



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών &
Μηχανικών Υπολογιστών

Τομέας Ηλεκτρονικής και Υπολογιστών

Διπλωματική Εργασία

Αφαίρεση Θορύβου Σκιάς από Αισθητήρες Depth Cameras



Εκπόνηση:

Μπούτης Πρόδρομος,
ΑΕΜ: 7734,
mpouprod@ece.auth.gr

Τζιαφάς Γεώργιος,
ΑΕΜ: 7784,
gtziafas@ece.auth.gr

Επίβλεψη:

Πέτρου Λουκάς,
Αναπληρωτής Καθηγητής ΑΠΘ,
ioukas@eng.auth.gr

Τσαρδούλιας Εμμανουήλ,
Μεταδιδακτορικός Ερευνητής ΑΠΘ,
etsardou@eng.auth.gr

Θεσσαλονίκη, Ιούνιος 2018

Περίληψη

Η ανάπτυξη της τεχνολογίας στο χώρο της μηχανικής όρασης, έχει προσφέρει πληθώρα εφαρμογών που επιχειρούν να απεικονίσουν και να επεξεργαστούν με υψηλή ευκρίνεια και απόδοση ένα τρισδιάστατο μοντέλο του φυσικού περιβάλλοντος. Μία μέθοδος που εφαρμόζεται σε προβλήματα τρισδιάστατης απεικόνισης και κυρίως σε εφαρμογές ρομποτικής, βασίζεται στην χρήση καμερών βάθους, οι οποίες εξοπλισμένες με στελέχη εκπομπής και απορρόφησης υπέρυθρης δομημένης ακτινοβολίας επιδιώκουν να απεικονίσουν πληροφορία βάθους της σκηνής σε μια δισδιάστατη εικόνα, αναφερόμενη ως εικόνα βάθους. Λόγω της αρχιτεκτονικής των αισθητήρων αυτών, καθώς και της φύσης των διαφόρων υλικών αντικειμένων που απεικονίζονται, τέτοιες εικόνες παρουσιάζουν δυνητικά μεγάλο ποσοστό θορύβου εντός τους, καθώς ολόκληρες περιοχές εντός τους δεν εμπεριέχουν μέτρηση. Στόχος της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη ενός αλγορίθμου, ενσωματωμένου ως πακέτα λογισμικού εντός του προγραμματιστικού περιβάλλοντος ROS, ο οποίος επιχειρεί να συνδυάσει το βάθος με την χρωματική πληροφορία που παρέχεται από την ίδια κάμερα, για να εξάγει συμπεράσματα σχετικά με το είδος του θορύβου και να συμπληρώσει με έγκυρη πληροφορία τις περιοχές μη μέτρησης της εικόνας βάθους. Αξιοποιώντας ήδη υλοποιημένους αλγορίθμους επεξεργασίας εικόνας καθώς και μαθηματικά εργαλεία λογισμού και αναλυτικής γεωμετρίας, ο αλγόριθμος επιδιώκει να ευθυγραμμίσει τις δυο εικόνες μεταξύ τους, να διασπάσει και να ομαδοποιήσει το θόρυβο με τις αντίστοιχες περιοχές έγκυρου βάθους και ίδιου χρώματος και, στη συνέχεια, να εξάγει μια όσο το δυνατόν ακριβέστερη εκτίμηση βάθους από τις περιοχές αυτές για να διορθώσει το θόρυβο. Ιδιαίτερο ενδιαφέρον δόθηκε στην αναγνώριση και στη αφαίρεση περιοχών θορύβου που οφείλονται σε ανακλαστικές επιφάνειες, καθώς αυτός αποτελεί συχνά μεγάλο ποσοστό του συνολικού θορύβου. Επίσης, στα πλαίσια της αξιολόγησης του αλγορίθμου μας, δημιουργήθηκε ένα σετ δεδομένων με εικόνες χρώματος, βάθους και εικόνες αλήθειας βάθους από τις κάμερες Microsoft Kinect V.1 και ASUS Xtion, στο οποίο διεξήχθησαν πειράματα μέτρησης ακρίβειας και απόδοσης.

Εκπόνηση:

Μπούτης Πρόδρομος,
ΑΕΜ: 7734,
mpouprod@ece.auth.gr

Τζιαφάς Γεώργιος,
ΑΕΜ: 7784,
gtziasfas@ece.auth.gr

Επίβλεψη:

Πέτρου Λουκάς,
Αναπληρωτής Καθηγητής ΑΠΘ,
loukas@eng.auth.gr

Τσαρδούλιας Εμμανουήλ,
Μεταδιδακτορικός Ερευνητής ΑΠΘ,
etsardou@eng.auth.gr

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών,
Τομέας Ηλεκτρονικής & Υπολογιστών,
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης,
Θεσσαλονίκη, Ιούνιος 2018



Aristotle University of Thessaloniki

School of Engineering

Department of Electrical & Computer
Engineering

Department of Electronics and Computer
Engineering

Bachelor's Thesis

Shadow Noise Elimination for Depth Sensor Cameras



Elaboration:

Mpoutis Prodromos,
AEM: 7734,
mpouprod@ece.auth.gr

Tziafas Georgios,
AEM: 7784
gtziafas@ece.auth.gr

Supervision:

Petrou Loukas,
Associate Professor AUTH,
loukas@eng.auth.gr

Tsardoulias Emmanouel,
Postdoctorate Researcher AUTH,
etsardou@eng.auth.gr

Thessaloniki, June 2018

Abstract

Due to the modern technological advances in the scientific field of Computer Vision, numerous highly efficient applications, aiming to reconstruct and process the physical environment as a 3D model have emerged. One popular 3D reconstruction method that is used, most commonly in robotic applications, focuses on the utilization of a depth camera. A depth camera, equipped with Infrared Structured Light emitting and sensing compounds, manages to produce a depth map of the scene and represent it as a 2D depth image. As a result of the camera architecture, in addition to the physical properties of materials whose depth is to be measured, depth images suffer from potentially high amount of noise, in the form of non-measured regions inside the frame. The objective of this thesis is the development of an algorithm, integrated as a collection of software modules in the ROS programming framework, that pursues to combine depth maps with the RGB image, captured from the same camera, in order to deduce knowledge about the source of the noise and interpolate the noisy regions of depth image with valid depth data. Via the employment of several already implemented image processing algorithms, as well as calculus and analytic geometry techniques, our algorithm attempts to align the two frames, separate and cluster the noise regions with the corresponding valid depth regions of the same color and extract an accurate estimation of the depth that these noise regions should have. Special interest is shown in the detection and correction of noise that is created by highly reflective surfaces of objects, as this type of noise is commonly the most dominant in depth images. Moreover, we have created a set of RGB-D data, obtained from the Microsoft Kinect v.1 and ASUS Xtion depth cameras, used for experimentation and accuracy testing purposes.

Elaboration:

Mpoutis Prodromos,
AEM: 7734,
mpouprod@ece.auth.gr

Tziasas Georgios,
AEM: 7784
gtziasas@ece.auth.gr

Supervision:

Petrou Loukas,
Associate Professor AUTH,
loukas@eng.auth.gr

Tsardoulias Emmanouel,
Postdoctorate Researcher AUTH,
etsardou@eng.auth.gr

School of Electrical & Computer Engineering,
Department of Electronics and Computer Engineering,
Aristotle University of Thessaloniki,
Thessaloniki, June 2018

Ευχαριστίες

Αρχικά, θα θέλαμε να ευχαριστήσουμε τους κ. Πέτρου και Συμεωνίδη για την εμπιστοσύνη, τη βοήθεια και την ανοχή ως προς τον εκτεταμένο χρόνο διεκπεραίωσης, λόγω προβλημάτων που συναντήσαμε κατά την εκπόνηση της παρούσας διπλωματικής εργασίας.

Επίσης, θέλουμε να πούμε ένα μεγάλο ευχαριστώ στο μεταδιδακτορικό ερευνητή Μάνο Τσαρδούλια για την αμέριστη βοήθεια, τη συνεχή επικοινωνία, την υπομονή και τη γενικότερη συνεργασία καθόλη τη διάρκεια, καθώς και στους Κωσταντίνο Παναγιώτου και Παναγιώτη Μουσουλιώτη για τις συζητήσεις, που μας βοήθησαν πολύ.

Τέλος, το μεγαλύτερο ευχαριστώ πηγαίνει στους γονείς, στα αδέρφια και στους φίλους μας, που ήταν δίπλα μας όλα αυτά τα χρόνια και μας στήριξαν με κάθε τρόπο.



Κατάλογος Πινάκων

Πίνακας 3.1.1.1: Τεχνικές Προδιαγραφές Microsoft Kinect v.1	21
Πίνακας 3.1.2.2: Τεχνικές Προδιαγραφές Κάμερας ASUS Xtion.....	23
Πίνακας 6.1.1: Τεχνικές Προδιαγραφές Υπολογιστή Διεξαγωγής Πειραμάτων.....	73
Πίνακας 6.2.1.1: Αποτελέσματα αναγνώρισης ανακλαστικών επιφανειών στο dataset της Kinect.....	75
Πίνακας 6.2.1.2: Αποτελέσματα αναγνώρισης ανακλαστικών επιφανειών στο dataset της Xtion.....	76
Πίνακας 6.3.1.1: Αποτελέσματα διόρθωσης θορύβου στο dataset της Kinect.....	84
Πίνακας 6.3.1.2: Αποτελέσματα διόρθωσης θορύβου στο dataset της Xtion	84

Κατάλογος Αλγορίθμων

Αλγόριθμος 3.2.2.1: Αλγόριθμος MeanShift	26
Αλγόριθμος 4.1.3.1: Αλγόριθμος Διόρθωσης ΕΠΑΜ εντός πλαισίου	40
Αλγόριθμος 4.2.1: Αλγόριθμος επεξεργασίας Πλαισίου Χρώματος και τμηματοποίησης....	41
Αλγόριθμος 4.3.1: Αλγόριθμος ελέγχου μορφολογίας ΕΠΑΜ	45
Αλγόριθμος 4.3.1.1: Αλγόριθμος τμηματοποίησης εικόνας με χρήση markers μέσω watershed για εύρεση ομοχρωματικών πιθανών ανακλαστικών επιφανειών	47
Αλγόριθμος 4.3.2.1.1: Αλγόριθμος κλεισίματος ακμών στα όρια ενός πλαισίου	49
Αλγόριθμος 4.3.2.3.1: Αλγόριθμος εύρεσης ανακλαστικών επιφανειών	50

Κατάλογος Σεναρίων

Σενάριο 4.1.2.1: Επίδειξη εφαρμογής μασκών	35
Σενάριο 4.4.1: Διάσπαση ΕΠΑΜ σε ενιαίες περιοχές ίδιου χρωματικού τμήματος	52



Κατάλογος Εικόνων

Εικόνα 1.0.1: Μικρομετρητής Βάθους Mituoyo – Series 527.....	1
Εικόνα 1.0.2: 3D Σαρωτής PRINTING SYSTEMS ScanMaster PLUS.....	2
Εικόνα 1.0.3: Παραδείγματα Μονοπτικών Μεθόδων Συνθήματος 3D Απεικόνισης.....	3
Εικόνα 1.0.4: Παράδειγμα Οπτικοποίησης 3D Πληροφορίας από Στερεοσκοπική Κάμερα.....	3
Εικόνα 1.0.5: Κάμερα Βάθους Intel RealSense F200.....	4
Εικόνα 1.0.6: Παράδειγμα Οπτικοποίησης 3D Πληροφορίας από Κάμερα Βάθους	5
Εικόνα 1.1.1: Θόρυβος Σκιάς Αντικειμένου σε Εικόνα Βάθους	6
Εικόνα 1.1.2: Συγχωνευμένος Θόρυβος Σκιάς διαφορετικών Αντικειμένων σε Εικόνα Βάθους.....	7
Εικόνα 1.1.3: Θόρυβος Ακραίας Απόστασης σε Εικόνα Βάθους	8
Εικόνα 1.1.4: Θόρυβος λόγω Ανακλαστικών Επιφανειών σε Εικόνα Βάθους	9
Εικόνα 1.1.5: Θόρυβος λόγω Απορροφητικής Επιφάνειας σε Εικόνα Βάθους	9
Εικόνα 1.1.6: Παράδειγμα Θορύβου σε Εικόνα Βάθους με σημειωμένο το είδος.....	10
Εικόνα 1.1.7: Θόρυβος Παραμόρφωσης Ακμών σε Εικόνα Βάθους	10
Εικόνα 1.2.1: Παράδειγμα Εισόδων – Εξόδου του λογισμικού μας	11
Εικόνα 3.1.1: Επίδειξη Μοντέλου Εξαγωγής Βάθους από Αισθητήρα Βάθους	20
Εικόνα 3.1.1.1: Κάμερα Βάθους Microsoft Kinect v.1	21
Εικόνα 3.1.1.2: Ενδογενή Χαρακτηριστικά Microsoft Kinect v.1 – Ορισμός	22
Εικόνα 3.1.1.3: Ενδογενή Χαρακτηριστικά Microsoft Kinect v.1 - Βαθμονόμηση	22
Εικόνα 3.1.2.1: Κάμερα Βάθους ASUS Xtion	23
Εικόνα 3.2.1.1: Χειρισμός πινάκων μέσω της NumPy.....	25
Εικόνα 3.2.2.1: Εφαρμογή του αλγορίθμου MeanShift	27
Εικόνα 3.2.3.1: Λειτουργία των threads.....	28
Εικόνα 3.4.2.1: Καταχώρηση βάθους στον RGB αισθητήρα	30
Εικόνα 3.4.4.1: Δημιουργία PointCloud2 από Ζεύγος Εικόνων Χρώματος – Βάθους	31
Εικόνα 4.0.1: Διάγραμμα Ροής Πληροφορίας Αλγορίθμου Διόρθωσης ΕΠΑΜ εντός Πλαισίου.....	32
Εικόνα 4.1.1: Εξαγωγή ΕΠΑΜ από Εικόνα Βάθους (α) Εικόνα Βάθους. (β) Μάσκα Θορύβου	33
Εικόνα 4.1.1.1: Βασική Επεξεργασία ΕΠΑΜ	34
Εικόνα 4.1.2.1: Εικόνες Σεναρίου 4.1.2.1	35
Εικόνα 4.1.3.1: Μάσκα και αντίστοιχα Πλαίσια Χρώματος και Βάθους	36
Εικόνα 4.1.3.2: Πλαίσια Βάθους, Θορύβου και Χρώματος μιας ΕΠΑΜ.....	38
Εικόνα 4.2.1: Στάδια διαδικασίας τμηματοποίησης	43
Εικόνα 4.3.0.1: Παρουσίαση Πλαισίων Χρώματος, Βάθους και μάσκας της ΕΠΑΜ για εξεταζόμενες ως προς τη μορφολογία ΕΠΑΜ	44
Εικόνα 4.3.1.1: Στάδια ταυτοποίησης ύπαρξης και μη ανακλαστικής επιφάνειας στα όρια της αρχικής εικόνας	46
Εικόνα 4.3.2.1.1: Αποτελέσματα διαδικασίας κλεισίματος ακμών στα όρια του πλαισίου	48
Εικόνα 4.3.2.3.1: Βήματα διαδικασίας ταυτοποίησης μίας επιφάνειας ως ανακλαστική	50
Εικόνα 4.4.1.1: Αποτύπωση του Σεναρίου 4.4.1 οπτικά	53
Εικόνα 4.4.2.1: Λειτουργία της διαδικασίας ΑΕΤ	54
Εικόνα 4.5.1.1: Παρουσίαση διαδικασία προσδιορισμού του ΔΒΚ	56
Εικόνα 4.5.2.1: Παρουσίαση διαδικασία προσδιορισμού του ΔΒΚ για ανακλαστικές	57
Εικόνα 4.6.1: Αναπαράσταση διανυσμάτων των 4 βασικών κατευθύνσεων (0° , 180° , 90° και 270°) και των αντίστοιχων χαρτών απόστασης	58
Εικόνα 5.0.1: Γράφος Διασύνδεσης Μηνυμάτων Εικόνας Πακέτου Αφαίρεσης Θορύβου... <td>61</td>	61

Εικόνα 5.0.2: Διάγραμμα Στελεχών Συστήματος Αφαίρεσης Θορύβου Εικόνας Βάθους	62
Εικόνα 5.1.1: Σύστημα Προεπεξεργασίας Βάθους	63
Εικόνα 5.1.2: Αλγόριθμος Προεπεξεργασίας Βάθους	64
Εικόνα 5.1.3: Αποτελέσματα της εφαρμογής της διαδικασίας <i>solidify_noise()</i>	65
Εικόνα 5.1.3: Αποτελέσματα της εφαρμογής της διαδικασίας <i>partition_noise()</i>	65
Εικόνα 5.2.1: Κλάση <i>Blob</i> – Βασικές Μέθοδοι	66
Εικόνα 5.2.2: Παραγωγή Πλαισίου Διορθωμένου Βάθους μιας <i>ΕΠΑΜ</i>	66
Εικόνα 5.2.3: Κλάση <i>Blob</i> – Περισσότερες Μέθοδοι	68
Εικόνα 5.3.1: Σύστημα Μετεπεξεργασίας Βάθους	69
Εικόνα 5.3.2: Αλγόριθμος Μετεπεξεργασίας Βάθους	69
Εικόνα 5.3.3: Εξομάλυνση Ακμών Καθαρής Εικόνας Βάθους (α) Εικόνα Βάθους Χωρίς, (β) Εικόνα Βάθους με Εξομάλυνση Ακμών	70
Εικόνα 6.2.1.1: Τμήματα ανακλαστικών επιφανειών που θα πρέπει και δεν θα πρέπει να αναγνωρισθούν	74
Εικόνα 6.2.2.1: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 7</i>	77
Εικόνα 6.2.2.2: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 1</i>	77
Εικόνα 6.2.2.3: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 15</i>	77
Εικόνα 6.2.2.4: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Xtion 8</i>	78
Εικόνα 6.2.2.5: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 24</i>	78
Εικόνα 6.2.2.6: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Xtion 11</i>	78
Εικόνα 6.2.2.7: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 28</i>	79
Εικόνα 6.2.2.8: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 14</i>	79
Εικόνα 6.2.2.9: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 6</i>	79
Εικόνα 6.2.2.10: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Xtion 6</i>	80
Εικόνα 6.2.2.11: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Xtion 14</i>	80
Εικόνα 6.2.2.12: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 26</i>	80
Εικόνα 6.2.2.13: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 13</i>	81
Εικόνα 6.2.2.14: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 3</i>	81
Εικόνα 6.2.2.15: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 25</i>	81
Εικόνα 6.2.2.16: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 10</i>	82
Εικόνα 6.2.2.17: <i>RGB</i> , <i>depth</i> και μάσκα ανακλαστικών επιφανειών της εικόνας <i>Kinect 17</i>	82
Εικόνα 6.3.2.1: Δείγμα <i>Kinect 32</i>	85
Εικόνα 6.3.2.2: Δείγμα <i>Xtion 19</i>	86
Εικόνα 6.3.2.3: Δείγμα <i>Xtion 1</i>	86
Εικόνα 6.3.2.4: Δείγμα <i>Xtion 4</i>	87
Εικόνα 6.3.2.5: Δείγμα <i>Kinect 12</i>	88
Εικόνα 6.3.2.6: Δείγμα <i>Kinect 15</i>	88
Εικόνα 6.3.2.7: Δείγμα <i>Kinect 21</i>	89
Εικόνα 6.3.2.8: Δείγμα <i>Xtion 12</i>	89
Εικόνα 6.3.2.9: Δείγμα <i>Xtion 10</i>	90
Εικόνα 6.3.2.10: Δείγμα <i>Xtion 16</i>	90
Εικόνα 6.3.2.11: Δείγμα <i>Xtion 20</i>	91
Εικόνα 6.3.2.12: Δείγμα <i>Kinect 23</i>	91
Εικόνα 6.3.2.13: Δείγμα <i>Kinect 22</i>	92
Εικόνα 6.3.2.14: Δείγμα <i>Kinect 22</i>	92
Εικόνα 6.3.2.15: Δείγμα <i>Xtion 9</i>	93
Εικόνα 6.3.2.16: Δείγμα <i>Kinect 13</i>	94
Εικόνα 6.3.2.17: Δείγμα <i>Kinect 3</i>	94
Εικόνα 6.3.2.18: Δείγμα <i>Kinect 10</i>	95
Εικόνα 6.3.2.19: Εικόνες βάθους του δείγματος 17 της <i>Kinect</i> πριν και μετά τη διόρθωση	95
Εικόνα 6.3.2.20: Εικόνες βάθους του δείγματος 31 της <i>Kinect</i> πριν και μετά τη διόρθωση	96

Εικόνα 7.1.1: Επίδειξη τρισδιάστατου μοντέλου σκηνής πριν και μετά τη διόρθωση.....	101
Εικόνα A.1.1: Χρωματικός κύβος <i>RGB</i> – Καμπύλες χρωματικής απορρόφησης κωνίων..	105
Εικόνα A.1.2: Σχηματική αναπαράσταση των <i>HSL</i> και <i>HSV</i>	106
Εικόνες A.1.3: Χρωματικά διαγράμματα των <i>CIE XYZ</i> , <i>CIELUV</i> και <i>CIELAB</i>	107
Εικόνα A.1.4: Μετασχηματισμός εικόνας σε μία σειρά από <i>colorspaces</i>	108
Εικόνα A.2.1: Ιστόγραμμα διακριτής και συνεχών μεταβλητών	109
Εικόνα A.2.2: Πρότυπο εικόνας με το ιστόγραμμα της πριν και μετά την εξισορρόπηση ..	109
Εικόνα A.2.3: Επίδειξη της απλής εξισορρόπησης και της προσαρμοσμένης με <i>clahe</i>	109
Εικόνα A.3.1: Εφαρμογή διαφορετικών λογικών <i>thresholding</i> σε εικόνα της κλίμακας του γκρι	111
Εικόνα A.3.2: Εφαρμογή διαδικασίας <i>thresholding</i> με τοπικά προσδιοριζόμενο όριο	112
Εικόνα A.3.3: Εφαρμογή <i>thresholding</i> με χρήση της λογικής <i>Otsu</i>	113
Εικόνα A.4.1: Πρότυπη εικόνα και αποτελέσματα <i>erosion</i> και <i>dilation</i>	114
Εικόνα A.4.2: Αποτέλεσμα του <i>opening</i> και του <i>closing</i>	114
Εικόνα A.4.3: Αποτελέσματα για <i>gradient</i> , <i>top hat</i> και <i>black hat</i>	115
Εικόνα A.5.1: Πυρήνας συνέλιξης διαστάσεων 5x5.....	115
Εικόνα A.5.2: Εφαρμογή της <i>filter2D</i>	116
Εικόνα A.5.3: Εφαρμογή της <i>blur – boxFilter</i>	116
Εικόνα A.5.4: Εφαρμογή της <i>GaussianBlur</i>	117
Εικόνα A.5.5: Εφαρμογή της <i>medianBlur</i>	117
Εικόνα A.5.6: Εφαρμογή της <i>bilateralFilter</i>	118
Εικόνα A.6.1: Πυρήνες για οριζόντια και κάθετη συνέλιξη – Υπολογισμός <i>gradient</i> σε φίλτρα <i>Sobel</i>	119
Εικόνα A.6.2: Πυρήνες για οριζόντια (αριστερά) και κάθετη συνέλιξη σε φίλτρα <i>Scharr</i> ...	119
Εικόνα A.6.2: Πυρήνες για εφαρμογή της μεθόδου <i>Laplacian of Gaussian</i> (<i>LoG</i>)	120
Εικόνα A.6.3: Εφαρμογή των <i>Laplacian</i> και <i>Sobel</i> (στη x και y κατεύθυνση)	120
Εικόνα A.6.4: Εφαρμογή του <i>Canny Edge Detector</i>	121
Εικόνα A.7.1: Εφαρμογή της διαδικασίας <i>skeletonization</i>	122
Εικόνα A.8.1: Εφαρμογή αλγορίθμου <i>floodfill</i>	123
Εικόνα A.9.1: Εφαρμογή του <i>watershed</i>	124
Εικόνα A.10.1: Εφαρμογή του αλγορίθμου <i>connectedComponents</i>	125



Περιεχόμενα

1.	Εισαγωγή	1
1.1	Περιγραφή του Προβλήματος	5
1.2	Στόχος της Διπλωματικής Εργασίας	11
1.3	Δομή του Κειμένου	12
2.	Υπάρχουσες Υλοποιήσεις.....	13
2.1	Ταξινόμηση Πηγών Θορύβου Σκιάς – Συμπαγές Γέμισμα	13
2.2	Χρήση RGB Πληροφορίας	16
2.2.1	Χρήση Πληροφορίας Ακμών – Διόρθωση Παραμόρφωσης Συνόρου	17
2.2.2	Χρήση Πληροφορίας Χρώματος	17
3.	Εργαλεία Υλικού και Λογισμικού.....	20
3.1	Υλικό.....	20
3.1.1	Microsoft Kinect v.1	21
3.1.2	ASUS Xtion	23
3.2	Γλώσσα Προγραμματισμού Python	24
3.2.1	Η Βιβλιοθήκη NumPy και η δομή της εικόνας	24
3.2.2	Αλγόριθμος MeanShift	25
3.2.3	Η βιβλιοθήκη Threading	27
3.3	OpenCV	28
3.4	Robot Operating System (ROS)	29
3.4.1	Η βιβλιοθήκη rospy	29
3.4.2	To OpenNI Framework.....	29
3.4.4	To πακέτο depth_image_proc	31
4.	Υλοποίηση Αλγορίθμου	32
4.1	Χαρακτηριστικά μιας ΕΠΑΜ	32
4.1.1	Περιγράμματα	33
4.1.2	Μάσκες	34
4.1.3	Πλαισία	35
4.2	Τμηματοποίηση Χρώματος	41
4.3	Αναγνώριση Ανακλαστικών Επιφανειών.....	43
4.3.1	ΕΠΑΜ Προσκολλημένες στα Όρια του Συνολικού Πλαισίου	45
4.3.2	ΕΠΑΜ στο Κέντρο του Συνολικού Πλαισίου	47
4.3.2.1	Κλείσιμο Ακμών στα Όρια του Πλαισίου	48
4.3.2.2	Αναγνώριση Ανακλαστικών Επιφανειών στο Κέντρο του Πλαισίου	49
4.4	Ομαδοποίηση των ΕΠΑΜ με Περιοχές Έγκυρου Βάθους ίδιου Χρωματικού Τμήματος.....	51
4.4.1	Δημιουργία ενιαίων περιοχών ίδιου τμηματικού χρώματος.....	52
4.4.2	Εξαγωγή Μάσκας Έγκυρου Βάθους	53
4.5	Υπολογισμός Διανύσματος Κατεύθυνσης Βάθους Έγκυρης Περιοχής	54
4.5.1	Διάνυσμα Κατεύθυνσης Βάθους από Μάσκα Έγκυρου Βάθους.....	55
4.5.2	Διάνυσμα Κατεύθυνσης Βάθους από Κάδρο Πλαισίου.....	56
4.6	Καταχώρηση Τιμών Βάθους στην ΕΠΑΜ	57
5.	Περιγραφή Συστήματος	61
5.1	Σύστημα Προεπεξεργασίας Βάθους	63
5.2	Σύστημα Αφαίρεσης Θορύβου	66
5.3	Σύστημα Μετεπεξεργασίας Βάθους	68

6. Πειράματα.....	71
6.1 Λήψη Σετ Δεδομένων και Χαρακτηριστικά Πειραμάτων	71
6.2 Εντοπισμός Ανακλαστικών Επιφανειών	73
6.2.1 Πίνακας Αποτελεσμάτων Εντοπισμού Ανακλαστικών Επιφανειών	73
6.2.2 Επίδειξη Εικόνων Εντοπισμού Ανακλαστικών Επιφανειών	76
6.3 Εξουδετέρωση Θορύβου.....	82
6.3.1 Πίνακας Αποτελεσμάτων Εξουδετέρωσης Θορύβου	83
6.3.2 Επίδειξη Εικόνων Εξουδετέρωσης Θορύβου	85
6.4 Αξιολόγηση Αποτελεσμάτων.....	96
7. Συμπεράσματα – Ανοιχτά Θέματα.....	99
7.1 Συμπεράσματα.....	100
7.2 Ανοιχτά Θέματα	101
Παράρτημα Α: OpenCV.....	103
Βιβλιογραφία.....	126

Ακρωνύμια Εγγράφου

Ενιαία Περιοχή Άκυρης Μέτρησης	→ ΕΠΑΜ
Συμπαγής Περιοχή Άκυρης Μέτρησης	→ ΣΕΠΑΜ
Μάσκα Έγκυρου Βάθους	→ ΜΕΒ
Διάνυσμα Κατεύθυνσης Βάθους	→ ΔΚΒ
Εικόνα Αληθούς Βάθους	→ ΕΑΒ
Μάσκα Ορθών Ανακλαστικών	→ ΜΟΑ
Μέσο Τετραγωνικό Σφάλμα	→ MSE
Τυπική Απόκλιση	→ STD
Υποφερτό Όριο Σφάλματος Βάθους	→ ΥΟΣΒ
Απομάκρυνση Εξωκείμενων Τιμών	→ ΑΕΤ
Λόγος Ορθά Βαμμένης Περιοχής	→ ΛΟΒΠ



1. Εισαγωγή

Τα τελευταία χρόνια, η ραγδαία ανάπτυξη που έχει επιτευχθεί στον χώρο του υπολογιστικού υλικού, ανοίγοντας νέους ορίζοντες στην εκτέλεση τεράστιου μεγέθους υπολογισμών μεγάλης πολυπλοκότητας σε πραγματικό χρόνο, όσο και στη καθιέρωση νέων αλγορίθμικών τεχνικών για την επεξεργασία και ανάλυση ψηφιακού σήματος (βλ. μηχανική μάθηση), έχει συγκεντρώσει σημαντικό βαθμό ενδιαφέροντος της επιστημονικής κοινότητας στον χώρο της μηχανικής όρασης, ένα διεπιστημονικό πεδίο που ασχολείται με το πως οι υπολογιστές μπορούν να εξάγουν γνώση υψηλού επιπέδου μέσα από την επεξεργασία ψηφιακών εικόνων και βίντεο. Οι διάφορες πτυχές της μηχανικής όρασης αφορούν μεθόδους και τεχνικές με τις οποίες η ψηφιακή εικόνα αποκτάται, επεξεργάζεται, αναλύεται και εν τέλει μετασχηματίζεται σε ένα μεταδεδομένο υψηλού επιπέδου, σε αριθμητική, συμβολική ή γενική μορφή συμπεράσματος. Περιοχές εφαρμογής του τομέα αυτού αποτελούν οι: *3D Απεικόνιση, Ανακατασκευή Σκηνής, Αναγνώριση Αντικειμένου – Γεγονότος, Εκτίμηση 3D Στάσης - Εκτίμηση Κίνησης, Αποκατάσταση Εικόνας κ.α.*

Με τον όρο *3D Απεικόνιση (3D Reconstruction)*, εννοούμε τη διαδικασία κατά την οποία απεικονίζουμε σε ένα τρισδιάστατο μοντέλο το σχήμα και την μορφή πραγματικών αντικειμένων στο φυσικό κόσμο. Αυτό επιτυγχάνεται με ενεργές και παθητικές μεθόδους. Οι πρώτες αλληλεπιδρούν ενεργώς με τα αντικείμενα προς απεικόνιση, είτε μηχανικά είτε ραδιομετρικά, χρησιμοποιώντας εξοπλισμό για να εξάγουν έναν χάρτη βάθους. Με δεδομένο τον χάρτη αυτό και με τη χρήση αριθμητικών τεχνικών προσέγγισης, το αντικείμενο κατασκευάζεται με βάση το χρησιμοποιούμενο μοντέλο προσέγγισης. Ένα παράδειγμα ενεργής μεθόδου με μηχανική τεχνολογία θα χρησιμοποιούσε ένα μετρητή βάθους για να υπολογίσει την απόσταση από ένα αντικείμενο τοποθετημένο σε μια περιστροφική πλάκα (βλ. Εικόνα 1.0.1). Πιο εφαρμοσμένες ραδιομετρικές μέθοδοι εκπέμπουν ακτινοβολία προς το αντικείμενο και μετρούν το ανακλώμενό του μέρος. Παραδείγματα τέτοιας τεχνολογίας ποικίλουν από την χρήση πηγών χρωματικού ορατού φωτός, Time-of-Flight λείζερ μέχρι χρήση μικροκυματικής και υπέρηχων (βλ. Εικόνα 1.0.2).



Εικόνα 1.0.1: Μικρομετρητής Βάθους Mitutoyo – Series 527



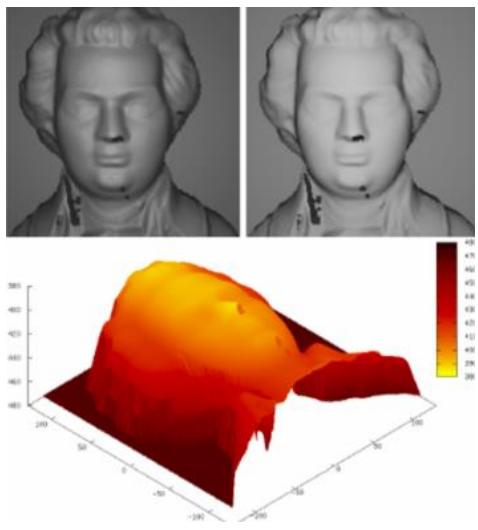


Εικόνα 1.0.2: 3D Σαρωτής PRINTING SYSTEMS ScanMaster PLUS

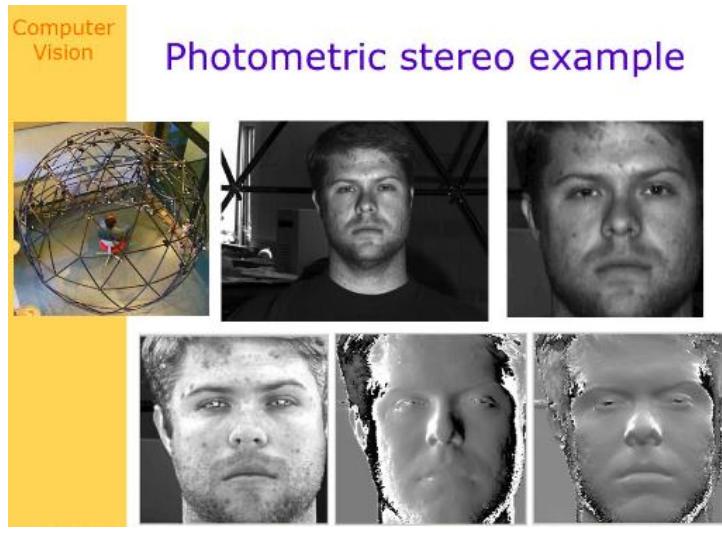
Οι παθητικές μέθοδοι της 3D Απεικόνισης δεν αλληλεπιδρούν με το απεικονιζόμενο αντικείμενο, αλλά χρησιμοποιούν αισθητήρες για να μετρήσουν την ακτινοβολία η οποία ανακλάστηκε ή εκπέμφθηκε από την επιφάνεια του αντικειμένου. Κατόπιν, από την ακτινοβολία αυτή εξάγουν πληροφορία βάθους για τη συγκεκριμένη επιφάνεια με χρήση λογισμικού *Κατανόησης Εικόνας*. Η πληροφορία αυτή απεικονίζεται ως μια 2D ψηφιακή εικόνα, κάθε pixel της οποίας συγκρατεί ένα κωδικοποιημένο δεδομένο για την απόσταση του σημείου της απεικονιζόμενης επιφάνειας από τον αισθητήρα μέτρησης. Ανάλογα με το είδος της κωδικοποίησης της πληροφορίας και την τεχνολογία που χρησιμοποιήθηκε, αλγόριθμοι εφαρμόζονται για να συνθέσουν ένα 3D μοντέλο όλων των σημείων που απεικονίζονται στην 2D αυτή εικόνα (x-y-z συντεταγμένες στον χώρο). Παραδείγματα τέτοιων μεθόδων αποτελούν οι:

- **Μονοπτικές Μέθοδοι Συνθήματος:** Οι μέθοδοι αυτές χρησιμοποιούν μόνο μία κατάλληλα διαμορφωμένη ψηφιακή εικόνα τραβηγμένη από μία οπτική γωνία (χρήση μόνο μίας κάμερας), για να εξάγουν 3D πληροφορία. Από την εικόνα αυτή αξιοποιούν 2D χαρακτηριστικά (τα οποία συχνά αποκαλούνται συνθήματα) για να μετρήσουν τρισδιάστατο σχήμα, γι' αυτό και συχνά στη βιβλιογραφία αναφέρονται ως *Shape-From-X* μέθοδοι, όπου X είναι ένα δισδιάστατο χαρακτηριστικό, όπως σιλουέτες, σκίαση ή υφή (*Shape-From-Shading*, *Photometric Stereo*, *Shape-of-Texture* κλπ, βλ. Εικόνα 1.0.3).
- **Διοπτρική Στερεοσκοπική Όραση:** Η μέθοδος αυτή εξάγει 3D γεωμετρικές πληροφορίες για ένα αντικείμενο από πολλές διαφορετικές εικόνες, τραβηγμένες παράλληλα από δύο κάμερες ή από την ίδια κάμερα σε διαφορετικές οπτικές γωνίες, αξιοποιώντας γνώση από τη μελέτη της ανθρώπινης όρασης. Με την χρήση δύο πανομοιότυπων αισθητήρων με παράλληλο οπτικό άξονα για τη δισδιάστατη απεικόνιση ενός αντικειμένου από διαφορετικές οπτικές γωνίες, η πληροφορία βάθους μπορεί, με την χρήση τριγωνομετρικών σχέσεων, να υπολογιστεί από τη διαφορά (*Disparity*). Η 3D πληροφορία οπτικοποιείται ως μία δισδιάστατη ψηφιακή εικόνα διαφοράς (*Disparity Image*) (βλ. Εικόνα 1.0.4).

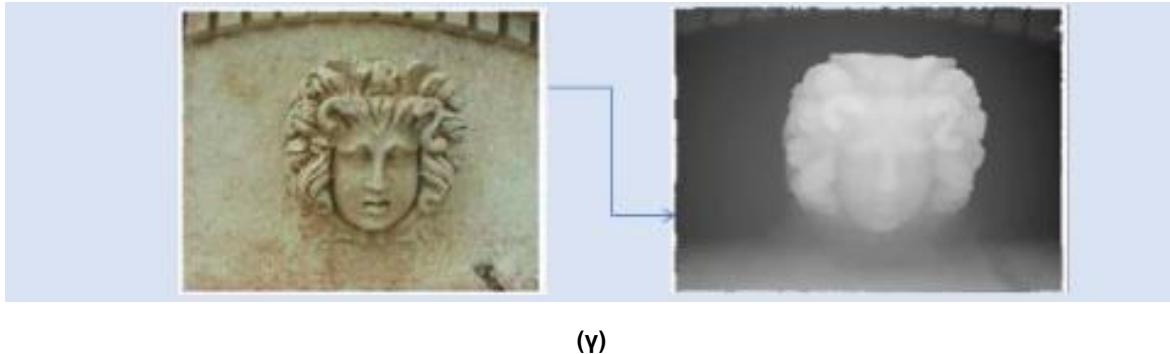




(α)



(β)



(γ)

Εικόνα 1.0.3: Παραδείγματα Μονοπτικών Μεθόδων Συνθήματος 3D Απεικόνισης
 (α) Σχήμα από Σκίαση, (β) Σχήμα από Σιλουέτα, (γ) Σχήμα από Υφή



(α)

(β)

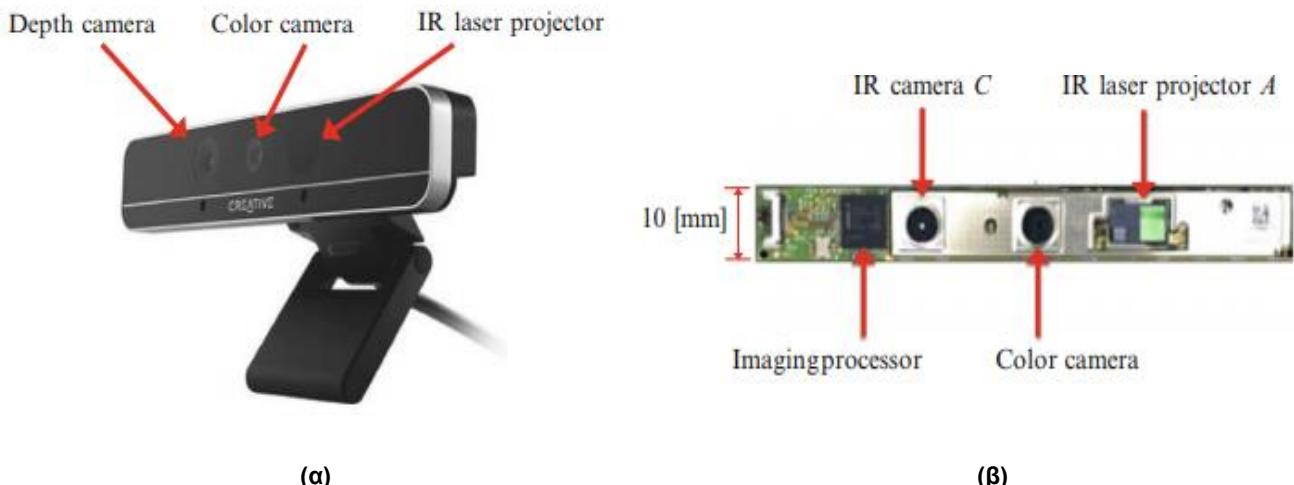
(γ)

Εικόνα 1.0.4: Παράδειγμα Οπτικοποίησης 3D Πληροφορίας από Στερεοσκοπική Κάμερα
 (α) Εικόνα από Αριστερό Αισθητήρα, (β) Εικόνα από Δεξί Αισθητήρα, (γ) Εικόνα Διαφοράς



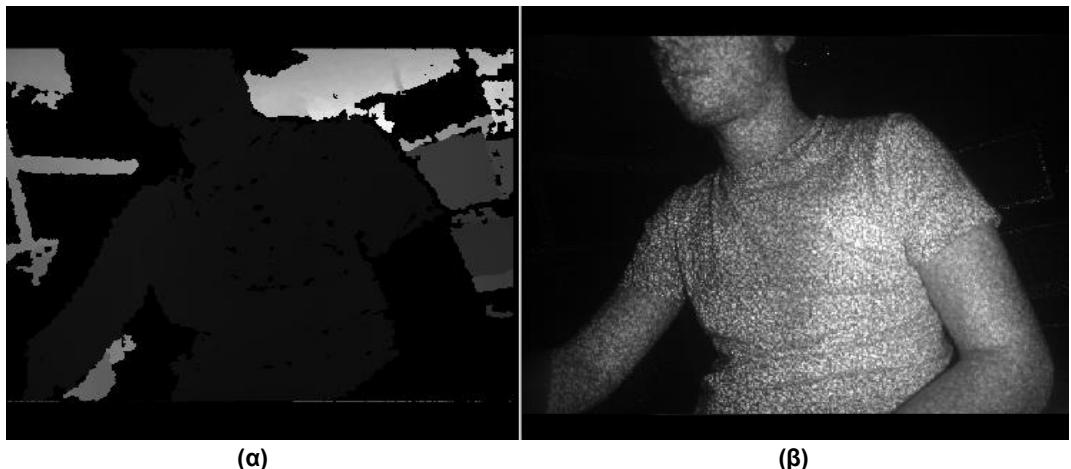
Μια ακόμα παθητική μέθοδος, η οποία αποτελεί και τον πυρήνα ενδιαφέροντος της διπλωματικής μας εργασίας, βασίζεται στην χρήση μιας *Κάμερας Βάθους* (*Depth Camera*) εξοπλισμένη με έναν *Αισθητήρα Βάθους* (*Depth Sensor*), ο οποίος, κάνοντας χρήση της τεχνολογίας *Υπέρυθρου Δομημένου Φωτός* (*InfraRed Structured Light*), υπολογίζει, για κάθε σημείο του αντικειμένου προς απεικόνιση, την απόσταση του από τον οριζόντιο οπτικό άξονα της κάμερας (*Bάθος*). Ο συγκεκριμένος τύπος κάμερας διαθέτει, εκτός από τον αισθητήρα χρώματος (*RGB Sensor*), έναν εκπομπό υπέρυθρης ακτινοβολίας (*IR Emitter*) και έναν αισθητήρα απορρόφησης υπέρυθρης ακτινοβολίας (*IR Sensor*) αντιδιαμετρικά του εκπομπού. Συνοπτικά, η κάμερα λειτουργεί εκπέμποντας ένα προκαθορισμένο μοτίβο από υπέρυθρες ακτίνες φωτός. Το φως απορροφάται από τα διάφορα αντικείμενα που υπάρχουν στον χώρο και μετριούνται από τον αισθητήρα βάθους. Με δεδομένη τη γνώση της απόστασης των δύο στελεχών της κάμερας (εκπομπού και αισθητήρα), η διαφορά ανάμεσα στην παρατηρούμενη και στην αναμενόμενη θέση των απολίξεων των ακτινών χρησιμοποιείται για τον υπολογισμό του βάθους κάθε pixel, σε αναφορά με τον RGB αισθητήρα, με την χρήση τριγωνομετρικών σχέσεων. Περισσότερες πληροφορίες για τα μαθηματικά μοντέλα εξαγωγής της πληροφορίας βάθους δίνονται στην παράγραφο 3.1, όπου εξετάζονται αναλυτικά οι κάμερες βάθους που χρησιμοποιήθηκαν για την διεξαγωγή των πειραμάτων μας.

Η πληροφορία αυτή οπτικοποιείται ως μια 2D μονοχρωματική ψηφιακή εικόνα, με το χρώμα του κάθε pixel της, να αντιστοιχεί στην απόσταση του αντίστοιχου σημείου από την κάμερα, με σκούρες αποχρώσεις να αντιστοιχούν σε κοντινότερα και ανοιχτές σε πιο απομακρυσμένα αντικείμενα. Η βιβλιογραφία αναφέρεται στην εικόνα αυτή ως *Χάρτη Βάθους* (*Depth Map*) ή *Εικόνα Βάθους* (*Depth Image*).



