9.4. Εντοπισμός οπλισμού στις νότες

Τελευταίο στάδιο για να ολοκληρώσουμε τον εντοπισμό της τονικότητας των νοτών είναι η πρόσθεση του οπλισμού των νοτών στις νότες τις οποίες αναλογεί. Σε αυτό το στάδιο δεν λαμβάνουμε υπόψιν την κλίμακα καθώς θα ενώσουμε όλες τις πληροφορίες στο τελευταίο στάδιο.

Σε αυτό το στάδιο καταγράφουμε τις αλλοιώσεις των νοτών που υφίστανται στο εσωτερικό του κομματιού και όχι λόγω της κλίμακας.

Η μουσική θεωρία περιγράφει ότι αν μια νότα αλλοιώνεται, τότε η ίδια αλλοίωση ισχύει για όλες τις νότες ίδιας τονικότητας που ακολουθούν στο ίδιο μέτρο. Θα εφαρμόσουμε ακριβώς αυτόν τον κανόνα στο πρόγραμμά μας.

Χρησιμοποιούμε δύο πίνακες **(intone1, intone2),** που έχουν μέγεθος ίσο με τον αριθμό των νοτών σε κάθε τμήμα που μελετάμε. Ο πρώτος πίνακας έχει τιμή 0 αν η νότα δεν έχει οπλισμό, 1 για δίεση, 2 για ύφεση και 3 για αναίρεση. Ο δεύτερος πίνακας έχει αρχικά τιμή μηδέν σε όλο το μήκος του.

Ξεκινώντας σαρώνουμε όλες τις νότες με σειρά σύμφωνα με την οριζόντια θέση τους. Όταν εντοπίσουμε μια νότα η οποία έχει μη μηδενική τιμή στον πίνακα intone1 τότε:

1. Εντοπίζουμε όλες τις νότες που έχουν ίδια τονικότητα με αυτή που εξετάζουμε, βρίσκονται στο ίδιο μέτρο (δηλαδή έχουν ίδια τιμή στον πίνακα bar_change) και έχουν μεγαλύτερη τετμημένη από την τρέχουσα.
2. Αν η τιμή intone1 των νοτών που βρήκαμε είναι μηδέν τότε στην αντίστοιχη θέση του πίνακα intone2 θέτουμε την τιμή του οπλισμού της τρέχουσας νότας. Εξετάζουμε την τιμή του intone1 διότι αν δεν είναι μηδέν σημαίνει ότι η νότα έχει ήδη δικό της οπλισμό και άρα δεν ισχύει ο προηγούμενος
3. Επαναλαμβάνουμε αυτή τη διαδικασία μέχρι να τελειώσουν οι νότες.

Ο τελικός πίνακας intone που περιλαμβάνει τις τελικές αλλοιώσεις των νοτών είναι η ένωση των δύο πινάκων κρατώντας σε κάθε θέση το μη μηδενικό στοιχείο εκ των δύο αν υπάρχει.

9.5. Προσδιορισμός χρονικής διάρκειας

Τελευταίο στάδιο είναι ο προσδιορισμός της χρονικής διάρκειας κάθε νότας. Μέχρι αυτό το σημείο οι αξίες των νοτών και των παύσεων έχουν τιμές 4 για ολόκληρη, 2 για μισή, 1 για τέταρτο, ½ για όγδοο κοκ. Αυτοί οι αριθμοί είναι ο αριθμός των τετάρτων που αντιστοιχεί σε κάθε νότα.

Επιπλέον έχουμε εντοπίσει το τέμπο του κομματιού σε προηγούμενο στάδιο και έχουμε τιμές που αντιστοιχούν σε BPM, δηλαδή χτύπους ανά λεπτό. Η ακριβής διάρκεια κάθε νότας αντιστοιχεί στην τιμή note_value*60/tempo. Αν δηλαδή η νότα έχει αξία 1 και το τέμπο είναι 120 η διάρκειά της είναι μισό δευτερόλεπτο.

Χρειαζόμαστε αυτές τις αξίες διότι η παραγωγή του αρχείου midi απαιτεί απόλυτο προσδιορισμό της διάρκειας της νότας και όχι σχετικό. Επιπλέον σε περίπτωση που δεν εντοπιστεί τέμπο, έχουμε θέσει ένα προκαθορισμένο στα 80 bpm.

9.6. Παραδείγματα εφαρμογής της μουσικής σημειολογίας

Τέλος παραθέτουμε μερικά παραδείγματα εφαρμογής όλων των κανόνων που περιγράψαμε παραπάνω.

Εικόνα 9.6.1

Ενδεικτικά στην εικόνα (9.6.1) βλέπουμε πως επιδρά η αλλαγή κλειδιών στην τονικότητα

Εικόνα 9.6.2

Αντίστοιχα στην εικόνα (9.6.2) πετυχημένα εντοπίζονται και εφαρμόζονται δύο αλλαγές κλειδιών. Παρατηρούμε επιπλέον πως οι νότες ντο φα και σολ (C F G) αλλοιώνονται σύμφωνα με την κλίμακα.

Εικόνα 9.6.3

Εδώ βλέπουμε πως στο πάνω πεντάγραμμο η νότα σι (B) αρχικά επαυξάνεται και έπειτα αναιρείται (9.6.3).

Εικόνα 9.6.4

Τέλος στην εικόνα (9.6.4) βλέπουμε αντίστοιχα παραδείγματα διαδοχικών αλλοιώσεων που έχουν εντοπιστεί σωστά.

10
Πειραματικά Αποτελέσματα

10.1. Εισαγωγή

Θα δοκιμάσουμε το σύστημα μας σε 4 σελίδες που καλύπτουν όλο το εύρος των στοιχείων που περιέχει μια παρτιτούρα. Η μουσική τυπογραφία αυτών των σελίδων είναι η πιο συνηθισμένη (σχεδόν καθολική) για τις παρτιτούρες του πιάνου. Επιπλέον υπάρχει (περιορισμένη μεν) φθορά κάποιων συμβόλων λόγω εκτύπωσης, γεγονός που δοκιμάζει επιπλέον το σύστημά μας.

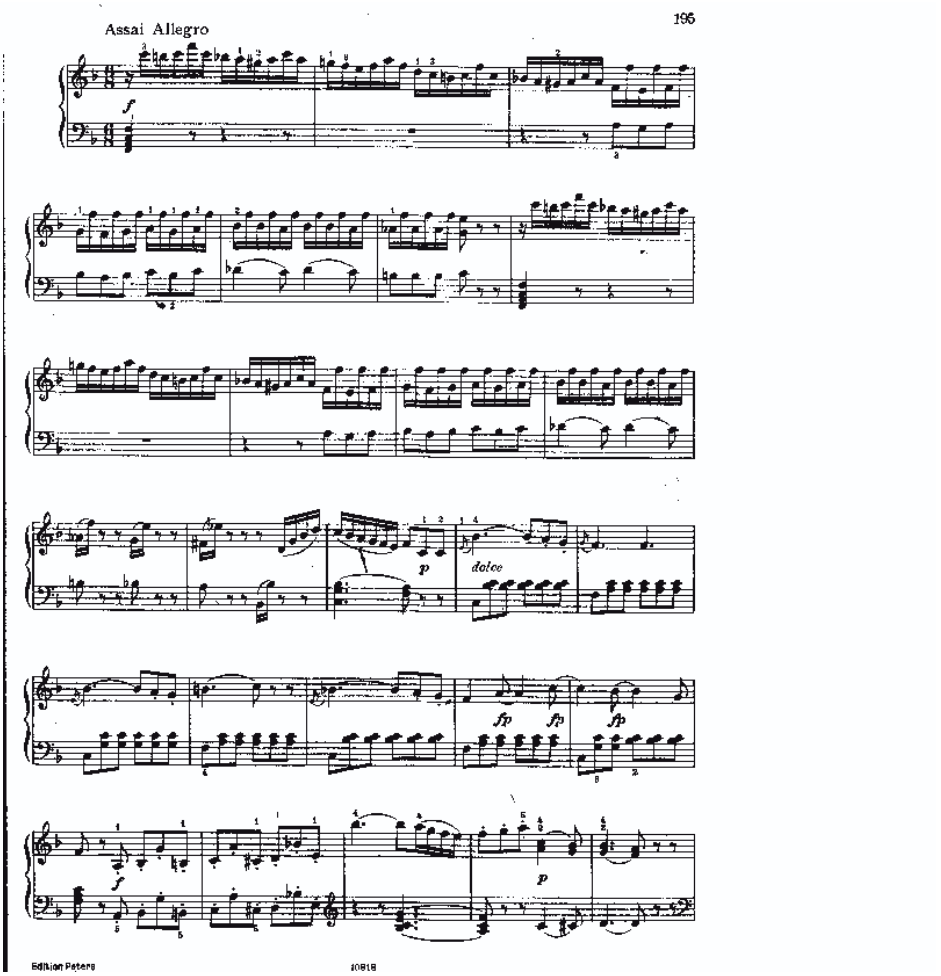
Ο πίνακας των στοιχείων που καταμετρούμε είναι:

1. **Full Head Notes Tonality:** Είναι ο αριθμός των νοτών αξίας τετάρτου η μικρότερης που εντοπίστηκαν στην σωστή τονικότητα
2. **Full Head Notes Duration:** Η διάρκεια αυτών των νοτών
3. **Half Notes:** Οι νότες αξίας μισού
4. **Whole Notes:** Οι νότες αξίας ολοκλήρου
5. **G Clefs:** Τα κλειδιά του σολ
6. **F Clefs:** Τα κλειδιά του φα
7. **Scales:** Η κλίμακα, δηλαδή ο εντοπισμού του οπλισμού και της κλίμακας που αντιπροσωπεύει
8. **Sharps:** Οι διέσεις των νοτών
9. **Flats:** Οι υφέσεις των νοτών
10. **Naturals:** Οι αναιρέσεις των νοτών
11. **Rests:** Οι παύσεις

Δεν συμπεριλάβαμε στοιχεία όπως οι συζεύξεις και οι δυναμικές μιας και δεν τα αναλύουμε εκτενώς στο σύστημά μας μέχρι τώρα αλλά και επειδή δεν παίζουν βασικό ρόλο στην διαδικασία της αναγνώρισης των παρτιτούρων.

10.2. Αποτελέσματα μετρήσεων

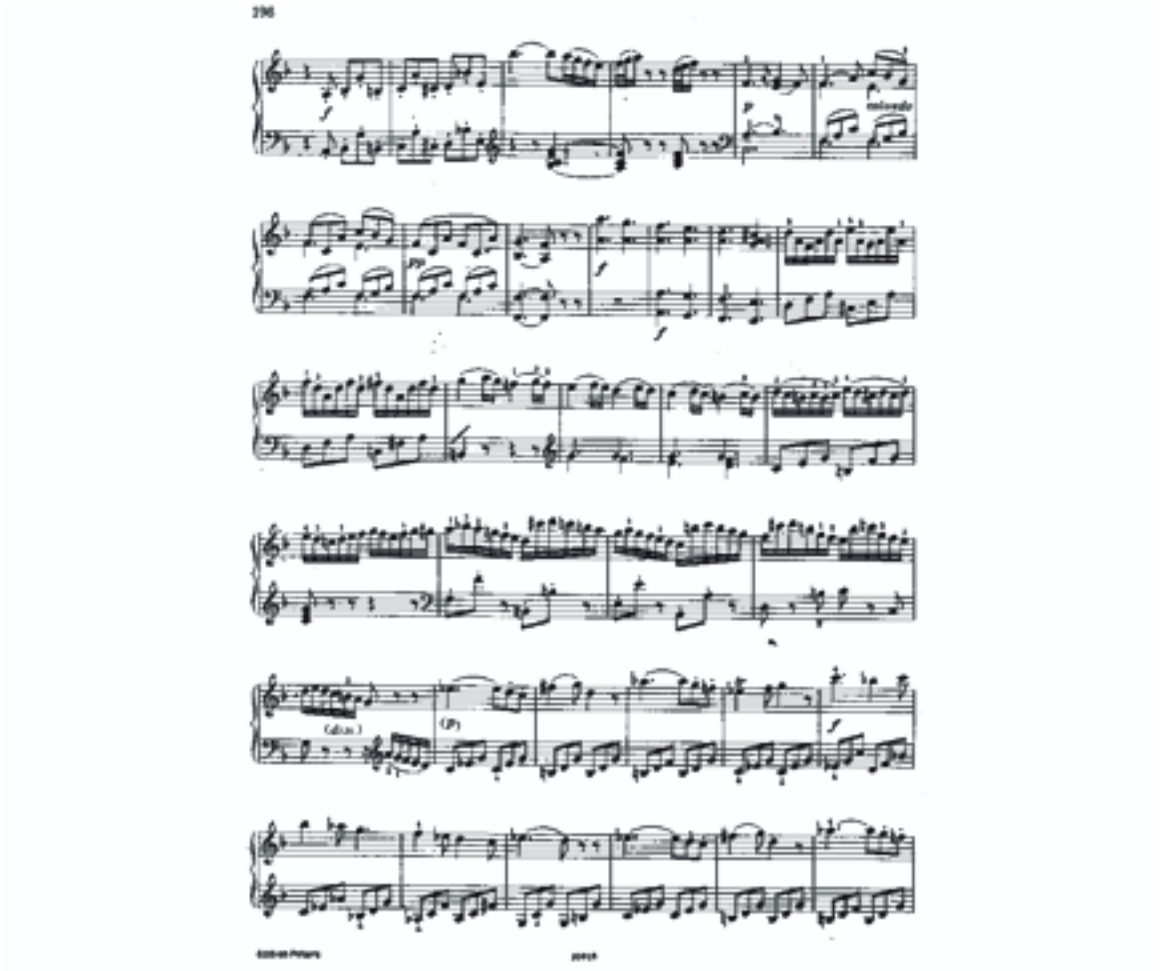
Εικόνα: Mozart5_1.png
Ανάλυση: 300dpi



mozart5_1.png

	Existing	Found	Percent (%)
FH Notes Tonality	347	347	100
FH Notes Duration	347	347	100
Half Notes	0	0	0
Whole Notes	0	0	0
G Clefs	7	7	100
F Clefs	7	7	100
Scale	12	12	100
Sharps	8	7	87,5
Flats	13	11	84,62
Naturals	11	11	100
Rests	46	45	97,83
Total	798	794	99,5

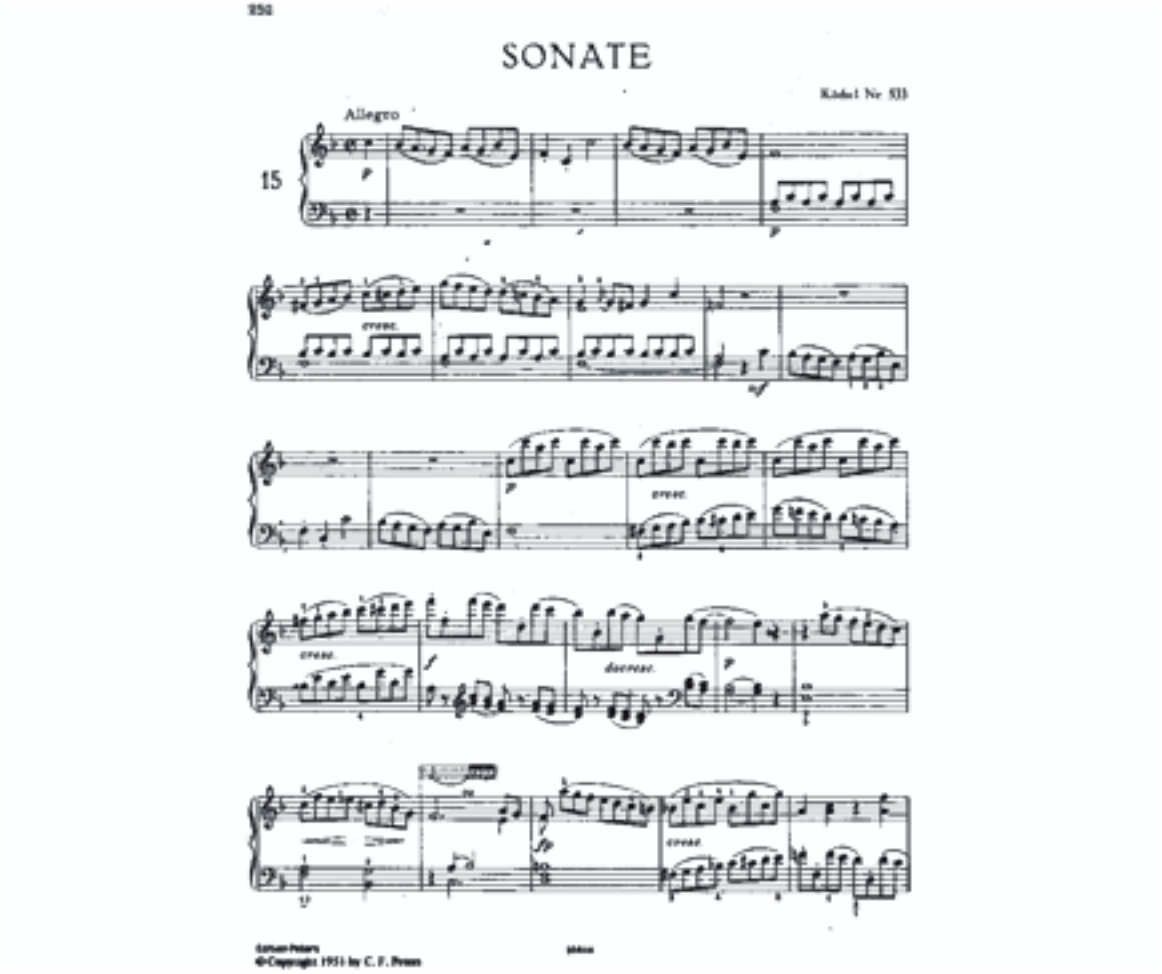
Εικόνα: Mozart5_2.png
Ανάλυση: 300dpi



mozart5_2.png

	Existing	Found	Percent (%)
FH Notes Tonality	375	373	99,47
FH Notes Duration	375	373	99,47
Half Notes	1	1	100
Whole Notes	0	0	0
G Clefs	11	11	100
F Clefs	6	6	100
Scale	12	12	100
Sharps	13	13	100
Flats	21	21	100
Naturals	26	25	96,15
Rests	40	40	100
Total	880	875	99,43