Εικόνα 1.0.5: Κάμερα Βάθους Intel RealSense F200
 (α) Βασικά Στελέγχη Κάμερας Βάθους, (β) Η Κάμερα "κάτω από την κουκούλα"





Εικόνα 1.0.6: Παράδειγμα Οπτικοποίησης 3D Πληροφορίας από Κάμερα Βάθους
 (a) Εικόνα Βάθους, (b) Εικόνα με το IR pattern

Οι αισθητήρες βάθους, εξαιτίας της σχετικά υψηλής αποδοτικότητας και πρακτικότητας λόγω μεγέθους, καθώς και της χαμηλής εμπορικής τους αξίας σε σχέση με αντίστοιχο υλικό μηχανικής όρασης, καθίστανται ιδιαίτερα δημοφιλείς σε πληθώρα εφαρμογών της 3D απεικόνισης, τόσο σε βιομηχανικό όσο και σε ακαδημαϊκό επίπεδο. Ενδεικτικά, αναφέρουμε την εικονική ανακατασκευή φυσικού χώρου σε εφαρμογές *Virtual* και *Augmented Reality*, την ανθεκτική στην απάτη *Αναγνώριση Ανθρώπινου Προσώπου*, τον εντοπισμό χειρονομιών και αναγνώριση εγγύτητας σε *Gaming* και *Security* εφαρμογές, καθώς και εφαρμογές ρομποτικής, πεδίο το οποίο εξετάζουμε πειραματικά και στα πλαίσια της εργασίας μας.

Τα συστήματα ρομπότ χρειάζεται να πλοηγούνται αυτόνομα στο χώρο, οπότε προκειμένου να υλοποιήσουν τις απαραίτητες λειτουργίες χαρτογράφησης, εντοπισμού θέσης, σχεδιασμού μονοπατιών και αποφυγής συγκρούσεων χρειάζονται ένα τρισδιάστατο μοντέλο του περιβάλλοντος. Πολύ διαδεδομένο εργαλείο, λοιπόν, για την απόκτηση 3D πληροφορίας για τα ρομπότ αποτελούν οι κάμερες βάθους, τους χάρτες βάθους των οποίων χρησιμοποιεί το λογισμικό τους για να ανακατασκευάσει το περιβάλλον τους.

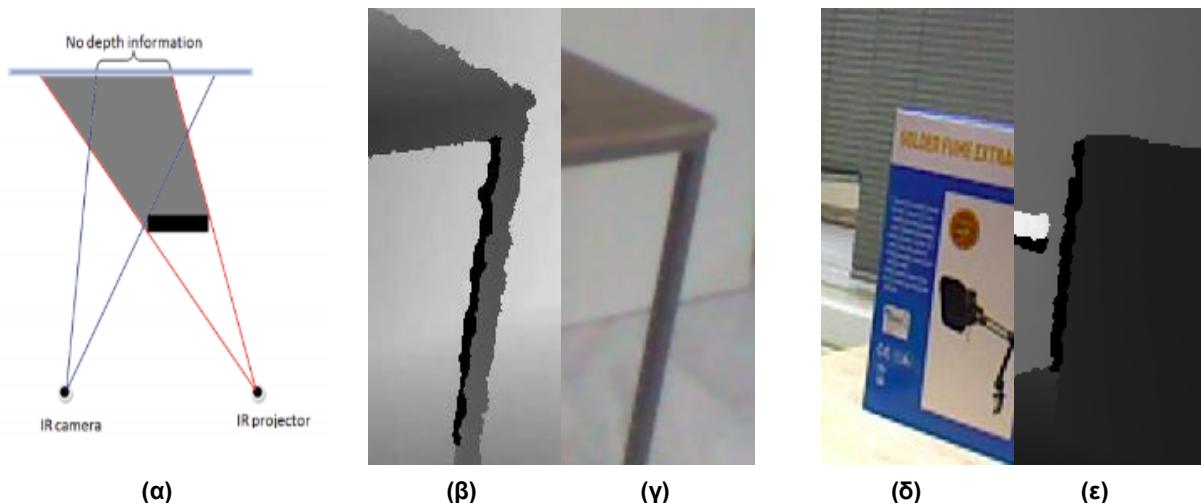
1.1 Περιγραφή του Προβλήματος

Όπως μπορεί ήδη να διακρίνει κάποιος από την εικόνα βάθους της εισαγωγικής παραγράφου, ο χάρτης βάθους που λαμβάνουμε από τον αισθητήρα και, κατά συνέπεια, η εκτίμηση της σκηνής που καλείται να απεικονίσει το σύστημα παρουσιάζει ελαττώματα. Χαρακτηριστικά στις εικόνες βάθους βλέπουμε περιοχές μαύρου χρώματος, στις οποίες, λόγω της αρχής λειτουργίας της τεχνολογίας του αισθητήρα ή και τού υλικού των αντικειμένων, ο αισθητήρας βάθους της κάμερας δεν έχει καταφέρει να λάβει μέτρηση. Σε τέτοιες περιοχές αναφερόμαστε συχνά με τον όρο θόρυβο, ή αφαιρετικά θόρυβο “σκιάς”, καθότι οι συνηθέστερες περιοχές μη μέτρησης στην εικόνα παραπέμπουν, στην



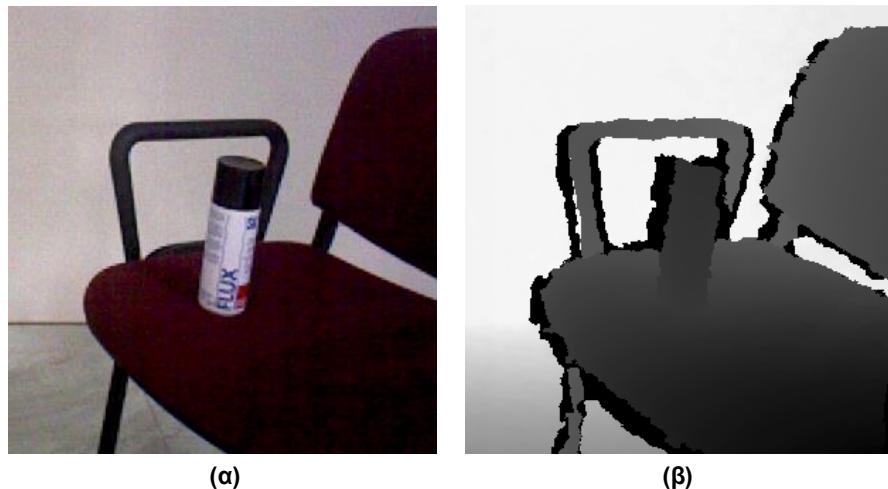
ανθρώπινη διαίσθηση, στο φαινόμενο της φυσικής σκιάς του οπτικού φωτός. Περισσότερες πληροφορίες για την κατηγοριοποίηση των διαφορετικών πηγών – ειδών θορύβου που έχουν γίνει σε ερευνητικό επίπεδο δίνονται στο επόμενο κεφάλαιο, ωστόσο συνοπτικά για λόγους αποσαφήνισης των προβλημάτων που αντιμετωπίζουμε αναφέρουμε τα εξής:

1. **Θόρυβος Έμφραξης ή Φυσικής Σκιάς:** Όπως το φως απορροφάται από ένα αντικείμενο στον χώρο, με αποτέλεσμα το ανθρώπινο μάτι να προβάλει το σχήμα του σαν σκιά στο επίπεδο, έτσι και στην περίπτωση του δομημένου IR φωτός, η ακτινοβολία απορροφάται από ένα αντικείμενο με αποτέλεσμα το σχήμα του, προβαλλόμενο στο επίπεδο της εικόνας, όταν πάει να αναγνωστεί από τον αισθητήρα βάθους στο επίπεδο της κάμερας, να μην εμπεριέχει τις σημεία του εκπεμπόμενου μοτίβου ώστε να πραγματοποιηθεί ο διαφορικός υπολογισμός. Συνεπώς, η μέτρηση θεωρείται άκυρη και ο αισθητήρας αποτυπώνει την περιοχή αυτή ως μαύρη, συμβολίζοντας την έλλειψη πληροφορίας βάθους. Οι περιοχές αυτές φέρουν σχήμα που προσεγγίζει την ακμή του αντικειμένου που προκάλεσε το θόρυβο και βρίσκονται “προσκολλημένες” σε αυτό. Αυτός ο θόρυβος είναι ο συνηθέστερος στις εικόνες βάθους, καθώς ακόμα και σε ένα σύστημα που λαμβάνει μια ροή εικόνων και ακυρώνει τον λεγόμενο παροδικό θόρυβο (θόρυβος που οφείλεται στην απόσταση των αντικειμένων και δεν παραμένει σταθερός για κινούμενο σύστημα αναφοράς), ο θόρυβος φυσικής σκιάς θα συνεχίσει να υπάρχει, αφού υπάρχει σταθερή απόσταση των στελεχών εκπομπής και μέτρησης του φωτός στην κάμερα βάθους. Συμβολικά το πρόβλημα καθώς και παραδείγματα τέτοιου θορύβου φαίνονται στις παρακάτω εικόνες.



Εικόνα 1.1.1: Θόρυβος Σκιάς Αντικειμένου σε Εικόνα Βάθους
 (a) Συμβολική Επίδειξη του Προβλήματος, (b),(d) RGB Εικόνα από Kinect, (c),(e) Αντίστοιχες Εικόνες Βάθους



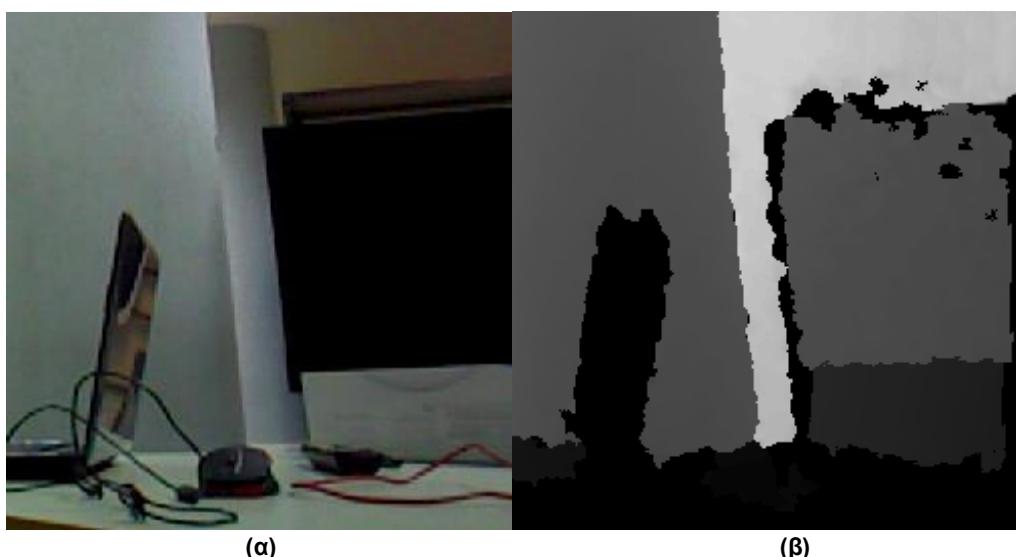


(α)

(β)

Εικόνα 1.1.2: Συγχωνευμένος Θόρυβος Σκιάς διαφορετικών Αντικειμένων σε Εικόνα Βάθους
 (α) RGB Εικόνα από ASUS Xtion, (β) Αντίστοιχη Εικόνα Βάθους

2. **Θόρυβος Ακραίας Απόστασης:** Όταν τα αντικείμενα που απεικονίζονται στη σκηνή βρίσκονται σε υπερβολικές κοντινές ή μακρινές αποστάσεις σε σχέση με τον αισθητήρα, τότε ολόκληρες επιφάνειες απεικονίζονται επίσης στον χάρτη ως μαύρες περιοχές μη έγκυρης μέτρησης. Τα αντικείμενα, στην περίπτωση αυτή, είτε θα βρίσκονται πολύ κοντά στον εκπομπό, με αποτέλεσμα να είναι εκτός της λήψης του αισθητήρα (βλ. Κάτω δεξιά περιοχή στην Εικόνα 1.1.3-β), είτε θα βρίσκονται εκτός της εμβέλειας βάθους που πληροί τις προϋποθέσεις κατασκευής της συγκεκριμένης κάμερας (βλ. πάτωμα στην Εικόνα 1.1.3-δ).



(α)

(β)





(γ)



(δ)

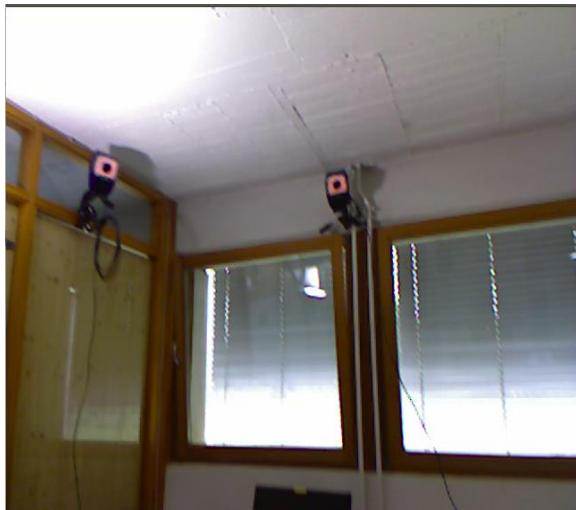
Εικόνα 1.1.3: Θόρυβος Ακραίας Απόστασης σε Εικόνα Βάθους

(α) RGB Εικόνα από ASUS Xtion, (β) Αντίστοιχη Εικόνα Βάθους με Θόρυβο Λόγω Μικρής Απόστασης, (γ) RGB Εικόνα από Kinect, (δ) Αντίστοιχη Εικόνα Βάθους με Θόρυβο Λόγω Μεγάλης Απόστασης

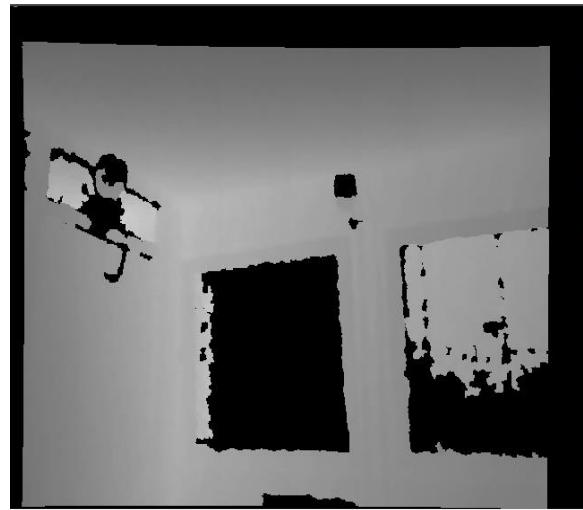
3. Θόρυβος που οφείλεται σε Ανακλαστικές ή Απορροφητικές Επιφάνειες: Βασική πηγή θορύβου στις εικόνες βάθους καθιστούν, επίσης, αντικείμενα που αποτελούνται από υλικά που αντανακλούν την υπέρυθρη ακτινοβολία, με αποτέλεσμα ο αισθητήρας να λαμβάνει μηδενική μέτρηση, καθώς και από πολύ απορροφητικά υλικά, που καθιστούν αδύνατη τη διάκριση του μοτίβου των υπέρυθρων τελειών από τον αισθητήρα. Στην πρώτη περίπτωση ανήκουν λείες ανακλαστικές επιφάνειες (καθρέφτες, παράθυρα, οθόνες υπολογιστών κλπ.), υλικά τα οποία βρίσκονται συχνά σε εσωτερικά περιβάλλοντα, ενώ, στη δεύτερη, αντικείμενα βαθιού μαύρου χρώματος. Όσον αφορά τις ανακλαστικές επιφάνειες, διακρίνουμε τις περιπτώσεις που ο αισθητήρας έχει λάβει κάποια ή κάποιες μετρήσεις εντός τους (π.χ. εάν ένα αντικείμενο βρίσκεται πολύ κοντά ακριβώς πίσω από ένα παράθυρο) και αυτές που ολόκληρη η επιφάνεια δεν έχει έγκυρες τιμές (π.χ. οθόνη υπολογιστή σε λειτουργία). Η περίπτωση του θορύβου ανακλαστικής επιφάνειας αποτελεί ιδιαίτερη περίπτωση

θορύβου στους χάρτες βάθους, καθώς, σαν πληροφορία βάθους, πρέπει να δίνεται αυτή του επιπέδου πάνω στο οποίο βρίσκονται και όχι οι σκόρπιες μετρήσεις βάθους εντός της επιφάνειας, μίας και η ανακατασκευή της επιφάνειας με βάση αυτό το βάθος και την χρωματική πληροφορία θα παράξει αποκλειστικά θόρυβο. Στην περίπτωση των απορροφητικών επιφανειών, ο αισθητήρας δεν λαμβάνει ή λαμβάνει ελάχιστες μετρήσεις που δεν καλύπτουν το περίγραμμα το αντικείμενου, με αποτέλεσμα η ανακατασκευή τους να είναι αδύνατη, οπότε η 3D πληροφορία πρέπει να εξαχθεί αποκλειστικά από τα RGB δεδομένα. Στις εικόνες που ακολουθούν, βλέπουμε παραδείγματα θορύβου ανακλαστικών και απορροφητικών επιφανειών (βλ. Παράθυρα στην Εικόνα 1.1.4, μαύρη θήκη στην εικόνα 1.1.5).





(α)



(β)

Εικόνα 1.1.4: Θόρυβος λόγω Ανακλαστικών Επιφανειών σε Εικόνα Βάθους
(α) RGB Εικόνα από Kinect, (β) Αντίστοιχη Εικόνα Βάθους



(α)



(β)

Εικόνα 1.1.5: Θόρυβος λόγω Απορροφητικής Επιφάνειας σε Εικόνα Βάθους
(α) RGB Εικόνα από XTION, (β) Αντίστοιχη Εικόνα Βάθους

Στις παρακάτω εικόνες, βλέπουμε μερικά ακόμα παραδείγματα θορύβου σημειωμένα με χρώμα πάνω στην εικόνα βάθους ανάλογα με το είδος τους.

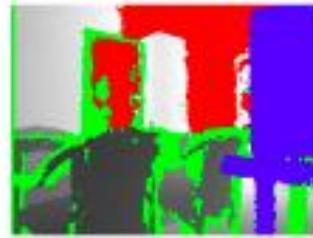




(α)



(β)



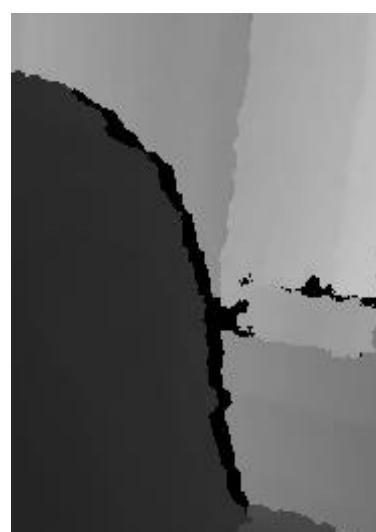
(γ)

Εικόνα 1.1.6: Παράδειγμα Θορύβου σε Εικόνα Βάθους με σημειωμένο το είδος (παραμένη από το έργο των [3])
(α) RGB Εικόνα από Kinect, (β) Αντίστοιχη Εικόνα Βάθους, (γ) Εικόνα Βάθος με σημειωμένο πράσινο το θόρυβο φυσικής σκιάς, ανακλαστικών και απορροφητικών επιφανειών, κόκκινο θόρυβο λόγω ακραίας μακρινής και μπλε λόγω ακραίας κοντινής απόστασης.

Ακόμη, ένα ελάττωμα που παρουσιάζουν οι αισθητήρες βάθους είναι η παραμόρφωση των ακμών των αντικειμένων στον αντίστοιχο χάρτη. Λόγω της διαρκούς εκπομπής και μέτρησης του μοτίβου υπέρυθρης ακτινοβολίας από την κάμερα, σημεία επάνω στα σύνορα επιφανειών αντικειμένων μπορεί να αντιστοιχίζονται διαδοχικά σε διαφορετικές τελείες του μοτίβου, με αποτέλεσμα συχνά να αποδίδονται στην εικόνα βάθους, όχι με συνεχή και λείο τρόπο, όπως στην RGB εικόνα, αλλά με παρουσία ανωμαλιών. Αυτό το φαινόμενο δεν αποτελεί σημαντικό πρόβλημα σε εφαρμογές που αποσκοπούν στον εντοπισμό αντικειμένων, στην διάκριση αντικειμένων προσκηνίου της σκηνής ή γενικά στην αποτύπωση τρισδιάστατης πληροφορίας για μια σκηνή, αλλά επηρεάζουν σημαντικά εφαρμογές που βασίζονται στην επεξεργασία πολύ συγκεκριμένων εντοπισμένων αντικειμένων ή γενικά στην απόδοση εικόνας με υψηλή ευκρίνεια. Στην παρακάτω εικόνα, βλέπουμε ένα χαρακτηριστικό παράδειγμα παραμόρφωσης της ακμής ενός αντικειμένου (τσάντα) σε μια εικόνα βάθους.



(α)



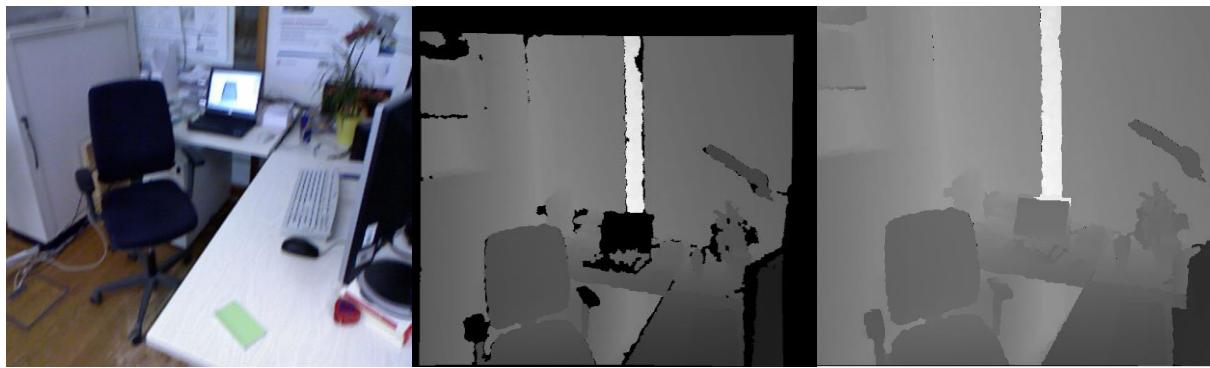
(β)

Εικόνα 1.1.7: Θόρυβος Παραμόρφωσης Ακμών σε Εικόνα Βάθους
(α) RGB Εικόνα από ASUS Xtion, (β) Αντίστοιχη Εικόνα Βάθους



1.2 Στόχος της Διπλωματικής Εργασίας

Το κεντρικό ζητούμενο της εργασίας αποτέλεσε ο σχεδιασμός ενός λογισμικού, το οποίο, δεδομένων εικόνων από τους αισθητήρες της κάμερας, επιδιώκει να διορθώσει όσο το δυνατόν ορθότερα το θόρυβο στην εικόνα βάθους. Πιο συγκεκριμένα, το ενδιαφέρον επικεντρώθηκε στη διόρθωση με έγκυρη πληροφορία βάθους όλων των περιοχών του χάρτη που εμπειριέχουν άκυρη μέτρηση. Επιχειρούμε αρχικά να εντοπίσουμε το θόρυβο στην εικόνα, να κάνουμε μια διάσπαση του σε συμπαγείς ενιαίες περιοχές που παραπέμπουν σε ξεχωριστά αντικείμενα της σκηνής, να τις ταξινομήσουμε με βάση την πηγή τους και, ανάλογα, να αξιοποιήσουμε τον κατάλληλο αλγόριθμο για να τις συμπληρώσουμε με έγκυρη πληροφορία. Προσπαθούμε, λοιπόν, να εξάγουμε κάποια γνώση για τις περιοχές θορύβου με βάση τη γεωμετρία τους και να αξιοποιήσουμε αυτή τη γνώση για να τις ομαδοποιήσουμε με τις σωστές αντίστοιχες περιοχές έγχρωμης πληροφορίας. Από τις περιοχές αυτές, εξάγεται η πληροφορία για τη διόρθωση του βάθους, με αναφορά την έγχρωμη εικόνα της κάμερας.



(α)

(β)

(γ)

Εικόνα 1.2.1: Παράδειγμα Εισόδων – Εξόδου του λογισμικού μας

(α) Έγχρωμη Εικόνα από Kinect, (β) Αντίστοιχη Εικόνα Βάθους, (γ) Εικόνα Βάθους Διορθωμένη από τον αλγόριθμό μας

Προκειμένου να το καταφέρουμε αυτό, χρησιμοποιήσαμε τις κάμερες βάθους Microsoft Kinect V.1 και ASUS Xtion για να λάβουμε RGBD δεδομένα, τα οποία προεπεξεργαστήκαμε κατάλληλα και σχεδιάσαμε έναν αλγόριθμο που εφαρμόζει βασικά εργαλεία ψηφιακής επεξεργασίας σήματος, μαθηματικές σχέσεις αναλυτικής γεωμετρίας, λογισμού και εργαλεία στατιστικής προκειμένου να παράξουμε “καθαρούς” χάρτες βάθους για κάθε ζεύγος τέτοιων πηγαίων δεδομένων. Στη συνέχεια, επιχειρούμε να υλοποιήσουμε αυτόν τον αλγόριθμο σε λογισμικό συμβατό με το λειτουργικό σύστημα ROS (Robot Operating System), έτσι ώστε να μπορεί να χρησιμοποιηθεί σε εφαρμογές πραγματικού χρόνου σε διανεμημένα συστήματα, με τη μορφή της δομής ενός πακέτου. Το πακέτο μας αποσκοπεί στο να χρησιμοποιεί τον driver μιας τέτοιας κάμερας, προκειμένου να δίνει μια ροή καθαρών εικόνων βάθους σε λογική χρονική απόκριση, εικόνες τις οποίες, κατόπιν, μπορεί να δεχθεί ως είσοδο κάποια άλλη μονάδα επεξεργασίας που εκτελεί κάποιον αλγόριθμο υλοποιημένο σε ROS, με σκοπό να εξάγει κάποιο συμπέρασμα για το περιβάλλον (π.χ. αναγνώριση αντικειμένου – στόχου) και να εκτελέσει κάποια άλλη ενέργεια (π.χ. κίνηση ρομποτικού ενεργοποιητή).



Ακόμη, λόγω της μεγάλης δυσκολίας που αντιμετωπίσαμε στο να βρούμε προεπεξεργασμένα δεδομένα με εικόνες αλήθειας (ground truth), καθώς και τη μη διεξαγωγή πειραμάτων σε εικόνες αλήθειας στις περισσότερες δημοσιευμένες εργασίες επί του θέματος, στόχο της διπλωματικής μας εργασίας αποτέλεσε επίσης η δημιουργία και η διάθεση ενός σετ δεδομένων από τις δύο προαναφερθείσες κάμερες, καθώς και η offline διόρθωση των εικόνων βάθους τους και παραγωγή των αντίστοιχων εικόνων αλήθειας για το μεγαλύτερο μέρος του σετ, προκειμένου να μπορούν να διεξαχθούν εύκολα πειράματα με χρήση εργαλείων υψηλού επιπέδου, χωρίς να χρειαστεί ο σχεδιαστής να έρθει σε επαφή με υλικό.

1.3 Δομή του Κειμένου

Μετά την παραπάνω περίληψη, προχωρούμε στην εισαγωγή, όπου παραθέτουμε τις μεθόδους 3D απεικόνισης. Επίσης, περιγράφουμε το πρόβλημα του θορύβου των εικόνων που προέρχονται από αισθητήρες βάθους, καθώς και τους στόχους που τίθενται.

Στο κεφάλαιο 2, παρουσιάζουμε τις υπάρχουσες μεθόδους αντιμετώπισης του προβλήματος, κατηγοριοποιώντας τες ανάλογα με τον τρόπο που επιδιώκεται η διόρθωση.

Ακολουθεί το κεφάλαιο 3, όπου αναφέρουμε τα χρησιμοποιούμενα εργαλεία, τόσο σε hardware (κάμερες Kinect και Xtion), όσο και σε software (Python και OpenCv, Robot Operating System).

Στη συνέχεια, στο κεφάλαιο 4, προχωρούμε σε ενδελεχή ανάλυση της σχεδίασης του λογισμικού. Ορίζουμε το αντικείμενο της βασικής οντότητας θορύβου, ως *ΕΠΑΜ* και περιγράφουμε όλα τα χαρακτηριστικά της, αλλά και τις διαδικασίες οι οποίες δημιουργήθηκαν με βάση το είδος της (ανακλαστική ή μη), ώστε να επιτευχθεί η διόρθωση. Κατόπιν, στο κεφάλαιο 5, παραθέτουμε μία αναλυτική περιγραφή του συστήματος, το οποίο διαιρούμε σε τρία υποσυστήματα, αυτά της προεπεξεργασίας, αφαίρεσης θορύβου και μετεπεξεργασίας. Για κάθε ένα από τα τελευταία, αναλύουμε τις λειτουργικότητες και τις διαδικασίες που περιλαμβάνουν για να πετύχουν τους στόχους τους.

Το κεφάλαιο 6 αποτελεί την παρουσίαση όλων των πειραμάτων που διεξήχθησαν, περιλαμβάνοντας μετρικές για την ικανότητα αναγνώρισης μίας *ΕΠΑΜ* ως ανακλαστική επιφάνεια, την ακρίβεια της διόρθωσης, καθώς και το χρόνο εκτέλεσης.

Ακολουθεί το κεφάλαιο 7, όπου πραγματοποιείται ο σχολιασμός των αποτελεσμάτων και επισημαίνονται οι επιτυχίες και οι αστοχίες της προσέγγισής μας, ενώ προτείνεται και μία σειρά μελλοντικών επεκτάσεων που αφορούν τις τελευταίες.

Για λόγους πληρότητας, παρουσιάζουμε αναλυτικά μία σειρά από διαδικασίες παρεχόμενες από την βιβλιοθήκη OpenCV της Python, τις οποίες χρησιμοποιήσαμε ευρέως στην υλοποίησή μας, σε μορφή παραρτήματος.

Τέλος, παραθέτουμε τη βιβλιογραφία στην οποία ανατρέξαμε. Στο σημείο αυτό, επισημαίνουμε ότι σε όλα τα παραπάνω κεφάλαια, παρατίθενται σειρές εικόνων για οπτικοποίηση της επίδρασης των διαδικασιών και των αποτελεσμάτων. Ακόμη, παρατίθενται συχνά αλγόριθμοι (στα αγγλικά) με τη μορφή εικόνων, καθώς και μερικά σενάρια λειτουργιών που επιθυμούμε να εξηγήσουμε.



2. Υπάρχουσες Υλοποιήσεις

Το φαινόμενο το οποίο μελετάμε, είναι η ύπαρξη "μαύρων" περιοχών στον χάρτη βάθους, των λεγόμενων σκιών, περιοχές στις οποίες ουσιαστικά έχουμε έλλειψη πληροφορίας, η οποία μεταφράζεται στον χάρτη ως μαύρα pixels (μηδενική gray-scale τιμή). Καθ' όλη την έκταση της βιβλιογραφίας, βλέπουμε ότι γίνεται διαχωρισμός των περιοχών θορύβου, τις οποίες καταχρηστικά ονομάζουμε **τρύπες**, βάση δύο αιτιών. Η βασική αιτία εμφάνισης θορύβου σκιάς οφείλεται στην γεωμετρία των αντικειμένων που υπάρχουν σε μια δεδομένη εικόνα. Ένα αντικείμενο το οποίο βρίσκεται πιο κοντά στην κάμερα αντανακλά την υπέρυθρη ακτινοβολία στον IR sensor, ο οποίος όταν θα επιχειρήσει να αναγνώσει την πληροφορία σε περιοχή της σκηνής πίσω από αυτό το αντικείμενο, θα συναντήσει σκιά. Η δεύτερη αιτία είναι η ύπαρξη αντικειμένων που λειτουργούν ως ανακλαστικές – διαχυτικές επιφάνειες (οθόνες, καθρέφτες κλπ.), οι οποίες δεν ανακλούν την *IR* ακτινοβολία, με αποτέλεσμα να μεταφράζονται από τον αισθητήρα βάθους ως τρύπες. Στο κεφάλαιο αυτό, θα προχωρήσουμε στην περιγραφή μίας σειράς υπάρχουσων υλοποιήσεων και αντιμετωπίσεων του θέματος που πραγματεύμαστε. Οι τελευταίες ταξινομούνται σε εκείνες που αντιμετωπίζουν τον παραπάνω θόρυβο ενιαία και επιχειρούν ένα συμπαγές γέμισμα των τρυπών και σε εκείνες που εκμεταλλεύονται την πληροφορία της έγχρωμης RGB εικόνας, ώστε να διορθώσουν το θόρυβο ή να εξάγουν σημαντικά δεδομένα για τη συγκεκριμένη, συνδυάζοντας και την υπάρχουσα πληροφορία βάθους.

2.1 Ταξινόμηση Πηγών Θορύβου Σκιάς – Συμπαγές Γέμισμα

Για την απόκτηση της πληροφορίας βάθους, μία από τις πρώτες μεθόδους αποτέλεσε η σάρωση με laser (laser scanning). Η συνηθέστερη αρχή λειτουργίας είναι η *Time – of – Flight*. Οι [1] στο σύγγραμμά τους προχωρούν σε ενδελεχή μελέτη της αρχής λειτουργίας, των μεθόδων και των εφαρμογών των *Time – of – Flight* καμερών. Οι τελευταίες παράγουν ένα χάρτη βάθους, εκπέμποντας ένα υπέρυθρο κύμα στην κατεύθυνση ενός αντικειμένου και λαμβάνουν, με έναν αισθητήρα, την επιστρεφόμενη συνιστώσα. Ο απαιτούμενος χρόνος για την επιστροφή του κύματος καθορίζει την τιμή του βάθους. Οι χάρτες, ωστόσο, παρουσιάζουν μια σειρά προβλημάτων ακριβείας, η οποία σχετίζεται με την υψηλή ταχύτητα του φωτός. Αξιοσημείωτα αποτελέσματα επιτυγχάνονται, όταν ευθυγραμμίζονται γεωμετρικά στερεοσκοπικά δεδομένα με αυτά της *Time – of – Flight* κάμερας. Η παραπάνω λογική μπορεί να επεκταθεί σε συστήματα που ενσωματώνουν στερεοσκοπική και *Time – of – Flight* κάμερα. Ένας προτεινόμενος αλγόριθμος λαμβάνει ως αρχικά σημεία τα ζευγάρια των προβολών της εικόνας βάθους της *Time – of – Flight* κάμερας πάνω στις προκύπτουσες από τη στερεοσκοπική εικόνες και προχωρά σε επέκταση αυτών, εκμεταλλεύμενος την πληροφορία της εξεταζόμενης κάθε φορά γειτονιάς (*seed – growing* αλγόριθμος).

Προχωρώντας σε ανάλυση και ταξινόμηση των πηγών θορύβου, θα περιγράψουμε την εργασία των [2], οι οποίοι παρουσιάζουν μια ευρύτερη επισκόπηση του θορύβου που εμφανίζεται στο χάρτη βάθους κατά τη λήψη με την Kinect. Παρόμοια είναι η προσέγγιση και για άλλες κάμερες βάθους. Τα μοντέλα προσομοίωσης του θορύβου



κατηγοριοποιούνται σε γεωμετρικά, εμπειρικά και στοχαστικά. Όσον αφορά τα γεωμετρικά (*pin – hole*), το *Disparity – Depth* μοντέλο πραγματοποιεί μια προσέγγιση για την πληροφορία του βάθους, βασιζόμενο στη σχετική μετατόπιση (*disparity*) εξαιτίας κάποιου αντικειμένου – εμποδίου του εκπεμπόμενου προτύπου υπέρυθρης ακτινοβολία ως προς ένα επίπεδο αναφοράς. Στην κατηγορία των γεωμετρικών, συμπεριλαμβάνεται και το λεγόμενο *Shadow Model*, το οποίο βασίζεται στον υπολογισμό του βάθους μέσω πληροφορίας που προκύπτει από τη δημιουργούμενη σκιά. Τα εμπειρικά μοντέλα συνδυάζουν τις παρατηρήσεις από αρκετά πειράματα, με αποτέλεσμα να εμφανίζουν περισσότερους βαθμούς ελευθερίας. Τέλος, υπάρχουν δύο χρησιμοποιούμενα στατιστικά μοντέλα που βασίζονται στο αντίστοιχο γεωμετρικό (*pin – hole*). Το πρώτο, αυτό της χωρικής αβεβαιότητας (*Spatial Uncertainty*), αποδεικνύει ότι ένα pixel μιας δισδιάστατης εικόνας απεικονίζεται στον τρισδιάστατο χώρο εντός ενός ελλειψοειδούς. Το δεύτερο είναι στοχαστικό και αφορά διατάξεις με περισσότερες Kinect (*multi – Kinect set – up*). Οι συγγραφείς κατατάσσουν το θόρυβο σε τρεις βασικές κατηγορίες:

1. Χωρικός Θόρυβος (*Spatial Noise*): ορατός σε ένα frame
2. Παροδικός Θόρυβος (*Temporal Noise*): ορατός σε μια σειρά από διαδοχικά frames
3. Θόρυβος Παρεμβολής (*Interference Noise*): προκύπτει σε διατάξεις με πολλαπλές Kinect

Ο χωρικός παρουσιάζει εξάρτηση από την απόσταση από την κάμερα, τη γεωμετρία της εικόνας, την επιφάνεια των αντικειμένων πάνω στα οποία προσπίπτει η υπέρυθρη ακτινοβολία και την τεχνολογία του αισθητήρα. Ο παροδικός δεν έχει μελετηθεί ενδελεχώς, ωστόσο έχει αποδειχθεί πειραματικά ότι οι μετρούμενες αποκλίσεις από frame σε frame έχουν να κάνουν με τον τρόπο λειτουργίας της κάμερας και είναι αρκετά συσχετισμένες. Ο θόρυβος λόγω παρεμβολής εμφανίστηκε ως αποτέλεσμα της ανάγκης να χρησιμοποιήσουμε περισσότερες της μίας Kinect, για να υπερκεράσουμε προβλήματα που σχετίζονται με το περιορισμένο πεδίο θέασης και κάποια είδη σκιών. Για να ελέγχουμε το θόρυβο στην πηγή, προχωρούμε σε διάφορα είδη πολυπλεξίας. Η χωρική αντιμετώπιση (*Space Division Multiplex*) λαμβάνει γεωμετρικά ανεξάρτητες όψεις, αλλά παρουσιάζει ικανοποιητικά αποτελέσματα σε περιορισμένο αριθμό διατάξεων. Μια άλλη προσέγγιση είναι η χρονική (*Time Division Multiplex*), όπου κάθε Kinect λειτουργεί μονάχα εντός ενός χρονικού παραθύρου (*time slot*). Τέλος, το *Pattern Division Multiplex* επιτρέπει επικαλυπτόμενα μοτίβα υπέρυθρης ακτινοβολίας, με την κάθε Kinect να αναλαμβάνει την απομόνωση του δικού του προτύπου από τα υπόλοιπα.

Όσο αφορά την αντιμετώπιση του προβλήματος του θορύβου, βασικός στόχος είναι η δημιουργία τεχνικών, οι οποίες θα επιχειρήσουν να γεμίσουν τις τρύπες με όσο το δυνατόν ορθότερη πληροφορία. Στη βιβλιογραφία, εμφανίζονται τόσο προσεγγίσεις που γεμίζουν με συμπαγή καθολικό τρόπο τις τρύπες (βλ. [3]), όσο και άλλες που στοχεύουν περισσότερο σε μία συγκεκριμένη κατηγορία θορύβου (βλ. [4], [5], [6]). Οι πρώτες μέθοδοι επιδιώκουν ένα “αφελές” (*naïve*) γέμισμα, βασιζόμενο κυρίως σε γεωμετρικά μαθηματικά μοντέλα και στην πληροφορία της γειτονιάς ενός pixel, η οποία εξάγεται μέσω συγκεκριμένων φίλτρων. Η απλούστερη κατηγορία φίλτρων είναι τα *median*, τα οποία λαμβάνουν ένα συγκεκριμένο παράθυρο γειτόνων για να υπολογίσουν το μέσο όρο γειτονιάς. Μια άλλη ευρέως χρησιμοποιούμενη στη βιβλιογραφία κατηγορία φίλτρων είναι τα *bilateral*. Στα φίλτρα αυτά, η απόχρωση που αντιστοιχεί σε ένα pixel εξάγεται από ένα σταθμισμένο μέσο της γειτονιάς (με χρήση για παράδειγμα γκαουσιανής κατανομής). Ως αποτέλεσμα, τα βάρη δεν εξαρτώνται μόνο από την ευκλείδεια απόσταση μεταξύ των pixels, αλλά και από την ευρύτερη φωτομετρική ομοιότητα. Η χρήση των παραπάνω φίλτρων καθώς και άλλων



εκτός αυτών ενσωματώνεται σε ευρύτερες τεχνικές προεπεξεργασίας και εφαρμογής απλών *image – processing* εργαλείων για ταξινόμηση και ειδική μεταχείριση τρυπών, μερικές από τις οποίες θα αναλύσουμε παρακάτω.

Στην εργασία των [3] προτείνονται δύο μέθοδοι γεμίσματος των περιοχών θορύβου, με την χρήση ενός μορφολογικού μετασχηματισμού της εικόνας. Στην πρώτη μέθοδο, ο αλγόριθμος δημιουργεί δύο δομικά στοιχεία, που αποσκοπούν στο να εντοπίσουν τις περιοχές έγκυρης πληροφορίας βάθους και τις αντίστοιχες θορύβου. Τα στοιχεία αυτά είναι πυρήνες (kernels) 11x11 στοιχείων, οι οποίοι εφαρμόζονται στην εικόνα σε τέσσερις διαφορετικές κατεύθυνσεις (N-S, W-E, NW-SE, NE-SW), με σκοπό να εντοπίσουν μεταβάσεις από περιοχές πληροφορίας (πρώτο δομικό στοιχείο) σε περιοχές θορύβου (δεύτερο δομικό στοιχείο) και αντίστροφα. Οι τρύπες γεμίζουν με τη μέγιστη τιμή βάθους που βρέθηκε στη συγκεκριμένη κατεύθυνση της μετάβασης, με τη λογική ότι οι σκιές που δημιουργούνται προβάλλονται πάντα στο παρασκήνιο. Στη δεύτερη μέθοδο, η σάρωση γίνεται μόνο οριζόντια, σαρώνοντας τον χάρτη γραμμή προς γραμμή. Σε κάθε σάρωση, ο αλγόριθμος δημιουργεί ένα προφίλ των γκρι τιμών της γραμμής pixel προς pixel και εντοπίζει μεταβάσεις από γκρι τιμή σε θόρυβο και αντίστροφα. Για κάθε τέτοια μετάβαση, οι τρύπες γεμίζονται με τη μέγιστη τιμή βάθους που βρέθηκε στο συγκεκριμένο προφίλ της μετάβασης, ακολουθώντας την προηγούμενη λογική. Οι δύο μέθοδοι καταφέρνουν πράγματι να απομακρύνουν τη σκιά από την εικόνα, ωστόσο τα πειράματα που πάρθηκαν δεν είναι αντιπροσωπευτικά, καθώς φαίνεται να υπάρχει πρόβλημα σε συγκεκριμένες γεωμετρίες διαφορετικών αποστάσεων μεταξύ αντικειμένων στη σκηνή.

Οι [4] επικεντρώνονται στην απαλοιφή θορύβου που δημιουργείται από σκιές στο παρασκήνιο λόγω παρεμβολής αντικειμένων, ενώ παράλληλα προσπαθούν να χρησιμοποιήσουν κατάλληλα την πληροφορία του θορύβου για την εξαγωγή αντικειμένων. Οι συγγραφείς χρησιμοποιούν ένα μαθηματικό μοντέλο, εξαγόμενο από την μελέτη της γεωμετρίας του προβλήματος παρεμβολής στην κάμερα, για να ταξινομήσουν τη δημιουργία σκιάς σε τρεις κατηγορίες: τις προσκείμενες στο παρεμβαλλόμενο αντικείμενο σκιές, τις παρακείμενες μεταξύ προσκηνίου και παρασκηνίου σκιές και τις σκιές ευθυγράμμισης με την RGB εικόνα. Το μοντέλο μελετά τις ακμές των περιοχών έγκυρης πληροφορίας βάθους και τρύπας, για να κατασκευάσει τους *classifiers* χρησιμοποιώντας το μήκος τους και συγκρίνει το εκτιμώμενο μήκος της τρύπας σε κάθε frame για να την κατατάξει. Το πρόβλημα της μεθόδου, αν και παρουσιάζονται ενδιαφέρουσες εφαρμογές στην εξαγωγή αντικειμένων προσκηνίου, είναι ότι το *ground truth* κατασκευάζεται χειροκίνητα με τη βοήθεια του μαθηματικού μοντέλου, οπότε οι μετρικές αξιολόγησης του *classification* που πραγματοποιείται δεν δίνουν έγκυρα αποτελέσματα για όλες τις πηγές θορύβου.

Η επόμενη εργασία [5], παρόμοια με την [4], ασχολείται αποκλειστικά με την εύρεση φυσικών σκιών και το γέμισμά τους. Μελετώντας τη γεωμετρία του φαινομένου κατά το οποίο η ύπαρξη ενός αντικειμένου στο προσκήνιο δημιουργεί σκιές και χρησιμοποιώντας αντίστοιχο μαθηματικό μοντέλο, αναπτύσσεται μια μετρική, η διακρίνουσα σκιάς, η οποία βασίζεται στην απόσταση συνόρων αντικειμένου και περιοχών σκιάς. Στη συνέχεια, ο χάρτης βάθους σαρώνεται γραμμή προς γραμμή για την αναζήτηση των οριακών pixels αντικειμένου (περιοχή έγκυρης πληροφορίας βάθους) και τρύπας. Για κάθε σημείο εντός της περιοχής αυτής, υπολογίζεται η παραπάνω διακρίνουσα και το στοιχείο χαρακτηρίζεται ως αντικείμενο ή θόρυβος. Τελικά, αφού έχουν εξερευνηθεί οι τρύπες εντός του χάρτη, κάθε pixel περιοχής σκιάς γεμίζεται με ένα πολωμένο προς τις μεσαίες τιμές μέσο όρο των τιμών βάθους 6 γειτονικών pixels στην οριζόντια διάσταση.