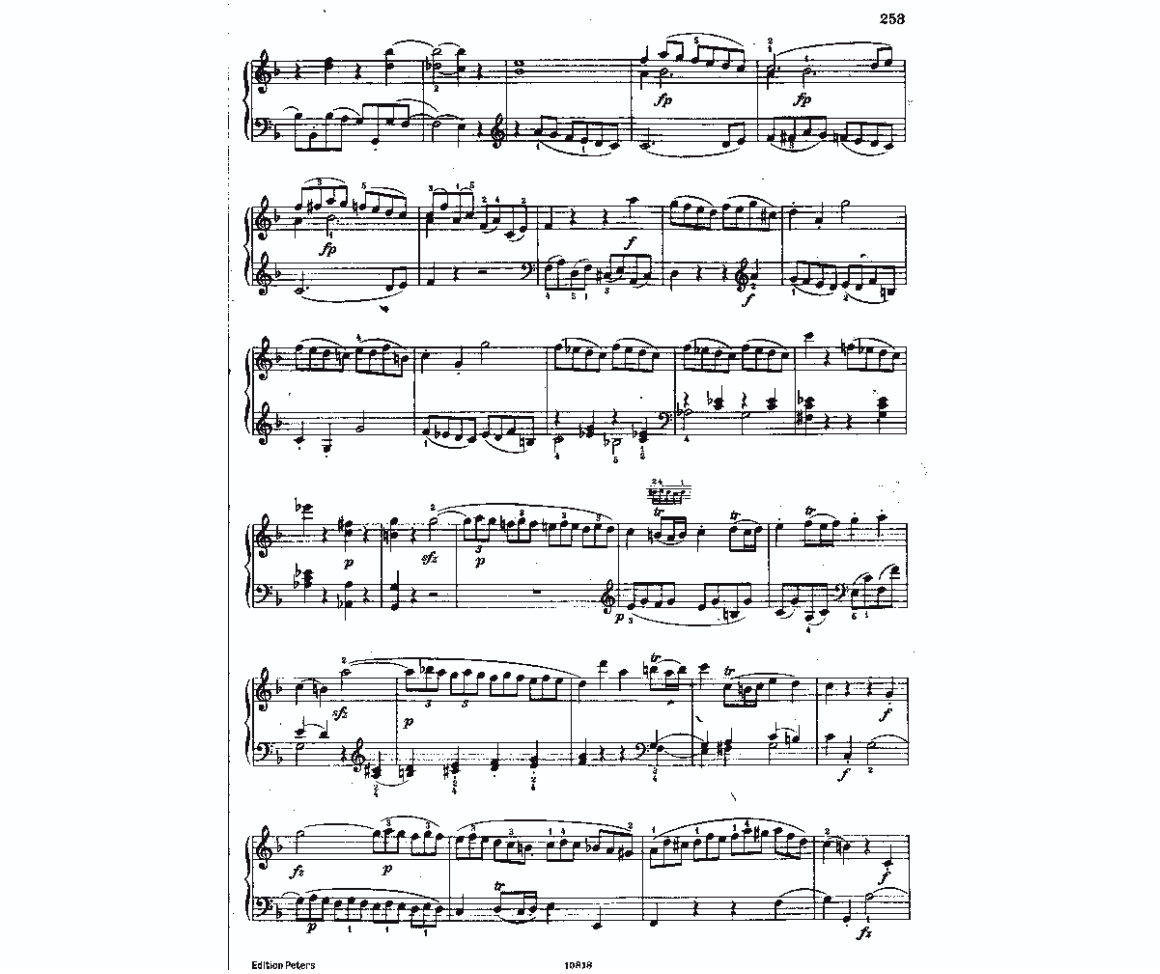
Εικόνα: Mozart3_1.png
Ανάλυση: 300dpi



mozart3_1.png

	Existing	Found	Percent (%)
FH Notes Tonality	245	245	100
FH Notes Duration	245	245	100
Half Notes	16	16	100
Whole Notes	12	12	100
G Clefs	7	7	100
F Clefs	6	6	100
Scale	10	10	100
Sharps	10	10	100
Flats	3	3	100
Naturals	6	6	100
Rests	17	15	88,24
Total	577	575	99,65

Εικόνα: Mozart3_2.png
Ανάλυση: 150dpi



mozart3_2.png

	Existing	Found	Percent (%)
FH Notes Tonality	311	311	100
FH Notes Duration	311	311	100
Half Notes	22	19	84,21
Whole Notes	2	2	100
G Clefs	12	12	100
F Clefs	8	7	88,89
Scale	12	10	83,33
Sharps	12	11	91,67
Flats	12	10	83,33
Naturals	18	15	83,33
Rests	34	34	100
Total	754	742	98,41

10.3. Σχολιασμός αποτελεσμάτων

Τα αποτελέσματα των μετρήσεων είναι αρκετά καλά δεδομένου ότι προκύπτει ελάχιστη απώλεια πληροφορίας. Το ποσοστό επιτυχίας ορίζεται ως προς την αναγνώριση δεδομένων και όχι ως προς το πόσο σωστά ακούγεται τελικά το κομμάτι όταν ανακατασκευαστεί.

Ένα άλλο μέτρο καταγραφής της επιτυχίας ενός OMR θα μπορούσε να είναι και ο χρόνος που εκτελείται σωστά το κομμάτι σε σχέση με τη διάρκεια του κομματιού. Για παράδειγμα αν σε μια νότα τετάρτου δεν εντοπιστεί σωστά ο οπλισμός τότε στο συνολικό σφάλμα προστίθεται η χρονική διάρκεια αυτής της νότας. Αυτό το κριτήριο είναι μεν πολύ πιο αυστηρό από τα τεχνικά κριτήρια που εφαρμόσαμε (διότι για παράδειγμα ένα λάθος κλειδί θα κοστίσει όλη τη γραμμή) αλλά είναι πολύ πιο κοντινό στο χρήστη και στην διαισθητική αντίληψη του ηχητικού αποτελέσματος.

Τα σφάλματα που παρουσιάζει το σύστημά μας κυρίως αφορούν 3 κατηγορίες περιπτώσεων:

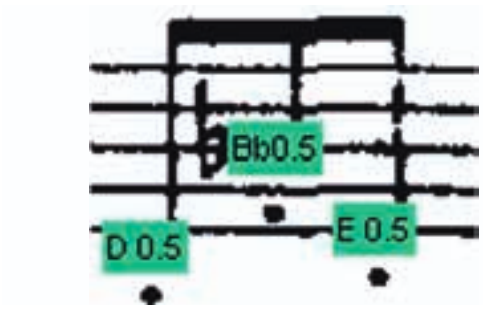
1. **Συνδεδεμένα σύμβολα.** Συχνά στις παρτιτούρες εμφανίζονται σύμβολα συνδεδεμένα μεταξύ τους που «τυπικά» δεν θα έπρεπε να είναι. Συνήθως αυτό συμβαίνει με τις συζεύξεις αλλά και με τον οπλισμό.

Εικόνα 10.3.1

Στην εικόνα (10.3.1) βλέπουμε ένα ενδεικτικό σφάλμα συνδεδεμένων στοιχείων. Η σύζευξη ακουμπάει την αναίρεση και έτσι η νότα σι διατηρεί την αλλοίωσή της ενώ δεν θα έπρεπε. Άλλες μορφές συνδεδεμένων στοιχείων είναι η ύπαρξη δύο διέσεων μαζί.

Υπάρχουν διάφοροι τρόποι ανάκτησης τέτοιου είδους σφαλμάτων. Μπορεί να χρησιμοποιηθεί διευρυμένο training set (όχι όμως για το παράδειγμα της σύζευξης), μπορούμε να εφαρμόσουμε template matching στα στοιχεία που έχουμε εντοπίσει ως συνδεδεμένα ή να εφαρμόσουμε περαιτέρω segmentation. Το πλέον ενδεικτικό για το παράδειγμα της εικόνας είναι η εφαρμογή segmentation στην εικόνα των ενωμένων στοιχείων και η ταξινόμηση του αποτελέσματος.

2. **Σφάλματα λόγω φθοράς.** Σε πολλές περιπτώσεις αντί για συνδεδεμένα σύμβολα προκύπτουν σπασμένα σύμβολα που οφείλονται είτε αρχικά στην ποιότητα της σελίδας, είτε στη διαδικασία της δυαδικοποίησης. Στα δικά μας παραδείγματα τα σφάλματα οφείλονται στην αρχική εικόνα μιας και εφαρμόζουμε δυαδικοποίηση με χαμηλό threshold και άρα παίρνουμε γενικά μεγάλες πυκνότητες. Ενώ έχουμε καταφέρει να ανακατασκευάσουμε σπασμένα stems (10.3.2) όταν η φθορά είναι κοντά στην κεφαλή τότε συνήθως έχουμε σφάλμα (10.3.3).



Εικόνα 10.3.2

Εικόνα 10.3.3

Γενική λύση στο πρόβλημα της φθοράς δεν υπάρχει μιας και το πρόβλημα είναι απρόβλεπτο. Ωστόσο στη συγκεκριμένη περίπτωση θα μπορούσαμε να αναζητήσουμε αν όλα τα stems κατά μήκος ενός beam αντιστοιχούν σε νότα, και αν όχι να τα προεκτείνουμε μέχρι να ενωθούν με μία. Γενικά όμως το ζητούμενο είναι όσο το δυνατόν καλύτερη και λιγότερο φθαρμένη αρχική εικόνα.

3. **Λάθη του ταξινομητή.** Η τελευταία κατηγορία αφορά τα λάθη που έγιναν κατά την ταξινόμηση. Στην περίπτωσή μας είναι ελάχιστα και αφορούν κυρίως τις νότες αξίας μισού. Αυτό συμβαίνει διότι σε αντίθεση με όλα τα άλλα σύμβολα που μπαίνουν στη διαδικασία της ταξινόμησης, οι μισές νότες έχουν μη προκαθορισμένο σχήμα. Ο ταξινομητής μας έχει εκπαιδευθεί σε αρκετές περιπτώσεις συμβόλων αλλά προφανώς υπάρχουν περιθώρια βελτίωσης.

Συνοψίζοντας μπορούμε να πούμε ότι τα σφάλματα που εμφανίζονται είναι λογικό να δυσκολεύουν το πρόγραμμα και την αποτελεσματικότητά του. Να σημειώσουμε ότι τα περισσότερα εμπορικά προγράμματα παρουσιάζουν τα αντίστοιχα σφάλματα και ότι η αποτελεσματικότητα του συστήματός μας είναι μεγαλύτερη από πολλά εξ'αυτών. Εξαίρεση αποτελεί το PhotoScore της Neuratron (χρησιμοποιεί το OMR σύστημα SharpEye) το οποίο καταφέρνει να επιλύει τα περισσότερα προβλήματα φθοράς αλλά έχει επίσης και πολύ καλύτερα εκπαιδευμένο νευρωνικό δίκτυο.

11
Μελλοντική δουλειά

Η δουλειά που έχουμε κάνει μέχρι τώρα αφορά κυρίως την υλοποίηση όλων των βασικών πτυχών ενός συστήματος OMR. Είναι μια αρκετά στέρεα βάση ανάπτυξης μιας πλήρους εφαρμογής OMR. Βασικές όψεις της μετέπειτα δουλειάς για τη βελτίωση και την ολοκλήρωση του συστήματός μας είναι:

11.1. Υλοποίηση – εισαγωγή σε C

Αναπτύξαμε το σύστημά μας σε Matlab προκειμένου να έχουμε γρήγορη δοκιμή αλγορίθμων και τεχνικών και αντίστοιχα γρήγορα αποτελέσματα. Ωστόσο η ανάπτυξη του συστήματος μας σε περιβάλλον C με χρήση των βιβλιοθηκών OpenCV που ενδείκνυνται για επεξεργασία εικόνας είναι εφικτή δεδομένου ότι κάνουμε περιορισμένη χρήση των βιβλιοθηκών του Matlab. Συγκεκριμένα πέρα από τις συναρτήσεις labeling (bwlabel) και connected components (bwconncomp) η συντριπτική πλειοψηφία των συναρτήσεων είναι κατασκευασμένες από εμάς. Βέβαια το Matlab υλοποιεί βέλτιστα αυτές τις συναρτήσεις και τη δέσμευση μνήμης που απαιτούν ωστόσο αντίστοιχες δυνατότητες παρέχει και το OpenCV.

11.2. Εξαγωγή του αποτελέσματος σε MusicXML

Αρχικός στόχος της εργασίας ήταν η εξαγωγή του αποτελέσματος σε MIDI. Για αυτό το λόγο κατά την εξέλιξη του συστήματος συγκρατήσαμε μόνο τις πληροφορίες που απαιτούνται για το ηχητικό αποτέλεσμα, τις οποίες περιγράφουμε στο παράρτημα. Η εξαγωγή του αποτελέσματος σε MusicXML δηλαδή η ανακατασκευή της παρτιτούρας με μια πρότυπη μουσική γραμματοσειρά απαιτεί την αποθήκευση περισσότερων πληροφοριών όπως είναι το μήκος των beams, ο προσανατολισμός των stems κλπ. Με τη σταδιακή αποδόμηση της εικόνας που εφαρμόσαμε αυτό είναι εφικτό αλλά απαιτεί μια νέα δομή αποθήκευσης των πληροφοριών της αναγνώρισης.

11.3. Δημιουργία Γραφικού Περιβάλλοντος

Η φιλοσοφία των εφαρμογών OMR δεν είναι απλά η εξαγωγή του τελικού αποτελέσματος αλλά και η παράθεσή του προκειμένου να μπορεί ο χρήστης να κάνει διορθώσεις. Έτσι, για να είναι μια εφαρμογή OMR εργαλείο σε μια διαδικασία ψηφιοποίησης μουσικού αρχείου, πρέπει αρχικά να παρέχει όσο το δυνατόν καλύτερα αποτελέσματα, αφετέρου να δίνει στο χρήστη τη δυνατότητα να κάνει όσες διορθώσεις χρειάζεται προκειμένου το ψηφιακό αποτέλεσμα να προσεγγίζει το αναλογικό.

11.4. Περαιτέρω ανάπτυξη του συστήματος

Τελευταίο κομμάτι είναι η βελτίωση του συστήματος κάθε αυτού. Βασικά βήματα είναι η προσθήκη των συζεύξεων, δυναμικών, πεντάλ, επαναλήψεων, των grace notes και των αλλαγών

tempo που συμβαίνουν κατά τη διάρκεια του κομματιού.

Ανακύπτει, επίσης, και η δυνατότητα εκ νέου παραμετροποίησης και προσαρμογής του συστήματος για της αναγνώρισης χειρόγραφων παρτιτούρων αν και είναι ένα πεδίο που προκύπτουν πολλά νέα ζητήματα μιας και ο τρόπος γραφής παρουσιάζει τεράστιες διαφοροποιήσεις.

Επιπλέον δοκιμάσαμε το σύστημά μας δίνοντας ως είσοδο φωτογραφία παρτιτούρας από κινητή συσκευή. Οι αλγόριθμοι staff_detection και page_dewarp δούλεψαν με μεγάλη ακρίβεια και το βασικό πρόβλημα κυρίως τέθηκε στη δυαδικοποίηση μιας και σε αυτή την περίπτωση χρειάζονται νέες παράμετροι. Ωστόσο τα αποτελέσματα ήταν αρκετά ενθαρρυντικά για την επέκταση του συστήματος και σε αυτό το πεδίο.

Τέλος, απαιτείται περαιτέρω εκπαίδευση του νευρωνικού δικτύου προκειμένου να βελτιωθούν τα αποτελέσματα αλλά και να προστεθούν και άλλες μουσικές γραμματοσειρές. Ενδιαφέρον παρουσιάζει η δυνατότητα δημιουργίας μιας διαδικτυακής δεξαμενής δεδομένων από χρήστες που χρησιμοποιούν την εφαρμογή έτσι ώστε το νευρωνικό δίκτυο να τροφοδοτείται συνέχεια με νέα δεδομένα. Αυτή η λογική χρησιμοποιείται στην ανοιχτή εφαρμογή Audiveris, η οποία αν και υπολείπεται στη διαδικασία του segmentation έχει εισάγει αυτή την κοινοτική τροφοδότηση και εκπαίδευση του συστήματος.