Οι [6] προτείνουν έναν αλγόριθμο απάλειψης θορύβου, ο οποίος βασίζεται σε classification του θορύβου σε χωρικό (*Spatial* – ορατός σε ένα frame) και παροδικό (*Temporal* – ορατό σε συνεχόμενα frames) γύρω από τις ακμές της εικόνας. Εκμεταλλευόμενοι την αρχική έγχρωμη εικόνα, εξάγουν τις ακμές με βάση την υφή. Ταυτόχρονα, γεμίζουμε τις τρύπες του χάρτη βάθους με τη χρήση ενός *joint bilateral* φίλτρου. Με τη βοήθεια των παραπάνω, διαχωρίζουμε το χάρτη βάθους σε περιοχές ακμών και ομαλές. Στη συνέχεια, απομακρύνουμε τον spatial θόρυβο από τις περιοχές ακμών, βρίσκοντας την κυρίαρχη τιμή βάθους, την οποία αποδίδουμε σε εκείνα τα pixels των οπίων η αρχική τιμή βάθους απέχει πολύ από την κυρίαρχη. Τέλος, η ίδια διαδικασία ακολουθείται και για τον temporal θόρυβο, μόνο που εδώ προβαίνουμε στην εφαρμογή της διαδοχικά για κάθε frame μιας ακολουθίας, αφού προσδιορίσουμε κάθε περιοχή ακμών για όλα τα frames της. Συνδυάζοντας τις πλέον επεξεργασμένες περιοχές ακμών και τις ομαλές, προκύπτει ένας χάρτης βάθους με καλύτερη αποτύπωση των σχημάτων των αντικειμένων της εικόνας και λιγότερη απώλεια πληροφορίας.

2.2 Χρήση RGB Πληροφορίας

Όπως έχει ήδη αναφερθεί, πέρα από τις βασικές τετριμμένες τεχνικές επεξεργασίας χαρτών βάθους για αποσφαλμάτωση των περιοχών θορύβου, υπάρχουν ποικίλες καινοτόμες προσεγγίσεις, οι οποίες επιδιώκουν τόσο να εκμεταλλευτούν τα περισσότερα δυνατά στοιχεία πληροφορίας, όσο και να διατηρήσουν την απόδοση της κάμερας σε υψηλά επίπεδα. Ο συνδυασμός αυτός έχει οδηγήσει την επιστημονική κοινότητα σε μεθόδους επεξεργασίας χαρτών βάθους που επιχειρούν να αξιοποιήσουν πληροφορία που παρέχεται από τον RGB αισθητήρα της κάμερας σχετικά με τα αντικείμενα που παρουσιάζονται στην σκηνή. Όπως καταλαβαίνουμε, πληροφορία όπως χρωματική ομοιότητα ή ύπαρξη διακριτών ακμών μεταξύ διαφορετικών περιοχών της σκηνής σηματοδοτούν την ύπαρξη διακριτών αντικειμένων στον χώρο. Αυτή τη γνώση σε συνδυασμό με τον χάρτη βάθους επιχειρούν να θέσουν σε εφαρμογή πολλές υπάρχουσες υλοποιήσεις.

Οστόσο, ένα στάδιο προεπεξεργασίας που απαιτείται, προτού γίνει εφικτή η χρήση της έγχρωμης εικόνας, είναι η ευθυγράμμιση της με τον χάρτη βάθους της Kinect, καθώς και η λεγόμενη βαθμονόμηση (*calibration*) της κάμερας. Γενικά στο τομέα της μηχανικής όρασης, η βαθμονόμηση είναι απαραίτητο βήμα για την τρισδιάστατη ανακατασκευή της σκηνής. Οι διαφορετικές κάμερες πρέπει να ρυθμιστούν τόσο εσωτερικά, όσο και σε σχέση με εξωτερικές σχετικές παραμέτρους οπτικής. Στη συνέχεια, με την χρήση γεωμετρικών μετασχηματισμών, αντιστοιχίζουν ένα προς ένα σημεία της έγχρωμης και της εικόνας βάθους. Χαρακτηριστικά στο [7], υλοποίηση βέβαια που επικεντρώνεται στην εξαγωγή χειρονομιών ανθρώπινων χεριών, επιδιώκεται να ευθυγραμμιστούν οι δύο εικόνες με χρήση γενετικών αλγορίθμων για τον εντοπισμό σημείων – κλειδιών, τόσο στην έγχρωμη, όσο και στην εικόνα βάθους και προτείνεται ένα μαθηματικό μοντέλο μετασχηματισμών των 2D συντεταγμένων κάθε εικόνας για αντιστοίχιση τόσο μεταξύ τους, όσο και σε ένα κοινό 3D πλαίσιο. Η μέθοδος αυτή γίνεται ανθεκτική στο θόρυβο που παράγουν οι γενετικοί αλγόριθμοι, με την χρήση απλών εργαλείων επεξεργασίας εικόνας, όπως εύρεση γωνιών ή ακμών στην εικόνα. Τα μαθηματικά εργαλεία που παρέχονται για την αντιστοίχιση της πληροφορίας βάθους με την RGB εικόνα ενδέχεται να κριθούν σημαντικά



στην ανάπτυξη μεθόδων της απαλοιφής θορύβου σκιάς.

2.2.1 Χρήση Πληροφορίας Ακμών – Διόρθωση Παραμόρφωσης Συνόρου

Η ύπαρξη των φαινομένων παρεμβολής αντικειμένου, σκίασης ή *scattering*, πέρα από την εισαγωγή θορύβου σε περιοχές του χάρτη βάθους, οδηγεί, δυστυχώς, σε ένα ακόμα πολύ σημαντικό πρόβλημα υποβάθμισης της λειτουργίας των καμερών βάθους, την παραμόρφωση των ακμών μεταξύ διαφορετικών αντικειμένων. Προκειμένου το εκτιμώμενο βάθος που παράγει η Kinect να καθίσταται χρήσιμο για τις διάφορες εφαρμογές, το πρόβλημα έχει επιχειρηθεί να λυθεί στη βιβλιογραφία.

Οι [8], επιχειρούν να διορθώσουν αυτήν την παραμόρφωση εξερευνώντας και αξιοποιώντας πληροφορία για τις ακμές που εμφανίζονται στη σκηνή της RGB εικόνας. Ο βασικός στόχος είναι η απάλειψη της παραμόρφωσης ακμών, προκειμένου ο χάρτης βάθους και η RGB εικόνα να ευθυγραμμίζονται τέλεια. Σε πρώτη φάση, πραγματοποιείται κατάλληλη βαθμονόμηση της Kinect για την ευθυγράμμιση έγχρωμης εικόνας και εικόνας βάθους, καθώς και ένα υποτυπώδες γέμισμα των τρυπών. Στη συνέχεια, από την RGB πληροφορία παράγεται ένας χάρτης που εντοπίζει συνεχείς λεπτές κλειστές καμπύλες των αντικειμένων της σκηνής, εφαρμόζοντας το λεγόμενο *edge-guided inpainting*. Ο αλγόριθμος που χρησιμοποιείται για τη διαδικασία του *inpainting* μπορεί να εξηγηθεί μέσω της αντιστοίχισης με ένα μοντέλο θερμοροής, όπου τα γνωστά pixels έγκυρου βάθους είναι η πηγή θερμότητας και τα σημεία που αποτελούν τις κλειστές καμπύλες – ακμές λειτουργούν ως ένας τέλειος μονωτής με μηδενική αγωγιμότητα.

Αντίθετα, οι [9] πραγματεύονται την αφαίρεση της παραμόρφωσης ακμών στο χάρτη βάθους με χρήση *alpha matting*. Ο προτεινόμενος αλγόριθμος απαρτίζεται από τρία μέρη. Στο πρώτο, πραγματοποιείται εκτίμηση των διακριτών επιπέδων βάθους. Συγκεκριμένα, εκμεταλλεύομενο το *disparity map* που εξάγεται από ένα ζεύγος δεξιάς και αριστερής έγχρωμης εικόνας (*stereo image pair*), παράγεται ένας χάρτης βάθους και το ιστόγραμμα των αντίστοιχων τιμών του. Από το τελευταίο, υπολογίζονται τα τοπικά ελάχιστα και, με βάση αυτά, διαχωρίζεται το σύνολο των διαφορετικών τιμών βάθους σε μη επικαλυπτόμενες περιοχές. Στο δεύτερο μέρος, απομονώνεται το προσκήνιο από το παρασκήνιο, με δημιουργία τοπικών μασκών για κάθε μία από τις παραπάνω περιοχές, ώστε να αποσπαστεί το σχήμα των αντικειμένων της εικόνας (*alpha matting*). Για κάθε μία από τις μάσκες, προκύπτει ένας αντίστοιχος χάρτης βάθους, του οποίου τα απροσδιόριστα ως προς το βάθος pixels γεμίζονται με *exemplar based* αλγόριθμο. Τέλος, οι χάρτες αθροίζονται για να προκύψει ο τελικός. Ο αλγόριθμος βελτιώνει το πρόβλημα της παραμόρφωσης σε όλες τις εξεταζόμενες περιπτώσεις στατικής εικόνας.

2.2.2 Χρήση Πληροφορίας Χρώματος

Οι προσεγγίσεις αυτές, επιδιώκουν να χρησιμοποιήσουν τη χρωματική ομοιότητα μεταξύ



διαφορετικών περιοχών στην έγχρωμη εικόνα και να τη συνδυάσουν είτε με προαναφερθείσες τεχνικές γεμίσματος, είτε χρησιμοποιώντας τις ακμές. Στην υλοποίηση των [10], εφαρμόζεται μία εναλλακτική μέθοδος ευθυγράμμισης του χάρτη βάθους με την εικόνα, όχι μέσω γεωμετρικών μετασχηματισμών για ένα προς ένα αντιστοίχιση σημείων, αλλά με την χρήση ενός αλγορίθμου που λαμβάνει τα pixels της εικόνας ως τυχαίες μεταβλητές και χρησιμοποιεί τις εντροπίες των κατανομών τους, τόσο ξεχωριστά, όσο και συσχετιζόμενα, για την εξαγωγή μιας μετρικής κοινής πληροφορίας. Η συσχέτιση αυτή χρησιμοποιείται για τη κατηγοριοποίηση περιοχών σε features και παρέχει πληροφορία για τη μεταξύ τους συσχέτιση. Μετά την ευθυγράμμιση, ο χάρτης επαναδειγματοληπτείται σε ανάλυση 1280x960 και γύρω από κάθε περιοχή θορύβου δημιουργείται ένα παράθυρο, μεγέθους εξαρτώμενου από το μέγεθος της περιοχής απροσδιόριστου βάθους, όπου μελετάται η πληροφορία χρώματος στα αντίστοιχα features της RGB εικόνας. Με την χρήση του γνωστού bilateral φίλτρου, οι τιμές βάθους εισάγονται σε κάθε pixel της τρύπας. Τέλος, εφαρμόζεται ένα median φίλτρο για το *smoothing* της παραμόρφωσης που εισήγαγε η υπερδειγματοληψία.

Παρόμοια, οι [11], προεπεξεργάζονται τον χάρτη βάθους για την αναγνώριση των τρυπών και την κατηγοριοποίηση τους με βάση το μέγεθος. Μικρές τρύπες (με σταθερά κατώφλια) γεμίζονται με joint bilateral φίλτρα. Ωστόσο, εάν η τρύπα κατηγοριοποιηθεί ως μεγάλη, εφαρμόζεται μια διαστολή της για συμμετοχή έγκυρων τιμών βάθους γειτονικών περιοχών. Για κάθε τέτοια περιοχή, γίνεται η διαμόρφωση ενός μεγαλύτερου παραθύρου και στην ευθυγραμμισμένη RGB εικόνα, επιχειρείται μια ομαδοποίηση των pixels με βάση την τιμή χρώματος τους, με την χρήση *MeanShift* αλγορίθμου. Κάθε παραγόμενη ομάδα περιέχει σημεία εντός τρύπας, εκτός τρύπας ή και τα δύο. Η πρώτη περίπτωση αντιμετωπίζεται όπως οι τρύπες μικρού μεγέθους και οι δεύτερες δεν παρουσιάζουν ενδιαφέρον για διόρθωση θορύβου. Για την τρίτη περίπτωση, δημιουργείται ένα ιστόγραμμα τη διαφορετικών τιμών βάθους και στη συνέχεια εντοπίζεται το πλήθος των γειτονικών pixels ίδιας ομάδας, που ανήκουν στο ίδιο επίπεδο. Τέλος, κάθε επίπεδο γεμίζεται με την πληροφορία βάθους της ομάδας, σε συνδυασμό με κανονικοποιημένες στοχαστικές παραμέτρους.

Μια εναλλακτική προσέγγιση αυτής της τεχνικής ακολουθούν οι [12], όπου σε πρώτη φάση πάλι με *MeanShift* αλγόριθμο, πραγματοποιείται τμηματοποίηση της εικόνας σε περιοχές ίδιας πληροφορίας χρώματος. Για κάθε pixel που ανήκει σε περιοχή διαφορετικού χρώματος, η τιμή βάθους συμπληρώνεται με την χρήση σταθμισμένου μέσου όρου τιμών βάθους γειτόνων, που ανήκουν στην ίδια ομάδα. Ωστόσο, η υλοποίηση αυτή επιδιώκει, με την χρήση ενός *fast-guided* φίλτρου παρόμοιου με το [5], να εκμεταλλευτεί και την πληροφορία που παρέχουν οι ακμές στην έγχρωμη εικόνα. Η λογική είναι ότι παράγονται κατώφλια που περιορίζουν την ύπαρξη pixels θορύβου σε ομάδες ίδιου χρώματος εντός του ίδιου συνόρου.

Πέρα από την ομαδοποίηση περιοχών με βάση το χρώμα και το *inpainting* εντός ακμών, έχουν αποπειραθεί και πιο τολμηρές μέθοδοι στη βιβλιογραφία, οι οποίες χρησιμοποιούν εξελικτική πληροφορία για τις εικόνες που προκύπτουν από βίντεο.

Οι [13], επιχειρούν να γεμίσουν τις περιοχές απροσδιόριστου βάθους του χάρτη βάθους της Kinect, εντοπίζοντας και αξιοποιώντας κινούμενα αντικείμενα σε ακολουθίες video. Αρχικά, από την τελευταία, εξάγεται η έγχρωμη εικόνα και το background της. Ο συνδυασμός αυτών των δύο μας παρέχει τη λεγόμενη διαφορική (*Differential*) εικόνα, η οποία εκφράζει αν ένα pixel ανήκει στο παρασκήνιο ή σε κάποιο κινούμενο κομμάτι. Από



την παραπάνω εικόνα, παράγεται ένας πίνακας αναφοράς που μας πληροφορεί για το αν ένα pixel ανήκει σε κανονική περιοχή, σε κάποια τρύπα του χάρτη βάθους ή σε κινούμενο αντικείμενο. Στη συνέχεια, διορθώνεται ο προηγούμενος πίνακας, καθώς και εκείνος του βάθους, ώστε να συμπεριλάβουμε στα pixels με κακώς ορισμένη πληροφορία βάθους εκείνα που είναι γειτονικά των κινούμενων αντικειμένων. Τέλος, διορθώνεται ο χάρτης σε δύο στάδια, με το πρώτο να αφορά καθαρά τις αρχικές τρύπες και το δεύτερο τα κινούμενα αντικείμενα με τις γειτονικές τους περιοχές. Ο παραγόμενος χάρτης βάθους παρουσιάζει περισσότερη πληροφορία στις περιοχές που βρίσκονται στα όρια των κινούμενων αντικειμένων.

Προχωρώντας σε πιο προηγμένες τεχνικές, οι [14] δημιούργησαν και εκπαίδευσαν ένα βαθύ συνελικτικό νευρωνικό δίκτυο με στόχο την ανίχνευση των ακμών σε περιοχές με θόρυβο έμφραξης, τόσο από RGB – D, όσο και μόνο από RGB δεδομένα. Το δίκτυο, αποτελούμενο από τρία ζεύγη επιπέδων συνέλιξης και *pooling*, εξετάζει την εικόνα σε τμήματα 32x32 pixels για να αποφασίσει για την πιθανή ύπαρξη τέτοιων ακμών. Όπως είναι αναμενόμενο, στο τέλος, οι συγγραφείς παρουσιάζουν το tradeoff μεταξύ ακρίβειας και ταχύτητας εκτέλεσης, φαινόμενο των περισσότερων τέτοιων εφαρμογών.

Τέλος, αξίζει να αναφέρουμε την υλοποίηση των [15], οι οποίοι γεμίζουν τις τρύπες του χάρτη σταδιακά, βασιζόμενοι στο παρασκήνιο των έγχρωμων εικόνων. Αρχικά, με την χρήση πιθανοτικών μοντέλων τυχαίας διάσχισης χρώματος [16], προτείνεται μια χρωματική τμηματοποίηση της εικόνας, με βάση την οποία γίνεται ένα πρόχειρο γέμισμα τρυπών που οφείλονται σε ανακλαστικές επιφάνειες. Έπειτα, λαμβάνεται όλος ο υπόλοιπος θόρυβος ως περιοχές του παρασκηνίου. Η πληροφορία του βάθους προκύπτει με την μελέτη τοπικών περιοχών έγκυρου βάθους γύρω από τις περιοχές απροσδιόριστου βάθους, με τις τελευταίες να γεμίζονται, αρχικά, με χρήση αυτής της αφελούς σταθμισμένης πληροφορίας των γειτονιών. Έπειτα, καθώς τα frames διαδέχονται το ένα το άλλο στο βίντεο, το γέμισμα των χαρτών με βάση την ομοιότητα χρώματος και την πληροφορία συνδυάζονται, προκειμένου να διορθωθεί θόρυβος που οφείλεται στην παραμόρφωση των ακμών αντικειμένων. Ο αλγόριθμος εξελικτικά παράγει κατώφλια τιμών βάθους για τις διάφορες γειτονιές τρυπών, χρησιμοποιώντας την πληροφορία από την αναζήτηση χρωματικής ομοιότητας και εφαρμόζοντας ουσιαστικά classification στις περιοχές θορύβου ανάλογα με τον αριθμό των τοπικών μεγίστων που παρουσιάζουν στη κάθε γειτονιά. Ως αποτέλεσμα, η εικόνα τμηματοποιείται και η καινούργια εξαγόμενη από το χρώμα πληροφορία αποφασίζει το κατώφλι για κάθε τμηματοποιημένη γειτονιά, διορθώνοντας προσδευτικά την παραμόρφωση.

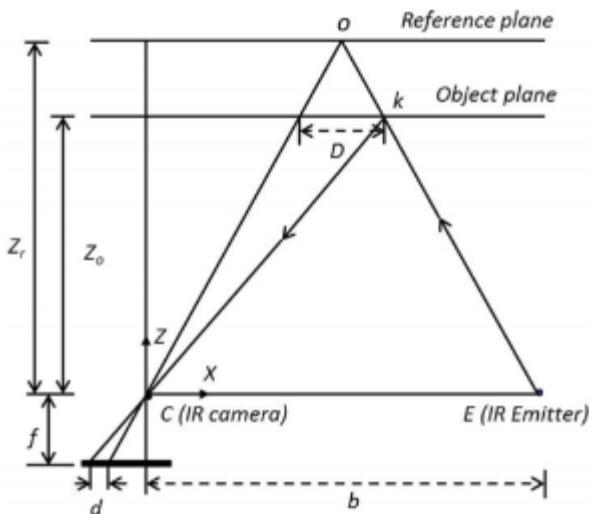


3. Εργαλεία Υλικού και Λογισμικού

Στο κεφάλαιο αυτό, θα περιγράψουμε σε βάθος το σύνολο των εργαλείων υλικού και λογισμικού που χρησιμοποιήσαμε, για να υλοποιήσουμε το σύστημα αφαίρεσης θορύβου σκιάς από εικόνα βάθους. Αρχικά, περιγράφουμε τις προδιαγραφές λειτουργίας των καμερών βάθους *Microsoft Kinect v.1.* και *ASUS Xtion*, από τις οποίες λάβαμε τα *RGBD* δεδομένα και διεξήγαμε τα πειράματα. Στη συνέχεια, περιγράφουμε τις βιβλιοθήκες της γλώσσας προγραμματισμού *Python*, τους υλοποιημένους αλγορίθμους επεξεργασίας εικόνας του προγραμματιστικού *API OpenCv 3.0*, και τα πακέτα διασύνδεσης και επεξεργασίας ψηφιακού σήματος του συστήματος *Robot Operating System*, εργαλεία τα οποία συνθέτουν το σύστημα λογισμικού μας.

3.1 Υλικό

Όπως αναφέραμε στην εισαγωγική παράγραφο, ο αλγόριθμος επεξεργασίας εικόνας που εφαρμόζουμε για την αφαίρεση του θορύβου σκιάς από τους χάρτες βάθους, εφαρμόζεται στον χάρτη ως μια 640×480 8-bit μονοχρωματική εικόνα από τον αισθητήρα βάθους που χρησιμοποιήσαμε. Η ανακατασκευή της πληροφορίας βάθους για κάθε pixel μιας τέτοιας εικόνας γίνεται εσωτερικά από το πρόγραμμα οδήγησης της αντίστοιχης κάμερας, έτσι ώστε λαμβάνοντας υπ' όψιν τις ενδογενείς παραμέτρους λειτουργίας και την αρχιτεκτονική των στελεχών της κάμερας, να αποδοθεί ως προς οποιοδήποτε σύστημα συντεταγμένων αναφοράς της κάμερας. Συνεπώς, το πρόγραμμα οδήγησης της κάμερας παρέχει στον αλγόριθμο μας έτοιμη αυτή την εικόνα, είτε στην μορφή ωμών δεδομένων απόστασης (πίνακας με float τιμές σε millimeters), είτε κλιμακοποιημένων κατάλληλα στις 255 ζώνες απόχρωσης της μονοχρωματικής εικόνας. Ενδεικτική παρουσίαση του μοντέλου για τον υπολογισμό του βάθους φαίνεται στην εικόνα 3.1.1.



Εικόνα 3.1.1: Επίδειξη Μοντέλου Εξαγωγής Βάθους από Αισθητήρα Βάθους



Παρακάτω ακολουθούν πίνακες με τα χαρακτηριστικά αυτά των καμερών βάθους που χρησιμοποιήθηκαν για τη λήψη εικόνων βάθους.

3.1.1 Microsoft Kinect v.1

Η κυκλοφορία της κάμερας Kinect της Microsoft έγινε τον Νοέμβριο του 2010. Ο αρχικός στόχος ήταν η δημιουργία μιας διεπαφής εισόδου για την πλατφόρμα του XBOX 360, αλλά η χρήση της επεκτάθηκε άμεσα και σε άλλους τομείς. Η κάμερα καθώς και οι τεχνικές προδιαγραφές της παρουσιάζονται στις δύο παρακάτω εικόνες, όπως έχουν δημοσιευτεί από τους [17].



Εικόνα 3.1.1.1: Κάμερα Βάθους Microsoft Kinect v.1

KINECT DEVICE SPECIFICATIONS [1]

Parameter	Values	
Spatial Resolution#	RGB / Depth / IR	640 pix × 480 pix
	X	1.70mm / pix / meter
	Y	1.64mm / pix / meter
Depth Range*	Default	0.8m–4.0m
	Near	0.4m–3.0m
Depth Resolution*	2mm to 40mm (depending on depth)	
Frame rate	30 fps	
Field Of View (FOV)	43° Vertical by 57° Horizontal	
Tilt Range	±27° Vertical	
Focal length [2]	Depth	5.453 ± 0.012mm
	RGB	4.884 ± 0.006mm
IR Wavelength [3]	830nm	

Πίνακας 3.1.1.1: Τεχνικές Προδιαγραφές Microsoft Kinect v.1

Πληροφορίες σχετικά με τα ενδογενή χαρακτηριστικά της κάμερας, τα οποία προκύπτουν από το καλιμπράρισμα των αισθητήρων της, παρέχονται αναλυτικά στο έργο των [18]. Εμείς παρουσιάζουμε ενδεικτικά έναν πίνακα με τον ορισμό των χαρακτηριστικών αυτών καθώς και των αντίστοιχων τιμών που προέκυψαν από το καλιμπράρισμα, τιμές τις οποίες χρησιμοποιήσαμε για την κάμερα μας κατά τη λήψη δεδομένων και εκτέλεση πειραμάτων.



Parameters	Descriptions
$f_{RGB}, cx_{RGB}, cy_{RGB}$	Intrinsic Matrix of color camera
k_1, k_2, k_3, p_1, p_2	Distortion vector of color camera
f_{IR}, cx_{IR}, cy_{IR}	Intrinsic Matrix of IR camera
k_1, k_2, k_3, p_1, p_2	Distortion vector of IR camera
R, T	Extrinsic matrix between color and IR cameras ¹
b	Baseline between IR camera and IR projector
d_{off}	Depth offset

Εικόνα 3.1.1.2: Ενδογενή Χαρακτηριστικά Microsoft Kinect v.1 – Ορισμός

Τα χαρακτηριστικά αυτά λαμβάνονται υπ' όψιν από το πρόγραμμα οδήγησης της κάμερας για τον ορθό υπολογισμό του βάθους και την κατασκευή των εικόνων βάθους, καθώς και για την επεξεργασία τους, για μεταφορά του χάρτη βάθους σε διαφορετικό σύστημα συντεταγμένων, ανακατασκευή σκηνής κ.α.

Parameters	IR Projector Covered	IR Projector Uncovered
RGB Intrinsic	$\begin{bmatrix} 517.055 & 0 & 315.008 \\ 0 & 517.679 & 264.155 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 514.120 & 0 & 310.744 \\ 0 & 513.841 & 262.611 \\ 0 & 0 & 1 \end{bmatrix}$
RGB Distortion	$\begin{bmatrix} 2.2658e-1 & -7.5265e-1 & 2.4148e-3 \\ & -1.9091e-3 & 8.3151e-1 \end{bmatrix}$	$\begin{bmatrix} 2.0456e-1 & -4.5719e-1 & 7.7826e-4 \\ & -3.8524e-3 & -5.5729e-1 \end{bmatrix}$
IR Intrinsic	$\begin{bmatrix} 580.606 & 0 & 314.758 \\ 0 & 580.885 & 252.187 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 596.270 & 0 & 321.490 \\ 0 & 597.689 & 250.363 \\ 0 & 0 & 1 \end{bmatrix}$
IR Distortion	$\begin{bmatrix} -1.8760e-1 & 1.013 & 2.3033e-4 \\ & -2.6935e-3 & -1.8375 \end{bmatrix}$	$\begin{bmatrix} -2.5917e-1 & 1.4193 & -3.9565e-3 \\ & -6.6566e-3 & -2.5772e-1 \end{bmatrix}$
R	$[25.06 \quad 0.65 \quad -2.1]$	$[23.41 \quad -3.16 \quad 15.48]$
$T(mm)$	N/A	$[82.63 \quad 1090.39]$
$b(mm), d_{off}$		1.19378
Projection Error(<i>pixel</i>)	0.68848	

Εικόνα 3.1.1.3: Ενδογενή Χαρακτηριστικά Microsoft Kinect v.1 - Βαθμονόμηση

Η οδήγηση της κάμερας και η λήψη των RGBD δεδομένων γίνεται εντός του προγραμματιστικού περιβάλλοντος ROS, με χρήση του OpenNI Framework¹ (βλ. Παράγραφο 3.4.2).

¹ <https://en.wikipedia.org/wiki/OpenNI>, <https://github.com/OpenNI/OpenNI>



3.1.2 ASUS Xtion



Εικόνα 3.1.2.1: Κάμερα Βάθους ASUS Xtion

(α) Η κάμερα, (β) Μια IR εικόνα από το μοτίβο τελειών δομημένου φωτός της κάμερας

Distance of Use	Between 0.8 and 3.5m
Field of View	58°H 45°V, 70°D
Depth Image Size	QVGA (320x240)

Πίνακας 3.1.2.2: Τεχνικές Προδιαγραφές Κάμερας ASUS Xtion

Δημοσιευμένες έρευνες για βαθμονόμηση της συγκεκριμένης κάμερας, αντίστοιχες με της Παραγράφου 3.1.1, δεν βρέθηκαν για το συγκεκριμένο τύπο (οι έρευνες έχουν γίνει σχεδόν αποκλειστικά για το μοντέλο ASUS Xtion Pro), για το λόγο αυτό παρουσιάζονται ενδεικτικά οι τιμές όπως εκτυπώνονται στην πλατφόρμα του συστήματος ROS από το πρόγραμμα οδήγησης της κάμερας, έπειτα από βαθμονόμηση την οποία εφαρμόσαμε οι ίδιοι.

Η οδήγηση της κάμερας και η λήψη των RGBD δεδομένων γίνεται εντός του προγραμματιστικού περιβάλλοντος ROS, με χρήση του OpenNI 2 Framework² (βλ. Παράγραφο 3.4.2).

² <https://github.com/OpenNI/OpenNI2>



3.2 Γλώσσα Προγραμματισμού Python

Η δημιουργία της Python³ ξεκίνησε στις αρχές του 1990 από τον Guido van Rossum, ο οποίος είχε, ως αρχικό στόχο, την υλοποίηση ενός μεταγλωττιστή για hackers Unix/C. Από την αρχή της κυκλοφορίας της, ωστόσο, κατέστη σαφές ότι, ως γλώσσα υψηλού επιπέδου, παρέχει ιδιαίτερη ευελιξία και ένα τεράστιο εύρος δυνατοτήτων, ικανό να την καταστήσει ισχυρή για την αντιμετώπιση ποικιλίας προγραμματιστικών προβλημάτων. Το γεγονός αυτό την ανέδειξε ως πρώτη επιλογή για την αντιμετώπιση του προβλήματος που πραγματεύεται η παρούσα διπλωματική. Στην παράγραφο αυτή, θα περιγράφουμε τα εργαλεία της Python, τα οποία χρησιμοποιήθηκαν για την υλοποίηση μας. Θα αναφερθούμε, αρχικά, στη βιβλιοθήκη Numpy⁴ και στον τρόπο που αυτή επιτρέπει τον χειρισμό μίας εικόνας. Ακόμη, θα αναλύσουμε τη λειτουργία του αλγορίθμου MeanShift και τον τρόπο υλοποίησης του σε Python από τους Chris M. Christoudias και Bogdan Georgescu⁵. Τέλος, θα επισημάνουμε τη λογική της βιβλιοθήκης threading και τον τρόπο με τον οποίο, με τη βοήθειά της, καθίσταται δυνατή η μείωση της ταχύτητας εκτέλεσης αλγορίθμων.

3.2.1 Η Βιβλιοθήκη Numpy και η δομή της εικόνας

Η Numpy αποτελεί τη σημαντικότερη βιβλιοθήκη της γλώσσας προγραμματισμού Python, επιτρέποντας την διαχείριση πολυδιάστατων λιστών και πινάκων, καθώς και διαθέτοντας μία σειρά από μαθηματικά εργαλεία υψηλού επιπέδου για τη διαχείρισή τους. Έχοντας ως πρόγονο της τη βιβλιοθήκη Numeric, η Numpy εισήχθη επίσημα το 2005 από τον Travis Oliphant και έχει εδραιωθεί ως open – source λογισμικό, με πολλούς συντελεστές να συνεισφέρουν στην προσθήκη νέων δυνατοτήτων. Η βασική λειτουργικότητα που μας παρέχει η Numpy στα πλαίσια της παρούσας διπλωματικής εργασίας, πέραν ενός μεγάλου εύρους μαθηματικών εργαλείων, έγκειται στο λεγόμενο ndarray, τη δομή που επιτρέπει τη δημιουργία πολυδιάστατων πινάκων, όπως αναφέραμε νωρίτερα. Ως αποτέλεσμα, γίνεται εφικτός ο χειρισμός binary και gray – scale εικόνων με μορφή δυσδιάστατων πινάκων, καθώς και έγχρωμων εικόνων των τριών ή τεσσάρων καναλιών πληροφορίας ως τρισδιάστατους και τετραδιάστατους πίνακες αντίστοιχα. Οι τελευταίοι είναι συμβατοί με προεκτάσεις άλλων γλωσσών, όταν τα στοιχεία ανήκουν στην ίδια δομή δεδομένων. Την παραπάνω λειτουργικότητα εκμεταλλεύεται το πακέτο SciPy, και το πακέτο αυτό με τη σειρά τους, οι OpenCV⁶ και scikit – image⁷, δίνοντας μεγάλη ευελιξία στη διαχείριση εικόνων. Περισσότερες λεπτομέρειες για τις OpenCV και scikit – image θα δοθούν στην παράγραφο 3.3 και στο Παράρτημα A.

³ [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

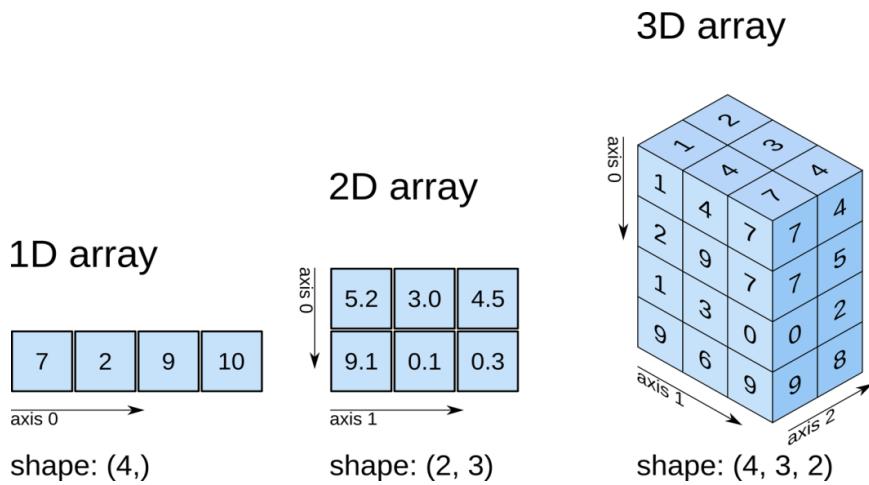
⁴ <http://www.numpy.org/>

⁵ <https://github.com/fjean/pymmeanshift>

⁶ <https://en.wikipedia.org/wiki/OpenCV>

⁷ scikit-image.org/





Εικόνα 3.2.1.1: Χειρισμός πινάκων μέσω της *Numpy*

3.2.2 Αλγόριθμος *MeanShift*

Παρακάτω, θα περιγράψουμε ενδελεχώς τη λειτουργία του αλγορίθμου MeanShift, και θα παραθέσουμε μία επίδειξη της εφαρμογής του. Πρωτίστως, ωστόσο, θα πρέπει να παραθέσουμε μερικούς ορισμούς, ώστε να αντιληφθούμε καλύτερα τη λειτουργία του. Συγκεκριμένα, θα αναφερόμαστε στην μη παραμετροποιημένη στατιστική ανάλυση (non-parametric statistics) ως τον τομέα εκείνο της Στατιστικής, ο οποίος ασχολείται όχι μόνο με οικογένειες πιθανοτικών κατανομών που είναι παραμετροποίησμες (όπως η μέση τιμή ή η διακύμανση), αλλά και με εκείνες των οποίων οι παράμετροι δεν μπορούν να προσδιοριστούν. Ο κλάδος αυτός επίσης ασχολείται με στατιστικά μεγέθη που δεν ακολουθούν κάποια κατανομή. Θα πρέπει να επισημανθεί ότι τα προβλήματα του τομέα αυτού δεν χαρακτηρίζονται από απουσία παραμέτρων, απλώς οι τελευταίες είναι ευέλικτες ως προς τον αριθμό και τη φύση τους, με αποτέλεσμα να μην είναι απαραίτητο να προσδιοριστούν προκαταβολικά. Μερικές μετρικές και τεχνικές που χρησιμοποιούνται σε τέτοιου είδους αναλύσεις είναι το ιστόγραμμα, η μη παραμετροποιημένη παλινδρόμηση (regression), ο αλγόριθμος ταξινόμησης $K - Means$ και τα *Support Vector Machines* (SVMs). Στη συνέχεια, θα ορίσουμε ένα διανυσματικό χώρο με τον όρο feature space, όπου κάθε διάνυσμα (feature vector) περιέχει μία σειρά αριθμητικών χαρακτηριστικών. Τέλος, έχουμε τη γνωστή συνάρτηση πυκνότητας πιθανότητας, η οποία για κάθε μία από τις δυνατές εισόδους επιστρέφει ένα μέτρο της συχνότητας εμφάνισης της εισόδου στο σύνολο των δεδομένων μας. Από την τελευταία, μπορούμε να εντοπίσουμε την επικρατούσα τιμή (mode), η οποία αποτελεί την τιμή των δεδομένων μας με τη μεγαλύτερη συχνότητα εμφάνισης.

Μετά από την παραπάνω εισαγωγή, είμαστε έτοιμοι να ορίσουμε τον αλγόριθμο MeanShift. Έτσι, όταν αναφερόμαστε στον τελευταίο, εννοούμε μία μη παραμετροποιημένη τεχνική ανάλυσης διανυσμάτων χαρακτηριστικών ενός προβλήματος, με στόχο το χωρικό προσδιορισμό του μεγίστου μίας συνάρτησης πυκνότητας (*mode – seeking algorithm*). Ο αλγόριθμος αυτός, που αρχικά παρουσιάστηκε από τους Fukunaga και Hostetler το 1975⁸,

⁸ https://en.wikipedia.org/wiki/Mean_shift



βρίσκει εφαρμογή κυρίως σε προβλήματα επεξεργασίας εικόνας και εντοπισμού ομογενών χρωματικών τμημάτων της τελευταίας. Η διαδικασία που ακολουθείται είναι επαναληπτική και εκκίνει με μία εκτίμηση (έστω \mathbf{x}) για ένα από τα στοιχεία του συνόλου δεδομένων μας. Με χρήση μίας συνάρτησης παραθύρου - πυρήνα (*kernel function*), προσδιορίζουμε το βάρος των γειτόνων του παρόντος στοιχείου στον επανυπολογισμό της νέας τιμής του. Συχνά, χρησιμοποιείται γκαουσιανός πυρήνας με την παρακάτω μορφή:

$$K(x_i - x) = e^{-c\|x_i - x\|^2}$$

όπου η c αποτελεί μία σταθερά. Ως αποτέλεσμα, η σταθμισμένη μέση τιμή της πυκνότητας εντός του παραθύρου υπολογίζεται ως

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x)x_i}{\sum_{x_i \in N(x)} K(x_i - x)}$$

όπου $N(\mathbf{x})$ είναι η γειτονιά γύρω από το σημείο με αρχική εκτίμηση \mathbf{x} , με μη μηδενικά βάρη στον πυρήνα K . Η διαφορά $m(\mathbf{x}) - \mathbf{x}$ αποτελεί τη μετακίνηση (shift) της εκτίμησής μας για το στοιχείο αυτό, στην οποία αναφερόμαστε με τον όρο mean shift. Πλέον, η νέα μας εκτίμηση για το στοιχείο είναι η $m(\mathbf{x})$ και η παραπάνω διαδικασία επαναλαμβάνεται έως τη σύγκλιση του $m(\mathbf{x})$ σε κάποια τιμή. Στην επόμενη εικόνα, βλέπουμε τη διαδικασία που μόλις περιγράψαμε σε μορφή ψευδοκώδικα.

Algorithm 1 MeanShift Algorithm

```

1: procedure MEANSHIFT(matrix, kernel_type, kernel_size, neighbors)
2:   for x in matrix do
3:     m(x) = 0
4:     repeat
5:       for xi in kernel_size do
6:         Calculate weight  $K(x - x_i)$  of xi for the computation of m(x)
7:         based on kernel_type
8:       end for
9:       Calculate m(x) as

```

$$m(x) = \frac{\sum_{x_i \in kernel} K(x - x_i)x_i}{\sum_{x_i \in kernel} K(x - x_i)} \quad (1)$$

```

9:     until x → m(x)
10:    end for
11: end procedure

```

Αλγόριθμος 3.2.2.1: Αλγόριθμος MeanShift

Για την εφαρμογή στο δικό μας πρόβλημα τμηματοποίησης εικόνων με βάση το χρώμα (*color segmentation*), χρησιμοποιήθηκε μία υλοποίηση από τους Chris M. Christoudias και



Bogdan Georgescu. Ο αλγόριθμος των παραπάνω εφαρμόζει τη διαδικασία που μόλις περιγράψαμε, λαμβάνοντας εκτός της εικόνας, την χωρική ακτίνα του παραθύρου του πυρήνα, το χρωματικό εύρος που θα θεωρηθεί ως ένα τμήμα σε μία περιοχή ενδιαφέροντος, καθώς και τον ελάχιστο αριθμό pixels που απαιτούνται για την σύσταση ενός τέτοιου τμήματος. Όσο αφορά τη λειτουργία του, ο πυρήνας αποτελείται από ένα συγκεκριμένο αριθμό υποπεριοχών, με την κάθε μία να συνιστά είτε ένα γκαουσιανό, είτε ομοιόμορφο (σε κάθε pixel αντιστοιχίζεται η ίδια τιμή), είτε κάποιο άλλο καθορισμένο από το χρήστη υποπυρήνα. Στη συνέχεια, ο συνολικός πυρήνας μπορεί να ρυθμιστεί με βάση μία συνάρτηση βαρύτητας, με δυνατότητα ορισμού από το χρήστη, για να εφαρμοστεί γύρω από ένα pixel ενδιαφέροντος. Για τον υπολογισμό του $m(x)$, οι συγγραφείς έχουν επιλέξει να μετασχηματίσουν την εικόνα στο CIELUV colorspace (βλ. Παράρτημα A, Παράγραφος A1), το οποίο, όπως αναφέρεται στο σχετικό παράρτημα, αναπαριστά τις χρωματικές διαφορές με πιο βολικό τρόπο, ενώ παρουσιάζει μεγαλύτερη ευελιξία στην χρωματική αναπαράσταση εικόνων με πολλαπλές πηγές φωτισμού κατά τη λήψη. Η χρήση του συγκεκριμένου colorspace μπορεί επίσης να μεταβληθεί ανάλογα με την εφαρμογή. Η διαδικασία υπολογισμού των $m(x)$ θα συνεχιστεί για όλα τα pixels μέχρι τη σύγκλιση των τιμών. Τέλος, μετά από μερικές διαδικασίες φίλτρων, η εικόνα μετασχηματίζεται στο RGB colorspace και επιστρέφεται από τον αλγόριθμο μαζί με έναν πίνακα διαστάσεων ίδιων με την εικόνα, που προσδιορίζει το χρωματικό label καθενός pixel, καθώς και τον συνολικό αριθμό αυτών των labels. Παρακάτω παραθέτουμε ένα παράδειγμα απλής εφαρμογής του αλγορίθμου.



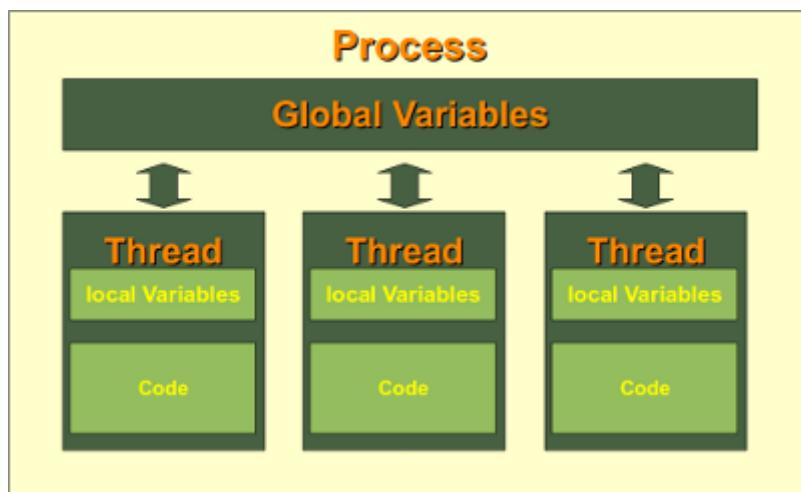
Εικόνα 3.2.2.1: Εφαρμογή του αλγορίθμου MeanShift

3.2.3 Η βιβλιοθήκη *Threading*

Στα πλαίσια της επιθυμίας μας για όσο το δυνατόν ταχύτερη ολοκλήρωση της διαδικασίας διόρθωσης του θορύβου μίας Ε.Π.Α.Μ., χρησιμοποιήθηκε η βιβλιοθήκη threading. Η τελευταία επιτρέπει την παραλληλοποίηση με τη γνωστή λογική των threads, όπου κάθε ένα από τα τελευταία αναλαμβάνει τη διεκπεραίωση μίας διαδικασίας σε ένα τμήμα των δεδομένων, η οποία είναι κοινή για το σύνολο αυτών. Το κάθε thread μπορεί να διατηρεί τοπικές μεταβλητές, αλλά και να ενημερώνει καθολικές, όπως φαίνεται στην παρακάτω εικόνα. Ύστερα από την ολοκλήρωση των απαραίτητων διεργασιών, ένα thread θα αναζητήσει ένα νέο τμήμα των δεδομένων που δεν έχει ακόμη υποστεί επεξεργασία,



έως ότου αυτή ολοκληρωθεί για το σύνολο τους. Η παραπάνω διαδικασία εφαρμόστηκε στα πλαίσια αυτής της διπλωματικής εργασίας, με το κάθε thread να αναλαμβάνει τη διαχείριση μίας E.P.A.M., έως ότου εξαλειφθεί ο θόρυβος από την εικόνα βάθους. Όπως θα γίνει κατανοητό στη συνέχεια, διαφορετική διαχείριση της εικόνας ως σύνολο δεδομένων με στόχο την παραλληλοποίηση δεν καθίσταται εφικτή, λόγω της επίλυσης του προβλήματος με βάση την πληροφορία χρώματος από γειτονικά σε κάθε E.P.A.M. pixels. Περισσότερες λεπτομέρειες για την εφαρμογή της παραλληλοποίησης θα δοθούν σε επόμενο κεφάλαιο.



Εικόνα 3.2.3.1: Λειτουργία των threads

3.3 OpenCV

Η OpenCV αποτελεί μία βιβλιοθήκη εργαλείων, τα οποία στοχεύουν στην αντιμετώπιση προβλημάτων πραγματικού χρόνου της μηχανικής όρασης. Αν και πρωταρχικά γραμμένη σε C++, η OpenCV έχει ενσωματωθεί και σε άλλες γλώσσες προγραμματισμού, όπως η Java και η Python. Όπως είναι αντιληπτό από τη φύση του προβλήματος το οποίο πραγματεύεται η παρούσα διπλωματική εργασία, η βιβλιοθήκη αυτή περιλαμβάνει την πλειοψηφία των εργαλείων τα οποία χρησιμοποιήθηκαν. Για λόγους συνέχειας της ροής της περιγραφής, λόγω του ειδικού ενδιαφέροντος της παρούσας διπλωματικής εργασίας, οι λεπτομέρειες για τα εργαλεία της OpenCV παρατίθενται στο Παράρτημα Α. Εκεί, αφού προχωρήσουμε σε μία ενδελεχή ανάλυση των χρησιμοποιούμενων στον τομέα της επεξεργασίας εικόνας χρωματικών χώρων (colorspaces), θα περιγράψουμε τη λειτουργία εργαλείων ποικίλων κατηγοριών όπως:

1. Εύρεση ιστογραμμάτων και εξισορρόπηση
2. Μορφολογικά φίλτρα (*thresholding, smoothing* και άλλα)
3. Ανίχνευση ακμών
4. Μορφολογία τμημάτων και αντικειμένων εικόνας (*skeletonization, floodfill, watershed*)
5. Διαχείριση και εύρεση χαρακτηριστικών ακμών και περιγραμμάτων



3.4 Robot Operating System (ROS)

Το ROS αποτελεί ένα ρομποτικό μεσολογισμικό, δηλαδή μια συλλογή από δομές ανάπτυξης λογισμικού και διασύνδεσης υλικού για ρομποτικές εφαρμογές. Παρότι δεν είναι ακριβώς λειτουργικό σύστημα, παρέχει υπηρεσίες σχεδιασμένες για ετερογενείς διανεμημένες υπολογιστικές μονάδες, όπως αφαιρετικότητα υλικού, έλεγχο συσκευών σε χαμηλό επίπεδο, υλοποίηση δημοφιλών λειτουργικοτήτων, επικοινωνία μηνυμάτων – δεδομένων μεταξύ διαφορετικών διαδικασιών και διαχείριση και οργάνωση πακέτων λογισμικού. Η εκτέλεση διαφόρων διαδικασιών στο σύστημα αυτό αντιπροσωπεύεται σαν μία αρχιτεκτονική γράφου, όπου οι διάφορες ενεργές μονάδες επεξεργασίας αναπαριστούν κόμβους, οι οποίοι μπορούν να δέχονται, να δημοσιεύουν και να κατανέμουν μηνύματα – προγραμματιστικές δομές δεδομένων, που αναφέρονται σε μετρήσεις από αισθητήρες, δεδομένα ελέγχου, κατάστασης, σχεδιασμού, ενεργοποίησης κλπ. Όλα τα ανεξάρτητα εργαλεία και οι βιβλιοθήκες λογισμικού είναι open-source και διαθέτονται τόσο για εμπορική, όσο και για ερευνητική χρήση. Ο πυρήνας του λογισμικού του ROS βασίζεται στην ύπαρξη πακέτων. Τα πακέτα αυτά υλοποιούν κεντρικές και διαδεδομένες λειτουργικότητες και εφαρμογές όπως drivers υλικού, μοντέλα ρομπότ, τύπους δεδομένων, σχεδιασμό μονοπατιών, ρομποτική αντίληψη, ταυτόχρονη χαρτογράφηση και εντοπισμό θέσης (SLAM), εργαλεία προσομοίωσης και άλλους αλγόριθμους.

Στη συνέχεια παρουσιάζουμε τα βασικά πακέτα ROS τα οποία χρησιμοποιήθηκαν στην ολοκλήρωση του συστήματος αφαίρεσης θορύβου σκιάς.

3.4.1 Η βιβλιοθήκη *rospy*

Η βιβλιοθήκη *rospy*⁹ είναι μια Python client βιβλιοθήκη για διασύνδεση με το ROS. Το API της *rospy* επιτρέπει τον προγραμματισμό κόμβων, μηνυμάτων, παραμέτρων, υπηρεσιών και ολόκληρων πακέτων λογισμικού εντός της διεπαφής του ROS. Η αρχιτεκτονική της βιβλιοθήκης αυτής έχει επιλεγεί με τέτοιο τρόπο, ώστε να ευνοεί την ταχύτητα υλοποίησης, σε σχέση με την απόδοση στην εκτέλεση, προκειμένου αλγόριθμοι να μπορούν να συνθέτονται και να δοκιμάζονται άμεσα. Πολλές από τις βασικότερες λειτουργικότητες του ROS είναι χτισμένες με βάση την *rospy*.

Στο σύστημα, χρησιμοποιούμε τη *rospy* για να προγραμματίσουμε έναν κόμβο, ο οποίος λαμβάνει μηνύματα εικόνων από την κάμερα, υλοποιεί την απαραίτητη επεξεργασία και δημοσιεύει, ως μήνυμα εικόνας, το διορθωμένο χάρτη βάθους.

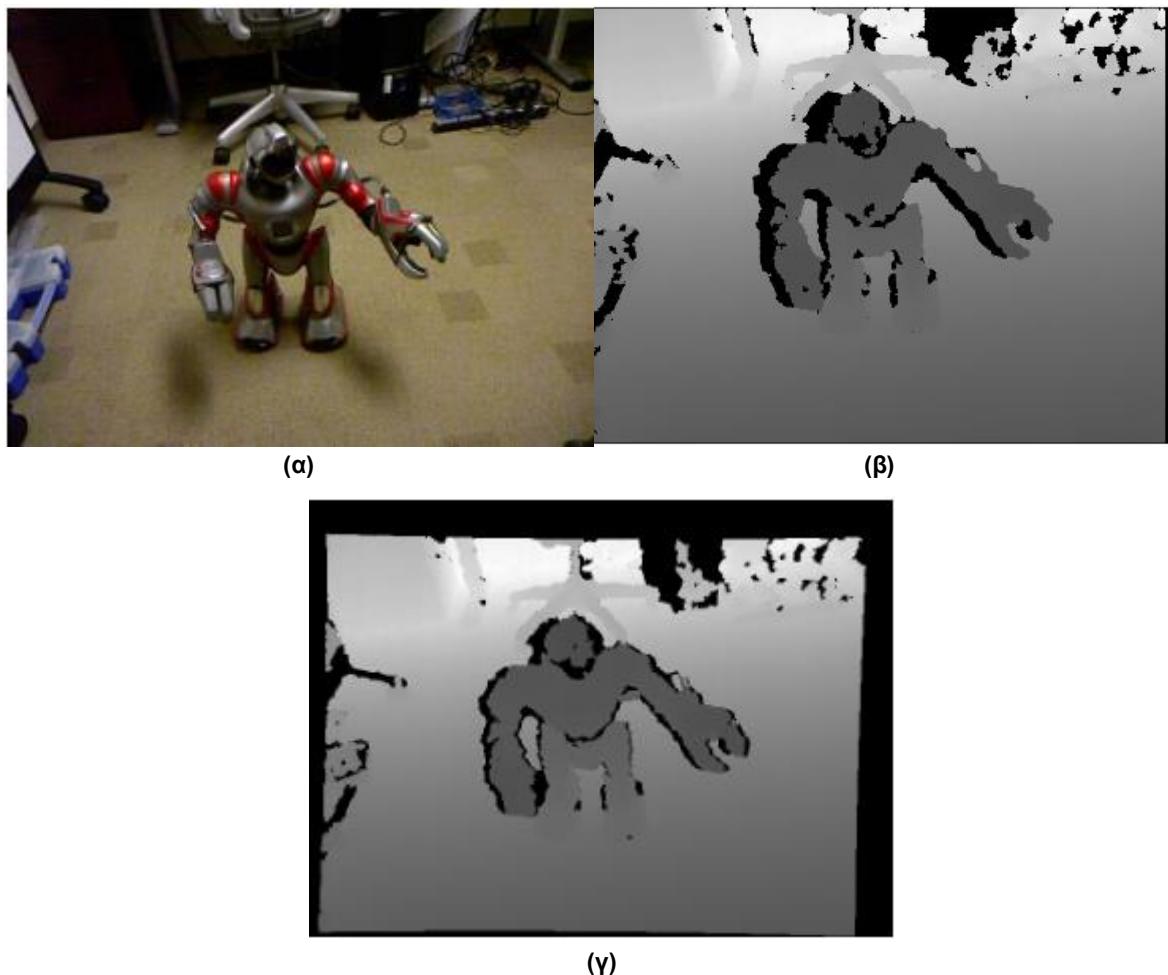
3.4.2 Το OpenNI Framework

Το OpenNI Framework αποτελεί μια open-source προγραμματιστική δομή που επιτρέπει την αλληλεπίδραση μεταξύ χρήστη και ψηφιακών συσκευών. Εντός της δομής, υπάρχουν πολλά πακέτα που υλοποιούν προγράμματα οδήγησης για διάφορες συσκευές, καθώς και αλγορίθμους αλληλεπίδρασης με τον χρήστη, (π.χ. εντοπισμός χειρονομιών ανθρώπινου χεριού).

⁹ wiki.ros.org/rospy



Το πακέτο *openni_launch* κάνει εκκίνηση μιας OpenNI – συμβατής συσκευής και θέτει σε εφαρμογή όλους τους κόμβους λήψης των μηνυμάτων, σε μορφή ιδανική για επεξεργασία και οπτικοποίηση. Αυτό το πακέτο χρησιμοποιούμε για την εκκίνηση της κάμερας Microsoft Kinect μέσα στο ROS. Εκτός από την εκκίνηση της κάμερας, το πακέτο εκτελεί όλες τις απαραίτητες λειτουργίες για την μετατροπή ωμών δεδομένων βάθους, εικόνων IR και RGB σε εικόνες βάθους, εικόνες διαφοράς, 3D RGB σημεία στον χώρο (δομή PointCloud2) κ.α. Με την εκτέλεση του πακέτου, ενεργοποιούμε την παράμετρο *depth_registration*, η οποία χρησιμοποιεί τις ενδογενείς παραμέτρους της κάμερας, για να πραγματοποιηθεί αντιστοίχιση του χάρτη βάθους πάνω στην έγχρωμη εικόνα. Η διαδικασία αυτή ονομάζεται *καταχώρηση* του βάθους στον RGB αισθητήρα της κάμερας. Στην παρακάτω εικόνα, βλέπουμε το αποτέλεσμα της καταχώρησης μιας εικόνας βάθους.



Εικόνα 3.4.2.1: Καταχώρηση βάθους στον RGB αισθητήρα
 (α) Έγχρωμη Εικόνα, (β) Αντίστοιχη Εικόνα βάθους, (γ) Αντίστοιχη Καταχωρημένη Εικόνα Βάθους

Το πακέτο *openni2_launch* χρησιμοποιείται, κατ' αντιστοιχία, για την κάμερα βάθους ASUS Xtion.



3.4.3 Το πακέτο depth_image_proc

Το πακέτο¹⁰ αυτό χρησιμοποιείται για την επεξεργασία και την μετατροπή εικόνων βάθους, όπως αυτές που παρέχονται από μία OpenNI Κάμερα. Οι λειτουργίες του πακέτου συμπεριλαμβάνουν τη σύνθεση εικόνων διαφοράς και PointCloud2, καθώς και την επαναπροβολή μίας εικόνας βάθους ως προς έναν εξωτερικό αισθητήρα.

Δεδομένου ενός ζεύγους καταχωρημένης εικόνας βάθους και αντίστοιχης έγχρωμης εικόνας από μία κάμερα, καθώς και του μηνύματος πληροφοριών της κάμερας (περιγραφή ενδογενών χαρακτηριστικών) που δημοσιεύεται από το πρόγραμμα οδήγησης, το πακέτο αυτό συνθέτει ένα τρισδιάστατο μοντέλο έγχρωμων σημείων του περιβάλλοντος της σκηνής που αποτυπώνεται. Αυτή τη λειτουργικότητα χρησιμοποιούμε εμείς τελικά στα πειράματα για τη σύνθεση PointCloud2 δομών από θορυβώδεις (*raw*) και καθαρές εικόνες βάθους, για να εξετάσουμε την ακρίβεια του αλγορίθμου μας σε πρακτική εφαρμογή. Παρακάτω, ακολουθεί μία εικόνα που επιδεικνύει το 3D μοντέλο που κατασκευάζει το ζεύγος εικόνων της κάμερας.



(α)



(β)



(γ)

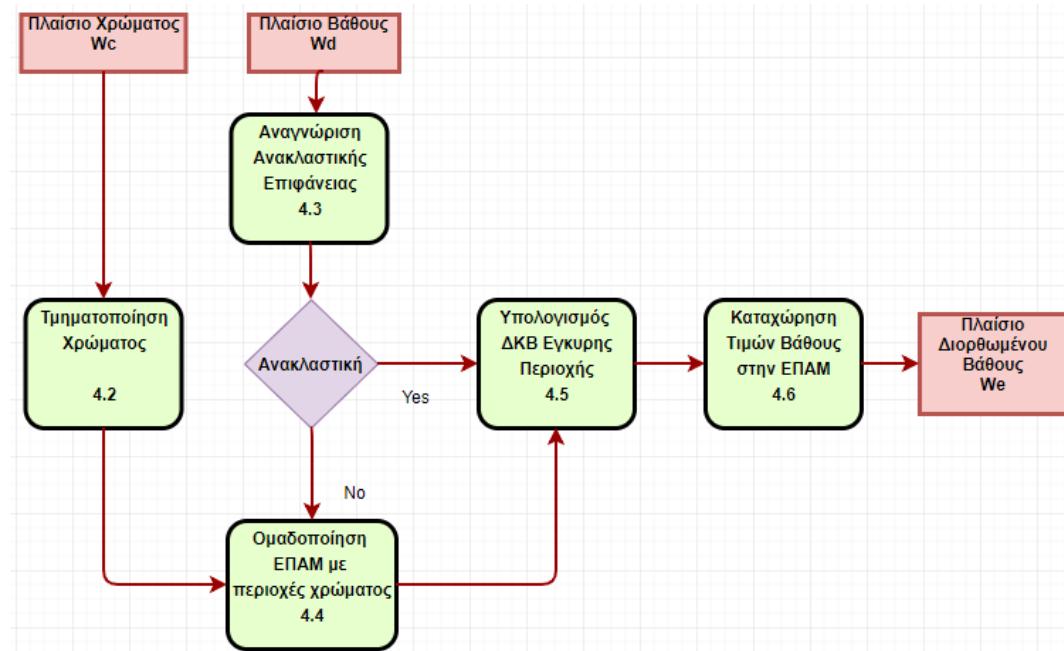
Εικόνα 3.4.4.1: Δημιουργία PointCloud2 από Ζεύγος Εικόνων Χρώματος – Βάθους
(α) Έγχρωμη Εικόνα, (β) Εικόνα Βάθους, (γ) PointCloud2

¹⁰ wiki.ros.org/depth_image_proc



4. Υλοποίηση Αλγορίθμου

Όπως έχουμε αναφέρει στην παράγραφο 1.2, αρχικός σκοπός της εργασίας μας αποτελεί ο σχεδιασμός ενός αλγορίθμου, ο οποίος, επεξεργαζόμενος τα RGBD δεδομένα της κάμερας, επιχειρεί να συμπληρώσει με έγκυρες τιμές κάθε διακριτή περιοχή θορύβου που εντοπίζεται στην εικόνα βάθους, όπως αυτή παρέχεται από τον αισθητήρα της κάμερας. Προκειμένου να αποσαφηνιστούν οι τεχνικές επεξεργασίας του αλγορίθμου μας, αρχικά, αναφερόμαστε στα βασικά χαρακτηριστικά της δομής μιας τέτοιας περιοχής, στην οποία αναφερόμαστε ως *Ενιαία Περιοχή Άκυρης Μέτρησης (ΕΠΑΜ)* και στη συνέχεια παρουσιάζουμε τα βασικά στάδια της διαδικασίας καταχώρησης τιμών βάθους σε αυτές αναλυτικά. Το παρακάτω διάγραμμα ροής περιγράφει αφορημένα τη διαδικασία που υλοποιεί ο αλγόριθμος.



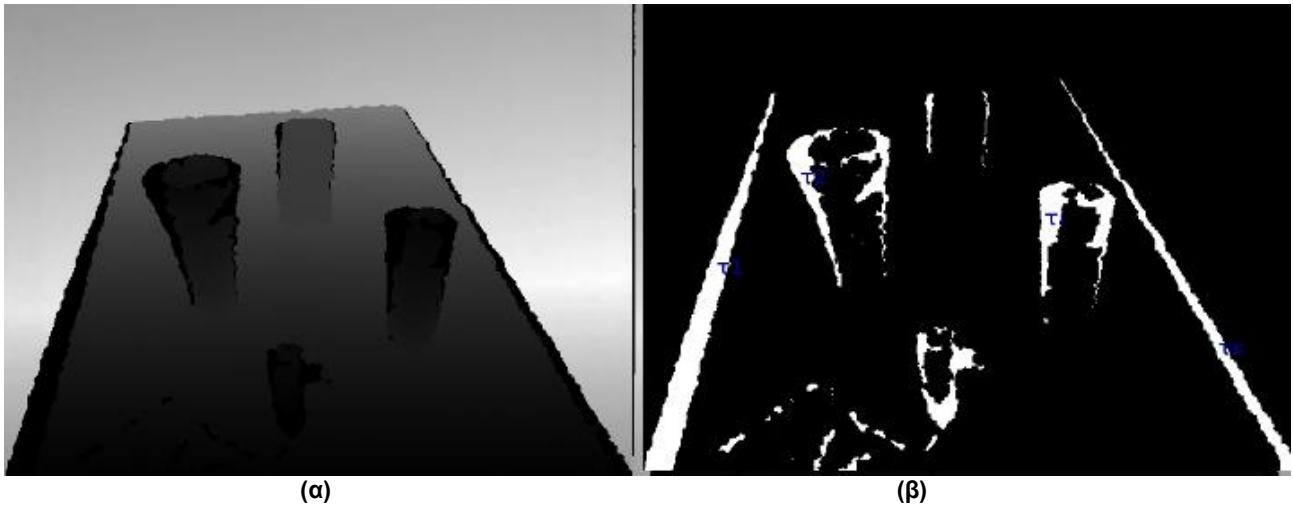
Εικόνα 4.0.1: Διάγραμμα Ροής Πληροφορίας Αλγορίθμου Διόρθωσης ΕΠΑΜ εντός Πλαισίου

4.1 Χαρακτηριστικά μιας ΕΠΑΜ

Αρχικά, από την εικόνα βάθους μπορούμε εύκολα να απομονώσουμε όλες τις περιοχές μη ύπαρξης μέτρησης σε μια μάσκα, όπως βλέπουμε στην εικόνα 4.1.1. Ορίζουμε λοιπόν το σύνολο T τέτοιων ενιαίων διακριτών περιοχών θορύβου πάνω στην εικόνα βάθους:

$$T = \{\tau_i \doteq (x, y) / Depth(x, y) = 0, i = 1, \dots, \kappa\}, \tau_i \cap \tau_j = \emptyset, \forall i \neq j$$





Εικόνα 4.1.1: Εξαγωγή ΕΠΑΜ από Εικόνα Βάθους
 (a) Εικόνα Βάθους. (β) Μάσκα Θορύβου

Για να παράξουμε την παραπάνω μάσκα, αρκεί να εφαρμόσουμε ένα φίλτρο που αποτυπώνει σε μια μαύρη εικόνα μόνο εκείνα τα pixels του βάθους που έχουν τιμή 0 (μη ύπαρξη μέτρησης). Ως αποτέλεσμα, κάθε ΕΠΑΜ ορίζεται ως εκείνο το υποσύνολο των σημείων της εικόνας βάθους που έχουν μηδενικό βάθος και είναι γειτονικά μεταξύ τους.

4.1.1 Περιγράμματα

Εφαρμόζοντας λοιπόν εντοπισμό εξωτερικών περιγραμμάτων επί της μάσκας θορύβου, μπορούμε να δημιουργήσουμε μια ιεραρχημένη λίστα με όλες τις τρύπες που υπάρχουν στην εικόνα ως περιγράμματα, όπου κάθε περίγραμμα ορίζεται ως το σύνολο των συντεταγμένων των σημείων που ανήκουν πάνω στο σύνορο της ΕΠΑΜ. Δηλαδή:

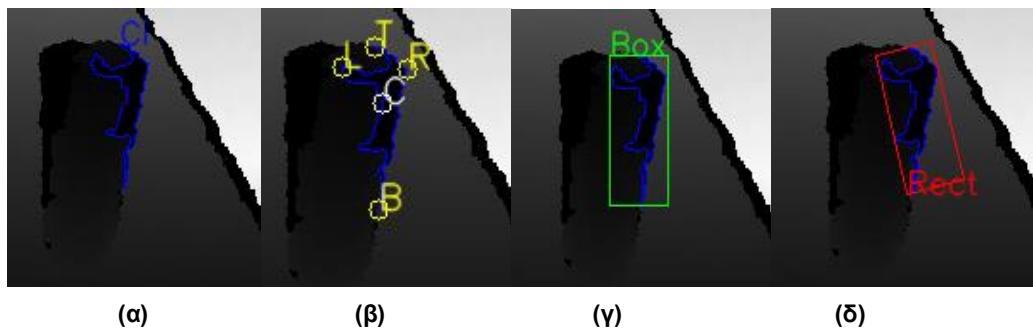
$$Contours = \{c_i \doteq (x, y) \in boundary(\tau_i), \forall \tau_i \in T, i = 1, \dots, \kappa\}$$

Εκτός από την θέση των σημείων στην εικόνα, με τη δομή του περιγράμματος έχουμε εύκολα πρόσβαση σε μια σειρά από γεωμετρικά χαρακτηριστικά της ΕΠΑΜ, τα οποία χρησιμοποιούμε τακτικά, μεταξύ άλλων:

- *T,L,R,B Ακραία Σημεία και Κέντρο Μάζας C* της ΕΠΑΜ
- Ευθύ Ορθογώνιο Παραλληλόγραμμο *Box* που εσωκλείει την ΕΠΑΜ χωρίς να λαμβάνει υπ' όψιν την περιστροφή της
- Στραμμένο Παραλληλόγραμμο ελαχίστου εμβαδού *Rect* που περιβάλλει την ΕΠΑΜ

Στην παρακάτω εικόνα βλέπουμε παραδείγματα τέτοιων χαρακτηριστικών πάνω σε ένα επεκταμένο πλαίσιο ΕΠΑΜ τι.





Εικόνα 4.1.1.1: Βασική Επεξεργασία ΕΠΑΜ

(α) Περιγραμμα Τρύπας, (β) Ακραία Σημεία και Κέντρο Μάζας Τρύπας, (γ) Ευθύ Ορθογώνιο Παρ/μο Τρύπας, (δ) Στραμμένο Παρ/μο Ελαχίστου Εμβαδού Τρύπας

Εκτός από αυτά τα χαρακτηριστικά, άλλα γεωμετρικά μεγέθη της ΕΠΑΜ όπως περίμετρος, εμβαδόν, αριθμός σημείων κ.α. είναι διαθέσιμα με τη γνώση του περιγράμματος.

4.1.2 Μάσκες

Εκτός από τη γνώση των συντεταγμένων κάθε ΕΠΑΜ εντός των πλαισίων, πολλές φορές μας είναι ιδιαίτερα χρήσιμη η αποτύπωση συγκεκριμένων χαρακτηριστικών της σε μία νέα εικόνα στη διάσταση του επεκταμένου πλαισίου, την οποία ονομάζουμε μάσκα. Η μάσκα αυτή να μπορεί να χρησιμοποιηθεί ως φίλτρο για την επεξεργασία συγκεκριμένων περιοχών μιας εικόνας πάνω σε μία άλλη εικόνα.

Για παράδειγμα, στο παρακάτω σενάριο, για ένα ζεύγος πλαισίων βάθους και χρώματος (depth, color), έχουμε αποτυπώσει με διαφορετικές τιμές σε μία μάσκα το εσωτερικό (άσπρο), το Box (πράσινο) και το Rect (κόκκινο) της ΕΠΑΜ. Όλες αυτές τις πληροφορίες τις έχουμε εξάγει από το πλαίσιο βάθους με την χρήση περιγραμμάτων. Με την χρήση της μάσκας επεξεργαζόμαστε επιθυμητές περιοχές της πάνω στο πλαίσιο χρώματος. Βλέπουμε ενδεικτικά το αποτέλεσμα του σεναρίου 4.1 για ένα γνωστό ζεύγος πλαισίων. Χαριστικά, ονομάζουμε την ΕΠΑΜ με τον όρο *hole*.



Scenario 1: Applying masks for known depth and color frames.

Step 1: Defining as *hole* the regions where *depth* = 0:

$hole = \text{where}(\text{depth} == 0)$

Step 2: Finding *Box* and *Rect* of *hole*.

Step 3: Drawing the above plus the *hole* in a mask:

$mask[\text{in Box}(hole)] = \text{Green}$

$mask[\text{in Rect}(hole)] = \text{Red}$

$mask[\text{in hole}] = \text{White}$

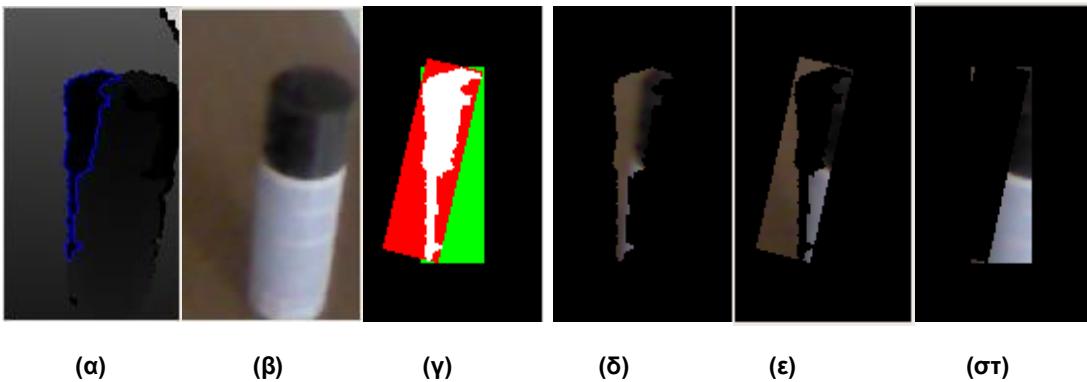
Step 4: Drawing the *color* segments of the *hole*, *Rect(hole)* and *Box(hole)*:

$hole_color_image = \text{color}[mask == \text{White}]$

$rect_color_image = \text{color}[mask == \text{Red}]$

$box_color_image = \text{color}[mask == \text{Green}]$

Σενάριο 4.1.2.1: Επίδειξη εφαρμογής μασκών



Εικόνα 4.1.2.1: Εικόνες Σεναρίου 4.1.2.1

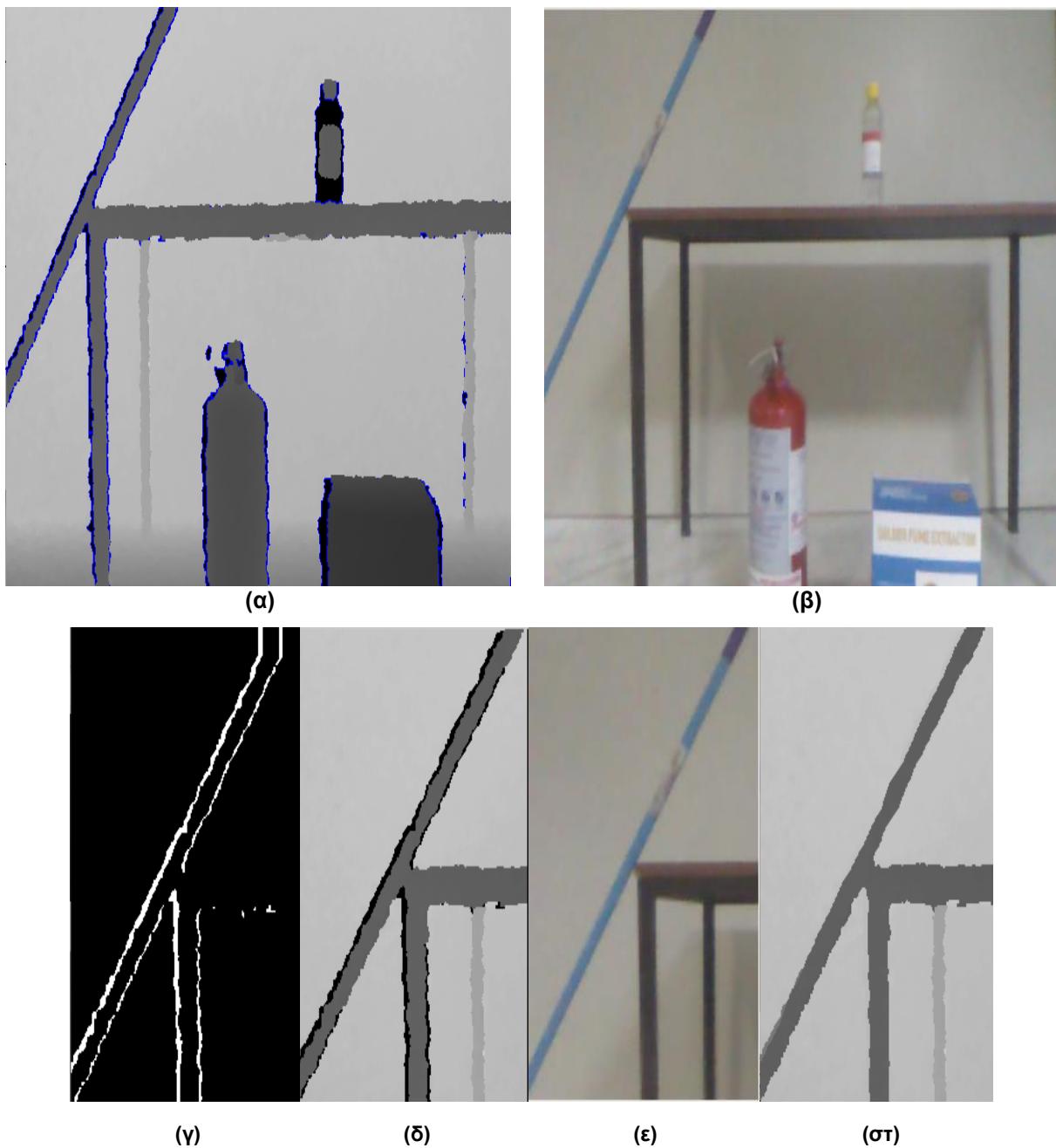
(α) Πλαίσιο θορύβου (*depth*), (β) Πλαίσιο χρώματος (*color*), (γ) Μάσκα (*mask*), (δ) Προβολή Τρύπας στο Χρώμα (*hole_color_image*), (ε) Προβολή *Rect* στο Χρώμα (*rect_color_image*), (στ) Προβολή *Box-Rect* στο Χρώμα (*box_color_image*)

4.1.3 Πλαίσια

Γνωρίζοντας τη θέση τη κάθε *ΕΠΑΜ* στην εικόνα με τη μορφή περιγράμματος $L(\tau)$, μπορούμε, όπως περιγράφουμε στο κεφάλαιο 3, να εξάγουμε το ορθογώνιο παραλληλόγραμμο *Box*(τ), να το επεκτείνουμε κατά μια επιλεγμένη κλίμακα και να αποκόψουμε από την εικόνα ένα $w \times h$ πλαίσιο W με κέντρο συμμετρίας το κέντρο μάζας της *ΕΠΑΜ*. Έτσι, για κάθε $i = 1, \dots, K$, ορίζουμε το πλαίσιο W μιας εικόνας *img* ως προς μια *ΕΠΑΜ* τ :

$$W_{img}(x, y) \doteq img(x + x_0, y + y_0), \quad \forall (x, y) \in \text{ExpandedBox}(\tau), \quad 0 \leq x \leq w, \quad 0 \leq y \leq h$$





Εικόνα 4.1.3.1: Μάσκα και αντίστοιχα Πλαίσια Χρώματος και Βάθους

(α) Εικόνα Βάθους με σημειωμένα τα περιγράμματα των ΕΠΑΜ, (β) Αντίστοιχη Έγχρωμη Εικόνα, (γ) Πλαίσιο γύρω από ΕΠΑΜ στη Μάσκα Θορύβου (W_h), (δ) Αντίστοιχο Πλαίσιο στο Βάθος (W_d), (ε) Αντίστοιχο Πλαίσιο στο Χρώμα (W_c), (στ) Αντίστοιχο Πλαίσιο με διορθωμένη την ΕΠΑΜ (W_e)

όπου (x_0, y_0) το πάνω αριστερά σημείο του επεκταμένου ορθογωνίου (με τις μικρότερες συντεταγμένες). Ουσιαστικά, το πλαίσιο μιας εικόνας *img* ως προς μια *ΕΠΑΜ t*, μετασχηματίζει μαθηματικά τις συντεταγμένες όλων των σημείων εντός του ορθογωνίου από το επίπεδο των αρχικών διαστάσεων 640×480 της εικόνας, στο ορθογώνιο $w \times h$ με σημείο αναφοράς $(0,0)$ το σημείο (x_0, y_0) του επεκταμένου ορθογωνίου της *ΕΠΑΜ*.

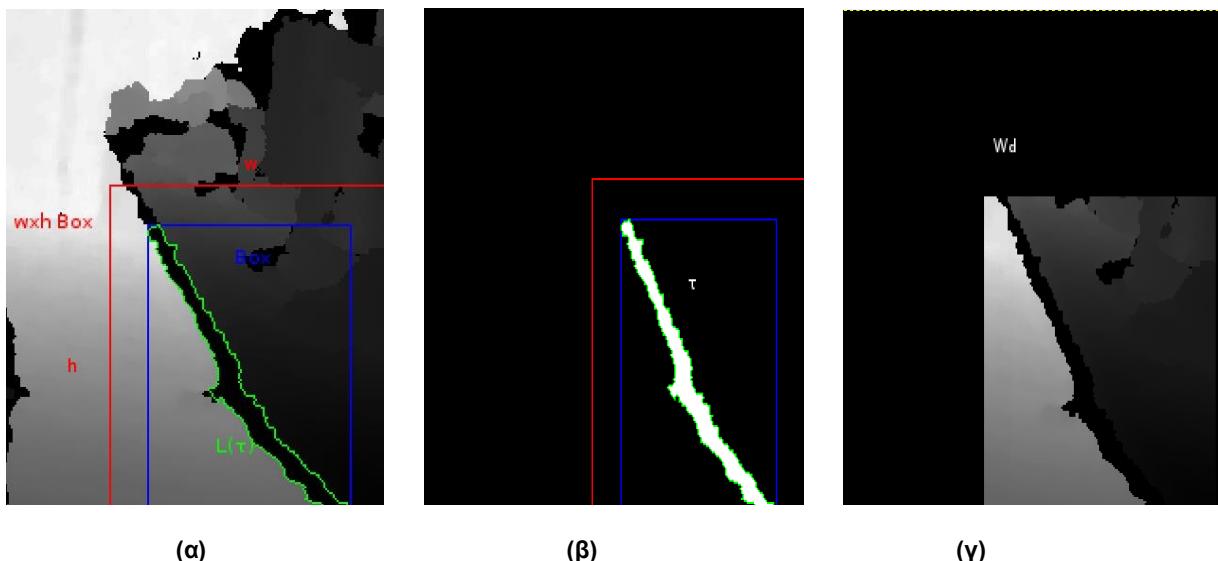


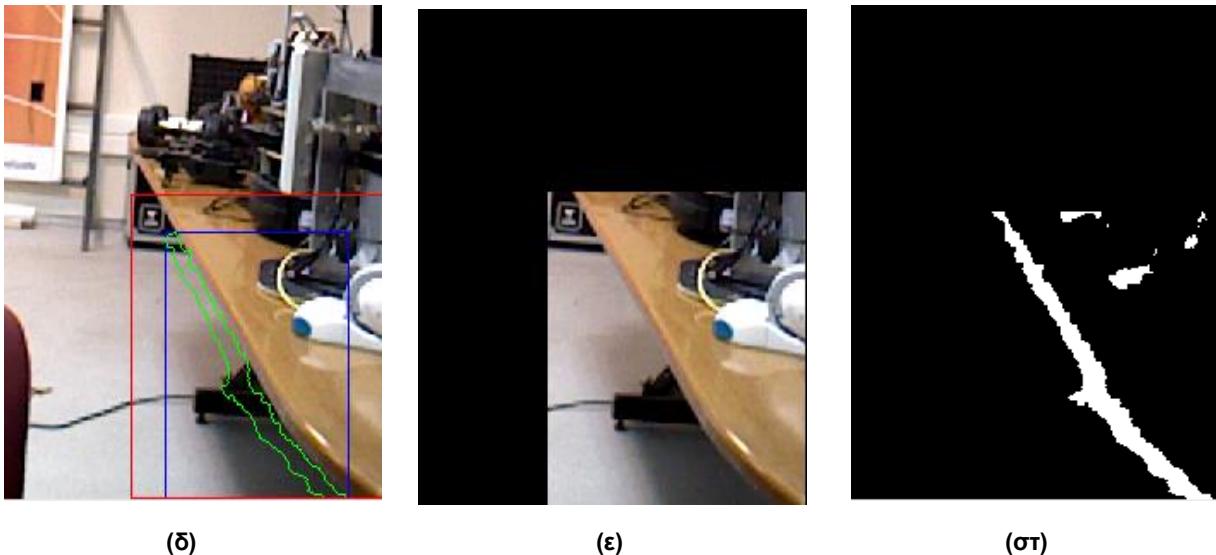
Ένα τέτοιο πλαίσιο στην εικόνα βάθους, το οποίο ορίζουμε ως $w \times h$ *topo* Πλαίσιο Βάθους W_d , καθώς και το αντίστοιχο $w \times h$ *rgb* πλαίσιο στην έγχρωμη εικόνα (*Πλαίσιο Χρώματος* W_c), θεωρούμε ότι αποτελούν είσοδο στον αλγόριθμό μας. Η επιλογή της κλίμακας επέκτασης του ορθογώνιου πλαισίου που συμπεριλαμβάνει την *ΕΠΑΜ*, γίνεται έτσι ώστε να συμπεριλαμβάνεται κατάλληλη γειτονική πληροφορία στο Πλαίσιο Χρώματος για να γίνει σωστά στη συνέχεια η τρηματοποίηση (βλ. Παράγραφο 4.2).

Στην εικόνα 4.1.3.1, βλέπουμε ένα παράδειγμα Πλαισίων Βάθους W_d , Πλαισίου Χρώματος W_c και ένα $w \times h$ *binary* Πλαίσιο Θορύβου W_t με διαστάσεις $w=198$, $h=265$, καθώς και τις 640×480 εικόνες βάθους και χρώματος από τις οποίες αποκόπηκαν, ενώ στην 4.1.3.2 βλέπουμε τη δημιουργία ενός πλαισίου W από μια περιοχή τ .

Σημειώνουμε ότι σε αυτό το σημείο θεωρούμε ότι τα Πλαίσια Βάθους-Χρώματος είναι απολύτως ευθυγραμμισμένα μεταξύ τους, με την έννοια ότι κάθε pixel εντός του Πλαισίου Βάθους είναι εκφρασμένο ως προς το αντίστοιχο pixel της έγχρωμης εικόνας, σαν η εικόνα βάθους να έχει εκφραστεί στο σύστημα αναφοράς του αισθητήρα χρώματος. Στην πραγματικότητα, αυτή η ευθυγράμμιση δεν μπορεί να είναι τέλεια, αλλά μπορεί να είναι ικανοποιητική με την χρήση ενδογενών χαρακτηριστικών που έχουν προκύψει από το καλιμπράρισμα των αισθητήρων της κάμερας και απαραίτητη διόρθωση (βλ. Παράγραφο 5.2.1).

Έξοδο του αλγορίθμου αποτελεί ένα $w \times h$ *topo* Πλαίσιο Βάθους, με συμπληρωμένη πληροφορία σε κάθε σημείο του εσωτερικού της *ΕΠΑΜ* (W_e). Η τελική εικόνα θα συντεθεί από τις κ διαδοχικές εφαρμογές του αλγορίθμου για κάθε *ΕΠΑΜ* που θα εντοπιστεί στην εικόνα βάθους, αντιγράφοντας στο εσωτερικό της τη διορθωμένη πληροφορία κάθε περιοχής τ εντός του αντίστοιχου πλαισίου W_e .





Εικόνα 4.1.3.2: Πλαίσια Βάθους, Θορύβου και Χρώματος μιας ΕΠΑΜ

(α) Περίγραμμα L , ορθογώνιο Box και επεκταμένο ορθογώνιο μιας ΕΠΑΜ, (β) Αντίστοιχα στοιχεία στη μάσκα της ΕΠΑΜ τ, (γ) Το Πλαίσιο Βάθους Wd που εξάγεται από το Box, (δ) Αντίστοιχα στοιχεία στην έγχρωμη εικόνα, (ε) Το Πλαίσιο Χρώματος Wc που εξάγεται από το Box, (στ) Το Πλαίσιο Θορύβου Wn που εξάγεται από το Box

Παρατηρώντας το διάγραμμα ροής της 4.0.1, μπορούμε να ξεχωρίσουμε τις εξής κεντρικές διενέργειες που πραγματοποιεί ο αλγόριθμος:

1. **Τμηματοποίηση Χρώματος:** Εφαρμόζοντας κατάλληλη επεξεργασία πάνω στο Πλαίσιο Χρώματος και επιλέγοντας τις παραμέτρους του αλγορίθμου MeanShift, πραγματοποιούμε τμηματοποίηση χρώματος πάνω στο πλαίσιο. Καταλήγουμε σε ένα ανανεωμένο Πλαίσιο Χρώματος Wc' , οι RGB τιμές κάθε σημείου του οποίου είναι ίδιες για κάθε χρωματικό τμήμα, καθώς και σε ένα πλαίσιο τμημάτων Ws , το οποίο για κάθε σημείο περιέχει απλά τον αύξοντα αριθμό του τμήματος που το σημείο αυτό ανήκει.
2. **Αναγνώριση Ανακλαστικής Επιφάνειας:** Αφού πραγματοποιήσουμε την επεξεργασία στο Πλαίσιο Χρώματος και, προτού εφαρμόσουμε την τμηματοποίηση, ελέγχουμε τη μορφολογία μίας ΕΠΑΜ, προχωρούμε σε μία σειρά μορφολογικών διαδικασιών για την εύρεση ανακλαστικών και απορροφητικών επιφανειών, μέσω της απομόνωσης μονοχρωματικών επιφανειών εντός του Πλαισίου.
3. **Ομαδοποίηση περιοχών ΕΠΑΜ με περιοχές χρώματος:** Δεδομένου ότι η ΕΠΑΜ του πλαισίου μας δεν αντιστοιχεί σε θόρυβο λόγω ανάκλασης, δεν μπορούμε να χρησιμοποιήσουμε την πληροφορία βάθους του πλαισίου για να συμπληρώσουμε όλη την περιοχή, καθώς αυτή μπορεί να καλύπτει διαφορετικά επίπεδα βάθους διαφορετικών αντικειμένων της σκηνής.

Έτσι, προβάλλουμε τα τμήματα χρώματος Ws πάνω στο Πλαίσιο Βάθους Wd και διακρίνουμε ποιες υποπεριοχές της ΕΠΑΜ αντιστοιχούν σε κάθε τμήμα. Σε κάθε τέτοια υποπεριοχή Ns , αντιστοιχίζουμε μια περιοχή έγκυρου βάθους Vs του πλαισίου (εκτός ΕΠΑΜ), η οποία ανήκει στο ίδιο τμήμα χρώματος s , φροντίζοντας να διασπούμε υποπεριοχές που περιλαμβάνουν απομονωμένα κομμάτια.



Καταλήγουμε με οιμάδες $Ps = (Ns, Vs)$ περιοχών θορύβου και τις αντίστοιχες περιοχές ίδιου χρωματικού τμήματος s και έγκυρης πληροφορίας βάθους, από τις οποίες οφείλουμε να εξάγουμε τις διορθωμένες τιμές υπολογίζοντας το *Διάνυσμα Κατεύθυνσης Βάθους (ΔΚΒ)* όπως θα δούμε στη συνέχεια του κεφαλαίου.

4. **Υπολογισμός ΔΚΒ Έγκυρης Περιοχής:** Με γνώμονα τη γεωμετρία της περιοχής έγκυρου βάθους V της κάθε οιμάδας P του προηγούμενου βήματος, ή ειδικά στην περίπτωση ανακλαστικής επιφάνειας, υπολογίζουμε το *Διάνυσμα Κατεύθυνσης Βάθους (ΔΚΒ)*, το μέτρο dz του οποίου αντιστοιχεί στο βήμα αλλαγής βάθους διαδοχικών σημείων της περιοχής και η γωνία θ του εκπροσωπεί την κατεύθυνση μεταβολής του βάθους κατά μήκος της έγκυρης περιοχής V .
5. **Καταχώρηση Τιμών Βάθους στην ΕΠΑΜ:** Η διεργασία αυτή γεμίζει κάθε οιμαδοποιημένη περιοχή θορύβου με πληροφορία βάθους, που εξάγεται από το ΔΚΒ της έγκυρης περιοχής. Η κατεύθυνση του διανύσματος διασπάται στις ορθοκανονικές συνιστώσες του συστήματος και, για κάθε κατεύθυνση, δημιουργείται ένας *Χάρτης Βάθους (XB)*, ο οποίος εμπεριέχει την τιμή με την οποία πρέπει να βαφτεί κάθε σημείο του πλαισίου, με βάση την απόσταση του σημείου από το ολικό ελάχιστο βάθος της έγκυρης περιοχής και το μέτρο του διανύσματος. Οι δύο χάρτες συνθέτονται διανυσματικά και οι τιμές της σύνθεσης του αντικαθιστώνται στο εσωτερικό της *ΕΠΑΜ* στο Πλαίσιο Βάθους. Η ολοκλήρωση του βήματος αυτού για κάθε περιοχή θορύβου N κάθε οιμάδας P της *ΕΠΑΜ* γεμίζει με πληροφορία όλο το εσωτερικό της, παρέχοντας ως έξοδο το Πλαίσιο Διορθωμένου Βάθους.