Βιβλιογραφία

Binarization

Bradley, D., Roth, G. Adaptive Thresholding Using Integral Image. Journal of Graphics Tools. Volume 12, Issue 2. pp. 13-21. 2007. NRC 48816.

Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". IEEE Trans. Sys., Man., Cyber. 9 (1): 62–66. doi:10.1109/TSMC.1979.4310076

Image Dewarping

N. Stamatopoulos, B. Gatos, I. Pratikakis and S. J. Perantonis, A Two-Step Dewarping of Camera Document Images, The Eighth IAPR Workshop on Document Analysis Systems

http://www.leptonica.com/dewarping.html

Staff Detection/Removal

A. Fornes, A. Dutta, A. Gordo, and J. Llados, "The ICDAR 2011 music scores competition: Staff removal and writer identification," ICDAR, pp. 1511-1515, 2011

C. Dalitz, M. Droettboom, B. Pranzas, and I. Fujinaga, "A comparative study of staff removal algorithms," IEEE Tran. on PAMI, vol. 30, no. 5, pp. 753-766, May 2008

D. Blostein and H. S. Baird, "A critical survey of music image analysis," in Structured Document Image Analysis, Baird, Bunke, and Yamamoto(Eds.), Eds. Springer, 1992, pp. 405-434

I. Fujinaga, "Staff detection and removal," in Visual Perception of Music Notation: On-Line and Off-Line Recognition, S. George, Ed. Idea Group Inc., 2004, pp. 1-39

D. Bainbridge and T. C. Bell, "Dealing with superimposed objects in optical music recognition," International Conference on Image Processing and its Applications, pp. 756-760, 1997.

N. P. Carter and R. A. Bacon. Automatic recognition of printed music. In Baird et al., pages 456-465

Wijaya, K., and D. Bainbridge. 1999. Staff line restoration. Seventh International Conference on Image Processing and Its Applications (Conf. Publ. No.465) 2: 760-4

Note Detection / Classification

Miyao, H., and Y. Nakano. 1996. Note symbol extraction for printed piano scores using neural networks. IEICE Transactions on Information and Systems E79-D (5): 548-54

Bainbridge, D. 1997. Extensible optical music recognition. Ph.D. Thesis, University of Canterbury, Christchurch, NZ

Carter, N. P. 1992. Segmentation and preliminary recognition of madrigals notated in white mensural notation. Machine Vision and Applications 5 (3): 223-30

Fujinaga, I., B. Alphonce, G. Diener, and B. Pennycook. 1992. Optical music recognition on NeXT workstation. Paper presented at the Second International Conference on Music Perception and Cognition

Neural Network

Hori, T., S. Wada, T. Howzan, S. Y. D. E. Kung, and W. Bastiaan Kleijn. 1999. Automatic music score recognition/play system based on decision based neural network. 1999 IEEE Third Workshop on Multimedia Signal Processing (Cat. No.99TH8451): 183-4

Lee, S., and J. Shin. 1994. Recognition of music scores using neural networks. Journal of the Korea Information Science Society 21 (7): 1358-66

Γενικά Θέματα

Pruslin, D. 1966. Automatic recognition of sheet music. Sc.D. Dissertation, Massachusetts Institute of Technology

Watkins, G. 1994. A fuzzy syntactic approach to recognizing hand-written music. Proceedings of the International Computer Music Conference: 297-302

Prerau, D. S. 1975. Do-Re-Mi: A program that recognizes music notation. Computer and the Humanities 9 (1): 25-29

Fotinea, S.-E., G. Giakoupis, A. Liveris, S. Bakamidis, and G. Carayannis. 2000. An optical notation recognition system for printed music based on template matching and high level reasoning. Paper read at The 6th Recherche d'Informations Assistée par Ordinateur, at Paris.

Florence Rossant and Isabelle Bloch "Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection" EURASIP Journal on Advances in Signal Processing 2007, 2007:081541

V. Poulain d',Andecy, J. Camillerapp and I. Leplumey, "Kalman Filtering for Segment Detection: Application to Music Scores Analysis," Proc. 12th Int',l Conf. Pattern Recognition, vol. 1, pp. 301-305, 1994.

http://kenschutte.com/midi

Παράρτημα

Παραθέτουμε σε αυτό το παράρτημα τον κώδικα των βασικών ενοτήτων του προγράμματος μας. Τα σχόλια είναι στα αγγλικά διότι είναι ακριβώς όπως είναι γραμμένος ο κώδικας στο Matlab. Αντίστοιχα παρατίθενται και εικόνες του αποτελέσματος μετά από κάθε ενότητα κώδικα.

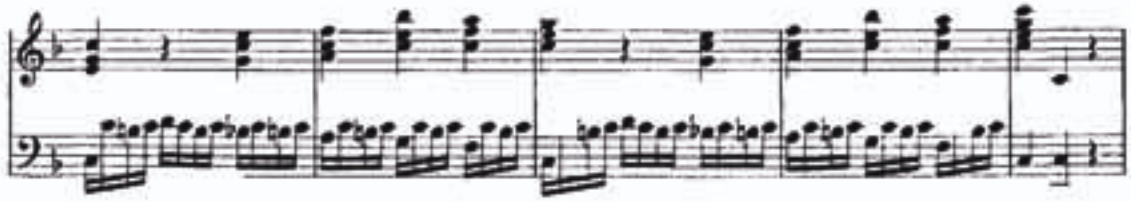
Περιεχόμενα Παραρτήματος

Helper Functions pos_th - ipos_th

We use the pos_th function and its inverse to quickly remove vertical runs that don't meet the input criteria of the function. To use for horizontal runs just input the rotated matrix Ib'.

```
function [posth,pix2cntr] = pos_th(Ib,thmax,thmin)
% the function returns the matrix posth with size equal with size of Ib.
% the matrix contains the center of all vertical runs within thmax and thmin
% and the value of its center is the length of the corresponding vertical
% run. The matrix pix2cntr gives each pixel the y coordinate of its center
[h,w]=size(Ib);
posth=double(zeros(h,w));
pix2cntr=double(zeros(h,w));

for j=1:w
    i=1;
    while i<=h
        if Ib(i,j)
            k=1;
            while (i+k<=h) && Ib(i+k,j)==1;
                k=k+1;
            end
            if k<=thmax && k>=thmin
                cntr=i-1+round(k/2);
                posth(cntr,j)=k;
                for l=i:i+k-1
                pix2cntr(l,j)=cntr;
                end
            end
            i=i+k;
        else
            i=i+1;
        end
    end
end

function [invpt] = ipos_th(pt)
% reconstructs image according to the centers and the run lengths
[h,w]=size(pt);
invpt=false(size(pt));

for j=1:w
    i=1;
    while i<=h
        if pt(i,j)~=0
            for l=i-(round(pt(i,j)/2)-1):i-(round(pt(i,j)/2)-1)+pt(i,j)-1
                invpt(l,j)=true;
            end
        i=i-(round(pt(i,j)/2)-1)+pt(i,j);
        else
        i=i+1;
        end
    end
end
```

STAFF DETECTION ALGORITHM



Calculate Only Staff Regions

In this part we detect the vertical regions of the picture that consist only from parts of the staff Lines

```
clearvars str_st
Iw=Ib | imclose(ipos_th(pos_th(Ib | Ir,2*th,0)),strel('line',round(gap/2),0));
mindist=4*gap; thr=3;

% we create the variable crit (short for criteria) which represents the
% sequence of vertical Black & White Runs Length that is allowed in
% order to detect a vertical window containing only staff parts
% *th = staff_thickness     gap = staff_distance*
crit=zeros(11,2);
crit(1,:)=[gap+2*thr,inf];
crit(11,:)=[gap+2*thr,inf];

crit(2:2:10,:)=repmat([0,2*th],[5,1]);
crit(3:2:9,:)=repmat([gap-thr,gap+thr],[4,1]);
```

```
CriteriaMat =
              MIN_RunLengt MAX_RunLengt
          W     23.00000          Inf
          B            0      6.00000
          W     14.00000     20.00000
          B            0      6.00000
          W     14.00000     20.00000
          B            0      6.00000
          W     14.00000     20.00000
          B            0      6.00000
          W     14.00000     20.00000
          B            0      6.00000
          W     23.00000          Inf
```