Ο παρακάτω ψευδοκώδικας περιγράφει συνοπτικά τη σύμπραξη των παραπάνω διεργασιών για την κατασκευή του Πλαισίου Διορθωμένου Βάθους *We*.



Algorithm 2 Depth Image Denoising

```
1: procedure DEPTHDENOISING(color-frame, depth-frame)
2:   Assigning color-frame as  $W_c$ 
3:   Assigning depth-frame as  $W_d$ 
4:   Assigning clean_depth_frame as  $W_e$ 
5:   Copying current  $W_d$  to  $W_e$ 
6:   if contour has regular shape then
7:     Marking regions in whith 0 depth as holes:

$$holes = \text{where}(W_d == 0)$$

8:     Marking valid depth region as valid_depth_regions:

$$valid\_depth\_regions = \text{where}(W_d > 0)$$

9:     Checking for reflective surfaces
10:    if hole is reflective then
11:      Calculating grad_vector using valid_depth_regions
12:      Estimating new depth values using valid_depth_regions and
        grad_vector as depth_map
13:      Assigning depth_map to  $W_e$ 
14:      return  $W_e$ 
15:    end if
16:    else
17:      Applying segmentation to  $W_c$ 
18:      Extracting discrete color regions as color_regions
19:      for each of the color_regions marked as regioni do
20:        Defining mask of relevant color regions as:

$$region\_mask = \text{where}(region_i)$$

21:        Extracting regions of valid depth values as:

$$valid\_depth\_regions = \text{where}(W_d[region\_mask] > 0)$$

22:        Defining the regions for which the depth need to be computed as:

$$zero\_depth\_regions = \text{where}(W_d[region\_mask] == 0)$$

23:        Calculating grad_vector using valid_depth_regions
24:        Estimating new depth values using valid_depth_regions and
        grad_vector as depth_map_regioni
25:        Assigning depth_map_regioni to depth_map to  $W_e$ 
26:      end for
27:      Assigning depth_map to  $W_e$ 
28:    end if
29:    return  $W_e$ 
30: end procedure
```

Αλγόριθμος 4.1.3.1: Αλγόριθμος Διόρθωσης ΕΠΑΜ εντός πλαισίου



4.2 Τμηματοποίηση Χρώματος

Στην παράγραφο αυτή, θα περιγράψουμε τη διαδικασία διεξαγωγής της τμηματοποίησης χρώματος, όπως αυτή διαμορφώθηκε μετά από πειραματισμούς με ποικιλία colorspaces και δοκιμές διαφορετικών παραμέτρων των χρησιμοποιούμενων διαδικασιών. Όπως αναφέραμε και νωρίτερα, για τη τμηματοποίηση χρησιμοποιήθηκε ο αλγόριθμος MeanShift, ο οποίος περιγράφηκε αναλυτικά στο προηγούμενο κεφάλαιο. Χρησιμοποιώντας, λοιπόν, ένα Πλαίσιο Χρώματος, λαμβάνουμε ένα νέο ίδιων διαστάσεων W_c' , όπου κάθε περιοχή κοινού χρώματος συνιστά ένα χρωματικό label. Προτού, ωστόσο, εφαρμόσουμε τον MeanShift, προηγείται ένα στάδιο επεξεργασίας του Πλαισίου Χρώματος, το οποίο θα περιγράψουμε παρακάτω.

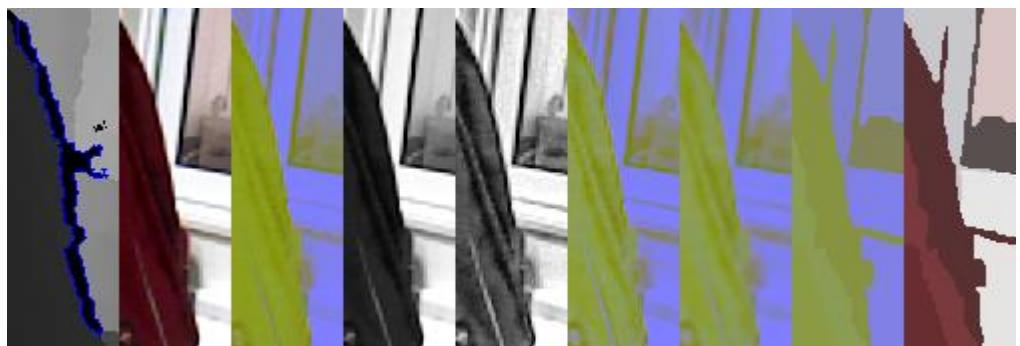
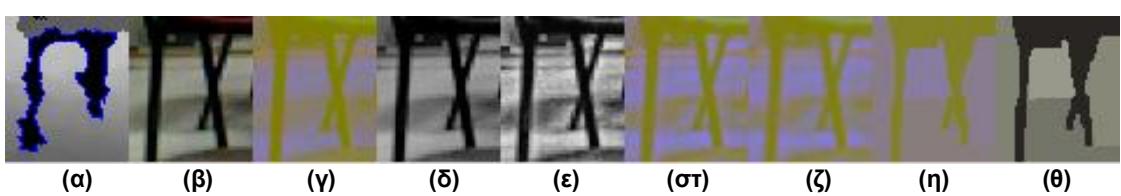
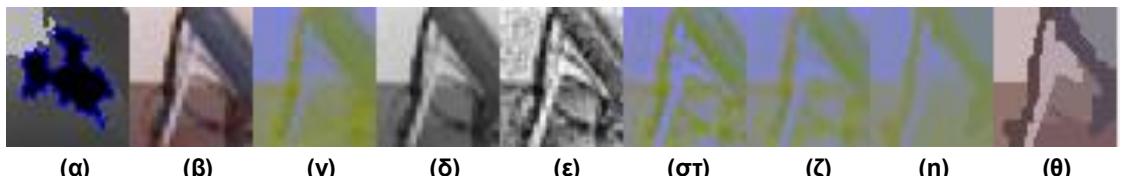
Αρχικά, ελέγχουμε το μέγεθος της υπό επεξεργασία $E\Gamma\Lambda M$, ώστε στην περίπτωση που αυτή έχει πολύ μικρή έκταση να βαφεί αποκλειστικά με το μέσο όρο των έγκυρων τιμών βάθους εντός του πλαισίου. Με τον τρόπο αυτό, πετυχαίνουμε ένα ικανοποιητικά ακριβές αποτέλεσμα, μειώνοντας σημαντικά το χρόνο εκτέλεσης, καθώς, λόγω της αρχιτεκτονικής των καμερών βάθους, τέτοιες περιπτώσεις $E\Gamma\Lambda M$ εμφανίζονται σε πληθώρα. Στη συνέχεια, πραγματοποιώντας πολυωνυμική παλινδρόμηση (*polynomial regression*), προσεγγίζουμε, με βάση το μέγεθος του Ενιαίου Ορθογωνίου Παραλληλογράμμου της $E\Gamma\Lambda M$, τον ελάχιστο αριθμό pixels που απαιτούνται για τη σύσταση ενός χρωματικού segment, παράμετρος η οποία απαιτείται για την εκτέλεση του MeanShift και καθορίζει σημαντικά την πληροφορία που μας παρέχει το νέο πλαίσιο W_c' . Κατόπιν, το Πλαίσιο μετασχηματίζεται από το RGB στο *LAB* colorspace (βλ. Παράρτημα A, Παράγραφος A1), ενώ το πρώτο κανάλι (*lightness* - φωτεινότητα) υφίσταται εξισορρόπηση ιστογράμματος με το αντικείμενο *clahe* (βλ. Παράρτημα A, Παράγραφος A2), ώστε να αυξηθεί η αντίθεση (*contrast*) με κατάλληλο τρόπο τόσο στο γενικό πλαίσιο, όσο και τοπικά εντός του. Για να ενισχύσουμε όσο το δυνατόν περισσότερο τις ακμές που οριοθετούν διαφορετικά τμήματα, εφαρμόζουμε *bilateral* φίλτρο, όπως ορίζεται στο Παράρτημα A, με παραμέτρους που εξαρτώνται από το μέγεθος των $E\Gamma\Lambda M$. Τέλος, ο αλγόριθμος MeanShift, έχοντας ως σταθερές για όλες τις $E\Gamma\Lambda M$ τη χωρική ακτίνα του παραθύρου και το χρωματικό εύρος για ένα label, μας αποδίδει το ζητούμενο πλαίσιο W_c' .

Algorithm 3 Segmentation Preprocess

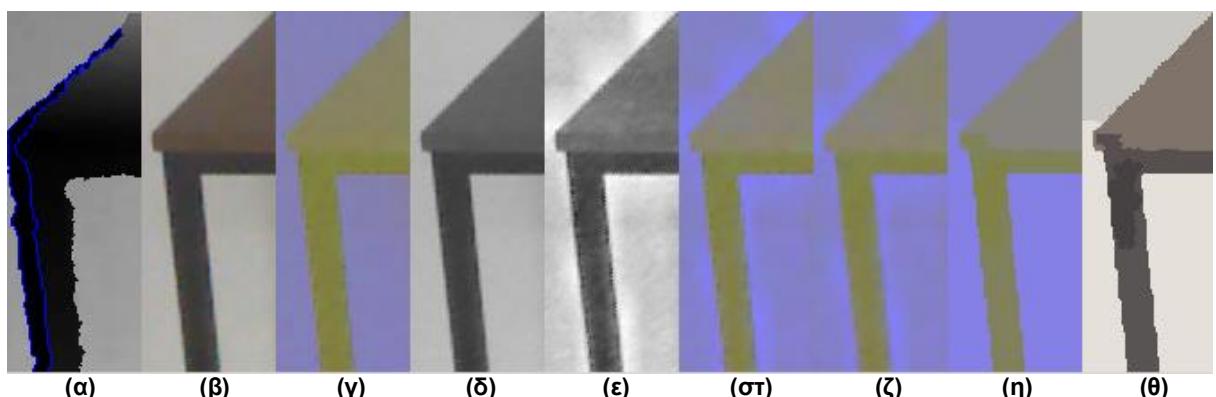
```
1: procedure SEGMPREPROCESS(contour, contour_rgb_frame, min_area)
2:   if contour_area < min_area then
3:     Fill the pixels inside the contour with the average of its neighbors
4:   end procedure
5: end if
6: Apply polynomial regression to define min_den
7: Convert contour_rgb_frame from RGB to LAB colorspace
8: Perform clahe equalization to L channel
9: Apply bilateral filter to the new LAB frame based on the contour_area
10: Execute MeanShift algorithm on the LAB frame using min_den for
    non reflective surfaces
11: end procedure
```

Αλγόριθμος 4.2.1: Αλγόριθμος επεξεργασίας Πλαισίου Χρώματος και τμηματοποίησης

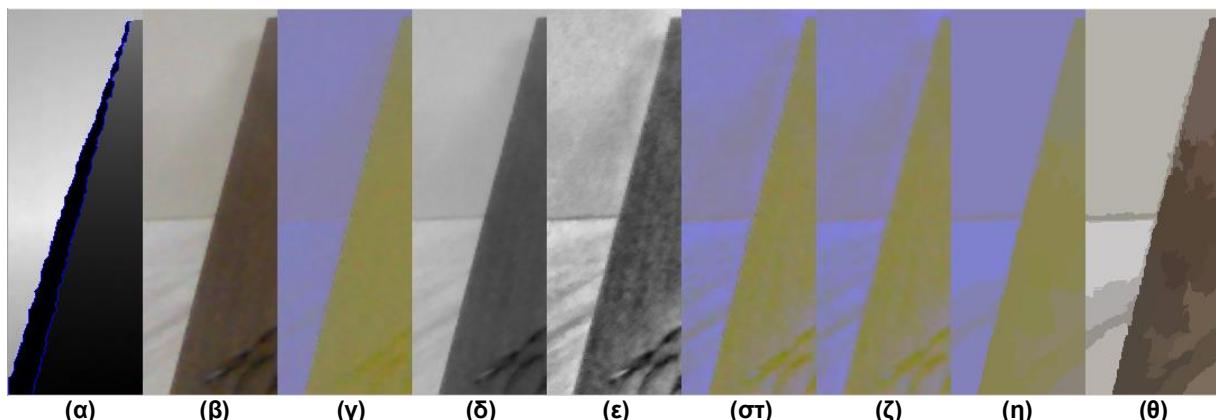




(α) (β) (γ) (δ) (ε) (στ) (ζ) (η) (θ)

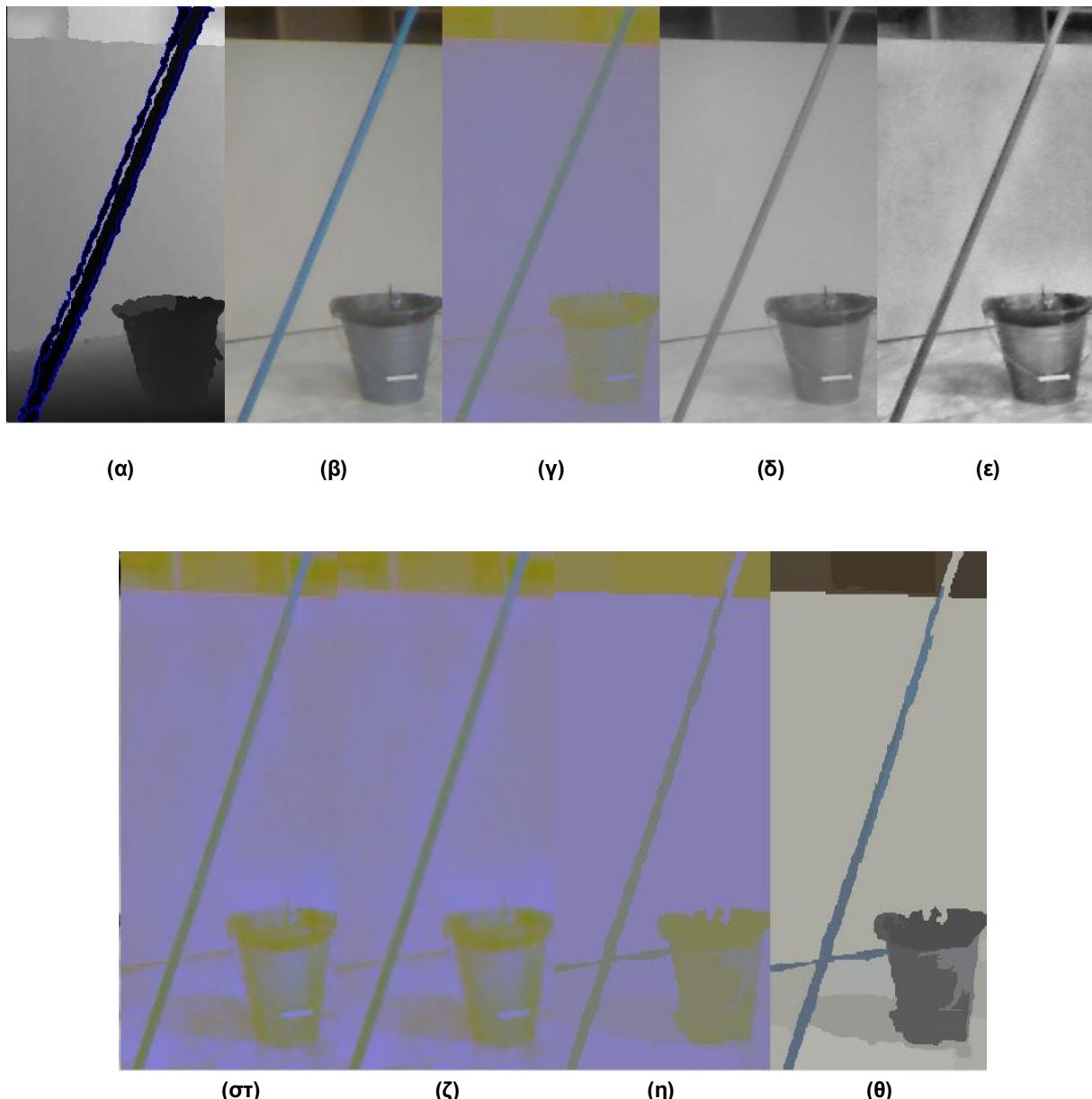


(α) (β) (γ) (δ) (ε) (στ) (ζ) (η) (θ)



(α) (β) (γ) (δ) (ε) (στ) (ζ) (η) (θ)





Εικόνα 4.2.1: Στάδια διαδικασίας τμηματοποίησης

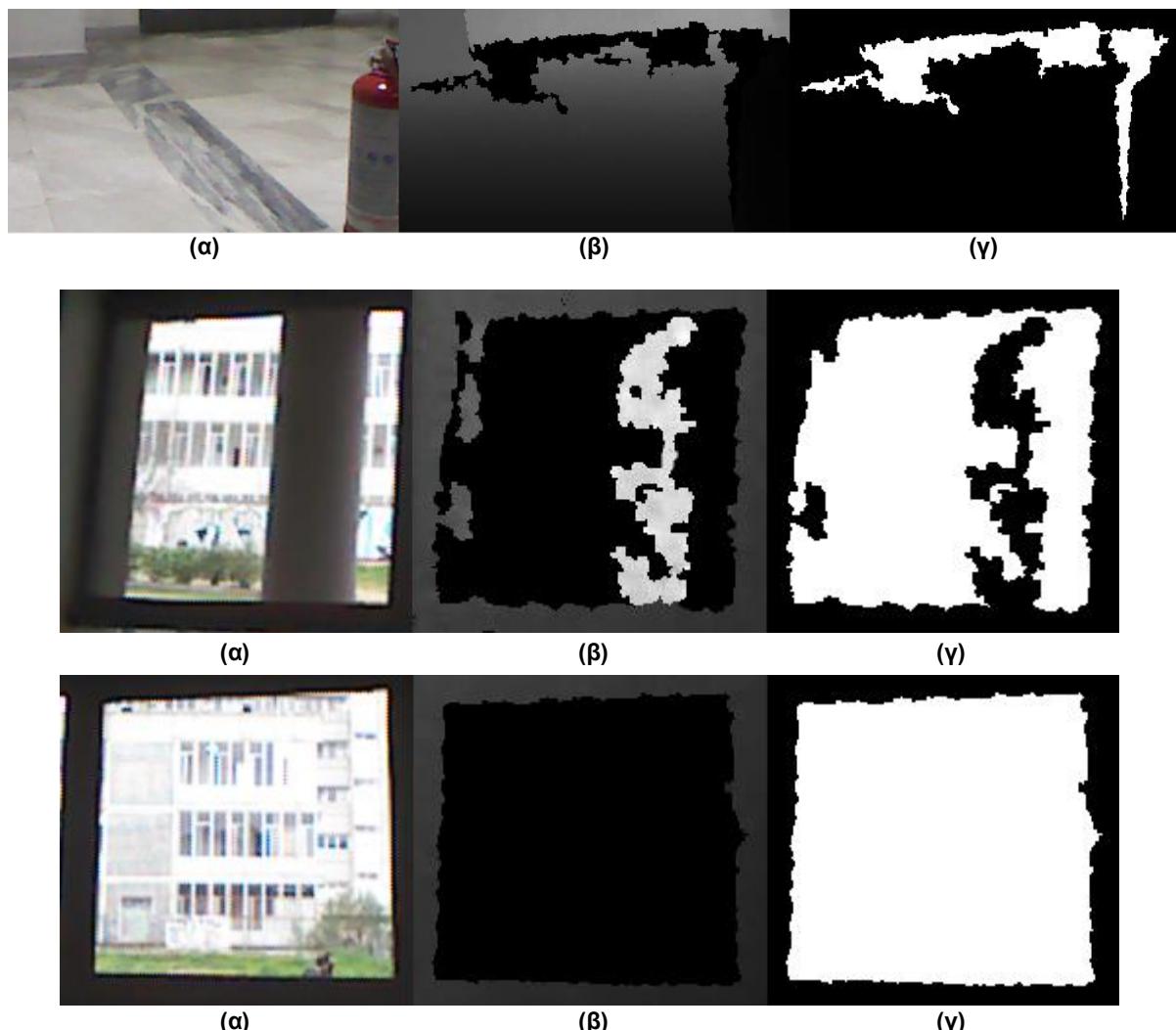
(α) Πλαίσιο Βάθους, (β) Πλαίσιο Χρώματος (RGB), (γ) Πλαίσιο Χρώματος (LAB), (δ) Κανάλι L , (ε) Κανάλι L μετά την εξισορρόπηση, (στ) Πλαίσιο Χρώματος (LAB) μετά την εξισορρόπηση, (ζ) Πλαίσιο Χρώματος (LAB) μετά την εφαρμογή bilateral φίλτρου, (η) Πλαίσιο Χρώματος (LAB) μετά την τμηματοποίηση, (θ) Πλαίσιο Χρώματος (RGB) μετά την τμηματοποίηση

4.3 Αναγνώριση Ανακλαστικών Επιφανειών

Με δεδομένο πλέον το πλαίσιο, όπως αυτό προέκυψε από την προεπεξεργασία της τμηματοποίησης, μπορούμε να προχωρήσουμε στη διαδικασία αναγνώρισης ανακλαστικών επιφανειών. Συγκεκριμένα, για κάθε πλαίσιο, εφόσον αυτό είναι ικανοποιητικά μεγάλο σε μέγεθος, θα προχωρήσουμε στον υπολογισμό μίας μετρικής ανωμαλίας της EΠΑΜ, ώστε να αποφύγουμε την πιθανότητα αναγνώρισης της τελευταίας



ως ανακλαστική επιφάνεια, καθώς μία τέτοια εκ φύσεως μπορεί να έχει καθορισμένο σχετικά ομοιόμορφο σχήμα (παραλληλόγραμμο, έλλειψη, κύκλος) . Για εκείνες μεγάλου μεγέθους, οι οποίες συχνά προκύπτουν σε απομακρυσμένες επιφάνειες με κλίση παράλληλη ως προς τη κατεύθυνση λήψης της κάμερας (για παράδειγμα στο έδαφος) ή για γειτονικές *ΕΠΑΜ* που τείνουν να ληφθούν ως ένα περίγραμμα λόγω ζητημάτων λειτουργίας και κατασκευής της κάμερας, η παραπάνω μετρική υπολογίζεται ως το πηλίκο του εμβαδού του Box που περιβάλει την *ΕΠΑΜ*, όπως αυτό ορίστηκε στην Παράγραφο 4.1.1, προς το εμβαδόν αυτής. Για *ΕΠΑΜ* μικρού και μέσου μεγέθους, η παραπάνω συνδυάζεται με μία μετρική, η οποία καθορίζεται ως ένα πηλίκο με αριθμητή τον αριθμό των σημείων που σχηματίζουν το ελάχιστο σε έκταση πολύγωνο που προσεγγίζει την *ΕΠΑΜ* και παρονομαστή το μέγεθός της. Τα ποικίλα όρια που καθορίζουν την κατά μέγεθος ταξινόμηση των *ΕΠΑΜ*, αλλά και εκείνα των αντίστοιχων μετρικών τους προσδιορίστηκαν πειραματικά, με τις μικρότερες από αυτές να λαμβάνουν πιο ανεκτικούς όρους ως προς το τι θεωρείται ανώμαλο περίγραμμα. Στη συνέχεια, για μία *ΕΠΑΜ* που δεν ξεπερνά τα δύο όρια που τίθενται για τις παραπάνω μετρικές, πραγματοποιείται έλεγχος ως προς το κατά πόσο η επιφάνεια της είναι σημαντικά προσκολλημένη στα όρια του συνολικού πλαισίου, καθώς πρόκειται να χειριστούμε τις συγκεκριμένες διαφορετικά. Κατόπιν, ξεκινά ο έλεγχος του κατά πόσο αυτή αποτελεί ανακλαστική επιφάνεια.



Εικόνα 4.3.0.1: Παρουσίαση Πλαισίων Χρώματος (α), Βάθους (β) και μάσκας της *ΕΠΑΜ* (γ) για εξεταζόμενες ως προς τη μορφολογία *ΕΠΑΜ* (η πρώτη απορρίπτεται, οι άλλες δύο ελέγχεται για τη πιθανότητα να αποτελεί ανακλαστική επιφάνεια)



Algorithm 4 Possible Reflective Surfaces

```
1: procedure      CHECKFORREFLECTIVES(contour,           big_box_thresh,
   medium_area_thresh   medium_anomaly_thresh,   medium_box_thresh,
   small_area_thresh     small_anomaly_thresh     small_box_thresh,
   bound_thresh)
2:   if contour_area < medium_area_thresh then
3:     Compute
   anomaly_measure = 
$$\frac{\text{approximate\_polygon\_vertices}}{\text{contour\_area}}$$

4:   end if
5:   Compute
   box_condition = 
$$\frac{\text{contour\_area}}{\text{bounding\_rectangle\_area}}$$

6:   small_condition = (contour_area < small_area_thresh
and box_condition < small_box_thresh and anomaly_measure
< small_anomaly_thresh)
7:   medium_condition = (contour_area < medium_area_thresh
and box_condition < medium_box_thresh and anomaly_measure
< medium_anomaly_thresh)
8:   big_condition = (contour_area > medium_area_thresh and
box_condition < big_box_thresh)
9:   if small_condition or medium_codition or big_condition then
10:    if bound_thresh < pixels of contour at any of the bounds then
11:      Check for possible reflective using MarkerBasedWatershed
12:    else
13:      Check for possible reflective using DetectReflectives
14:    end if
15:  end if
16: end procedure
```

Αλγόριθμος 4.3.1: Αλγόριθμος ελέγχου μορφολογίας ΕΠΑΜ

4.3.1 ΕΠΑΜ Προσκολλημένες στα Όρια του Συνολικού Πλαισίου

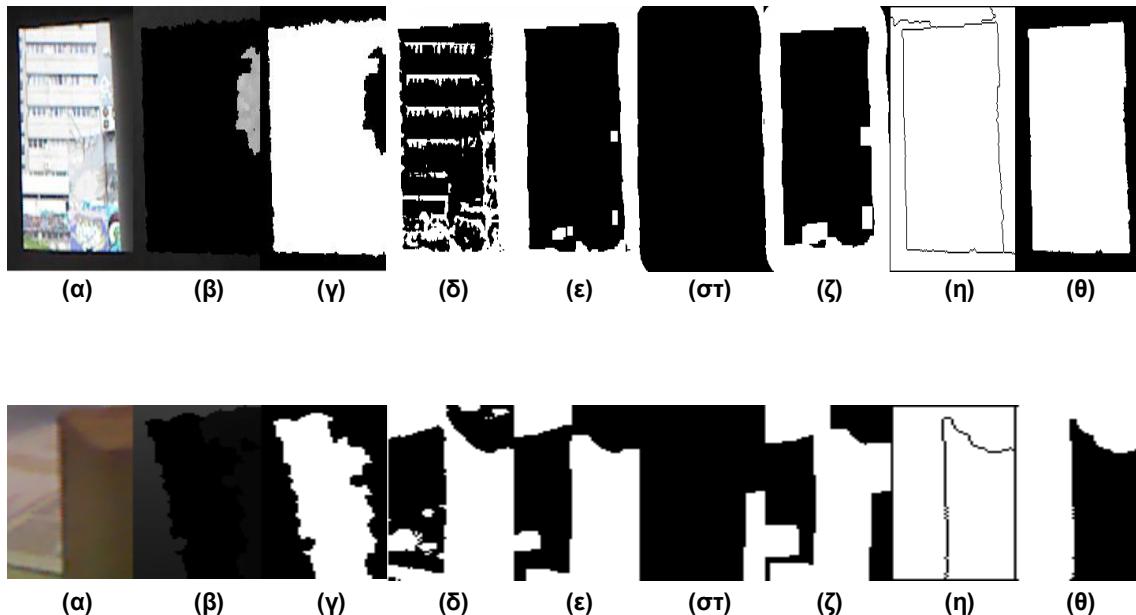
Για τις συγκεκριμένες *ΕΠΑΜ*, αποδείχτηκε πειραματικά, όπως θα δούμε παρακάτω, ότι με τη βοήθεια του αλγορίθμου watershed, μπορούμε να αναγνωρίσουμε ανακλαστικές επιφάνειες με μεγαλύτερη ακρίβεια. Ο περιορισμός από τα όρια της εικόνας επιτρέπει στον αλγόριθμο να απομονώσει τις δύο κυρίαρχες επιφάνειες του πλαισίου, εντοπίζοντας μόνο εκείνη της πιθανής ανακλαστικής και του παρασκηνίου. Η διαδικασία που ακολουθήθηκε είναι εκείνη που προτείνεται από το documentation της OpenCV για την τμηματοποίηση εικόνας με τη βοήθεια δεικτών (*markers*).

Αρχικά, το πλαίσιο μας, όπως αυτό προκύπτει από τη διαδικασία της τμηματοποίησης, υφίσταται smoothing με bilateral φίλτρο, ώστε πιθανές ακμές που προκύπτουν λόγω θορύβου ή εντός μίας ομοχρωματικής επιφάνειας να εξαλειφθούν. Στη συνέχεια,



πραγματοποιείται thresholding με τη λογική *Otsu* (βλ. Παράρτημα A, Παράγραφος A3), για να προκύψει μία μάσκα με την εκτίμηση των αντικειμένων του προσκηνίου. Η εξαγωγή πιθανού θορύβου πραγματοποιείται με τη χρήση μορφολογικού opening. Κατόπιν, απομονώνουμε το τμήμα της προκύπτουσας μάσκας που ανήκει σύγουρα στο background, με χρήση της διαδικασίας του *dilation* (βλ. Παράρτημα A, Παράγραφος A4). Αντίστοιχα, για εκείνα του προσκηνίου, χρησιμοποιούμε thresholding (βλ. Παράρτημα A, Παράγραφος A3) με όριο που προκύπτει με βάση τη μέγιστη τιμή του distance transform, δηλαδή της μέγιστης τιμής ενός πίνακα αποστάσεων από μη μηδενικό pixel, της μάσκας που προέκυψε από το opening (βλ. Παράρτημα A, Παράγραφος A4). Ως αποτέλεσμα των παραπάνω, μπορούμε να αφαιρέσουμε τις δύο προκύπτουσες μάσκες για να εντοπίσουμε το τμήμα της μάσκας για το οποίο δεν μπορούμε να εξάγουμε πληροφορία ως προς το που ανήκει. Πλέον, μπορούμε να ομαδοποιήσουμε τα ενωμένα στοιχεία εντός της μάσκας, με τη βοήθεια της διαδικασίας *connected components* (βλ. Παράρτημα A, Παράγραφος A10), ώστε να λάβουμε τα ζητούμενα markers. Τέλος, ο αλγόριθμος watershed θα εντοπίσει τα περιγράμματα των αντικειμένων του προσκηνίου.

Το τελικό πλαίσιο περιέχει μόνο το περίγραμμα του κυρίαρχου αντικειμένου του πλαισίου, καθώς και το περίγραμμα του συνολικού πλαισίου. Δημιουργώντας τη μάσκα με το μεγαλύτερο σε έκταση περίγραμμα, λαμβάνουμε το κυρίαρχο αντικείμενο του πλαισίου. Υπολογίζοντας το ποσοστό του τελευταίου το οποίο ανήκει στην *EPAM*, καθορίζεται το κατά πόσο αυτή είναι όντως ανακλαστική. Παρακάτω παρουσιάζουμε μία σειρά εικόνων από όλα τα στάδια που μόλις περιγράψαμε.



Εικόνα 4.3.1.1: Στάδια ταυτοποίησης ύπαρξης (πάνω) και μη (κάτω) ανακλαστικής επιφάνειας στα όρια της αρχικής εικόνας

(α) Πλαίσιο Χρώματος, (β) Πλαίσιο Βάθους, (γ) Μάσκα EPAM, (δ) Μάσκα μετά από *Otsu thresholding* στο gray scale πλαίσιο, (ε) Μάσκα (δ) μετά από μορφολογικό ανοιγμα, (στ) Μάσκα σύγουρου προσκηνίου, (ζ) Μάσκα απροσδιόριστης περιοχής, (η) Μάσκα διασυνδεμένων pixels, (θ) Μάσκα κυρίαρχης επιφάνειας (πιθανή ανακλαστική)



Algorithm 5 Marker Based Watershed

```
1: procedure MARKERBASEDWATERSHED(processed_lab_frame, contour,  
    reflective_thresh)  
2:   reflective_flag = 0  
3:   Convert processed_lab_frame to gray scale  
4:   Apply Otsu thresholding to the gray_scale_frame  
5:   Remove noise from the thresholded_frame using morphological opening  
6:   Use dilation to the opened_frame to extract the background area  
7:   Find the distance from non zero pixel for each pixel in opened_frame  
8:   Apply thresholding to the distance_map using its max_distance to find  
     the foreground area  
9:   Mark as unknown the common area of the background_mask and  
     foreground_mask  
10:  Define the desired markers as the connected components of the  
     foreground_mask  
11:  Ignore the markers of the pixels of the unknown_region  
12:  Execute WatershedAlgorithm using the markers found  
13:  Find the contour of the Watershed_frame with the maximum area  
14:  Compute  
  
reflective_measure = 
$$\frac{\text{common\_area}(\text{maximum\_area\_contour}, \text{contour\_area})}{\text{contour\_area}}$$
  
15:  if reflective_measure > reflective_thresh then  
16:    reflective_flag = 1  
17:  end if  
18:  return reflective_flag  
19: end procedure
```

Αλγόριθμος 4.3.1.1: Αλγόριθμος τρηματοποίησης εικόνας με χρήση *markers* μέσω *watershed* για εύρεση ομοχρωματικών πιθανών ανακλαστικών επιφανειών

4.3.2 ΕΠΑΜ στο Κέντρο του Συνολικού Πλαισίου

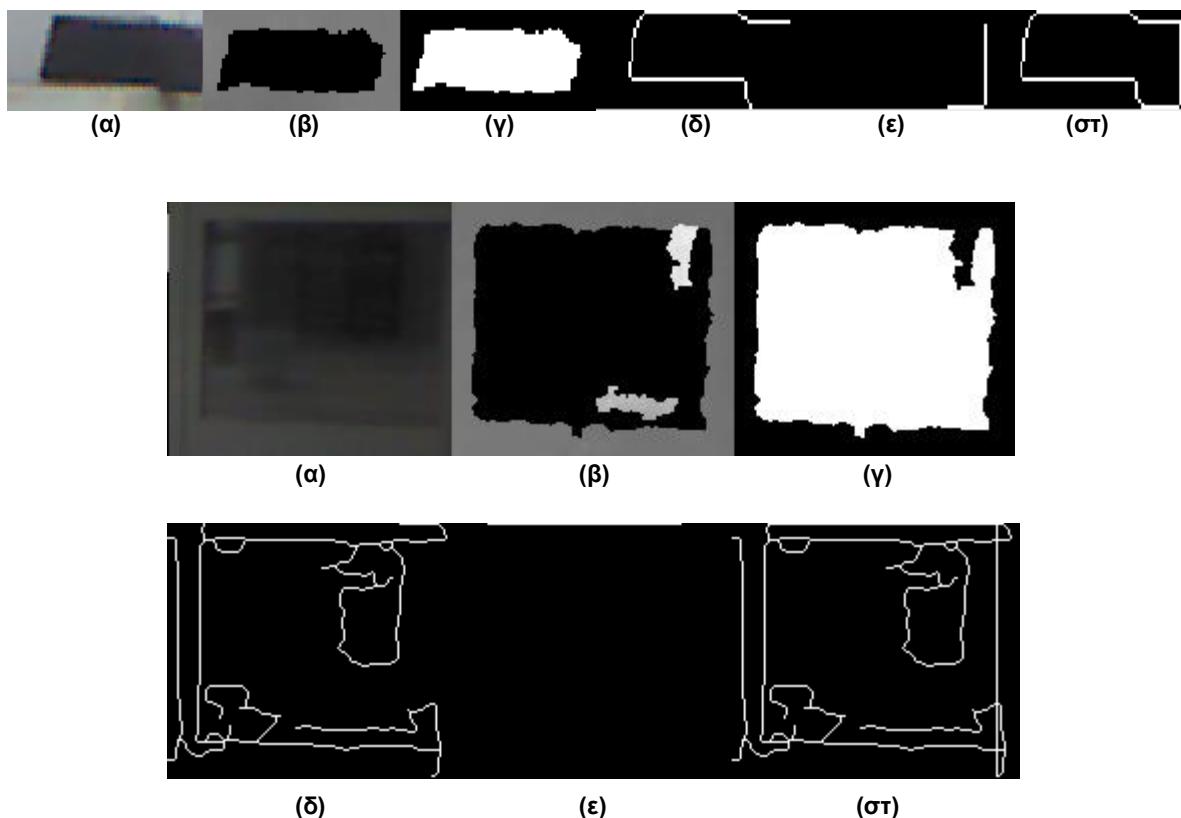
Για *ΕΠΑΜ* που δεν είναι προσκολλημένες στα όρια της εικόνας, αποδείχθηκε ότι η παραπάνω διαδικασία δεν είναι ιδιαίτερα αποτελεσματική. Ως αποτέλεσμα, προχωρήσαμε στη διαχείριση των πλαισίων με μία σειρά μορφολογικών μετασχηματισμών, διαδικασία την οποία στο σύνολο της θα περιγράψουμε στην παράγραφο 4.3.2.2. Προτού προχωρήσουμε σε αυτό, ωστόσο, θα αναλύσουμε μία βασική για αυτήν συμπληρωματική διαδικασία, το κλείσιμο των ανιχνευόμενων ακμών κοντά στα όρια του πλαισίου.



4.3.2.1 Κλείσιμο Ακμών στα Όρια του Πλαισίου

Σε ένα εξεταζόμενο πλαίσιο, η ανίχνευση ακμών με οποιαδήποτε από τις μεθόδους που περιγράψαμε στο προηγούμενο κεφάλαιο δεν αποδίδει κλειστά περιγράμματα, όταν αυτά τείνουν να βρίσκονται πάνω στα όριά του, σε αντίθεση με εκείνα που προκύπτουν ως αποτέλεσμα του αλγορίθμου watershed (βλ. Παράρτημα A, Παράγραφος A9), όπως δείξαμε παραπάνω. Το φαινόμενο αυτό είναι αρκετά συχνό όταν απομονώνουμε ένα πλαίσιο γύρω από μία *ΕΠΑΜ*, καθώς συχνά εμφανίζονται μόνο τμήματα αντικειμένων της συνολικής εικόνας εντός των ορίων του πλαισίου. Για το λόγο αυτό, δημιουργήθηκε η διαδικασία κλεισίματος ακμών στα όρια του πλαισίου, την οποία θα περιγράψουμε ευθύς αμέσως.

Αρχικά, λαμβάνοντας τη μάσκα των ανιχνευμένων ακμών, χρησιμοποιούμε τη διαδικασία connected components, ώστε να αναγνωρίσουμε μία σειρά συνδεδεμένων pixels. Στη συνέχεια, για pixels κοντά σε οποιαδήποτε από τα τέσσερα όρια του πλαισίου, εντοπίζουμε εκείνα με μόλις ένα γείτονα στη ίδια σειρά συνδεδεμένων pixels, δηλαδή εκείνα στην άκρη ενός ενιαίου περιγράμματος, δημιουργώντας μία λίστα. Κατόπιν, για κάθε ένα από αυτά τα περιγράμματα, συγκρατούμε το πλήθος εκείνων με pixels εντός της λίστας, τα οποία ενώνουμε, εφόσον είναι περισσότερα από ένα, ανάλογα με τη σχετική τους θέση, χρησιμοποιώντας οριζόντιες και κάθετες ευθείες.



Εικόνα 4.3.2.1.1: Αποτελέσματα διαδικασίας κλεισίματος ακμών στα όρια του πλαισίου

(α) Πλαίσιο Χρώματος, (β) Πλαίσιο Βάθους, (γ) Μάσκα *ΕΠΑΜ*, (δ) Περίγραμμα προς κλείσιμο, (ε) Προσθήκη διαδικασίας κλεισίματος, (στ) Τελικό περίγραμμα



Algorithm 6 Bound Edges Closing

```
1: procedure BOUNDEDGESCLOSING(detected_edges_frame)
2:   Find the connected components of the detected_edges_frame
3:   for each of the bounds do
4:     Find the pixels close to the bound
5:     for each of the current_bound_pixels do
6:       if current_pixel_neighboors_with_same_marker = 1 then
7:         Add pixel to to_connect_list
8:       end if
9:     end for
10:    end for
11:    for each of the unique markers do
12:      Find pixels of to_connect_list with the same marker
13:      repeat
14:        Connect first two pixels of same_marker_list based on their po-
           sition using horizontal and vertical line segments
15:      Remove the first pixel from same_marker_list
16:      until same_marker_list=1
17:    end for
18: end procedure
```

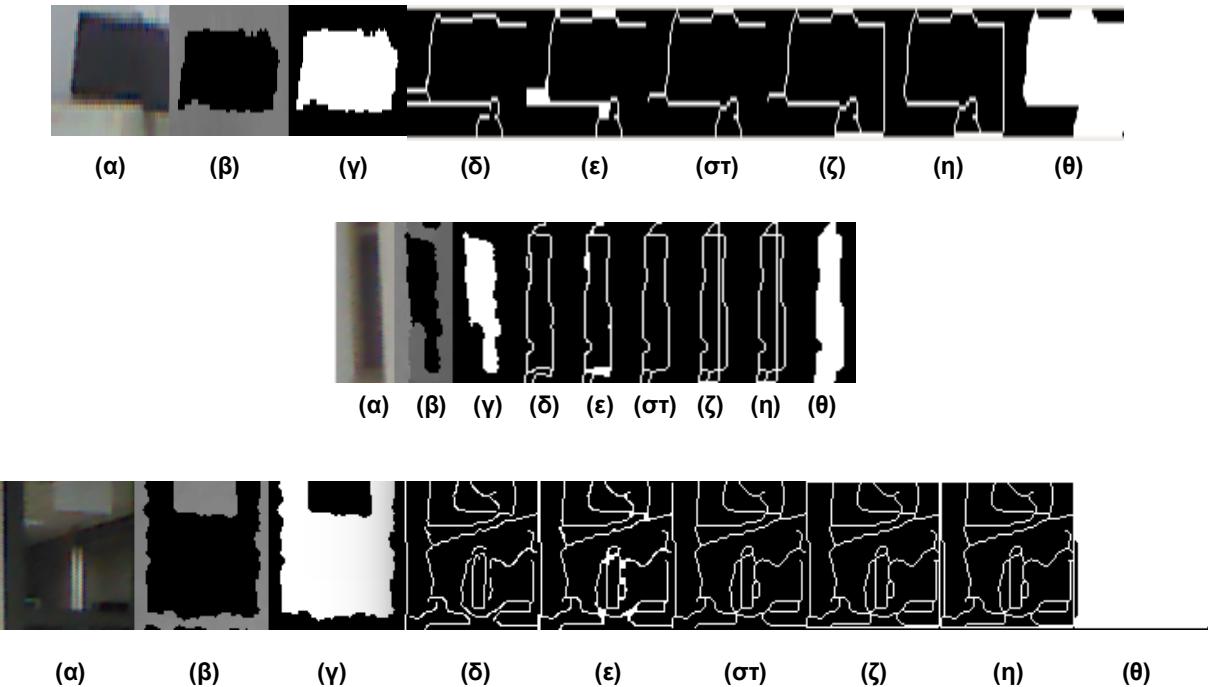
Αλγόριθμος 4.3.2.1.1: Αλγόριθμος κλεισμάτος ακμών στα όρια ενός πλαισίου

4.3.2.2 Αναγνώριση Ανακλαστικών Επιφανειών στο Κέντρο του Πλαισίου

Όπως αναφέραμε νωρίτερα, για τον εντοπισμό αυτών των επιφανειών, πειραματικά αποδείχθηκε ότι η λογική του αλγορίθμου watershed δεν αρκεί. Ως αποτέλεσμα, χρησιμοποιήσαμε μία σειρά μορφολογικών μετασχηματισμών, για να απομονώσουμε την κυρίαρχη επιφάνεια ενδιαφέροντος και σε τελικό στάδιο να την ταξινομήσουμε ως ανακλαστική ή όχι. Πιο συγκεκριμένα, αρχικά, λαμβάνουμε το αντίστοιχο gray scale πλαίσιο και εφαρμόζουμε τη λογική *Laplacian of Gaussian (LoG)* (βλ. Παράρτημα A, Παράγραφος A6) για την ανίχνευση ακμών. Στη συνέχεια, η διαδικασία μορφολογικού closing έρχεται για να κλείσει τυχόν ασυνέχειες στις προκύπτουσες ακμές, με πυρήνα και πάλι βασισμένο στο μέγεθος του πλαισίου. Η πιθανότητα δημιουργίας ακμών με πλάτος περισσότερων του ενός pixel αποφεύγεται με τη βοήθεια του skeletonization (βλ. Παράρτημα A, Παράγραφος A7). Οι προκύπτουσες ακμές είναι πλέον έτοιμες να κλείσουν με τη βοήθεια της διαδικασίας κλεισμάτος ακμών που περιγράψαμε παραπάνω. Ως τελευταίο βήμα αυτής της μορφολογικής επεξεργασίας, χρησιμοποιούμε τον αλγόριθμο floodfill (βλ. Παράρτημα A, Παράγραφος A8), ώστε να αφαιρεθούν από την binary εικόνα μη κλειστά περιγράμματα. Τέλος, για τη μάσκα των κλειστών περιγραμμάτων, συγκρίνουμε το ποσοστό των pixels της EPAM που βρίσκονται εντός της σε σχέση με το συνολικό τους αριθμό και, εφόσον αυτό ξεπερνά ένα ικανοτοιητικό όριο, κατατάσσουμε την EPAM ως ανακλαστική επιφάνεια. Ακολουθεί μία σειρά εικόνων που αποδίδει βήμα προς βήμα την μορφολογική επεξεργασία που μόλις περιγράψαμε. Στην πρώτη και τρίτη περίπτωση, οι



ΕΠΑΜ αναγνωρίζονται ως ανακλαστικές επιφάνειες, ενώ στη δεύτερη αυτό ορθά δεν συμβαίνει.



Εικόνα 4.3.2.3.1: Βήματα διαδικασίας ταυτοποίησης μίας επιφάνειας ως ανακλαστική

(α) Πλαίσιο Χρώματος, (β) Πλαίσιο Βάθους, (γ) Μάσκα Βάθους, (δ) Ευρισκόμενα περιγράμματα στην έγχρωμη εικόνα, (ε) Περιγράμματα μετά από μορφολογικό κλείσιμο, (στ) Περιγράμματα μετά από skeletonization, (ζ) Περιγράμματα μετά από κλείσιμο, (η) Περιγράμματα μετά την αφαίρεση μη κλειστών, (θ) Μάσκα πιθανής ανακλαστικής επιφάνειας

Algorithm 7 Detect Reflective Surfaces Away From Bounds

```

1: procedure DETECTREFLECTIVES(color_image, contour, refl_bound)
2:   reflective_flag = 0
3:   Converting color_image to gray_scale_image
4:   Extracting contours from gray_scale_image image using LoG
5:   Using morphological closing at the edges_image
6:   Perform skeletonization to get one_pixel_width_edges_image
7:   Using floodfill to isolate the closed contour
8:   Compute

```

$$\text{reflective_measure} = \frac{\text{closed_contour_area}}{\text{contour_area}}$$

```

9:   if reflective_measure > refl_bound then
10:      reflective_flag = 1
11:   end if
12:   return reflective_flag
13: end procedure

```

Αλγόριθμος 4.3.2.3.1: Αλγόριθμος εύρεσης ανακλαστικών επιφανειών



4.4 Ομαδοποίηση των ΕΠΑΜ με Περιοχές Έγκυρου Βάθους ίδιου Χρωματικού Τμήματος

Σε αυτήν την παράγραφο περιγράφουμε τη διαδικασία αντιστοίχησης κάθε περιοχής μίας ΕΠΑΜ που ανήκει σε διαφορετικό τμήμα χρώματος με την αντίστοιχη περιοχή έγκυρου βάθους, από την οποία οφείλουμε να εξάγουμε πληροφορία για να γεμίσουμε την περιοχή. Με τον τρόπο αυτό διασπούμε μια ενιαία περιοχή μη μέτρησης από τον αισθητήρα στα διαφορετικά αντικείμενα που απέτυχε να αποδώσει, προβάλλοντας πάνω της τα τμήματα χρώματος.

Για την αντιστοίχηση ΕΠΑΜ σε αντικείμενα, θεωρούμε γνωστά ένα πλαίσιο βάθους (*Depth*) που συμπεριλαμβάνει μια ΕΠΑΜ και το αντίστοιχο πλαίσιο RGB κατάλληλα τμηματοποιημένης πληροφορίας (*Segments*).

Έστω, λοιπόν, ότι το σύνολο των τμημάτων χρώματος που υπάρχουν στο πλαίσιο τμημάτων χρώματος είναι

$$labels \doteq l = \{1, 2, \dots, \Lambda\}$$

όπου το label περιγράφει απλά έναν αύξοντα αριθμό ταυτότητας του συγκεκριμένου χρωματικού τμήματος. Έτσι, για κάθε τμήμα χρώματος ορίζουμε τα σύνολα:

$$S_l \doteq \{(x, y) / Segments(x, y) = l\}, \quad l = 1, \dots, \Lambda$$

$$N_l \doteq \{(x, y) \in S_l / Depth(x, y) = 0\}, \quad l = 1, \dots, \Lambda$$

$$V_l \doteq \{(x, y) \in S_l / Depth(x, y) > 0\}, \quad l = 1, \dots, \Lambda$$

δηλαδή το S αποτελεί το σύνολο των σημείων που ανήκουν στο ίδιο τμήμα χρώματος, N το υποσύνολο του S εντός και V το υποσύνολο του S εκτός της ΕΠΑΜ. Εμείς επιθυμούμε να διασπάσουμε το N σε n υποσύνολα διακριτών μη γειτονικών περιοχών θορύβου, δηλαδή για κάθε l :

$$N_l = N_{l_1} \cup N_{l_2} \cup \dots \cup N_{l_n}, \quad N_{l_i} \cap N_{l_j} = \emptyset, \quad \forall i \neq j$$

καθώς και να αντιστοιχίσουμε κάθε τέτοιο υποσύνολο σε ένα υποσύνολο του V , το οποίο περιέχει εκείνα ακριβώς τα σημεία έγκυρης πληροφορίας βάθους με τα οποία οφείλει να βαφτεί η συγκεκριμένη διασπασμένη περιοχή θορύβου. Ουσιαστικά, θέλουμε για κάθε $l = 1, \dots, \Lambda$ να φτιάξουμε ομάδες:

$$P_{l_i} = N_{l_i} \cup V_{l_i}, \quad i = 1, \dots, \lambda, \quad P_{l_i} \cap P_{l_j} = \emptyset, \quad \forall i \neq j$$

Η παραπάνω διαδικασία μπορεί να περιγραφεί από τα εξής βήματα:

1. Εύρεση όλων των ενιαίων περιοχών του ίδιου χρωματικού τμήματος l που περιέχουν σημεία εντός και εκτός ΕΠΑΜ στο πλαίσιο βάθους, για κάθε l



2. Αντιστοίχηση ενιαίων περιοχών θορύβου N με ενιαίες γειτονικές υποψήφιες περιοχές έγκυρου βάθους V
3. Εξαγωγή του VI για κάθε τέτοια περιοχή, από τα σημεία του τμήματος εκτός τρύπας τα οποία είναι γειτονικά στο θόρυβο.

Το τελευταίο βήμα αντιπροσωπεύει την εξαγωγή της MEB , για την οποία θα υπολογίσουμε στη συνέχεια το διάνυσμα κατεύθυνσης βάθους, με βάση το οποίο θα γεμίσουμε την κάθε συμπαγή περιοχή θορύβου.

4.4.1 Δημιουργία ενιαίων περιοχών ίδιου τμηματικού χρώματος

Το βήμα αυτό περιγράφει τη διαδικασία εύρεσης όλων των ενιαίων περιοχών ίδιου χρωματικού τμήματος εντός και εκτός της $EPAM$. Ουσιαστικά, για κάθε διαφορετικό χρωματικό *label* που υπάρχει εντός της, προβάλλουμε το σύνολο S των σημείων του τμήματος σε μια μάσκα, όπου με την χρήση περιγραμμάτων το διαχωρίζουμε σε λ ενιαίες περιοχές $S_1, S_2, \dots, S_\lambda$. Με τον τρόπο αυτό τελικά διασπούμε την $EPAM$ σε ενιαία μέρη ίδιου χρωματικού τμήματος, για κάθε ένα από τα οποία υπάρχει μια γειτονική περιοχή του ίδιου χρώματος.

Έτσι, για κάθε $l = 1, 2, \dots, \Lambda$ αποκτούμε:

$$S_l \doteq S^l = S_1^l \cup S_2^l \cup \dots \cup S_\lambda^l, \quad S_i^l \cap S_j^l = \emptyset, \quad \forall i \neq j, \quad l = 1, \dots, \Lambda$$

όπου για το κάθε τέτοιο σύνολο έχουμε, χρησιμοποιώντας τον ίδιο συμβολισμό με την προηγούμενη παράγραφο:

$$S_i^l = N_i^l \cap V_i^l, \quad i = 1, \dots, \lambda, \quad l = 1, \dots, \Lambda$$

Το σενάριο 4.4.1.1 επιδείχνει τον τρόπο διάσπασης του συνόλου σε ενιαίες περιοχές, ενώ η εικόνα 4.4.1.1 παρουσιάζει το αποτέλεσμα του σεναρίου για διαφορετικές περιπτώσεις χρωματικών τμημάτων και $EPAM$.

Scenario 2: Dividing a hole in discrete regions of similar color

```
Defining th depth region as Depth, color labels as labels and a random
blob as blobj
for l in labels inside blobj do
```

Defining the depth regions or interest as:

```
cluster = Depth[labels == l]
```

Finding existing contours in the *cluster*:

```
cluster_contours = findContours(cluster)
```

```
for i in cluster_contours do
```

Defining the points inside as *points_i*:

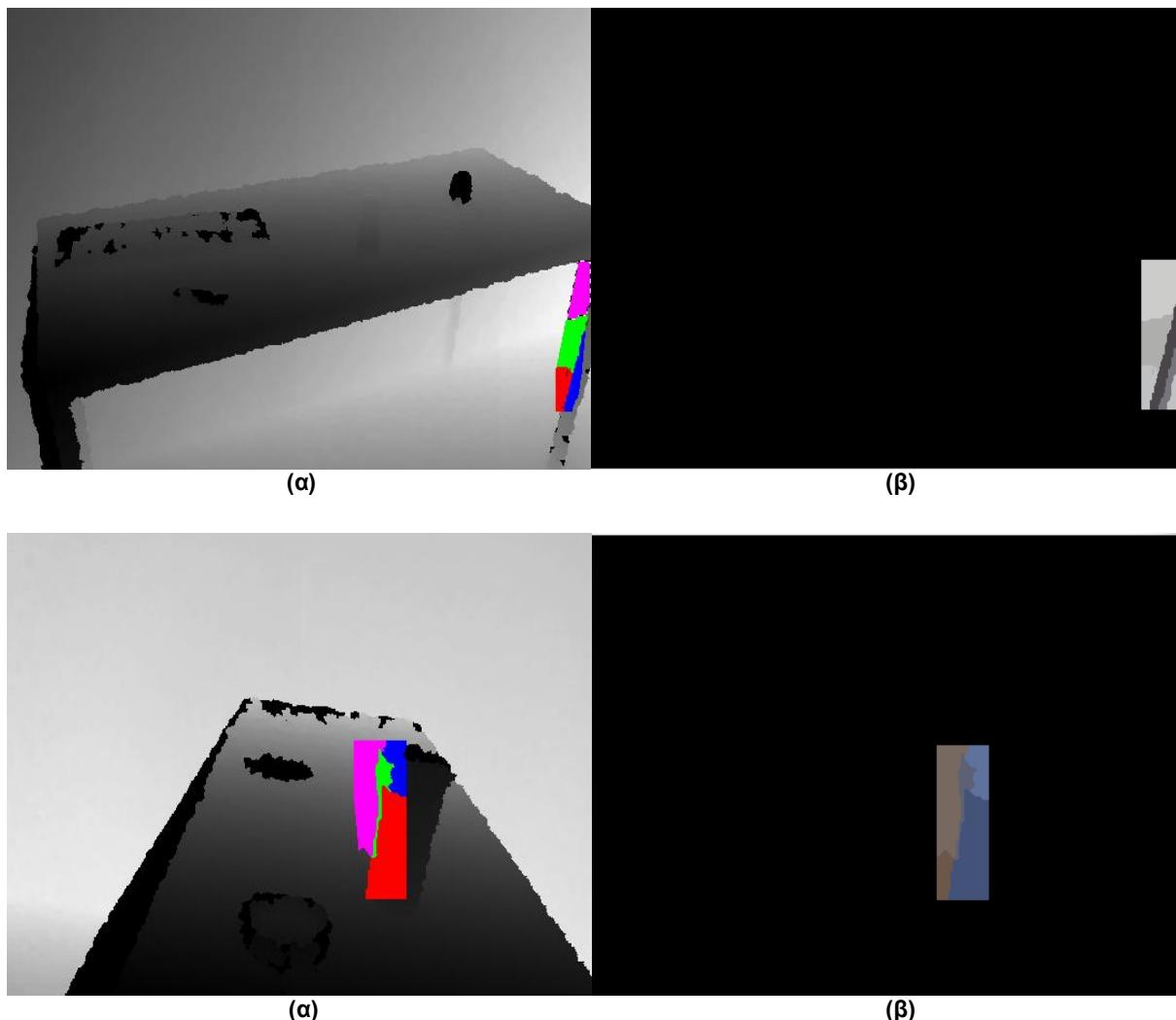
```
pointsi = points.inside(i)
```

```
end for
```

```
end for
```

Σενάριο 4.4.1: Διάσπαση $EPAM$ σε ενιαίες περιοχές ίδιου χρωματικού τμήματος





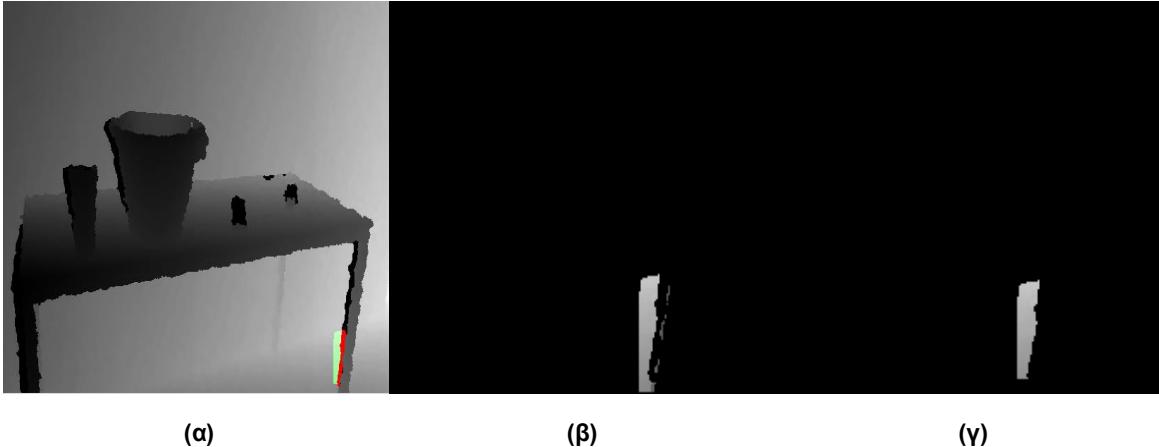
**Εικόνα 4.4.1.1: Αποτύπωση του Σεναρίου 4.4.1 οπτικά
(a) Υπάρχοντα χρωματικά τμήματα εντός του Πλαισίου, (β) Πλαίσιο Χρώματος**

4.4.2 Εξαγωγή Μάσκας Έγκυρου Βάθους

Για κάθε *EPAM* που δεν αναγνωρίζεται ως ανακλαστική επιφάνεια, η πρώτη αναγκαία διαδικασία είναι να εξάγουμε εκείνες τις περιοχές έγκυρου βάθους που την περιβάλλουν είτε αυτή αναγνωριστεί ως ανακλαστική επιφάνεια, είτε όχι. Για να συμβεί αυτό, υπολογίζουμε το *Rect*, όπως το ορίσαμε παραπάνω, και το επεκτείνουμε κατά ένα αριθμό *pixels*, ώστε να αποφύγουμε προβλήματα που έχουν να κάνουν με τη μορφολογία της *EPAM*. Στη συνέχεια, η τελευταία, ανάλογα με το μέγεθος της, θα προχωρήσει στη διαδικασία της διόρθωσης. Όταν αυτή είναι ιδιαίτερα μικρή, επιλέγουμε, όπως θα δούμε παρακάτω, να τις αποδώσουμε ως τιμή σε όλα τα *pixels* αυτή του μέσου όρου της γειτονιάς της, δηλαδή αυτού του *Rect*. Ωστόσο, προτού συμβεί αυτό, παρεμβάλλεται μία διαδικασία απομάκρυνσης εξωκείμενων τιμών (*AET*). Η συγκεκριμένη, αφού χωρίσει το σύνολο των τιμών του *Rect*, σε δύο ομάδες με βάση το αν ξεπερνούν ή όχι τη μέση του εύρους των εμφανιζόμενων τιμών. Οι ομάδες συγκρίνονται ως προς τους πληθυσμούς τους και, στην περίπτωση που κάποια από τις δύο είναι σημαντικά μικρότερη από την άλλη, θα αποκλειστεί από τη μάσκα έγκυρου βάθους ως εξωκείμενη τιμή. Η *AET*



εφαρμόζεται και στις περιπτώσεις διόρθωσης με βάση την πληροφορία χρώματος, όπου καταφέρνει σε σημαντικό βαθμό να αποβάλλει rixels που προέκυψε να ανήκουν στο ίδιο με το παρόν label καθαρά λόγω αστοχίας της τμηματοποίησης, ενώ στην πραγματικότητα ανήκουν σε εντελώς διαφορετικό επίπεδο βάθους. Παρακάτω, παραθέτουμε εικόνες που αποτυπώνουν τη δράση της AET.



Εικόνα 4.4.2.1: Λειτουργία της διαδικασίας AET

(α) Πλαίσιο Βάθους με χρωματισμένο ένα χρωματικό label μίας EPAM, (β) Μάσκα έγκυρης πληροφορίας, (γ) Μάσκα έγκυρης πληροφορίας μετά την AET

4.5 Υπολογισμός Διανύσματος Κατεύθυνσης Βάθους Έγκυρης Περιοχής

Στην παράγραφο αυτή θα περιγράψουμε τη μέθοδο με την οποία αναλύουμε την πληροφορία βάθους που εμπεριέχεται εντός της περιοχής της MEB που έχουμε εξάγει στο προηγούμενο βήμα και προσπαθούμε να τη μεταφέρουμε κατάλληλα στην περιοχή θορύβου που θέλουμε να διορθώσουμε. Ουσιαστικά η MEB διαθέτει για κάθε EPAM ή τμήμα EPAM που θέλουμε να διορθώσουμε, εκείνη ακριβώς την περιοχή που ανήκει στην ίδια χρωματική ομάδα και συνορεύει με την αυτή. Στις απλές περιπτώσεις το αντικείμενο, κομμάτι του οποίου αποτελεί στον χάρτη βάθους θόρυβο σκιάς, βρίσκεται σε σταθερό επίπεδο βάθους, οπότε μια μέση τιμή της έγκυρης περιοχής αρκεί για τη διόρθωση του θορύβου. Στις περισσότερες περιπτώσεις όμως, το αντικείμενο βρίσκεται σε κατευθυνόμενο επίπεδο βάθους. Στις παρακάτω εικόνες παρουσιάζουμε ενδεικτικά τις δύο περιπτώσεις για λόγους αποσαφήνισης.

Όπως γίνεται φανερό, στην περίπτωση κατευθυνόμενου επιπέδου, οι τιμές βάθους στην MEB παρουσιάζουν μια κατευθυνόμενη μεταβολή. Η μεταβολή αυτή μπορεί να μοντελοποιηθεί ως ένα διάνυσμα, το Διάνυσμα Κατεύθυνσης Βάθους (ΔΚΒ), μέτρο του οποίου αποτελεί η στοιχειώδης μεταβολή του βάθους (έστω dz) μεταβαίνοντας από rixel σε pixel της MEB και γωνία η διεύθυνση της μεταβολής του βάθους εντός της περιοχής. Στην παρακάτω εικόνα παρουσιάζουμε το ΔΚΒ των σημειωμένων περιοχών στις εικόνες:

Βλέπουμε ότι το ΔΚΒ υπολογίζεται ως προς το γνωστό καρτεσιανό σύστημα συντεταγμένων, με διεύθυνση οριζόντιου άξονα από αριστερά προς τα δεξιά (άξονας x) και



κατακόρυφου από κάτω προς το πάνω (άξονας y). Η γωνία του λοιπόν υπολογίζεται με την ανθωρολογιακή φορά και από τον άξονα x.

Στη συνέχεια περιγράφουμε πως υπολογίζουμε το ΔΚΒ από τη ΜΕΒ για διαφορετικές κατηγορίες ΕΠΑΜ.

4.5.1 Διάνυσμα Κατεύθυνσης Βάθους από Μάσκα Έγκυρου Βάθους

Στην περίπτωση αυτή, η ΜΕΒ διαθέτει μόνο μια κλειστή καμπύλη η οποία διασχίζεται κατακόρυφα και οριζόντια χωρίς να μεταβαίνει εκτός της. Συνθέτοντας την περιοχή αυτή με την κλειστή καμπύλη του τμήματος ΕΠΑΜ που εξετάζουμε, προκύπτει μία ενιαία συμπαγής περιοχή (μία περιοχή που αποτελεί τμήμα της συγκεκριμένης ομάδας χρώματος). Παράδειγμα τέτοιας ΜΕΒ φαίνονται στις εικόνες παρακάτω.

Σε ένα πλαίσιο γύρω από την περιοχή αυτή, μεγέθους R x C, ορίζουμε για κάθε γραμμή και για κάθε στήλη του πλαισίου τους αντίστοιχους συντελεστές μεταβολής βάθους:

$$dx_i = \frac{1}{C} \sum_j (Depth[i+1, j] - Depth[i, j])$$
$$dy_j = \frac{1}{R} \sum_i (Depth[i, j+1] - Depth[i, j])$$

όπου τα dx_i , dy_j εκφράζουν τη στοιχειώδη μεταβολή του βάθους στην οριζόντια και στην κατακόρυφη κατεύθυνση αντίστοιχα, με φορά από αριστερά προς τα δεξιά και από πάνω προς τα κάτω. Για τον υπολογισμό των μεγεθών αυτών, καθώς και των παρακάτω, έχουμε παραβλέψει όλα τα σημεία του πλαισίου τα οποία έχουν μηδενικό βάθος, καθώς και τις μεταβολές των αλμάτων από κενά σε έγκυρα βάθη, έτσι ώστε να περιοριστεί ο υπολογισμός στα σημεία μόνο εντός της περιοχής της ΜΕΒ ανεξαρτήτως σχήματος. Αυτός ο τρόπος υπολογισμού των μεταβολών κατεύθυνσης είναι που καθιστούσε ανέφικτη την χρήση των έτοιμων εργαλείων υπολογισμού κατεύθυνσης μεταβολής και μας οδήγησε στον ορισμό του ΔΚΒ.

Η μέση στιγμιαία μεταβολή βάθους σε κάθε κατεύθυνση ορίζεται από τις τιμές κάθε στοιχειώδους μεταβολής:

$$dx = \frac{1}{R} \sum_i dx_i = \frac{1}{RC} \sum_i \sum_j (Depth[i+1, j] - Depth[i, j])$$
$$dy = \frac{1}{C} \sum_i dy_j = \frac{1}{RC} \sum_j \sum_i (Depth[i, j+1] - Depth[i, j])$$

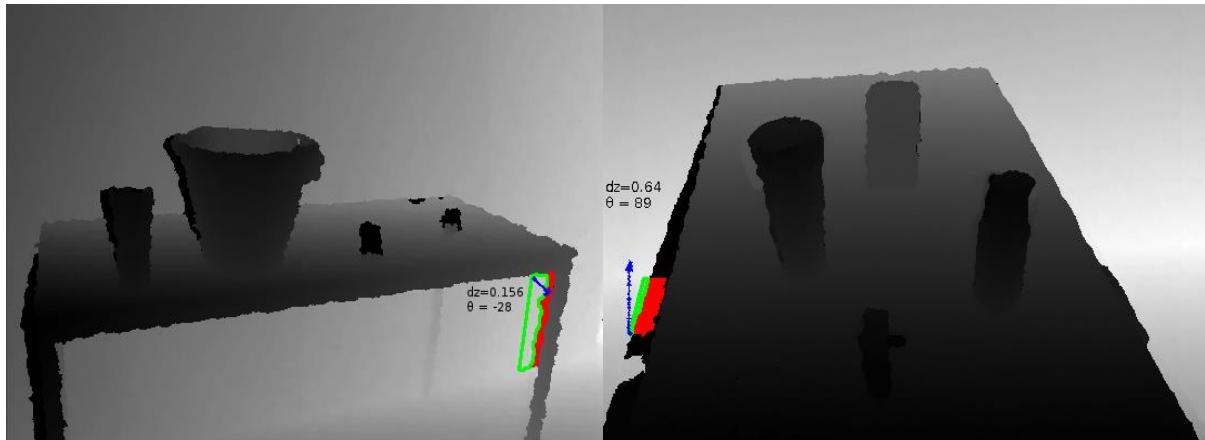


και τελικά το ΔΚΒ:

$$dz = \sqrt{dx^2 + dy^2}, \quad \theta = \arctan(-y/x)$$

όπου το πρόσημο – στον ορισμό της γωνίας προκύπτει απ' την αντίθετη φορά ορισμού του ΔΚΒ στην κατακόρυφη διεύθυνση από τον τρόπο εξαγωγής των μεγεθών στο πλαίσιο.

Κάποιες περιπτώσεις διαφορετικής κατεύθυνσης ή (και) διαφορετικού μεγέθους μεταβολής βάθους παρουσιάζονται ενδεικτικά στις παρακάτω εικόνες.



Εικόνα 4.5.1.1: Παρουσίαση διαδικασία προσδιορισμού του ΔΒΚ (με πράσινο αποδίδονται η ΜΕΒ και με κόκκινο το αντίστοιχο χρωματικό τμήμα μίας ΕΠΑΜ)

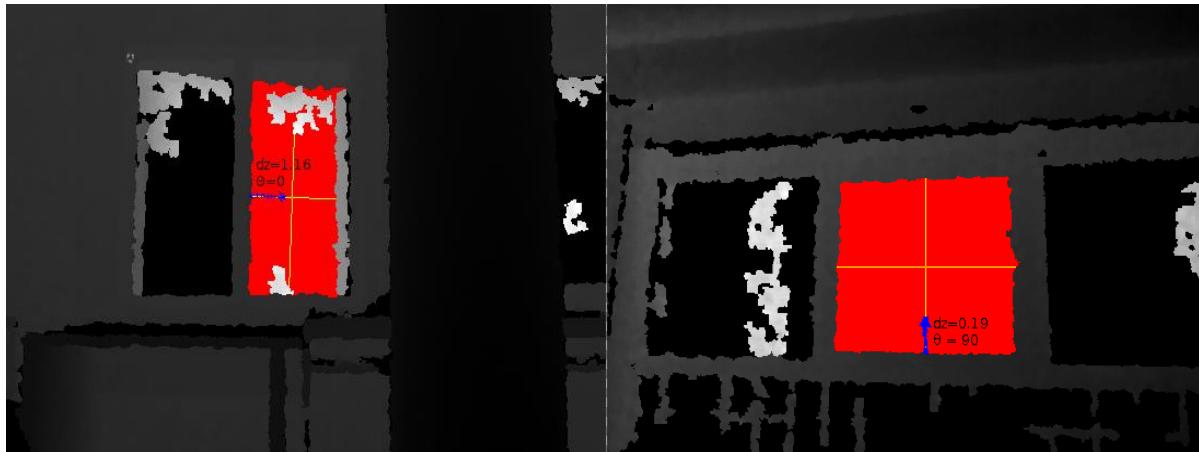
4.5.2 Διάνυσμα Κατεύθυνσης Βάθους από Κάδρο Πλαισίου

Υπάρχουν περιπτώσεις, που μια ΕΠΑΜ δεν έχει περάσει λόγω της ταξινόμησής της σε ομάδα χρώματος, και η πληροφορία βάθους με την οποία θα διορθωθεί οφείλει να εξαχθεί από ολόκληρο το πλαίσιο βάθους της ΕΠΑΜ. Στις περιπτώσεις αυτές, ανήκουν ΕΠΑΜ που αποτελούν ανακλαστικές επιφάνειες. Εδώ δεν υπάρχει ΜΕΒ και το ΔΚΒ προκύπτει ειδικά από το “κάδρο” του πλαισίου, υπολογίζοντας μόνο τις μεταβολές βάθους στις τέσσερις ακραίες περιοχές της ΕΠΑΜ. Πιο συγκεκριμένα, αφού βρεθεί το κέντρο μάζας και το επεκταμένο Box, σχεδιάζουμε σε μία μάσκα τις δύο ευθείες που περνούν από το πρώτο και έχουν διευθύνσεις που ορίζονται από τα σημεία του δεύτερου, όπως φαίνονται στις παρακάτω εικόνες. Στη συνέχεια, απομονώνουμε τις περιοχές των γραμμών εκείνες που βρίσκονται στο εσωτερικό της ΕΠΑΜ. Ως αποτέλεσμα, προκύπτει μία σειρά γραμμών με δύο εξ αυτών να διέρχονται ακόμα από το κέντρο μάζας που υπολογίστηκε. Για αυτές, λοιπόν, εντοπίζουμε να σημεία τα σημεία που τις οριοθετούν. Τα τελευταία θα αποτελέσουν τα T, L, R, B Ακραία Σημεία, που θα χρησιμοποιηθούν για τον υπολογισμό του ΔΚΒ. Ακολουθεί η διαδικασία υπολογισμού του ΔΚΒ, η οποία αφού προσδιορίσει την κατεύθυνση στην οποία η μεταβολή είναι κυρίαρχη θα υπολογίσει τη γωνία του διανύσματος με τη μαθηματική λογική από το αντίστοιχο ζεύγος σημείων (T και B ή L και R), που παρουσιάστηκε και στην προηγούμενη παράγραφο. Από το ζεύγος αυτό, τα προσδιοριστεί και το μέτρο ως:



$$dz = \frac{depth_max - depth_min}{\sqrt{(x_max - x_min)^2 + (y_max - y_min)^2}}$$

Δεδομένου, πλέον, του ΔΚΒ, μπορούμε να προχωρήσουμε στη καταχώρηση τιμών βάθους.

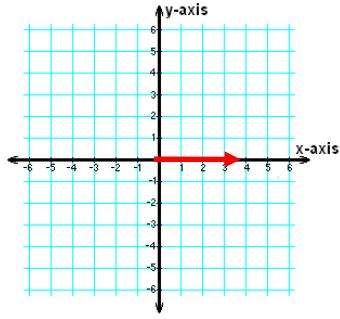


Εικόνα 4.5.2.1: Παρουσίαση διαδικασία προσδιορισμού του ΔΒΚ για ανακλαστικές (με κίτρινο αποδίδονται οι ευθείες που περιγράψαμε και με κόκκινο η αντίστοιχη ΕΠΑΜ)

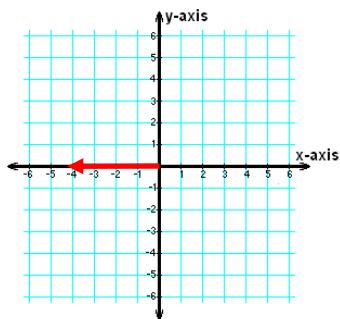
4.6 Καταχώρηση Τιμών Βάθους στην ΕΠΑΜ

Όσο αφορά την καταχώρηση των υπολογίζομενων τιμών βάθους, αυτή ξεκινά με εκείνες που η έκταση τους δεν ξεπερνά ένα κατώτερο όριο έκτασης. Για τέτοιες ΕΠΑΜ, που εμφανίζονται συχνά σε εικόνες βάθους λόγω προσωρινής αστοχίας της κάμερας, προτιμήσαμε, δεδομένου του μικρού τους μεγέθους, την άμεση απόδοση του μέσου όρου των έγκυρων βαθών γειτονικών pixels ως νέα τιμή, όπως αυτά οριθμεύονται από τη ΜΕΒ. Για να αποφύγουμε την ύπαρξη λίγων σε αριθμό τιμών με σημαντική διαφορά από τον μέσο όρο της πλειοψηφίας, χρησιμοποιούμε τη διαδικασία απομάκρυνσης εξωκείμενων τιμών, συγκρίνοντας τον αριθμό των pixels της ΜΕΒ για όσα έχουν τιμές άνω και κάτω του μέσου του εύρους. Σχετικά με τις ανακλαστικές επιφάνειες αλλά και τις πιθανές περιοχές κοινής χρωματικής πληροφορίας για ένα τμήμα μίας ΕΠΑΜ, δεδομένου του ΔΚΒ, εξάγουμε ένα Χάρτη Αποστάσεων (ΧΑ). Ο τελευταίος προκύπτει από τη διάσπαση του διανύσματος σε μία οριζόντια και μία κάθετη συνιστώσα από τις παρακάτω, με την κάθε διεύθυνση να εκπροσωπεύται από έναν τίνακα αυξανόμενο προς τη ανάλογη φορά.

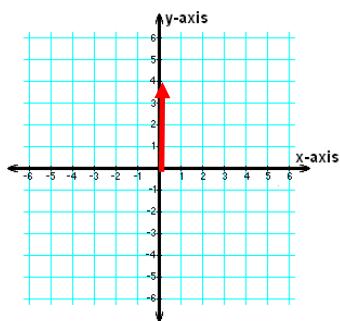




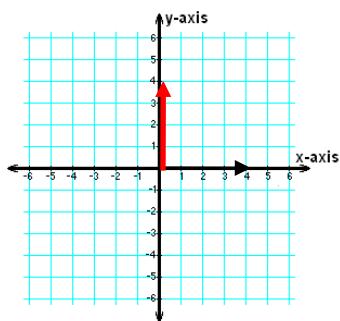
0	1	2	...	M
0	1	2	...	M
0	1	2	...	M
0	1	2	...	M
0	1	2	...	M



M	...	2	1	0
M	...	2	1	0
M	...	2	1	0
M	...	2	1	0
M	...	2	1	0



N	N	N	N	N
...
2	2	2	2	2
1	1	1	1	1
0	0	0	0	0



0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
...
N	N	N	N	N

Εικόνα 4.6.1: Αναπαράσταση διανυσμάτων των 4 βασικών κατευθύνσεων (0° , 180° , 90° και 270°) και των αντίστοιχων χαρτών απόστασης

Ως αποτέλεσμα, ανάλογα με το τεταρτημόριο στο οποίο η γωνία τους διανύσματος ανήκει, δημιουργείτε ο ανάλογος ΧΑ, σύμφωνα με τη λογική ανάλυση διανυσμάτων και τις σχέσεις:



$$distance_{first\ quarter} = \sqrt{(\cos \theta * \begin{bmatrix} 0 & 1 & 2 & \dots & M \\ 0 & 1 & 2 & \dots & M \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 1 & 2 & \dots & M \end{bmatrix})^2 + (\sin \theta * \begin{bmatrix} N & N & N & \dots & N \\ \dots & \dots & \dots & \dots & \dots \\ 2 & 2 & 2 & \dots & 2 \\ 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix})^2}$$

$$distance_{second\ quarter} = \sqrt{(\cos \theta * \begin{bmatrix} M & \dots & 2 & 1 & 0 \\ M & \dots & 2 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ M & \dots & 2 & 1 & 0 \end{bmatrix})^2 + (\sin \theta * \begin{bmatrix} N & N & N & \dots & N \\ \dots & \dots & \dots & \dots & \dots \\ 2 & 2 & 2 & \dots & 2 \\ 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix})^2}$$

$$distance_{third\ quarter} = \sqrt{(\cos \theta * \begin{bmatrix} M & \dots & 2 & 1 & 0 \\ M & \dots & 2 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ M & \dots & 2 & 1 & 0 \end{bmatrix})^2 + (\sin \theta * \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 1 \\ 2 & 2 & 2 & \dots & 2 \\ \dots & \dots & \dots & \dots & \dots \\ N & N & N & \dots & N \end{bmatrix})^2}$$

$$distance_{fourth\ quarter} = \sqrt{(\cos \theta * \begin{bmatrix} 0 & 1 & 2 & \dots & M \\ 0 & 1 & 2 & \dots & M \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 1 & 2 & \dots & M \end{bmatrix})^2 + (\sin \theta * \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 1 \\ 2 & 2 & 2 & \dots & 2 \\ \dots & \dots & \dots & \dots & \dots \\ N & N & N & \dots & N \end{bmatrix})^2}$$

Στη συνέχεια, για να σταθμίσουμε το χάρτη σε σχέση με τη τιμή που καταλαμβάνει σε αυτόν το pixel με την ελάχιστη τιμή βάθους της περιοχής των έγκυρων που μας αφορά, αφαιρούμε από όλες τις τιμές του χάρτη εκείνη του παραπάνω pixel ως εξής:

$$distance_{theta} = distance_{theta} - distance_{theta}[minimum_depth_position]$$

Η διαδικασία ως το σημείο αυτό είναι πανομοιότυπη, τόσο για μία ανακλαστική επιφάνεια, όσο και για οποιαδήποτε γειτονική περιοχή ενός χρωματικού τμήματος μίας ΕΠΑΜ. Για την τελική διόρθωση, όσο αφορά την πρώτη, δημιουργούμε στις νέες τιμές τη βαθμιαία μεταβολή των βαθών, βάση του μέτρου του διανύσματος κατεύθυνσης βάθους:

$$depth_values[depth_values = 0] = minimum_depth + distance_{theta}[depth_values = 0] * |region_gradient|$$

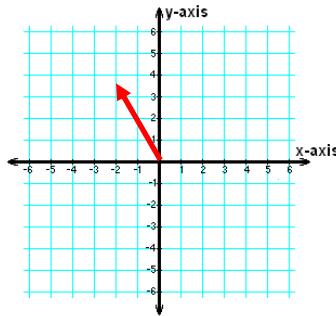
Η λογική είναι παρόμοια και για τις περιπτώσεις διόρθωσης με βάση την RGB πληροφορία, με την κάθε έγκυρη περιοχή να συνεισφέρει στον υπολογισμό των νέων τιμών ανάλογα με τον αριθμό των pixels που την απαρτίζουν, όπως φαίνεται στη σχέση που ακολουθεί:

$$depth[depth = 0] = \sum_i^{regions > min_area} \frac{region_pixels_i}{total_pixels} * (dmin_i + distance_{theta_i}[depth = 0] * |region_gradient_i|)$$

με το *min_area* να αντιπροσωπεύει ένα πολύ χαμηλό όριο, κάτω από το οποίο θεωρούμε ότι η επίδραση της περιοχής είναι μηδαμινή και το *total_pixels*, το συνολικό αριθμό των εμπλεκόμενων στους υπολογισμούς pixels.



Ως παράδειγμα υπολογισμών, παραθέτουμε τον υπολογισμό του χάρτη απόστασης για διάνυσμα γωνίας 120° , με το υπόλοιπο της διαδικασίας να θεωρείται τετριμμένο.

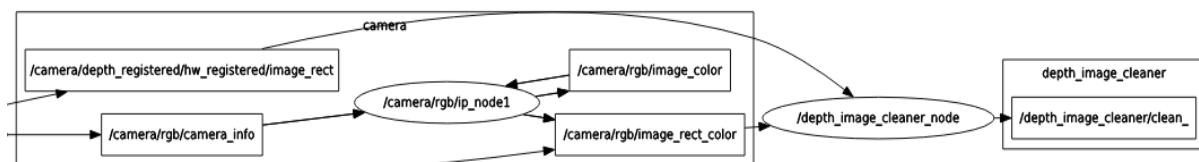


$$\begin{aligned}
 distance_{120^\circ} &= \sqrt{(\cos(120^\circ) * \begin{bmatrix} M & \dots & 2 & 1 & 0 \\ M & \dots & 2 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ M & \dots & 2 & 1 & 0 \end{bmatrix})^2 + (\sin(120^\circ) * \begin{bmatrix} N & N & N & \dots & N \\ \dots & \dots & \dots & \dots & \dots \\ 2 & 2 & 2 & \dots & 2 \\ 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix})^2} \\
 &= \sqrt{\left(\begin{bmatrix} \dots & \dots & -1 & -\frac{1}{2} & 0 \\ \dots & \dots & -1 & -\frac{1}{2} & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & -1 & -\frac{1}{2} & 0 \end{bmatrix}\right)^2 + \left(\begin{bmatrix} \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \sqrt{3} & \sqrt{3} & \sqrt{3} & \dots & \sqrt{3} \\ \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} & \dots & \frac{\sqrt{3}}{2} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}\right)^2} \\
 &= \sqrt{\begin{bmatrix} \dots & \dots & 1 & \frac{1}{4} & 0 \\ \dots & \dots & 1 & \frac{1}{4} & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & 1 & \frac{1}{4} & 0 \end{bmatrix} + \begin{bmatrix} \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 3 & 3 & 3 & \dots & 3 \\ \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \dots & \frac{3}{4} \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}} \\
 &= \sqrt{\begin{bmatrix} \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & 4 & \frac{5}{4} & 3 \\ \dots & \dots & \frac{7}{4} & 1 & \frac{3}{4} \\ \dots & \dots & 1 & \frac{1}{4} & 0 \end{bmatrix}} = \left(\begin{bmatrix} \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & 2 & \frac{\sqrt{5}}{2} & \sqrt{3} \\ \dots & \dots & \frac{\sqrt{7}}{2} & 1 & \frac{\sqrt{3}}{2} \\ \dots & \dots & 1 & \frac{1}{2} & 0 \end{bmatrix}\right)
 \end{aligned}$$



5. Περιγραφή Συστήματος

Στο κεφάλαιο αυτό, θα περιγράψουμε αναλυτικά το σύστημα λογισμικού που υλοποιήθηκε για την αφαίρεση θορύβου σκιάς από εικόνα βάθους, όπως αυτή παρέχεται από τον αντίστοιχο αισθητήρα μιας κάμερας βάθους. Συνοπτικά, το συνολικό σύστημα μπορεί να αναπαρασταθεί ως ένας κόμβος εντός του προγραμματισμού περιβάλλοντος ROS, ο οποίος επικοινωνεί με το λογισμικό οδήγησης της κάμερας για τη λήψη των δεδομένων και κατασκευάζει έναν ορθό χάρτη βάθους για το συγκεκριμένο ζεύγος εικόνων και το επιστρέφει στο γράφο ως ένα νέο μήνυμα εικόνας. Στην παρακάτω εικόνα, βλέπουμε τον γράφο της διασύνδεσης του πακέτου λογισμικού μας στο πακέτο οδήγησης της κάμερας Kinect, με σημειωμένα τα μηνύματα εικόνας που συμμετέχουν στην επικοινωνία.



Εικόνα 5.0.1: Γράφος Διασύνδεσης Μηνυμάτων Εικόνας Πακέτου Αφαίρεσης Θορύβου

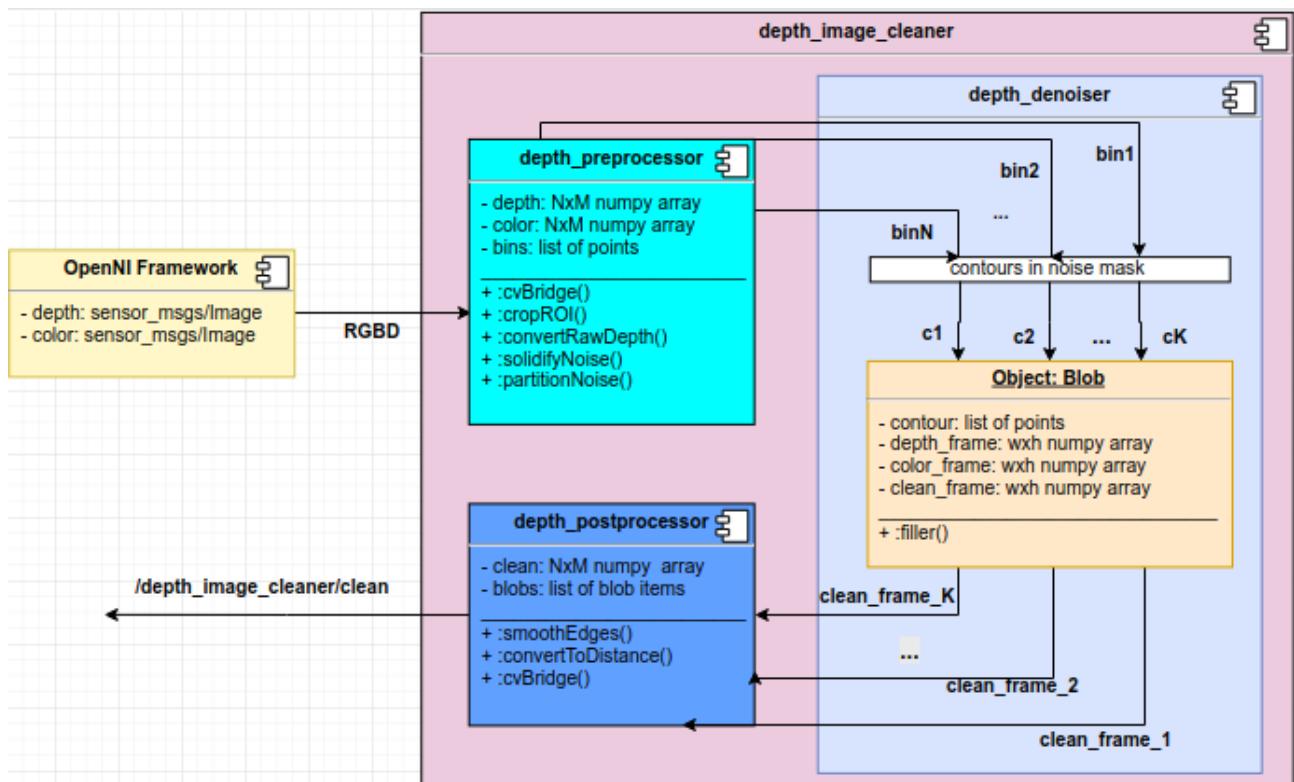
Μια αρχική διάκριση του λογισμικού, η οποία βασίζεται στο είδος των βασικών δομών δεδομένων που επεξεργάζεται το σύστημα, το διαχωρίζει σε τρία επίπεδα:

1. **Επίπεδο Επικοινωνίας Μηνυμάτων:** Πρόκειται ουσιαστικά για το λογισμικό το οποίο συνδέει το module επεξεργασίας εικόνων βάθους (*depth_image_cleaner*) με το ROS. Στο επίπεδο αυτό, ουσιαστικά το σύστημα περιμένει τη δημοσίευση νέων μηνυμάτων εικόνας από τον κόμβο οδήγησης της κάμερας, συγχρονίζει αντίστοιχα μηνύματα μεταξύ τους για τη δημιουργία του ζεύγους, υλοποιεί την διασύνδεση με το εσωτερικό επίπεδο για την αποστολή και την αποδοχή των δεδομένων και δημοσιεύει την διορθωμένη εικόνα βάθους για χρήση από εξωτερικά πακέτα του ROS. Το αντίστοιχο λογισμικό αναπτύχθηκε με την χρήση του API διασύνδεσης *rospy* (βλ. Παράγραφο 3.4.1) και η δομή δεδομένων αναφοράς είναι τα *sensor_msgs/Image* μηνύματα.
2. **Επίπεδο Επεξεργασίας Ζεύγους Εικόνων:** Στο επίπεδο αυτό, το σύστημα έχει δεχθεί ένα ζεύγος εικόνων, εφαρμόζει έναν αλγόριθμο προεπεξεργασίας στην εικόνα βάθους για την τροποποίηση του σε μορφή που είναι κατάλληλη για επεξεργασία από τον υλοποιημένο αλγόριθμο του κεφαλαίου 4 (δημιουργία αντικειμένων *EPAM*) και τέλος συνδυάζει τα διαδοχικά αποτελέσματα του αλγορίθμου για τη σύνθεση της συνολικής διορθωμένης εικόνας βάθους. Το λογισμικό εδώ επεξεργάζεται 640 x 480 εικόνες ως δομή δεδομένων αναφοράς.
3. **Επίπεδο Επεξεργασίας *EPAM*:** Για κάθε ζεύγος εικόνων, από το προηγούμενο επίπεδο έχουν δημιουργηθεί έστω *K* συνολικά αντικείμενα *EPAM*, κάθε ένα από τα



οποία διαθέτει γεωμετρικές πληροφορίες για την κάθε τέτοια περιοχή, καθώς και τα δύο πλαίσια χρώματος W_c και βάθους W_d της *ΕΠΑΜ* που δέχεται ο αλγόριθμος μας ως είσοδο. Στο επίπεδο αυτό, εφαρμόζεται η διαδικασία που περιγράψαμε για την εξαγωγή ενός νέου πλασίου W_e διορθωμένου βάθους. Τα K πλαίσια αυτά συνθέτουν το συνολικό χάρτη βάθους στο παραπάνω επίπεδο. Το επίπεδο αυτό επεξεργάζεται αντικείμενα της κλάσης *ΕΠΑΜ*, την οποία έχουμε αναπτύξει εμείς για την αποδοτικότερη και πιο οργανωμένη εφαρμογή του αλγορίθμου.