Main part of the Algorithm

We check if the sequence of the crit matrix exists and if it does we store the points in the struct str_st

```
[ptw,ptwm]=pos_th(Iw,inf,0);
[ptwi,ptwim]=pos_th(~Iw,inf,0);

pta=ptw + ptwi;
ptam=ptwm+ptwim;

i=1; j=1; n=1; si=1;
first=true; stf=zeros(5,2);
while j<=width
    try
    if n==1 && Iw(i,j)
        i=i+pta(ptam(i,j),j);
    end

    k=pta(ptam(i,j),j);

    if n==7
        cnt=i;
    end

    if k>=crit(n,1) && k<=crit(n,2)
        n=n+1;
        if n~=12
        i=i+pta(ptam(i,j),j);
        end
    else
        if n<=2
            i=i+pta(ptam(i,j),j);
        end
        n=1;
        stf=zeros(5,2);
    end

    try
    if mod(n,2)==0 && n<12
       stf(n/2,:)=[ptam(i,j),j];
    end
    catch
    end

    if n==12
        if first
            str_st{1,1}=cnt;
            str_st{1,2}=j;
            str_st{1,3}=stf;
            first=false;
        else
            [mn,mni]=min(abs(cnt-[str_st{:,1}]));
            if mn<5*gap
                if j-str_st{mni,2}>mindist
                str_st{mni,1}=cnt;
                str_st{mni,2}=j;
```

```
                str_st{mni,3}=cat(2,str_st{mni,3},stf);
                end
                str_st{mni,4}=stf;
            else
                si=si+1;
                str_st{si,1}=cnt;
                str_st{si,2}=j;
                str_st{si,3}=stf;
            end
        end
        n=1;
        stf=zeros(5,2);
    end
    catch
    end
    if i>height
        i=1;
        n=1;
        stf=zeros(5,2);
        j=j+1;
    end
end

str_st=sortrows(str_st,1);
for k=1:size(str_st,1)
    pts=cat(2,str_st{k,3},str_st{k,4});
    str_st{k,1}=pts(1,2);
    str_st{k,2}=pts(1,size(pts,2));
    str_st{k,3}=pts;
end
```

Create Polynomials

Having the Points stored in the struct str_st we perform polyfit for every line to find the polynomials. Polynomial grade is set to 4

```
staffp=[];
polyn=4;
for k=1:size(str_st,1)
    for x=1:5
        pts=str_st{k,3};
        staffp=cat(1,staffp,polyfit(pts(x,2:2:size(pts,2)),pts(x,1:2:size(pts,2)),polyn));
    end
end
staffp=sortrows(staffp,polyn+1);
```

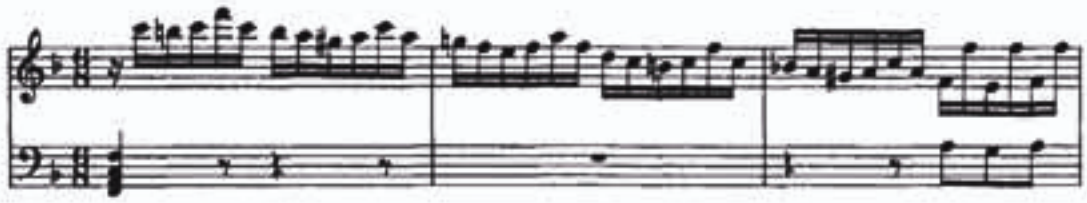
Create Staff Image

We perform polyval to create the staff image

```
staff_im=false(psize);
for k=1:size(staffp,1)
   for j=1:width
       staff_im(withinSize(round(polyval(staffp(k,:),j)),height),j)=true;
   end
end

figure; imshow(imoverlay(~Ib,bwmorph(staff_im,'thick',2),[0,0,1]));
```

Full Head Note Detection

In this part we detect the position and value of notes with black head (i.e. notes with value less than quarter)



Find Note Heads

First we detect the black heads of the Notes

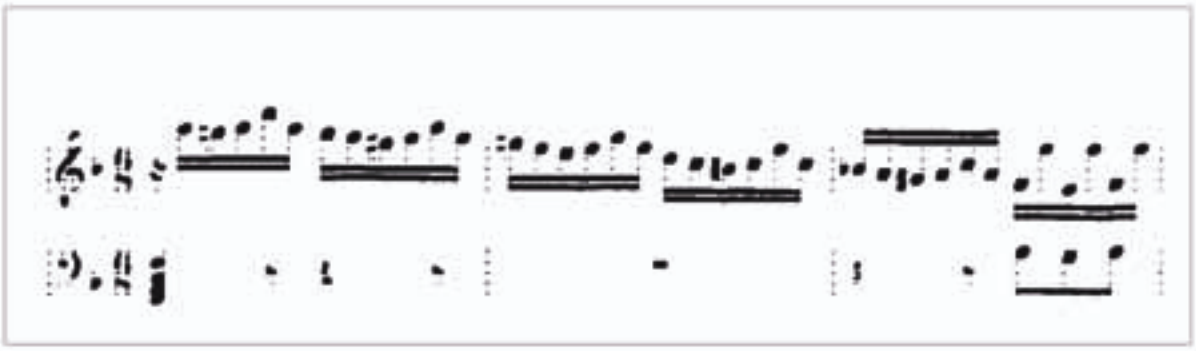
```
% we remove small white holes in case of bad printing of the sheet
ccs=bwconncomp(~Ib);
ss=regionprops(ccs,'BoundingBox');
bbss=cat(1,ss.BoundingBox);
Ibnh=Ib | CC_draw(ccs,bbss(:,3)<=gap*0.6 & bbss(:,4)<=gap*0.4);
% Ibnh stand for Ib no holes

note_xy=[];
Isr=Ibnh & ~(ipos_th(pos_th(Ibnh,2*th,0)) | ...
    ipos_th(pos_th(Ibnh',2*th,0))');
% Isr is a Ib without short vertical or horizontal runs
```

Detect Note Head Position

Within every segment of Isr we check cross correlation and opening results with template note. If they meet the criteria we keep the note in the note_xy variable

```
cc=bwconncomp(Isr);
s=regionprops(cc,'BoundingBox','Image');
bb=cat(1,s.BoundingBox);
note_i=false(length(s),1);
for k=1:length(s)
    if s(k).BoundingBox(3)>=size(note,2)*0.7 && ...
            s(k).BoundingBox(3)<=size(note,2)*2.5...
            && s(k).BoundingBox(4)>=0.6*size(note,1)
        im=s(k).Image;
        [tl,br]=bb2tlbr(s(k).BoundingBox);
        nim=Iexpand(imopen(Iexpand(im,1,1),...
            strel(imresize(note,0.8))),-1,-1);
        pks=xcorrpks(im & Ib(tl(1):br(1),tl(2):br(2)),...
            note,0.7,size(note,1)*0.65);
        if nnz(im & ~nim)<=size(pks,1)*12*gap && nnz(nim)>0
            note_xy=cat(1,note_xy,pks+repmat(floor(s(k).BoundingBox(1:2))...
                ,size(pks,1),1));
            note_i(k)=true;
        end
```

```
    end
end

% we store the note_xy points in image pks_im and we dilate them with the
% note template getting the In (note) image
pks_im=false(psize);
for k=1:size(note_xy,1);
    pks_im(round(note_xy(k,2)),round(note_xy(k,1)))=true;
end
In=imdilate(pks_im,strel(note));
```



Irr image

In image

Stems

Here we detect all the possible stems. First we perform closing between thin horizontal runs (cl) and then we perform opening with vertical structural element (lst)

```
cl=imclose(ipos_th(pos_th(ipos_th(pos_th(Ib,inf,2*th))',2*th,0))',strel('line',3*th,90)) &
-Ib;
Ist=imopen((Ib | cl) &
-imdilate(pks_im,strel(imresize(note,0.6))),strel('line',2.2*gap,90));

% we filter out the stems that dont touch the In image using imreconstruct
% function and the we run through every stem. If two stems touch the
% same In image and their vertical span overlap each other we remove the
% smallest. This way we manage to keep only the vertical lines that
% represent stems
Ist=imreconstruct(In,Ist);
Ilb=bwlabel(Ist);
cct=bwconncomp(Ist);
st=regionprops(cct,'BoundingBox','Centroid');
bbt=cat(1,st.BoundingBox);
cnt=cat(1,st.Centroid);
rmv=[];
for k=1:length(st)
    lbs=Ilb(imreconstruct(imreconstruct(CC_draw(cct,k),In),Ist));
    lbs=setdiff(unique(lbs),k);

    if ~isempty(lbs)
    clst=find(abs(cnt(lbs,1)-cnt(k,1))<=0.9*size(note,2));
    clst=lbs(clst);
    else
        clst=[];
    end
    if ~isempty(clst)
        for k2=1:size(clst,1)
          if bbt(clst(k2),4)<bbt(k,4)
              car= cmn_area(bbt(k,2),bbt(k,2)+bbt(k,4),...
                      bbt(clst(k2),2),bbt(clst(k2),2)+bbt(clst(k2),4));
            if car>0
                if car/bbt(clst(k2),4)>0.5
                    rmv=cat(1,rmv,clst(k2));
                end
            end
          end
        end
    end
end

Ist=Ist & ~CC_draw(cct,rmv);


lb=bwlabel(Ist);
ccl=bwconncomp(Ist);
sl=regionprops(ccl,'BoundingBox','Centroid');
st_cnt=cat(1,sl.Centroid);
```