Το λογισμικό των δυο εσωτερικών επιπέδων αναπτύχθηκε όπως αναφέρουμε αναλυτικά στο κεφάλαιο 3 σε γλώσσα προγραμματισμού *Python* με χρήση της βιβλιοθήκης *OpenCV 3*. Μια αφαιρετική αναπαράσταση του συστήματος μας αυτού (*depth_image_cleaner*) δίνεται στο παρακάτω διάγραμμα στελεχών.



Εικόνα 5.0.2: Διάγραμμα Στελεχών Συστήματος Αφαίρεσης Θορύβου Εικόνας Βάθους

Όπως βλέπουμε στο παραπάνω διάγραμμα, το συνολικό σύστημα μπορεί να διακριθεί στα εξής τρία υπο-συστήματα λογισμικού, τα οποία περιγράφουμε σε λεπτομέρεια στις επόμενες παραγράφους:

1. **Σύστημα Προεπεξεργασίας Βάθους:** Το σύστημα αυτό δέχεται τα συγχρονισμένα μηνήματα χρώματος – βάθους, τα μετατρέπει σε συμβατού τύπου εικόνες, αφαιρεί τον περιπτώ θόρυβο που οφείλεται στην καταχώρηση του βάθους, επιχειρεί να τμηματοποιήσει σε κομμάτια μεγάλες περιοχές θορύβου και δημιουργεί διαφορετικές ομάδες περιοχών θορύβου, οι οποίες θα δοθούν ως είσοδο στο επόμενο σύστημα,



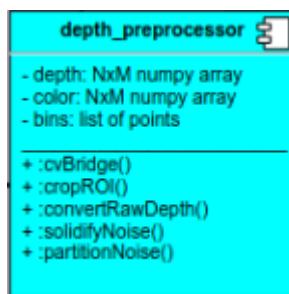
έτσι ώστε η εξουδετέρωση του θορύβου σε αυτές να επιτευχθεί παράλληλα.

2. **Σύστημα Αφαίρεσης Θορύβου:** Το σύστημα αυτό δέχεται τις ομάδες θορύβου και μοντελοποιεί κάθε *EPAM* ως ένα αντικείμενο στην κλάση *Blob*. Οι ιδιότητες και οι μέθοδοι της κλάσης αυτής, αποτελούν το σύνολο των μονάδων λογισμικού που υλοποιούν όλες τις επιμέρους διαδικασίες που περιγράφτηκαν στο κεφάλαιο 4, για την αφαίρεση του θορύβου σε ένα πλαίσιο γύρω από την *EPAM*.
3. **Σύστημα Μετεπεξεργασίας Βάθους:** Το σύστημα αυτό, έχοντας ως δεδομένο το σύνολο των αντικειμένων *EPAM* για κάθε ζεύγος εικόνων, συνθέτει τα διαφορετικά πλαίσια διορθωμένου βάθους για να κατασκευάσει τη συνολική εικόνα, την οποία, αφού προσαρμόσει με μια πολύ βασική εξομάλυνση στις ακμές των *EPAM*, την επαναφέρει στη μορφή μηνύματος δεδομένων βάθους και τη δημοσιεύει πίσω στο γράφο.

5.1 Σύστημα Προεπεξεργασίας Βάθους

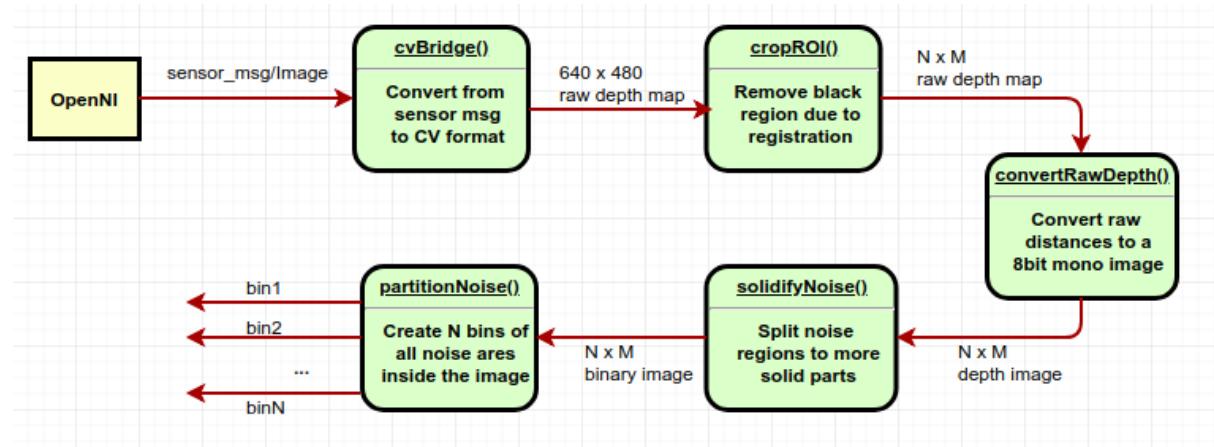
Το σύστημα προεπεξεργασίας βάθους αποτελεί μια κλάση του module *depth_image_cleaner*, η οποία είναι υπεύθυνη για την εφαρμογή κατάλληλων μετατροπών επί της δοθείσας εικόνας βάθους και τη δημιουργία διαφορετικών νημάτων για την εφαρμογή του αλγορίθμου αφαίρεσης θορύβου σε διαφορετικές περιοχές επί της αρχικής εικόνα. Μια συμβολική αναπαράσταση της κλάσης ακολουθεί στην εικόνα 5.1.1, ενώ το διάγραμμα ροής της εικόνας 5.1.2 παρουσιάζει τον αλγόριθμο προεπεξεργασίας που εφαρμόζεται στην εικόνα.

Ιδιότητες της κλάσης αυτής αποτελούν εκτός από τις δύο εικόνες του ζεύγους, την εικόνα βάθους στην οποία εφαρμόζεται ο αλγόριθμος προεπεξεργασίας και την έγχρωμη εικόνα, η οποία απλά μεταφέρεται στο επόμενο σύστημα για την αξιοποίησή της στην ομαδοποίηση *EPAM*, μια λίστα από περιγράμματα, κάθε ένα από τα οποία αντιπροσωπεύει τις συντεταγμένες των σημείων του συνόρου μιας *EPAM* στην εικόνα.



Εικόνα 5.1.1: Σύστημα Προεπεξεργασίας Βάθους





Εικόνα 5.1.2: Αλγόριθμος Προεπεξεργασίας Βάθους

Βήμα προς βήμα, η εφαρμογή του αλγορίθμου αναλύεται παίρνοντας με τη σειρά τις μεθόδους της κλάσης και την επεξεργασία που ασκούν στην εικόνα βάθους:

1. `cvBridge()`: Πρόκειται για τη διαδικασία μετατροπής των RGBD δεδομένων από τη μορφή μηνυμάτων αισθητήρα `sensor_msgs/Image` σε συμβατή στη βιβλιοθήκη της OpenCV `numpy image`. Η μετατροπή αυτή δημιουργεί μια 640×480 8-bit BGR `numpy` εικόνα από το μήνυμα εικόνας χρώματος και έναν 640×480 `float32` `numpy` πίνακα με τιμές βάθους σε χιλιοστά από το μήνυμα βάθους.
2. `CropROI()`: Η μέθοδος αυτή είναι υπεύθυνη για την αφαίρεση της μαύρης περιοχής κάδρου που υπάρχει στην εικόνα βάθους και η οποία οφείλεται στην καταχώρηση που έχει γίνει ήδη από το `OpenNI` πακέτο του χάρτη βάθους επί της εικόνας βάθους. Για το ίδιο καλιμπράρισμα των αισθητήρων μιας κάμερας (βλ. Παράγραφο 3.3.2), η καταχώρηση βάθους πάνω στην εικόνα χρώματος δημιουργεί το ίδιο κάδρο για την ίδια κάμερα. Ουσιαστικά, η μέθοδος αναγνωρίζει το είδος της κάμερας, εφαρμόζει το αντίστοιχο κόψιμο και εξάγει το εσωτερικό της εικόνας βάθους, που περιέχει την έγκυρη απεικόνιση της σκηνής. Το ίδιο εφαρμόζεται και στην εικόνα χρώματος του ζεύγους.
3. `convertRawDepth()`: Η μέθοδος αυτή μετατρέπει τα ωμά δεδομένα απόστασης του χάρτη βάθους σε μονοχρωματική 8-bit εικόνα, μορφή στην οποία πρέπει να βρίσκεται ο χάρτης για να εφαρμοστεί ο αλγόριθμος εξουδετέρωσης θορύβου. Το σύστημα εντοπίζει την ελάχιστη και τη μέγιστη ένδειξη βάθους που έχει γίνει για το συγκεκριμένο δείγμα και μετατρέπει κάθε μέτρηση βάθους σε μια από τις αντίστοιχες 255 αποχρώσεις της μονοχρωματικής εικόνας.
4. `solidifyNoise()`: Η μέθοδος αυτή επιδιώκει να εντοπίσει εντός της μάσκας θορύβου ΕΠΑΜ οι οποίες δεν είναι συμπαγείς. Με τον όρο συμπαγής, εννοούμε ΕΠΑΜ που η οριζόντια και κατακόρυφη διάσχιση της κλειστής καμπύλης του συνόρου της, δεν την αφήνει εκτός σε κανένα σημείο. Η περίπτωση αυτή εντοπίζεται με τον έλεγχο του εμβαδού του εγγεγραμμένου ορθογώνιου παραλληλόγραμμου, όταν αυτό υπερβαίνει κατά πολύ το εμβαδό της ίδιας της ΕΠΑΜ (του λάχιστον μια τάξη

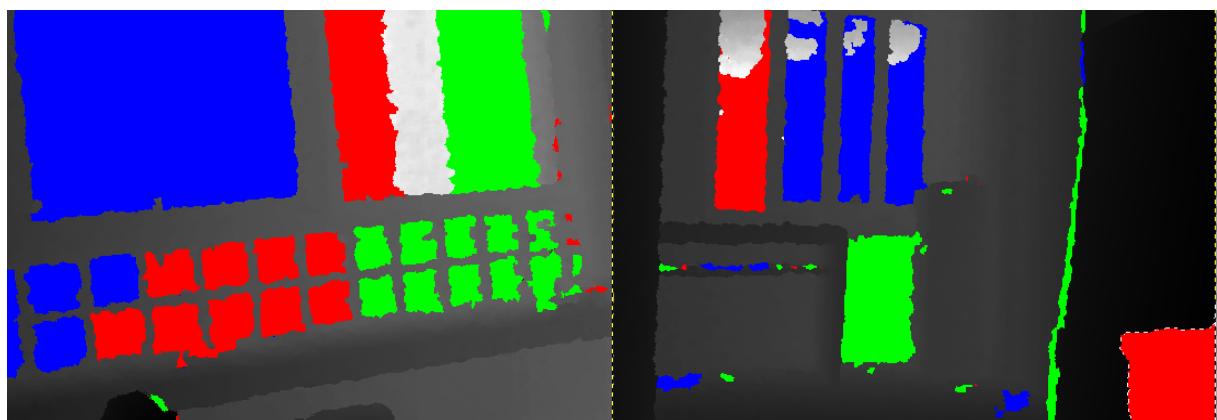


μεγέθους). Οι μη συμπαγείς ΕΠΑΜ που εντοπίζονται χωρίζονται σε τέσσερα κομμάτια, τα τέσσερα ορθογώνια που συνθέτουν το μεγάλο εγγεγραμένο. Ο αριθμός των σημείων της ΕΠΑΜ που υπάρχει σε κάθε συνδυασμό μικρών τετραγώνων εξάγει την αντίστοιχη τμηματοποίηση που οφείλει να γίνει σε αυτήν. Έτσι μεγάλες ΕΠΑΜ, οι οποίες συνήθως εμπεριέχουν συγχωνευμένο θόρυβο, διασπώνται σε μικρότερα κομμάτια, κάτι που, εκτός από καλύτερη τμηματοποίηση τοπικά σε κάθε κομμάτι και ΕΠΑΜ, σε αντίθεση με την εφαρμογή σε ολόκληρη, οδηγεί και σε ταχύτερη συνολικά τμηματοποίηση. Παραδείγματα τέτοιας διάσπασης θορύβου φαίνεται στα παρακάτω παραδείγματα.



Εικόνα 5.1.3: Αποτελέσματα της εφαρμογής της διαδικασίας `solidify_noise()`

5. `partitionNoise()`: Η μέθοδος αυτή επιδιώκει να διασπάσει το σύνολο των ΕΠΑΜ σε N ομάδες, προκειμένου οι ομάδες αυτές να μοιραστούν σε διαφορετικά νήματα επεξεργασίας και η διόρθωση θορύβου να πραγματοποιείται παράλληλα στην εικόνα, χρησιμοποιώντας N νήματα. Οι ΕΠΑΜ ταξινομούνται με βάση το εμβαδόν τους και κάθε ομάδα αναλαμβάνει αντίστοιχες περιοχές, έτσι ώστε το συνολικό εμβαδόν όλων των ΕΠΑΜ κάθε ομάδας να είναι περίπου το ίδιο.

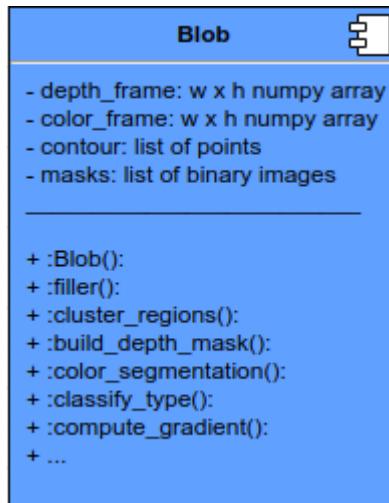


Εικόνα 5.1.3: Αποτελέσματα της εφαρμογής της διαδικασίας `partition_noise()`

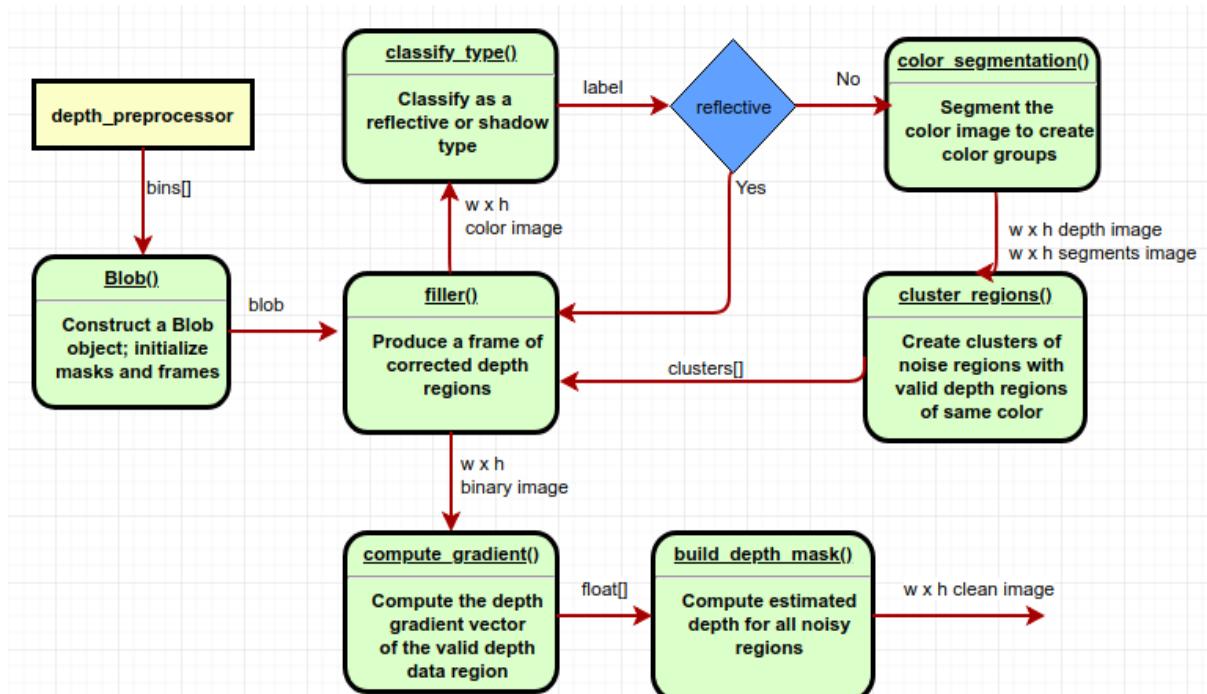


5.2 Σύστημα Αφαίρεσης Θορύβου

Το σύστημα αφαίρεσης θορύβου, αφού δεχτεί τις ομάδες διαφορετικών περιγραμμάτων από ΕΠΑΜ, δημιουργεί αντικείμενα στην κλάση *Blob*. Η κλάση αυτή, διαθέτει ως ιδιότητες όλες τις γεωμετρικές πληροφορίες της τρύπας, τόσο ως λίστες με σημεία, όσο και ένα σύνολο από μάσκες για όλα τα διαφορετικά χαρακτηριστικά της ΕΠΑΜ (ελάχιστο στραμμένο παραλληλόγραμμο, επεκταμένο ορθογώνιο, μάσκα θορύβου κλπ.), όπως περιγράφονται στην παράγραφο 4.1, τα $w \times h$ πλαίσια βάθους και χρώματος που την εσωκλείνουν, καθώς και τις $N \times M$ αρχικές εικόνες.



Εικόνα 5.2.1: Κλάση Blob – Βασικές Μέθοδοι



Εικόνα 5.2.2: Παραγωγή Πλαισίου Διορθωμένου Βάθους μιας ΕΠΑΜ



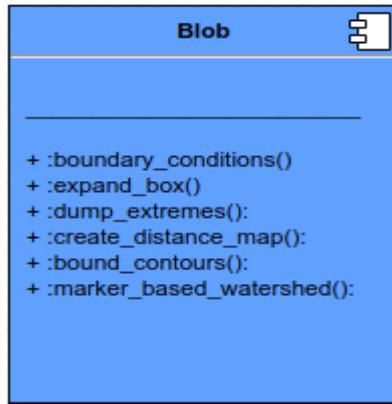
Όλες οι διαδικασίες που φαίνονται στην παραπάνω εικόνα είναι μέθοδοι της κλάσης blob, οπότε διαθέτουν όλες τις ιδιότητες της κλάσης. Στις ακμές που ενώνουν τις διαδικασίες, δηλώνουμε μόνο εκείνα τα στοιχεία που αποτελούν εισόδους – εξόδους στη συνάρτηση, αλλά κάθε μια έχει πρόσβαση στα γεωμετρικά χαρακτηριστικά, τις μάσκες και τα πλαίσια της ΕΠΑΜ.

Αναλυτικά, περιγράφουμε τη λειτουργία κάθε μεθόδου παρακάτω, κάνοντας αναφορές στις αντίστοιχες διαδικασίες που περιγράψαμε στο προηγούμενο κεφάλαιο, καθώς αυτόν τον αλγόριθμο υλοποιεί το σύστημα για κάθε ΕΠΑΜ:

1. filler(): Η μέθοδος αυτή, δεδομένου του αντικειμένου ΕΠΑΜ, εξάγει το τελικό πλαίσιο διορθωμένου βάθους. Είναι δηλαδή υπεύθυνη για το σύνολο του αλγορίθμου εξουδετέρωσης θορύβου, πραγματοποιώντας τους κατάλληλους ελέγχους και καλώντας την αντίστοιχη διαδικασία για όλα τα βήματα από την ταξινόμηση της πηγής του θορύβου ως το βάψιμο περιοχών.
2. color segmentation(): Στην μέθοδο αυτή δεδομένου του πλαισίου χρώματος μιας ΕΠΑΜ, πραγματοποιείται η κατάλληλη προεπεξεργασία και υλοποιείται η τμηματοποίηση της εικόνας σε ομάδες χρώματος, όπως εξηγείται αναλυτικά στην Παράγραφο 4.2.
3. cluster regions(): Η μέθοδος αυτή είναι υπεύθυνη για την ομαδοποίηση περιοχών ΕΠΑΜ με αντίστοιχες περιοχές βάθους ίδιου τμηματικού χρώματος, τη διάσπαση των περιοχών αυτών σε συμπαγή κομμάτια και την εξαγωγή της MEB για κάθε τέτοιο κομμάτι. Η διαδικασία αυτή υλοποιείται όπως περιγράφεται στην Παράγραφο 4.3.
4. classify type(): Σε αυτή τη μέθοδο, υλοποιείται η διαδικασία εντοπισμού ανακλαστικών επιφανειών της Παραγράφου 4.3 και πραγματοποιούνται κάποιοι ακόμα έλεγχοι για το είδος της ΕΠΑΜ, οι οποίοι επηρεάζουν τον τρόπο υπολογισμού του ΔΚΒ ή της εξαγωγής του πλαισίου διορθωμένου βάθους της. Οι έλεγχοι αυτοί και οι αντίστοιχες μεταβολές στο βάψιμο είναι αυτοί που αναλύονται στην Παράγραφο 4.5.
5. compute gradient(): Η μέθοδος αυτή εφαρμόζει τους μαθηματικούς τύπους της Παραγράφου 4.4 για τον υπολογισμό του ΔΚΒ της περιοχής έγκυρης βάθους από το βάθος της οποίας θα διορθωθεί η ΕΠΑΜ.
6. build depth mask(): Η μέθοδος αυτή υλοποιεί τη διαδικασία που περιγράφεται στην Παράγραφο 4.6 με τον υπολογισμό του χάρτη απόστασης και μεταφοράς της πληροφορίας βάθους από το ΔΚΒ της έγκυρης περιοχής στην περιοχή του τμήματος ΕΠΑΜ. Η έξοδος της διαδικασίας αυτής είναι μια ένα διορθωμένο πλαίσιο βάθους από το οποίο το σύστημα μετεπεξεργασίας βάθους θα συνθέσει τη συνολική διορθωμένη εικόνα.

Ακόμη, η κλάση διαθέτει μερικές ακόμα μεθόδους που υλοποιούν επιμέρους υπολογισμούς ή αλγορίθμους που χρειάζονται σε διάφορα σημεία της ροής του προγράμματος. Οι μέθοδοι αυτές φαίνονται στην εικόνα και περιγράφονται συνοπτικά παρακάτω:





Εικόνα 5.2.3: Κλάση Blob – Περισσότερες Μέθοδοι

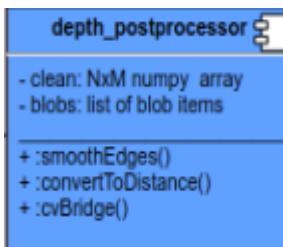
1. boundary conditions(): Η μέθοδος αυτή υπολογίζει και αποδίδει το επεκταμένο δεδομένο αριθμό pixels ένα μη περιστραμμένο ορθογώνιο Box (βλ. Παράγραφο 4.1.1)
2. expand_box(): Η συγκεκριμένη, σε αναλογία με τη προηγούμενη, επιστρέφει το επεκταμένο παραλληλόγραμμο ελάχιστης έκτασης Rect, που περιβάλει μία ΕΠΑΜ (βλ. Παράγραφο 4.1.1)
3. dump extremes(): Η ΑΕΤ που περιγράφεται στην παράγραφο 4.4.2 υλοποιείται μέσω της μεθόδου αυτής, αφαιρώντας, όπως αναφέραμε, πιθανές μικρές ομάδες pixels με εξωκείμενες τιμές βάθους
4. create distance map(): Η επιλογή των πινάκων σύνθεσης του ΧΑ που περιγράφεται στην 4.6 πραγματοποιείται με τη βοήθεια της τελευταίας, με την επιλογή των δύο αναγκαίων πινάκων να γίνεται με βάση το τεταρτιμόριο του ΔΚΒ και τη λογική ανάλυσης διανυσμάτων
5. bound contours(): Η μέθοδος αυτή χρησιμοποιείται για το κλείσιμο των ακμών που έχουν ανιχνευθεί και ακουμπούν στα όρια του μη περιστραμμένου ορθογωνίου που περιβάλλει την ΕΠΑΜ, ώστε να δημιουργηθεί η μάσκα του αντικειμένου που πιθανώς αποτελεί τμήμα ανακλαστικής επιφάνειας (βλ. Παράγραφο 4.3.2.1)
6. marker based watershed(): Η τελευταία αυτή μέθοδος εφαρμόζει τη λογική που περιγράφεται στην παράγραφο , για τη χρήση του αλγορίθμου watershed για την εύρεση ανακλαστικών επιφανειών στα όρια της εικόνας (βλ. Παράγραφο 4.3.1)

5.3 Σύστημα Μετεπεξεργασίας Βάθους

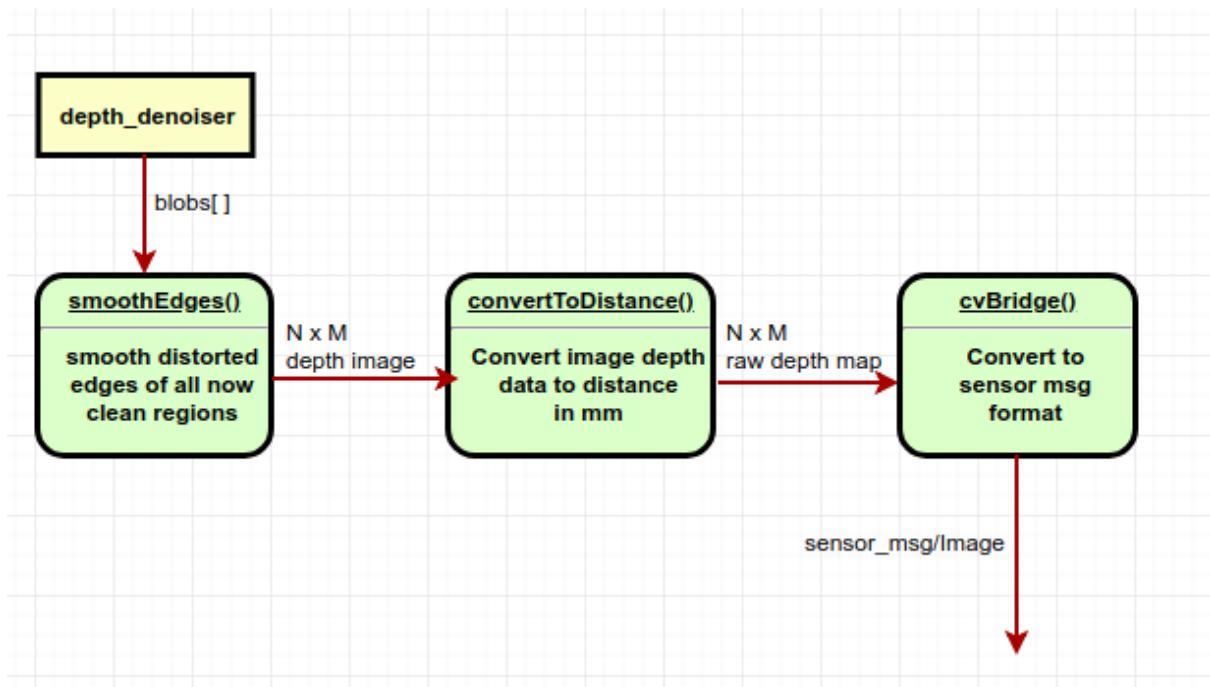


Το σύστημα μετεπεξεργασίας βάθους αποτελεί επίσης μια κλάση του module `depth_image_cleaner`, η οποία είναι υπεύθυνη για επαναφορά των δεδομένων σε μορφή μηνυμάτων αισθητήρα, αφού πρώτα εφαρμοστεί μια βασική απόπειρα εξομάλυνσης των ακμών της εικόνας βάθους. Μια συμβολική αναπαράσταση της κλάσης ακολουθεί στην εικόνα 5.3.1, ενώ το διάγραμμα ροής της εικόνας 5.3.2 παρουσιάζει τον αλγόριθμο που εφαρμόζεται στην εικόνα.

Ιδιότητες της κλάσης αυτής αποτελούν μια λίστα με το σύνολο των *ΕΠΑΜ* αντικειμένων που υπάρχουν και έχουν διορθωθεί στην εικόνα και η συνολική καθαρή εικόνα. Εξάγοντας από το κάθε αντικείμενο το πλαίσιο διορθωμένου βάθους, μετά την εφαρμογή της εξομάλυνσης, στην καθαρή εικόνα αντιγράφεται η αυθεντική εικόνα βάθους και στο περιεχόμενο κάθε *ΕΠΑΜ* αντιγράφεται το περιεχόμενο βάθους της αντίστοιχης περιοχής στο ανάλογο πλαίσιο διορθωμένου βάθους του αντικειμένου.



Εικόνα 5.3.1: Σύστημα Μετεπεξεργασίας Βάθους



Εικόνα 5.3.2: Αλγόριθμος Μετεπεξεργασίας Βάθους

Αντίστοιχα με το σύστημα προεπεξεργασίας, οι μέθοδοι της κλάσης αυτής είναι:



1. `smoothEdges()`: Η μέθοδος αυτή αποτελεί την εφαρμογή ενός απλού *median* φίλτρου με έναν 7x7 πυρήνα επί των πλαισίων διορθωμένου βάθους κάθε *EPAM*, μόνο για τις περιοχές εκείνες που αποτελούν το σύνορο της περιοχής θορύβου. Ουσιαστικά αυτή η διαδικασία εφαρμόζει μια βασική διόρθωση τοπικών ανωμαλιών που εμφανίζονται σε βαμμένες οριακές περιοχές, λόγω του σφάλματος που εισάγει η τρηματοποίηση χρώματος (π.χ. περιοχές εκτός του αντικειμένου στο οποίο ανήκει η σκιά που έχουν ομαδοποιηθεί στο παρασκήνιο). Παρακάτω βλέπουμε παραδείγματα εφαρμογής της εξομάλυνσης που εφαρμόζεται σε διαφορετικές περιπτώσεις ανωμαλιών.



Εικόνα 5.3.3: Εξομάλυνση Ακμών Καθαρής Εικόνας Βάθους
(a) Εικόνα Βάθους Χωρίς, (b) Εικόνα Βάθους με Εξομάλυνση Ακμών

2. `convertToDistance()`: Η διαδικασία αυτή αποτελεί την αντίστροφη από αυτή που εφαρμόστηκε στο σύστημα προεπεξεργασίας βάθους για την μετατροπή των ωμών δεδομένων απόστασης της κάμερας σε μονοχρωματική εικόνα. Εδώ, οι δύο οριακές τιμές μέτρησης του αισθητήρα χρησιμοποιούνται για την επαναφορά κάθε μονάδας απόχρωσης στην αντίστοιχη τιμή βάθους. Αυτό πραγματοποιείται έτσι ώστε ο δημοσιευτής μηνυμάτων του κόμβου να αποδίδει και ένα μήνυμα καθαρής εικόνας, που βρίσκεται στην ίδια μορφή με τα ωμά δεδομένα βάθους της.
3. `cvBridge()`: Εδώ πάλι πρόκειται για την μετατροπή των δύο εικόνων στη μορφή μηνυμάτων αισθητήρα του γράφου ROS, έτσι ώστε να μπορεί η διορθωμένη εικόνα βάθους να χρησιμοποιηθεί από κάποιο εξωτερικό σύστημα. Η διορθωμένη εικόνα βάθους δημοσιεύεται ως ένα `sensor_msgs/Image` μήνυμα.



6. Πειράματα

Στο κεφάλαιο αυτό, θα παρουσιάσουμε τα αποτελέσματα των διαφόρων πειραμάτων που διεξήχθησαν σε ζεύγη εικόνων βάθους και χρώματος, προκειμένου να μετρηθούν η ακρίβεια του αλγορίθμου που σχεδιάστηκε, καθώς και η απόδοση του τελικού συστήματος που υλοποιήθηκε. Αρχικά, περιγράφουμε τη διαδικασία λήψης των δεδομένων του σετ στο οποίο διεξήχθησαν τα πειράματα, αναλύουμε τους διάφορους συντελεστές που χρησιμοποιούμε για να μετρήσουμε τα αποτελέσματα των πειραμάτων και επιδεικνύουμε εικόνες με εξουδετερωμένο θόρυβο από χάρτες βάθους, προκειμένου να δοθεί μια σαφής εικόνα των αποτελεσμάτων. Τέλος, παρουσιάζουμε μια εφαρμογή που χρησιμοποιεί εικόνες βάθους στο περιβάλλον ROS (κατασκευή της δομής PointCloud2 από RGB-D δεδομένα) για να εξεταστεί στην πράξη η χρησιμότητα του συστήματός μας.

6.1 Λήψη Σετ Δεδομένων και Χαρακτηριστικά Πειραμάτων

Για τη δημιουργία του σετ, λήφθηκαν συνολικά 33 ζεύγη εικόνων από την κάμερα Kinect και 19 από την Xtion. Το σετ δηλαδή αποτελείται από συνολικά 52 δείγματα, κάθε ένα από τα οποία, σε ενδεδειγμένες χαρακτηριστικές περιπτώσεις διαφορετικού είδους θορύβου, εκτός από το ζεύγος εικόνων χρώματος – βάθους, διαθέτουν την αντίστοιχη *Εικόνα Αληθούς Βάθους (EAB)*, δηλαδή την εικόνα βάθους με ορθά διορθωμένο θόρυβο, καθώς και την *Ορθή Μάσκα Ανακλαστικών (OMA)*, δηλαδή μια μάσκα που εμπεριέχει όλες τις περιοχές θορύβου που οφείλονται σε ανακλαστικές επιφάνειες, προκειμένου να εξεταστεί ο αλγόριθμος εντοπισμού μας. Τα ζεύγη προέκυψαν από τη στιγμιαία σύλληψη κατά την οπτικοποίηση της ροής των εικόνων, όπως αυτές δημοσιεύονταν από τους αντίστοιχους οδηγητές του OpenNi framework. Ουσιαστικά, τα ζεύγη εικόνων βάθους και χρώματος είναι τα ωμά δεδομένα, όπως αυτά δίνονται από τους αισθητήρες της κάμερας, χωρίς κάποια προεπεξεργασία. Τα χαρακτηριστικά της μορφής των εικόνων αυτών είναι τα εξής:

- Οι έγχρωμες εικόνες είναι αποθηκευμένες ως 640 x 480 8-bit RGB εικόνες στο PNG format.
- Οι χάρτες βάθους είναι αποθηκευμένοι ως 640 x 480 8-bit μονοχρωματικές εικόνες στο PNG format.
- Οι δυο εικόνες είναι ήδη προ-αντιστοιχισμένες σύμφωνα με το depth registration module του OpenNi framework (βλ. Παράγραφο 3.4.3). Δηλαδή, τα pixels στις εικόνες βάθους και χρώματος είναι αντιστοιχισμένα ήδη 1:1.
- Δεν έχει χρησιμοποιηθεί κλίμακα κωδικοποίησης της απόστασης των διαφόρων pixels του χάρτη βάθους σε μονοχρωματική εικόνα, δηλαδή ένα pixel τιμής 255 στην εικόνα βάθους αντιστοιχεί στη μέγιστη απόσταση που μπορεί να καταγράψει ο αισθητήρας βάθους της κάμερας (π.χ. 3.6 - 4m για την Kinect), ενώ ένα pixel τιμής 1 αντιστοιχεί στην ελάχιστη απόσταση (0.4 - 0.8m για την Kinect). Ένα pixel με τιμή 0 αντιστοιχεί σε άκυρη μέτρηση – θόρυβο.

Για την εξαγωγή των δυο εικόνων αλήθειας του σετ, επιχειρήθηκαν διάφορες μέθοδοι



κατά τη διάρκεια της εκπόνησης της εργασίας. Αρχικά, επενδύθηκε χρόνος στην ιδέα του να χρησιμοποιηθεί κάποια άλλη κάμερα καλύτερης τεχνολογίας, που θα δώσει ορθότερους χάρτες βάθους για την απεικόνιση της ίδιας σκηνής, ή τουλάχιστον ενός σημαντικού κοινού μέρους σκηνών που αποτυπώνουν οι δυο κάμερες (βάθους και επιπλέον) όταν αυτές εφαρμόζονται κατακόρυφα.

Σε πρώτο χρόνο δοκιμάστηκε ως δεύτερη κάμερα η *Microsoft Kinect v.2*. τεχνολογίας *ToF*. Ωστόσο, πολύ γρήγορα διαπιστώθηκε ότι και οι χάρτες βάθους της κάμερας αυτής είναι θορυβώδεις για παρεμφερείς λόγους, και μάλιστα με τρόπο που οι δυο εικόνες βάθους δεν διέθεταν τις ίδιες περιοχές θορύβου, ούτε απαραίτητα αλληλεπικαλυπτόμενες, οπότε η σύγκριση τους θα μπορούσε να γίνει για πολύ μικρό κομμάτι της εικόνας και θεωρήθηκε μη επωφελής για τα πειράματα.

Για το λόγο αυτό, επιχειρήθηκε η χρήση μιας στερεοσκοπικής κάμερας, η *Bumblebee2 1394a*¹¹. Ωστόσο, κατά τη χρήση της κάμερας διαπιστώθηκε ότι υπάρχει έλλειψη συμβατότητας λογισμικού, καθώς υπάρχει μεν *ROS driver* για την κάμερα, αλλά δεν υπάρχει κάποιο υλοποιημένο πακέτο που να πραγματοποιεί διεπαφή ανάμεσα στα ωμά δεδομένα της κάμερας από το *ROS* και στο λογισμικό οδήγησης της κάμερας (*Triclops-3.4.0.8* σε λειτουργικό σύστημα *Microsoft Windows XP*). Εφαρμογές του προγράμματος οδήγησης της κάμερας στα *Windows* μπορούσαν να παρέχουν ένα αρχείο με συντεταγμένες *xyzrgb* δεδομένων στον χώρο (δομή παρόμοια με του *PointCloud2*), από την οποία μπορούσαμε να επιστρέψουμε στην εικόνα βάθους, αλλά διαπιστώθηκε ότι και τα αρχεία αυτά αποτελούταν συνολικά από σημεία που δεν αρκούσαν με τα ήδη υλοποιημένα εργαλεία (βιβλιοθήκη *PCL – Point Cloud Library*, πακέτο *pcl_ros*) να ανακατασκευάσουν *PointCloud2* με λιγότερο θόρυβο από την κάμερα βάθους. Προκειμένου να συγγραφεί το απαραίτητο λογισμικό διεπαφής, κάτι που αποκλίνει από το πυρήνα ενδιαφέροντος της εργασίας, αποφασίστηκε να χρησιμοποιηθεί άλλη μέθοδος εξαγωγής αλήθειας.

Εν τέλει, αποφασίστηκε πως είναι πιο ωφέλιμο να γίνει μια εμπειρική διόρθωση των εικόνων βάθους στο πρόγραμμα επεξεργασίας εικόνας *GIMP (GNU Image Manipulation Program)* χειροκίνητα από εμάς ως χρήστες. Ουσιαστικά, τα ζεύγη χρώματος - βάθους του σετ εισήχθησαν στο πρόγραμμα και εμείς προβάλλοντας τις ακμές των διαφόρων αντικειμένων της σκηνής όπως διακρίνονται πάνω στην *RGB* εικόνα πάνω στο χάρτη βάθους ως ακμές (πραγματοποιώντας με αυτόν τον τρόπο την 100% αληθή ομαδοποίηση που επιχειρούμε να κάνουμε με τον αλγόριθμο μας), διορθώνουμε κάθε περιοχή θορύβου με την σωστή πληροφορία βάθους της έγκυρης περιοχής, φτιάχνοντας ένα μοντέλο *Gaussian* κατανομής της γειτονιάς και μεταφέροντας το κατευθυνόμενα στη περιοχή θορύβου (το γνωστό στα διάφορα προγράμματα επεξεργασίας εικόνων εργαλείο της σφραγίδας), κάτι που επιχειρούμε με πιο αφελή τρόπο να πραγματοποιήσουμε εμείς στον αλγόριθμο μας, υπολογίζοντας το ΔΚΒ της έγκυρης περιοχής και βάφοντας με βάση αυτό το θόρυβο. Το dataset μπορεί να βρεθεί στη διεύθυνση (to be uploaded).

Στο σημείο αυτό παρουσιάζουμε ένα πίνακα με τις τεχνικές προδιαγραφές του υπολογιστή (*Toshiba Satellite L50-B-1K0*), στον οποίο διεξήχθησαν τα πειράματα και σημειώθηκαν τα αποτελέσματα των συντελεστών:

¹¹ <https://www.ptgrey.com/bumblebee2-firewire-stereo-vision-camera-systems>



Επεξεργαστής	<i>Intel Core i5-4120U</i>	Κάρτα Γραφικών	<i>AMD Radeon R7 M260</i>
Αριθμός Πυρήνων	2	Μνήμη Κάρτας Γραφικών	2 GB
Αριθμός Νημάτων	4	Μνήμη RAM	4 GB
Συχνότητα	1.70 GHz		

Πίνακας 6.1.1: Τεχνικές Προδιαγραφές Υπολογιστή Διεξαγωγής Πειραμάτων

6.2 Εντοπισμός Ανακλαστικών Επιφανειών

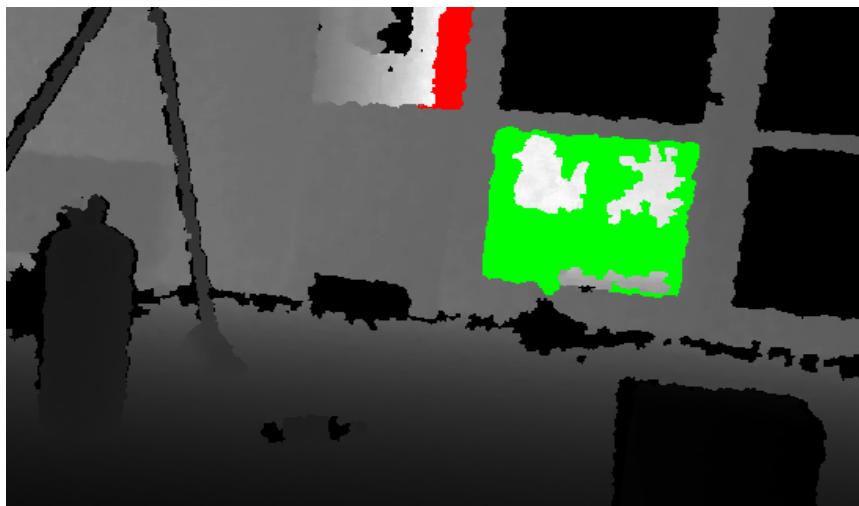
Στην παράγραφο αυτή, θα παρουσιάσουμε τα αριθμητικά αποτελέσματα καθώς και ενδεικτικά ζεύγη εικόνων σχετικά με την αναγνώριση ανακλαστικών επιφανειών στις εικόνες του dataset που δημιουργήσαμε για τις δύο κάμερες, Kinect και Xtion.

6.2.1 Πίνακας Αποτελεσμάτων Εντοπισμού Ανακλαστικών Επιφανειών

Όπως αναφέραμε και στην περιγραφή των διαδικασιών στο κεφάλαιο 4, η προσέγγισή μας περιλαμβάνει ένα συνδυασμό απομόνωσης μονοχρωματικών επιφανειών σε συνδυασμό με μία σειρά περιορισμών για τη μορφολογία του περιγράμματος μίας *ΕΠΑΜ*, καθώς η μορφολογία του τελευταίου πρέπει να την καθιστά συμπαγή και σχετικά καθορισμένου σχήματος (ορθογώνιο, τετράγωνο, έλλειψη, κύκλος). Ως αποτέλεσμα, πέρα από τις τελευταίες, είναι δυνατό να αναγνωρισθούν ως ανακλαστικές επιφάνειες μικρά σε σχέση με το πραγματικό μέγεθος τμήματα αυτής, καθώς η υπόλοιπη τυγχάνει να έχει αποκτήσει έγκυρη πληροφορία λόγω αντικειμένων που βρίσκονται κοντά και πίσω από αυτήν. Το φαινόμενο αυτό εμφανίζεται κυριών σε τζάμια ή καθρέφτες, όπως φαίνεται και στην παρακάτω εικόνα. Για τις δικές μας μετρήσεις ακρίβειας, θεωρούμε ότι ένα από τα παραπάνω τμήματα θα θεωρείται κομμάτι ανακλαστικής επιφάνειας και θα πρέπει να αναγνωριστεί ως τέτοιο, όταν η *ΕΠΑΜ* του αντιστοιχεί τουλάχιστον στο 40% της έκτασής της.

Όσον αφορά τα αριθμητικά αποτελέσματα, χρησιμοποιήσαμε τρεις κλασσικές μετρικές που εμφανίζονται σε προβλήματα δυαδικής αξιολόγησης, όπως το δικό μας, τις precision, recall και f – measure. Προτού προχωρήσουμε στην περιγραφή τους, θα επισημάνουμε μερικούς αρκετά γνωστούς όρους που αφορούν τον υπολογισμό τους. Συγκεκριμένα, ως true positives, αναφέρονται εκείνες οι *ΕΠΑΜ* που κατατάσσονται ως ανακλαστικές ή απορροφητικές επιφάνειες και αναγνωρίζονται ως τέτοιες. Αντίθετα, με τον όρο false negatives, προσδιορίζονται εκείνες που, ενώ είναι τέτοιες επιφάνειες, δεν κατέστη δυνατό να αναγνωριστούν. Επιπλέον, τις false positives θα αποτελούν εκείνες οι *ΕΠΑΜ*, οι οποίες θα βαφούν ως ανακλαστικές επιφάνειες, ενώ στην πραγματικότητα δεν είναι. Για τις ανάγκες της παρούσας διπλωματικής, δημιουργούμε τον όρο partially false positives για να απαριθμήσουμε εκείνες τις *ΕΠΑΜ* που είναι false positives, αλλά αποτελούν μικρά τμήματα (μικρότερα του 40%) ανακλαστικών επιφανειών.