1st (Initial Opening)

In,1st after filtering

Beams

Here we detect the possible beam segment. One of the criteria is small median of first derivative of vertical projection since their projection resembles step function. If we detect a beam we store the median of its vertical projection in between the parts that it's similar and store the final result in an image in which every non zero element represents the value of the projection at this certain point

```
Ibc=imclose(ipos_th(pos_th(imreconstruct(Ist,Isr)',inf,gap))' &
~CC_draw(cc,note_i),strel('line',4*th,90));
cc=bwconncomp(Ibc);
s=regionprops(cc,'Image','BoundingBox');

bm_i=false(length(s),1);
Ibmv=zeros(psize);
for k=1:length(s)
   if s(k).BoundingBox(3)>=1.1*size(note,2)
       im=s(k).Image;

       if var(diff(sum(im(:,th:size(im,2)-th),1)))<=15
           bm_i(k)=true;
           [tl,br]=bb2tlbr(s(k).BoundingBox);
           im=Ib(tl(1):br(1),tl(2):br(2)) & imdilate(im,strel('line',th,0));
           im=ipos_th(pos_th(im',inf,2.5*th))';
           pr=sum(im,1);
           dfs=diff(pr);

[~,dpks]=findpeaks(abs(dfs),'minpeakheight',hook_h*0.25,'minpeakdistance',round(0.6*gap));
           dpks=dpks(dpks>0.7*gap & dpks<length(dfs)*0.7*gap);

           dpks=cat(2,1,dpks,length(pr));
```

```
            bpr=zeros(length(pr),1);
            for ti=1:length(dpks)-1
                bpr(dpks(ti):dpks(ti+1))=median(pr(dpks(ti):dpks(ti+1)));
            end

            b=im .* repmat(bpr',size(im,1),1);

            Ibmv(tl(1):br(1),tl(2):br(2))=b;

        end
    end
end
```

Ibmv (intensity represents beam value)

Clean Out Stems

This part relates every note to its corresponding stem. It is used to sort out situations that two stems touch the same In segment but a different note within it. Most times it doesn't affect the result

```
note2st=zeros(size(note_xy,1),1);
st_i=false(length(s1),1);
for k=1:size(note_xy,1)
    w_note=lb(round(note_xy(k,2)-size(note,1)/1.7:note_xy(k,2)+size(note,1)/1.7),...
        round(note_xy(k,1)-size(note,2)/1.7:note_xy(k,1)+size(note,2)/1.7));
    p_note=pks_im(round(note_xy(k,2)-size(note,1)/1.7:note_xy(k,2)+size(note,1)/1.7),...
        round(note_xy(k,1)-size(note,2)/1.7:note_xy(k,1)+size(note,2)/1.7));


ls=unique(nonzeros(imdilate(p_note,strel(imresize(note,[1*size(note,1),1.05*size(note,2)]))
)...
        .*w_note));

    % Check if there is connection with stem found
    if numel(ls)>1
        imw=Ib(round(note_xy(k,2)-size(note,1)/1.7:note_xy(k,2)+size(note,1)/1.7),...
            round(note_xy(k,1)-size(note,2)/1.7:note_xy(k,1)+size(note,2)/1.7));
        wsk=bwmorph(w_note~=0,'thin',inf);
        wcs=bwmorph(wsk,'endpoints');

        wcs(1,:)=false; wcs(size(wcs,1),:)=false;
        wcs(1,:)=false; wcs(:,size(wcs,2))=false;
```

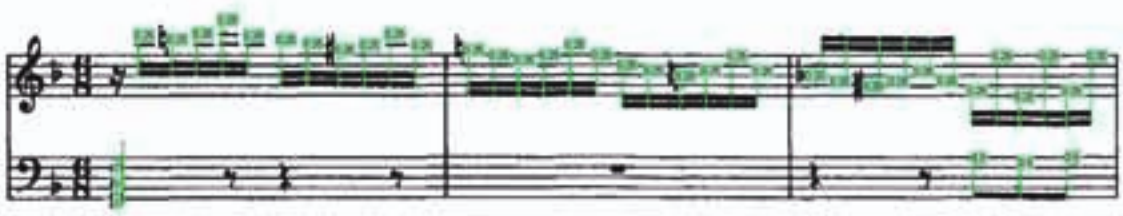
```
        if nnz(wcs)~=0
        [epr,epc]=find(wcs);
        [nr,nc]=find(p_note);
        isconn=zeros(length(epr),1);
        for ki=1:length(epr)
            t_pts=line2points([nr,nc],[epr(ki),epc(ki)]);

            if (imw(sub2ind(size(imw),t_pts(:,1),t_pts(:,2))))
                isconn(ki)=true;
            end
        end

        tmp_m=false(size(w_note));
        for ki=1:length(epr)
            if isconn(ki)
                tmp_m(epr(ki),epc(ki))=true;
            end
        end

        ls=unique(nonzeros(w_note.*tmp_m));
        end
    end

    if numel(ls)>1
        hts=zeros(numel(ls),1);
        for x=1:length(ls)
            hts(x)=s1(ls(x)).BoundingBox(4);
        end
        [mx,mxi]=max(hts);
        ls=ls(mxi);
    end
    if ~isempty(ls)
        note2st(k)=ls;
        st_i(ls)=true;
    end
end
```

Note Evaluation

The final part is the evaluation of Notes found. For every note in the note_xy variable we find the corresponding stem. If it touches a beam segment it take its value. Else we create an image along the stem where we believe its wing exists. Then we use a classifier to detect the value of the wing_im (wing = beams connected to single notes)

```
wing_im=false(psize);

load('wingnet.mat');
stem_ind=false(length(s1),1);
stem_up=false(length(s1),1);
Ins=ipos_th(pos_th(Ib,inf,1.5*th));

fnote2st=[];
fnote_xy=[];
for k=1:length(s1)
    if st_i(k)
        cnotes=find(note2st==k);
        if ~isempty(cnotes)
            stem_ind(k)=true;
            if nnz(abs(note_xy(cnotes,2)-s1(k).BoundingBox(2)<1.6*gap))==0
                stem_up(k)=true;
            end
            if stem_up(k)
                first_n=min(note_xy(cnotes,2))-round(0.6*size(note,1));
                stm_lmt=round(s1(k).BoundingBox(2));
            else
                first_n=max(note_xy(cnotes,2))+round(0.6*size(note,1));
                stm_lmt=round(s1(k).BoundingBox(2))+s1(k).BoundingBox(4);
            end

bm_sum=mode(nonzeros(Ibmv(min([first_n,stm_lmt]):max([first_n,stm_lmt]),round(s1(k).Centroi
d(1)))));

            if bm_sum<=1.5*th || isnan(bm_sum)
                if stem_up(k)
                    offs=0;
                    w=1;
                    while sum(w)~=0
                        w=Ins(stm_lmt+6*th+offs,...

round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                            round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
                            +round(0.7*size(note,2)));
                        offs=offs+1;
                    end
                    wing_w=Ins(stm_lmt:stm_lmt+offs+6*th,...
                        round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                        round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
                        +round(0.7*size(note,2)));
                    if size(wing_w,1)>2.5*gap
                        wing_im(stm_lmt:stm_lmt+offs+6*th,...

round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                            round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
```

```
                            +round(0.7*size(note,2)))=true;
                    end
                else
                    offs=0;
                    w=1;
                    while sum(w)~=0 && stm_lmt-6*th+offs>=first_n
                        w=Ins(stm_lmt-6*th+offs,...

round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                            round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
                            +round(0.7*size(note,2)));
                        offs=offs-1;
                    end
                    wing_w=Ins(stm_lmt-6*th+offs:stm_lmt,...
                        round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                        round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
                        +round(0.7*size(note,2)));
                    if size(wing_w,1)>2.5*gap
                        wing_im(stm_lmt-6*th+offs:stm_lmt,...

round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2):...
                            round(s1(k).BoundingBox(1))+s1(k).BoundingBox(3)+round(th/2)...
                            +round(0.7*size(note,2)))=true;
                    end
                end
                if size(wing_w,1)>2.5*gap
                    [mx,mxi]=max(sim(wingnet,im2stats2r2(wing_w,gap)'));
                    value=2^-(ceil(mxi/2));
                else
                    value=1;
                end
            else
                value=2^-(round_down(bm_sum/(hook_h),0.5));
            end
            fnote_xy=cat(1,fnote_xy,[note_xy(cnotes,:),ones(length(cnotes),1)*value]);
            fnote2st=cat(1,fnote2st,repmat(k,size(cnotes,1),1));
        end
    end
end
```



Segmentation and Classification

Here we segment the image and classify its components two times. First to detect sharps, flats, naturals and rest and a second time to classify the remaining elements

Image Preparation

We remove short vertical runs that touch the fnote_im, bars_im plus all short runs that lie on the staff_im Note that fnote_im is image containing all note elements, bars_im is image containing bars, Inobs is image without bars and slurs

```
num_staffs=size(staffp,1)/5;
around_sh=imreconstruct(imdilate(fnote_im | bars_im,strel('line',th,0))...
    ,ipos_th(pos_th(Inobs,2*th,0)));
Is=Inobs & ~fnote_im & ~around_sh;
sh=ipos_th(pos_th(Is,1.5*th,0).*imdilate(staff_im,strel('line',th,90)));
```

1st Segmentation

During first segmentation we connect small symbols together like broken whole notes. Most times the result is same as the initial image if there are no symbols to connect

```
lb=bwlabel(Is & ~sh);
ccl=bwconncomp(Is & ~sh);
sl=regionprops(ccl,'BoundingBox');
bbl=cat(1,sl.BoundingBox);
sh=sh | CC_draw(ccl,bbl(:,4)<=2.5*th);
cc=bwconncomp(sh);
s=regionprops(cc,'BoundingBox','Centroid');
lengthL=1.2*gap;
keep=[];
for k=1:length(s)
    if s(k).BoundingBox(3)<=lengthL
        lt=lb(round(s(k).Centroid(2)),floor(s(k).BoundingBox(1)));
        rt=lb(round(s(k).Centroid(2)),round(s(k).BoundingBox(1))+s(k).BoundingBox(3));
        if lt~=0 && rt ~=0 && rt~=lt

cmnar=cmn_area(sl(lt).BoundingBox(2),sl(lt).BoundingBox(2)+sl(lt).BoundingBox(4),...
               sl(rt).BoundingBox(2),sl(rt).BoundingBox(2)+sl(rt).BoundingBox(4));
            if ((bbcrit(sl(lt).BoundingBox,sl(rt).BoundingBox,[4*gap,gap],[4*gap,1.5*gap])
||...

bbcrit(sl(lt).BoundingBox,sl(rt).BoundingBox,[1.7*gap,1.2*gap],[1.7*gap,2.4*gap]))...
               && cmnar >0 && cmnar>sl(lt).BoundingBox(4)*0.3)...
```

```
            && sl(lt).BoundingBox(4)>3*th && sl(rt).BoundingBox(4)>3*th
            keep=cat(1,keep,k);
        end
    end
end
end
Isl=Is & ~sh | CC_draw(cc,keep);
sh=sh & ~CC_draw(cc,keep);
```

Get rests and arm

First classification We load our neural net (classnet) and we only extract sharps, flats, naturals and rest that our net chooses with confidence of 80%

```
cc1=bwconncomp(Isl);
s1=regionprops(cc1,'BoundingBox','Image','Centroid');
load('classnet.mat');
rest_arm_ind=3:11;
rest_arm=[];
for k=1:length(s1)
    if s1(k).BoundingBox(3)>=0.9*gap && s1(k).BoundingBox(4)>=0.7*gap
        [mx,mxi]=max(sim(classnet,im2stats2r2(s1(k).Image,gap)'));
        if mx>0.8 && nnz(mxi==rest_arm_ind)~=0
            rest_arm=cat(1,rest_arm,[k,mxi]);
        end
    end
end
```

2nd Segmentation

We remove elements that are detected at the first classification and we further connect larger symbols now in order to proceed to the final classification

```
if ~isempty(rest_arm)
Is2=Is1 & ~CC_draw(cc1,rest_arm(:,1));
else
Is2=Is1;
end

cc1=bwconncomp(Is2 & ~sh);
s1=regionprops(cc1,'BoundingBox');
lb=bwlabel(Is2 & ~sh);

cc=bwconncomp(sh);
s=regionprops(cc,'BoundingBox','Centroid');

lengthL=2*gap;
keep=[];
for k=1:length(s)
    if s(k).BoundingBox(3)<=lengthL
        lt=lb(round(s(k).Centroid(2)),floor(s(k).BoundingBox(1)));
        rt=lb(round(s(k).Centroid(2)),round(s(k).BoundingBox(1))+s(k).BoundingBox(3));
        if lt~=0 && rt ~=0 && rt~=lt
            if
bbcrit(s1(lt).BoundingBox,s1(rt).BoundingBox,[8*gap,5*gap],[9.5*gap,3.7*gap])...
              &&s1(lt).BoundingBox(4)>3*th && s1(rt).BoundingBox(4)>3*th
              keep=cat(1,keep,k);
            end
        end
    end
end

Is3=(Is2 & ~sh) | CC_draw(cc,keep);
```

Final Classification

We are ready now to classify the remaining elements

```
cc2=bwconncomp(Is3);
s2=regionprops(cc2,'BoundingBox','Image','Centroid');
bb2=cat(1,s2.BoundingBox);
clef_r_ind=[1,2,12:15];
clef_r=[];
for k=1:length(s2)
    if s2(k).BoundingBox(3)>=0.8*gap && s2(k).BoundingBox(4)>=2*gap
        [mx,mxi]=max(sim(classnet,im2stats2r2(s2(k).Image,gap)'));
        if mx>0.8 && nnz(mxi==clef_r_ind)==0
            clef_r=cat(1,clef_r,[k,mxi]);
        end
    end
end
```

Sort Elements

Finally, we store the extracted elements to their respecting class, where we store the center of their bounding boxes

```
if ~isempty(rest_arm)
sharp_xy=cat(1,s1(rest_arm(rest_arm(:,2)==3,1)).BoundingBox);
sharp_xy=bb_center(sharp_xy);
flat_xy=cat(1,s1(rest_arm(rest_arm(:,2)==4,1)).BoundingBox);
flat_xy=bb_center(flat_xy);
natural_xy=cat(1,s1(rest_arm(rest_arm(:,2)==5,1)).BoundingBox);
natural_xy=bb_center(natural_xy);
rest_xy=cat(2,cat(1,s1(rest_arm(rest_arm(:,2)>=7 & rest_arm(:,2)<=10,1))...
    .Centroid),2.^-(rest_arm(rest_arm(:,2)>=7 & rest_arm(:,2)<=10,2)-7));
whole_xy=cat(1,s1(rest_arm(rest_arm(:,2)==11,1)).Centroid);
else
sharp_xy=[]; flat_xy=[]; natural_xy=[]; rest_xy=[]; whole_xy=[];
end
% Here we offset the flats center since
% flat's position is the center of its hole
if ~isempty(flat_xy)
    flat_xy(:,2)=flat_xy(:,2)+0.6*gap;
end
gclef_xy=cat(1,s2(clef_r(clef_r(:,2)==1,1)).Centroid);
fclef_xy=cat(1,s2(clef_r(clef_r(:,2)==2,1)).Centroid);
```

Half and Whole Note Retrieval

We extract the images that are classified as whole or half notes and we cross correlate them with respecting templates in order to get the position of the "hollow" heads

```
hnote_ind=(clef_r(clef_r(:,2)>=12 & clef_r(:,2)<=14,1));
hnote_xy=[];
load('hnote.mat');
hnote=imresize(hnote,[size(note,1),size(note,2)]);
for k=1:length(hnote_ind)
    tc=xcorrpks(s2(hnote_ind(k)).Image,hnote,0.65,size(hnote,1)*0.5);

hnote_xy=cat(1,hnote_xy,tc+repmat(round(s2(hnote_ind(k)).BoundingBox(1:2)),size(tc,1),1));
    end
end

wnote_ind=(rest_arm(rest_arm(:,2)==11,1));
wnote_xy=[];
load('wnote.mat');
wnote=imresize(wnote,(gap+1*th)/size(wnote,1));
for k=1:length(wnote_ind)
    tc=xcorrpks(s1(wnote_ind(k)).Image,wnote,0.5,size(wnote,1)*0.5);

wnote_xy=cat(1,wnote_xy,tc+repmat(round(s1(wnote_ind(k)).BoundingBox(1:2)),size(tc,1),1));
end

if ~isempty(hnote_xy);
fnote_xy=cat(1,fnote_xy,[hnote_xy,ones(size(hnote_xy,1),1)*2]);
end

if ~isempty(wnote_xy);
fnote_xy=cat(1,fnote_xy,[wnote_xy,ones(size(wnote_xy,1),1)*4]);
end
```