Εικόνα 6.2.1.1: Τμήματα ανακλαστικών επιφανειών που θα πρέπει (πράσινο) και δεν θα πρέπει (κόκκινο) να αναγνωρισθούν

Προχωρώντας στις μετρικές, αυτή του precision προσδιορίζει την ακρίβεια της διαδικασίας στην εύρεση όσο το δυνατόν περισσότερων ορθών (true positives) στο σύνολο των αποτελεσμάτων (true positives και false positives):

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Αντίστοιχα, με τη μετρική recall εκφράζουμε τη δυνατότητα της διαδικασίας να αναγνωρίσει όσο το δυνατόν περισσότερα (true positives) από τα υπάρχοντα επιθυμητά χαρακτηριστικά (true positives και false negatives):

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Το f – measure λειτουργεί και αυτό σαν ένα μέτρο ακρίβειας, συνδυάζοντας τα δύο προηγούμενα και δίνοντας εποπτεία ως προς την ικανότητα των διαδικασιών να αναγνωρίζουν τα ζητούμενα (recall), διατηρώντας ταυτόχρονα χαμηλό το ποσοστό των λανθασμένα ευρισκόμενων (precision):

$$F - Measure = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Τέλος, ορίζουμε και μία δική μας μετρική, την partially false measure, η οποία θα μας επιτρέψει, κυρίως στο dataset της Kinect, να αποτυπώσουμε το πόσες από την ευρισκόμενες ανακλαστικές είναι τμήματα αυτών τα οποία δεν ικανοποιούν το όριο μεγέθους του 40% που θέσαμε παραπάνω ως:

$$\text{Partially False Measure} = \frac{\text{Partially False Positives}}{\text{False Positives}}$$



Εικόνα	Συνολικός Αριθμός Dataset	Ανακλαστικών Επιφανειών	False				Patially False		
			True Positives	False Negatives	False Positives	Measure	Precision	Recall	F - Measure
Kinect	1	4	4	0	0	0	1	1	1
	2	5	5	0	1	1	0.833	1	0.909
	3	2	2	0	0	0	1	1	1
	4	2	1	1	0	0	1	0.5	0.667
	5	1	1	0	0	0	1	1	1
	6	1	1	0	0	0	1	1	1
	7	5	5	0	0	0	1	1	1
	8	1	1	0	0	0	1	1	1
	9	0	0	0	0	0	-	-	-
	10	4	4	0	1	1	0.8	1	0.889
	11	0	0	0	0	0	-	-	-
	12	0	0	0	0	0	-	-	-
	13	6	6	0	0	0	1	1	1
	14	3	3	0	0	0	1	1	1
	15	1	1	0	0	0	1	1	1
	16	0	0	0	0	0	-	-	-
	17	4	3	1	0	0	1	0.75	0.857
	18	0	0	0	0	0	-	-	-
	19	1	1	0	0	0	1	1	1
	20	0	0	0	0	0	-	-	-
	21	0	0	0	0	0	-	-	-
	22	0	0	0	0	0	-	-	-
	23	0	0	0	1	0	0	-	-
	24	2	2	0	0	0	1	1	1
	25	3	3	0	0	0	1	1	1
	26	4	4	0	2	0	0.667	1	0.889
	27	3	3	0	0	0	1	1	1
	28	4	4	0	0	0	1	1	1
	29	4	4	0	2	2	0.667	1	0.8
	30	5	3	2	0	0	1	0.6	0.75
	31	0	0	0	0	0	-	-	-
	32	0	0	0	0	0	-	-	-
	33	1	1	0	0	0	1	1	1
<hr/>			Σύνολο	66	62	4	7	0.58	0.9
<hr/>								0.94	0.92

Πίνακας 6.2.1.1: Αποτελέσματα αναγνώρισης ανακλαστικών επιφανειών στο dataset της Kinect

Όπως φαίνεται από τον παραπάνω πίνακα για το dataset της Kinect, η εύρεση των ανακλαστικών επιφανειών είναι εξαιρετικά ακριβής, αλλά και επιλεκτική, αφήνοντας παρουσιάζοντας μικρό αριθμό false positives, τόσο ανά εικόνα, όσο και στο σύνολο αυτών. Την πλειοψηφία αυτών, μάλιστα, αποτελούν τμήματα ανακλαστικών επιφανειών.



Εικόνα Dataset Xtion	Συνολικός Αριθμός Ανακλαστικών Επιφανειών					Partially False			Precision	Recall	F - Measure
		True Positives	False Negatives	False Positives	Partially Positives						
0	0	0	0	0	0	-	-	-	-	-	-
1	0	0	0	0	0	-	-	-	-	-	-
2	0	0	0	0	0	-	-	-	-	-	-
3	0	0	0	0	0	-	-	-	-	-	-
4	0	0	0	0	0	-	-	-	-	-	-
5	0	0	0	0	0	-	-	-	-	-	-
6	1	1	1	0	0	1	1	1	1	1	1
7	1	1	1	0	0	1	1	1	1	1	1
8	4	4	4	0	0	1	1	1	1	1	1
9	2	2	2	0	0	1	1	1	1	1	1
10	0	0	0	1	0	0	-	-	-	-	-
11	25	24	24	0	0	1	0.96	0.98			
12	0	0	0	0	0	-	-	-	-	-	-
13	0	0	0	0	0	-	-	-	-	-	-
14	1	1	1	0	0	1	1	1	1	1	1
15	4	3	3	0	0	1	0.75	0.857			
16	0	0	0	0	0	-	-	-	-	-	-
17	0	0	0	0	0	-	-	-	-	-	-
18	0	0	0	0	0	-	-	-	-	-	-
Σύνολο	38	36	2	1	0	0.973	0.947	0.96			

Πίνακας 6.2.1.2: Αποτελέσματα αναγνώρισης ανακλαστικών επιφανειών στο dataset της Xtion

Όσο αφορά αυτό της Xtion, παρότι οι περισσότερες υπάρχουσες ανακλαστικές εμφανίζονται σε μία εικόνα, στη τελευταία παρατηρούμε την πλειοψήφια των διαφορετικών μορφολογιών *ΕΠΑΜ* που συναντήσαμε. Τα αποτελέσματα κυμαίνονται σε ελαφρώς καλύτερα επίπεδα από τα ήδη υψηλά της Kinect, με την εμφάνιση μόλις ενός false positive.

6.2.2 Επίδειξη Εικόνων Εντοπισμού Ανακλαστικών Επιφανειών

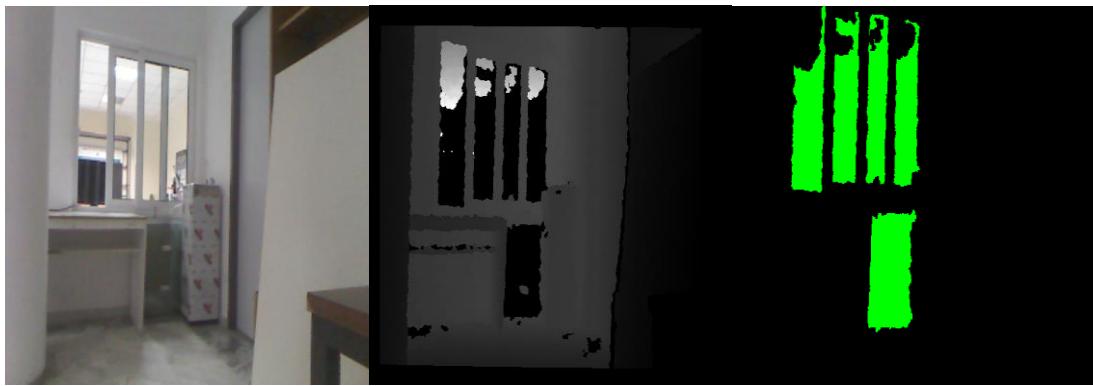
Στη συνέχεια, θα παρουσιάσουμε εποπτικά μερικές από τις εικόνες των δύο datasets, καταταγμένες σε επτά κατηγορίες. Πιο συγκεκριμένα, θα παραθέσουμε τριάδες εικόνων, με την πρώτη να δείχνει το έγχρωμο rgb πλαίσιο της κάμερας, τη δεύτερη το αντίστοιχο βάθους και την τρίτη μία μάσκα των εμπλεκόμενων επιφανειών, όπου:

1. Τα true positives, δηλαδή οι *ΕΠΑΜ* που ανήκουν στις επιφάνειες ενδιαφέροντος και αναγνωρίζονται ως τέτοιες, σημειώνονται με πράσινο χρώμα
2. Τα false positives, δηλαδή οι *ΕΠΑΜ* που δεν ανήκουν στις επιφάνειες
3. Τα false negatives, δηλαδή οι *ΕΠΑΜ* που ανήκουν στις επιφάνειες ενδιαφέροντος αλλά δεν αναγνωρίζονται ως τέτοιες, σημειώνονται με μπλε χρώμα

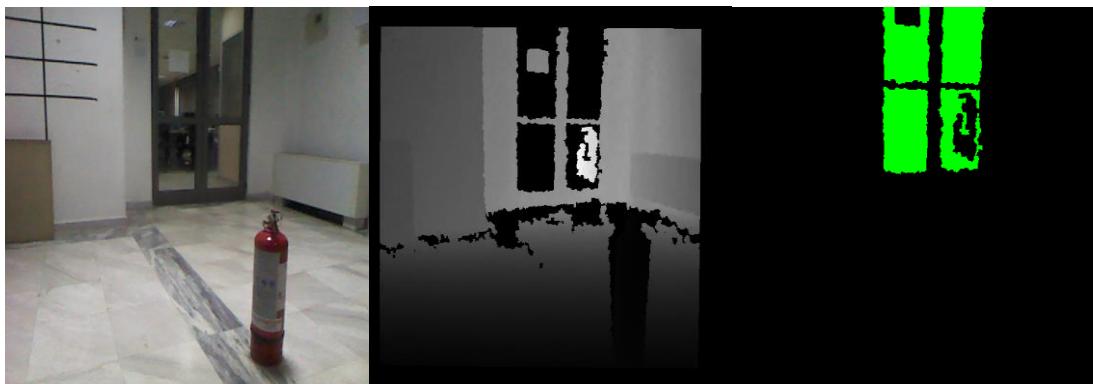
Έτσι, κατηγοριοποιούμε τις παρουσιαζόμενες εικόνες με βάση:



1. Την ύπαρξη ή όχι έγκυρης πληροφορίας για την ανακλαστική επιφάνεια



Εικόνα 6.2.2.1: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 7*



Εικόνα 6.2.2.2: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 1*

Όπως φαίνεται, ο αλγόριθμός μας δεν επηρεάζεται από την πιθανή ύπαρξη πληροφορίας βάθους του παρασκηνίου εντός της ανακλαστικής επιφάνειας, είτε αυτή είναι προσκολλημένη, είτε όχι στο περίγραμμά του.

2. Το μέγεθος των *EΠΑΜ*

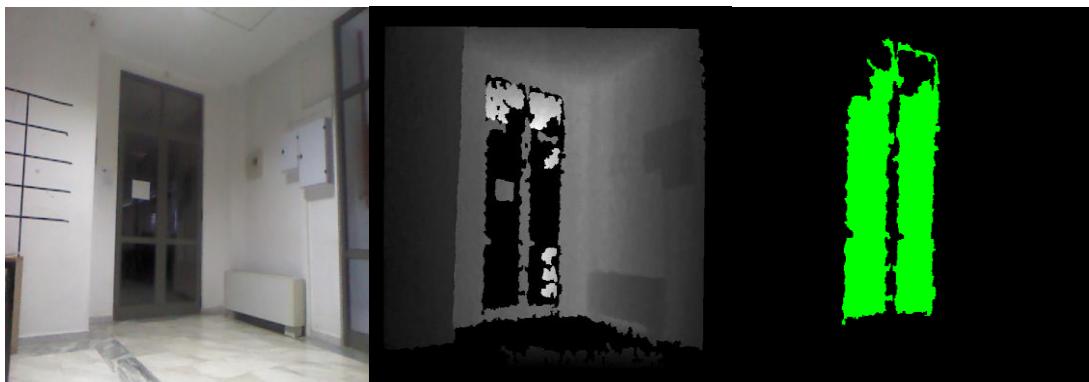


Εικόνα 6.2.2.3: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 15*





Εικόνα 6.2.2.4: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Xtion 8*



Εικόνα 6.2.2.5: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 24*

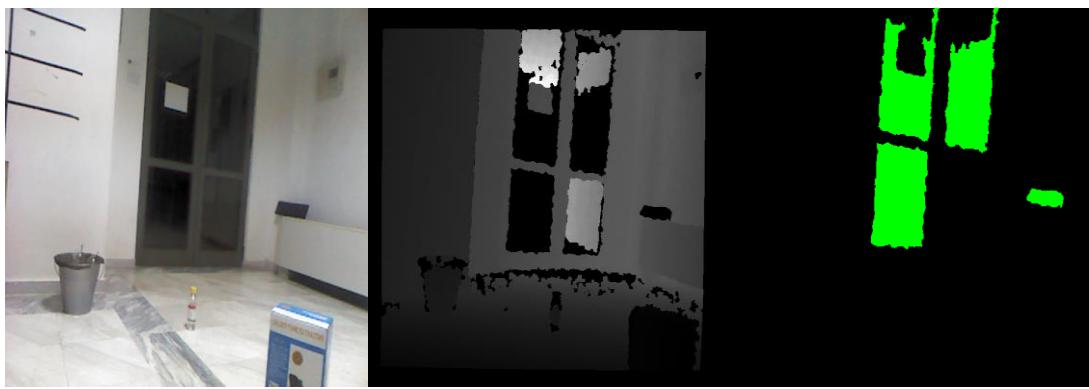
Σε περιπτώσεις διαφορετικών μεγεθών, η ορθή ρύθμιση των εμπλεκόμενων παραμέτρων έχει επιτρέψει την αναγνώριση τους ως ανακλαστικές επιφάνειες στη συντριπτική πλειοψηφία των περιπτώσεων.

3. Το σχήμα των ΕΠΑΜ



Εικόνα 6.2.2.6: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Xtion 11*

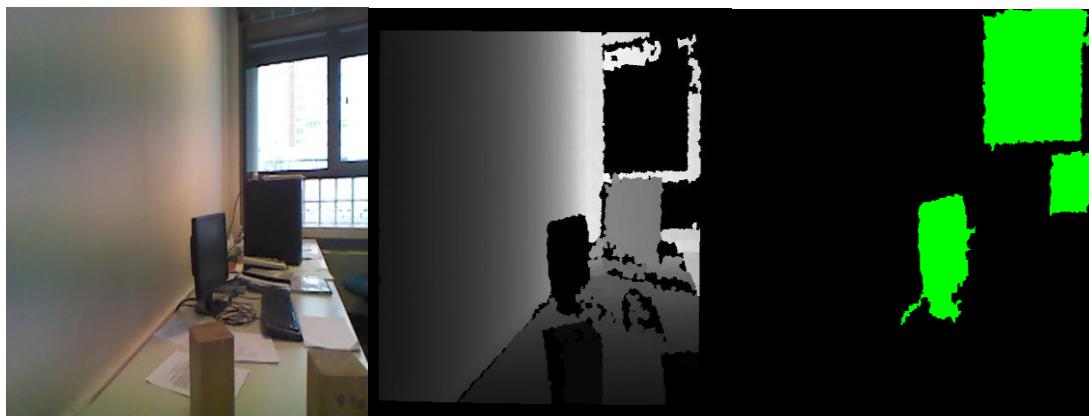




Εικόνα 6.2.2.7: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 28*

Έχοντας κάνει τη θεώρηση ότι οι περισσότερες από τις ανακλαστικές επιφάνειες που εμφανίζονται σε κλειστούς χώρους έχουν σχετικά καθορισμένο σχήμα, πετυχαίνουμε χαρακτηριστική ακρίβεια στην αναγνώριση. Ωστόσο, σε ορισμένες περιπτώσεις, κυρίως λόγω παροδικής αστοχίας της κάμερας (βλ. Εικόνα 6.2.2.6), η μορφολογία της αντίστοιχης ΕΠΑΜ μπορεί να παρουσιάσει σημαντικές ανωμαλίες, με αποτέλεσμα τον προκαταβολικό αποκλεισμό της από τον μετέπειτα έλεγχο για ανακλαστικές επιφάνειες.

4. Διαφορετικότητα των επιφανειών/αντικειμένων



Εικόνα 6.2.2.8: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 14*



Εικόνα 6.2.2.9: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 6*





Εικόνα 6.2.2.10: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Xtion 6*

Έχοντας σχεδιάσει τον αλγόριθμο με τέτοιο τρόπο, ώστε να εξετάζει μόνο ΕΠΑΜ, με σχετικά ομαλό σχήμα, καταφέρνουμε να αναγνωρίσουμε τόσο τέτοιες το περίγραμμά του προσεγγίζει τετράγωνο ή γενικά παραλληλόγραμμο (βλ. Εικόνα 6.2.2.8), όσο και τέτοιες των οποίων είναι ελλειψοειδές ή κυκλικό (βλ. Εικόνα 6.2.2.9).

5. Την απόσταση από την κάμερα



Εικόνα 6.2.2.11: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Xtion 14*



Εικόνα 6.2.2.12: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 26*





Εικόνα 6.2.2.13: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 13*

Όπως γίνεται εμφανές από την εικόνα 6.2.2.12, η απόσταση από την κάμερα είναι δυνατόν να επηρεάσει την αναγνώριση ανακλαστικών επιφανειών, με αποτέλεσμα την εμφάνιση false positives.

6. Διεύθυνση λήψης της κάμερας σε σχέση με αυτή των επιφανειών



Εικόνα 6.2.2.14: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 3*



Εικόνα 6.2.2.15: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 25*

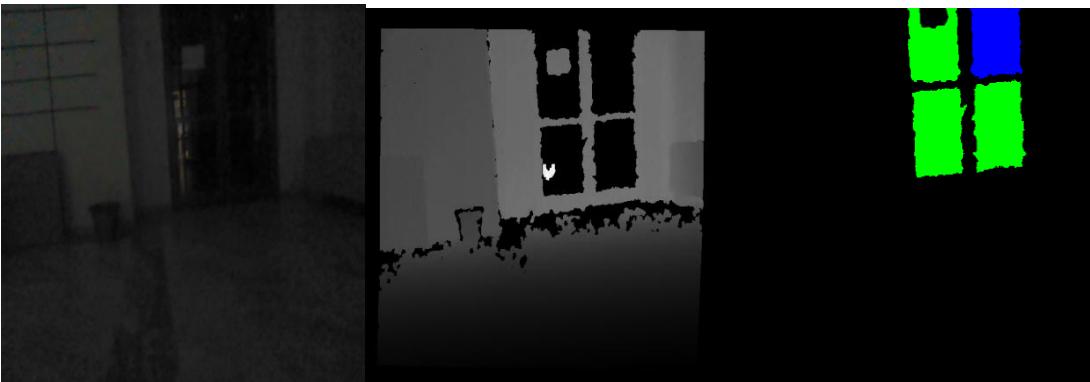
Είτε το επίπεδο που αποτυπώνεται είναι κάθετο στη διεύθυνση λήψης (βλ. Εικόνα 6.2.2.15), είτε υπό κάποια γωνία (βλ. Εικόνα 6.2.2.14), η ακρίβεια του αλγορίθμου μας δεν επηρεάζεται ιδιαίτερα.



7. Τον φωτισμό



Εικόνα 6.2.2.16: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 10*



Εικόνα 6.2.2.17: *RGB, depth και μάσκα ανακλαστικών επιφανειών της εικόνας Kinect 17*

Όπως ήταν αναμενόμενο, κακές συνθήκες φωτισμού, τοπικά η συνολικά στην εικόνα, μπορούν να επιφέρουν αστοχίες στην αναγνώριση των ανακλαστικών επιφανειών.

6.3 Εξουδετέρωση Θορύβου

Αντίστοιχα με την προηγούμενη παράγραφο, οι συντελεστές αξιολόγησης που χρησιμοποιήθηκαν για την μέτρηση της ακρίβειας του αλγορίθμου και της απόδοσης του συστήματος είναι οι εξής:

1. Μέσο Τετραγωνικό Σφάλμα (MSE): Το μέσο τετραγωνικό σφάλμα ανάμεσα στη καθαρή από εμάς εικόνα και στην ΕΑΒ εκφρασμένο σε μονάδες απόχρωσης μονοχρωματικής εικόνας. Συνεπώς, μια διαφορά d μονάδων στην εικόνα βάθους αντιστοιχεί σε σφάλμα $d / 255$. Η επιλογή αυτή έγινε καθώς κάθε μονάδα απόχρωσης στην εικόνα βάθους αντιστοιχεί σε συγκεκριμένο πραγματικό βάθος σε μονάδες μέτρου, όπου για κάθε κάμερα εκφράζεται στο διάστημα [0,255]. Συνεπώς, το MSE θα ανήκει στο διάστημα [0,1] και ένας πολλαπλασιασμός του την κλίμακα αντιστοίχησης της κάμερας δίνει το σφάλμα σε πραγματικό μήκος (π.χ.



λαμβάνοντας ως μέγιστη μέτρηση βάθους για την Kinect ~3.8m, μια εικόνα με MSE 0.1 μεταφράζεται ως μέσο σφάλμα ~38cm).

$$MSE = \frac{1}{M} \sum_{p=0}^{M-1} \left(\frac{Clean[p] - GT[p]}{255} \right)^2$$

όπου με p συμβολίζουμε όλα τα pixels που βρίσκονται εντός κάποιας *ΕΠΑΜ*, και M το σύνολο των pixels αυτών.

- Λόγος Ορθά Βαμμένης Περιοχής (ΛΟΒΠ):** Αποτελεί το λόγο του αριθμού των *ΕΠΑΜ* που εντοπίστηκαν στην εικόνα βάθους και διορθώθηκαν με μέγιστο εντός της περιοχής τους σφάλμα μικρότερο του Υποφερτού Ορίου Σφάλματος Βάθους (*ΥΟΣΒ*), το οποίο θεωρήσαμε 3 μονάδες απόχρωσης μονοχρωματικής εικόνας, (αντίστοιχα με πριν αυτό μεταφράζεται σε επιτρεπτό σφάλμα βάθους ~3.75cm στην Kinect). Έτσι, λόγοι κοντά στη μονάδα μεταφράζονται ως συγκεντρωμένα σφάλματα σε μεμονωμένες *ΕΠΑΜ*, ενώ κοντά στο μηδέν ως το σφάλμα υψηλά σε όλες τις περιοχές θορύβου.

$$\text{ΛΟΒΠ} = \frac{\text{Number of Blobs painted with } maxError \leq 3.75 \text{ cm}}{\text{Total Number of Blobs inside Depth Image}}$$

6.3.1 Πίνακας Αποτελεσμάτων Εξουδετέρωσης Θορύβου

Στην παράγραφο αυτή θα τοποθετήσουμε πίνακες των αποτελεσμάτων των συντελεστών αξιολόγησης που αναφέρθηκαν στην εισαγωγική παράγραφο σχετικά με την ακρίβεια του αλγορίθμου. Σε ότι αφορά τον χρόνο εκτέλεσης, επιλέχθηκε η εξής μέθοδος: Προκειμένου να εξερευνηθεί η καλύτερη δυνατή επίδοση του αλγορίθμου έγινε ρύθμιση των διαφόρων παραμέτρων που χρησιμοποιεί το λογισμικό μας (σχετικές με το μέγεθος του πλαισίου επεξεργασίας *ΕΠΑΜ*, την ποιότητα της τμηματοποίησης χρώματος κλπ.) έτσι ώστε να δοθεί η καλύτερα διορθωμένη έκδοση της συγκεκριμένης εικόνας βάθους και σημειώνεται ο αντίστοιχος χρόνος εκτέλεσης του προγράμματος μας. Μετά τον πίνακα αυτόν όμως ακολουθεί και ένας πίνακας που για κάθε δείγμα παρουσιάζει τον χρόνο εκτέλεσης στη *default* έκδοση παραμέτρων του αλγορίθμου μας (αυτή που εφαρμόζει το σύστημα που παρουσιάσαμε στην παράγραφο 5), το κέρδος σε χρόνο, καθώς και την απώλεια σε ακρίβεια, καθώς οι *default* τιμές αυτές για κάθε παράμετρο. Συνολικά, δηλαδή, θα παρουσιάσουμε δύο πίνακες για κάθε κάμερα, καθώς και μια στήλη με τη μέση τιμή των διαφόρων συντελεστών για όλα τα δείγματα της κάμερας.



- Microsoft Kinect: (range: ~0.6 to 3.8m)

Αριθμός Δείγματος	MSE (cm ²)	ΛΟΒΠ	Exec Time (sec)	Αριθμός Δείγματος	MSE (cm ²)	ΛΟΒΠ	Exec Time (sec)
1			0.9859	18			1.6016
2			1.0443	19			1.1189
3	0.066	0.7143	0.2940	20	6.665	0.9487	0.5270
4			0.5233	21	4.5185	0.8333	0.9699
5			0.4117	22	16.5990	0.3962	1.4580
6			0.3257	23	3.774	0.7865	0.7071
7	1.9800	0.8182	0.5897	24			0.8252
8	0.8335	0.8235	0.6478	25			0.9469
9			0.6253	26			1.0053
10	6.9360	0.7755	0.7278	27			1.0861
11			0.2101	28	37.5204	0.9359	1.1306
12	3.9501	0.8148	0.6770	29			1.0915
13	0.1523	0.7741	0.7480	30			1.4790
14			0.8732	31			1.7888
15	5.9851	0.9500	0.9902	32	1.6911	0.9792	0.8713
16			1.1577	33			0.9697
17			0.6515				
Σύνολο	6.9747	0.8115	0.8775				

Πίνακας 6.3.1.1: Αποτελέσματα διόρθωσης θορύβου στο dataset της Kinect

- ASUS Xtion: (range: ~0.8 to 3.5m)
-

Αριθμός Δείγματος	MSE (cm ²)	ΛΟΒΠ	Exec Time (sec)	Αριθμός Δείγματος	MSE (cm ²)	ΛΟΒΠ	Exec Time (sec)
1	20.0904	0.9583	0.4254	11			0.5433
2			0.5235	12	2.5260	0.9189	0.9434
3			1.6322	13			1.0487
4	0.2924	0.9857	0.8167	14			0.7548
5			0.5296	15			1.1629
6	5.8581	0.9194	0.9663	16	4.6627	0.9753	0.9962
7			0.2907	17			1.0871
8			0.2127	18			1.4080
9	0.1871	0.6818	0.3900	19	1.0804	0.9318	0.6200
10	18.9914	0.8926	1.1243				
Σύνολο:	6.7111	0.908	0.7984				

Πίνακας 6.3.1.2: Αποτελέσματα διόρθωσης θορύβου στο dataset της Xtion



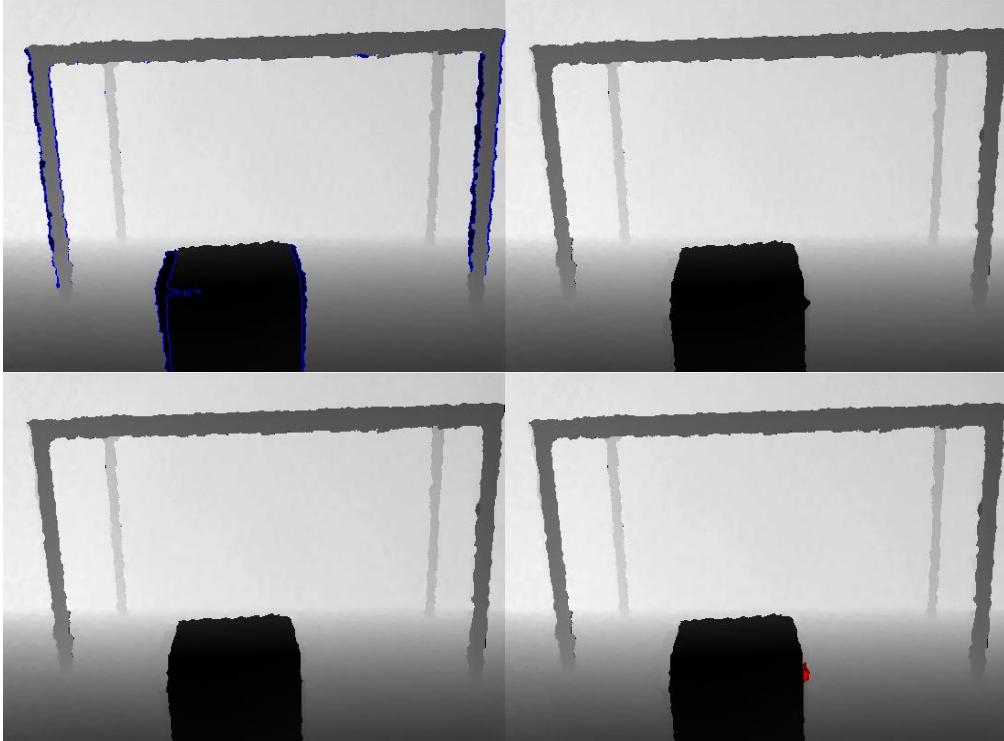
Οι εικόνες για τις οποίες δημιουργήσαμε το ground truth παρουσιάζουν ανάλογα αποτελέσματα και στις δύο κάμερες με MSE να κυμαίνεται λίγο κάτω από 7. Το ΛΟΠΒ αναδεικνύει ότι το πιθανώς υψηλό υπάρχον λάθος στη διόρθωση συγκεντρώνεται σε περιορισμένο αριθμό *ΕΠΑΜ* στο σύνολο των εικόνων και, κυρίως, σε εκείνες του dataset της Xtion.

6.3.2 Επίδειξη Εικόνων Εξουδετέρωσης Θορύβου

Στην παράγραφο αυτή, επιδεικνύουμε, όπως και στην προηγούμενη, εικόνες των ζευγών μαζί με την EAB, καθώς και μια εικόνα του διορθωμένου βάθους με εμφανώς σημειωμένες τις περιοχές που υπερβαίνουν το YOΣΒ για την οπτικοποίηση του μεγέθους του σφάλματος που εμφανίζουν οι έξοδοι του συστήματός μας σε σχέση με το *ground truth*. Διακρίνουμε πάλι διαφορετικές περιπτώσεις θορύβου και διαφορετικές συμπεριφορές του αλγορίθμου σε αυτές και παραλείπουμε τις επαναλαμβανόμενες περιπτώσεις αποτελεσμάτων στο σετ.

Οι εικόνες παρουσιάζονται με την εξής πάντα σειρά από αριστερά προς τα δεξιά και από πάνω προς τα κάτω: (α) Εικόνα Βάθους με σημειωμένο το θόρυβο, (β) Εικόνα Διορθωμένου Βάθους, (γ) EAB, (δ) Εικόνα με ενδεδειγμένες περιοχές που το *MSE* ξεπερνάει το *YOΣΒ* με μωβ χρώμα. Η απόχρωση του μωβ δίνεται αφαιρώντας από το R κανάλι της εικόνας το τριπλάσιο του σφάλματος για κάθε pixel. Έτσι, μωβ αποχρώσεις με ασθενή ένταση σημαίνουν μη υποφερτά σφάλματα στις περιοχές αυτές, με μεγαλύτερα σφάλματα για σκουρότερες αποχρώσεις.

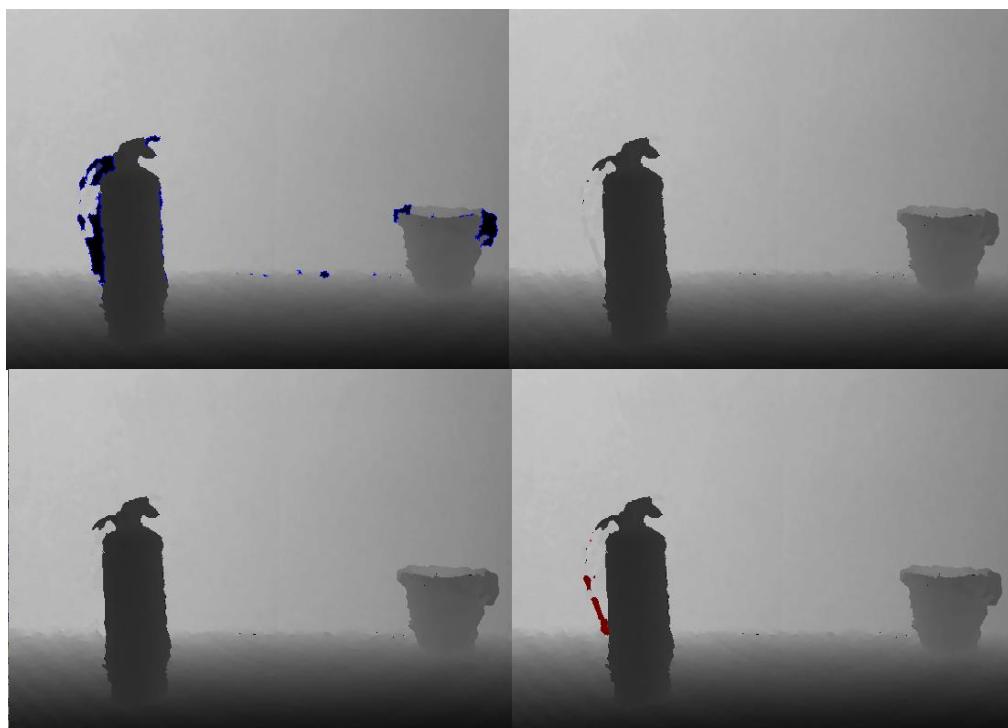
- Περίπτωση 1: Ύπαρξη μόνο απλής φυσικής σκιάς σε σταθερό επίπεδο.



Εικόνα 6.3.2.1: Δείγμα Kinect 32, MSE: 1.6911, ΛΟΒΠ: 0.9792

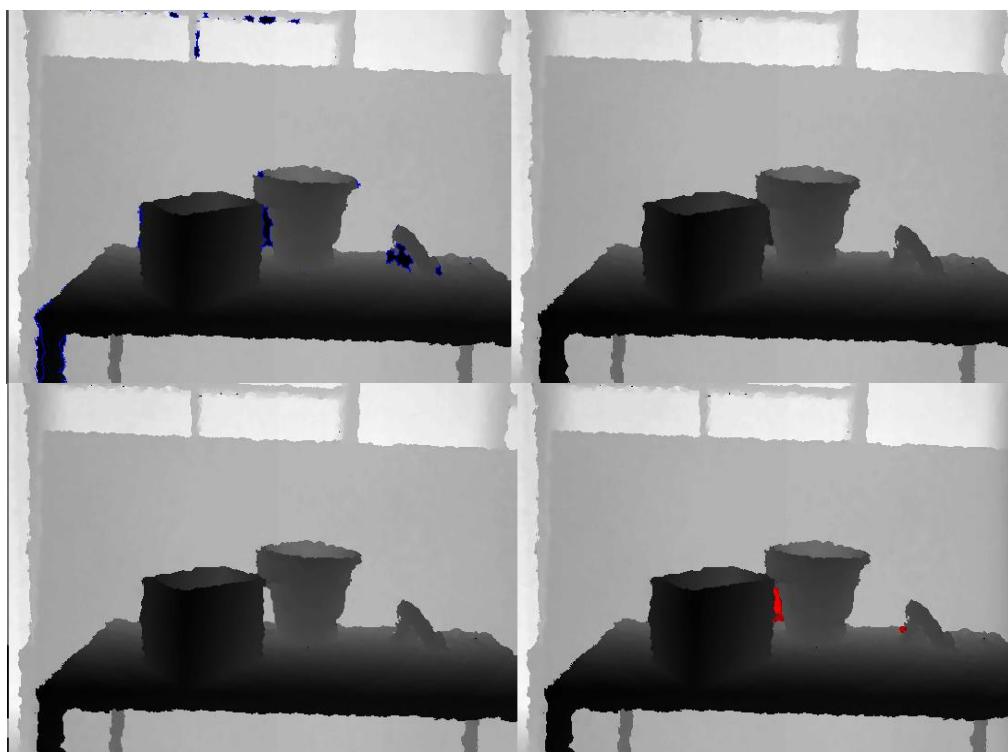


Εδώ, βλέπουμε μια ασήμαντη περιοχή σφάλματος που οφείλεται σε τοπικά εσφαλμένη τμηματοποίηση χρώματος πάνω στην ακμή του κουτιού.



Εικόνα 6.3.2.2: Δείγμα Xtion 19, MSE: 1.0804, ΛΟΒΠ: 0.9318

Στην περίπτωση αυτή βλέπουμε ότι το χερούλι του πυροσβεστήρα εντοπίστηκε από την τμηματοποίηση χρώματος και επιδιώχθηκε να διορθωθεί εσφαλμένα, καθώς, όπως φαίνεται και στην ΕΟΒ, καμία τιμή του δεν έχει αποτυπωθεί στην αρχική εικόνα βάθους

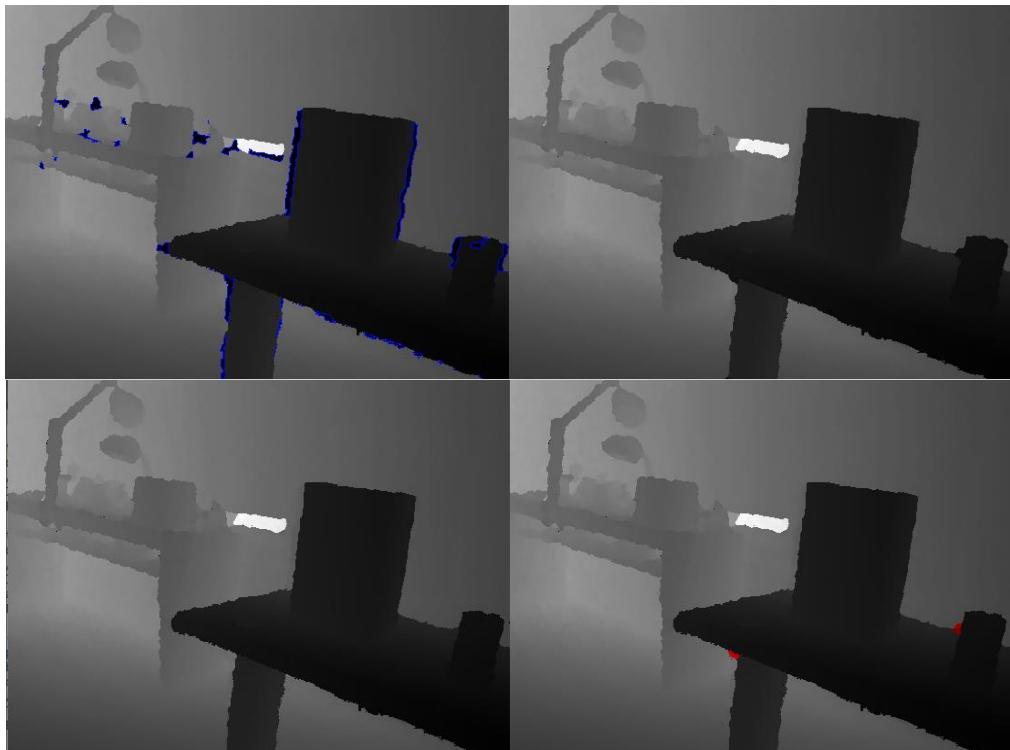


Εικόνα 6.3.2.3: Δείγμα Xtion 1, MSE: 20.0904, ΛΟΒΠ: 0.9583



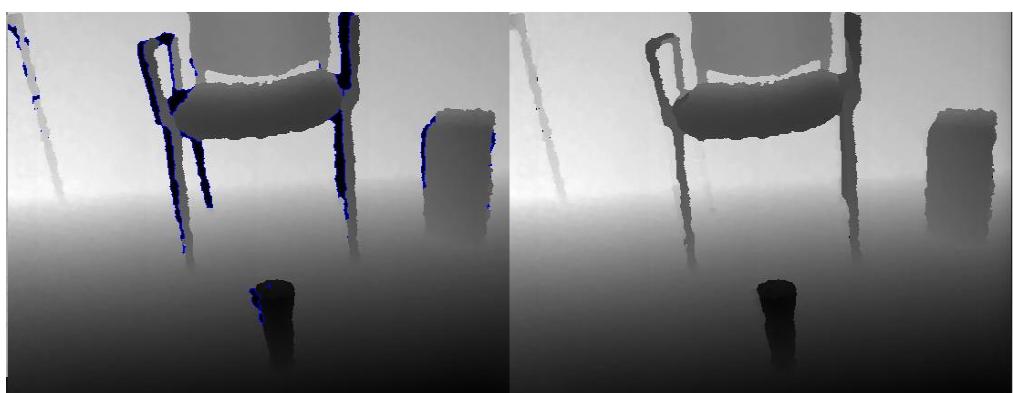
Παρατηρείται αποτυχία τμηματοποίησης λόγω κοντινής απόχρωσης τούχου – αντικειμένων.

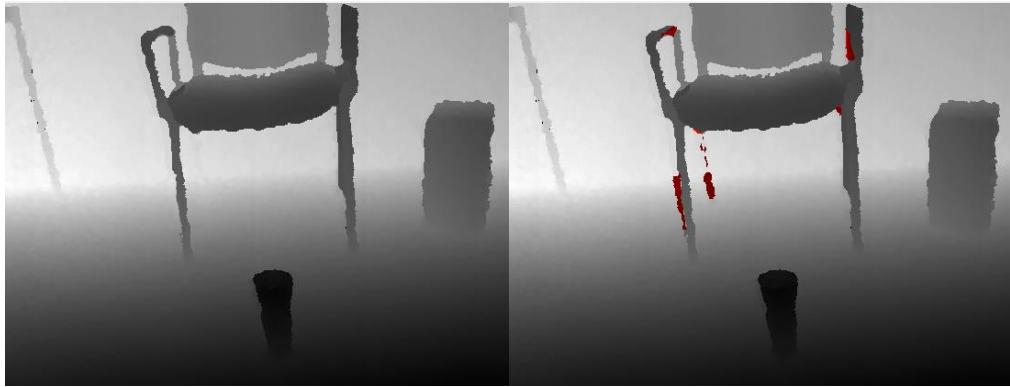
- Περίπτωση 2: Ύπαρξη μόνο φυσικής σκιάς σε κατευθυνόμενο επίπεδο.



Εικόνα 6.3.2.4: Δείγμα Xtion 4, MSE: 0.2924, ΛΟΒΠ: 0.9857

Έχουμε ασήμαντο σφάλμα στις ακμές, όπως και στην εικόνα 6.3.2.1.





Εικόνα 6.3.2.5: Δείγμα Kinect 12, MSE: 3.9501, ΛΟΒΠ: 0.8148

Εδώ, εμφανίζεται μία χαρακτηριστική περίπτωση σφάλματος τμηματοποίησης. Ορισμένες περιοχές που οφείλανε να βρίσκονται στο παρασκήνιο (φυσική σκιά επί των ποδιών της καρέκλας) έχουν αποδοθεί στο προσκήνιο, με αποτέλεσμα να επιχειρηθεί να διορθωθούν λανθασμένα με έγκυρη πληροφορία βάθους του τελευταίου.



Εικόνα 6.3.2.6: Δείγμα Kinect 15, MSE: 5.9851, ΛΟΒΠ: 0.95

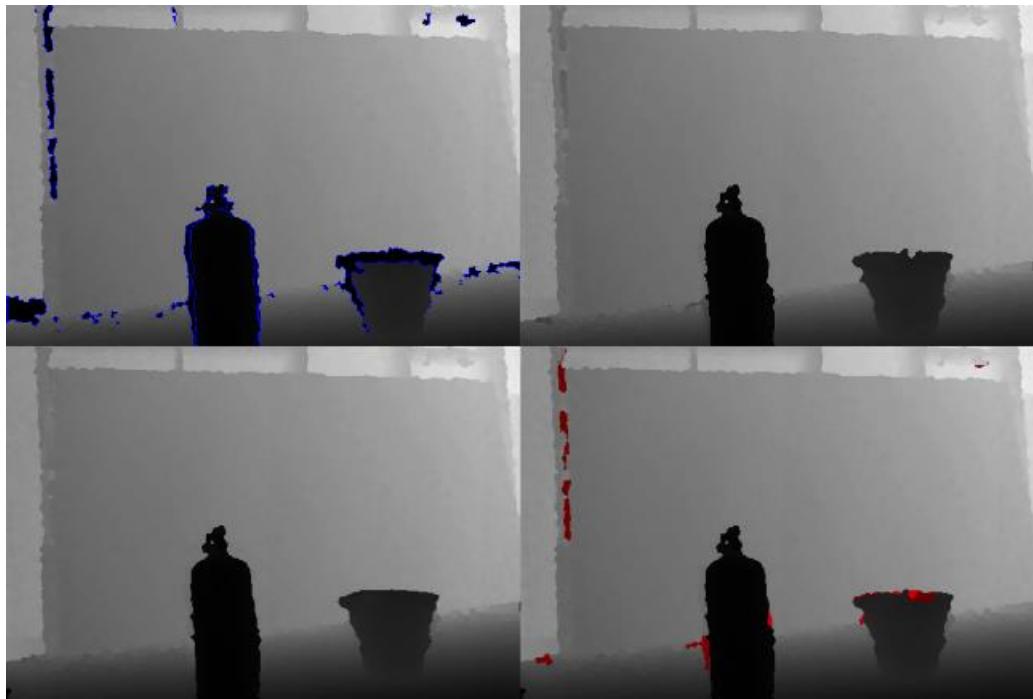
Στην εικόνα αυτή, έχουμε πλήρη επιτυχία ομαδοποίησης της περιοχής του μπουκαλιού παρά την κοντινή χρωματική απόχρωση. Η διόρθωση είναι επιτυχής χωρίς ιδιαίτερο σφάλμα συνολικά.

- Περίπτωση 3: Ύπαρξη μόνο συγχωνευμένης φυσικής σκιάς σε σταθερό / κατευθυνόμενο επίπεδο

Σε όλες τις παρακάτω περιπτώσεις με αυξανόμενο ποσοστό σφάλματος, βλέπουμε ότι όσο αυξάνεται ο όγκος του θορύβου και όσο φυσικές σκιές που οφείλονται σε διαφορετικά αντικείμενα συγχωνεύονται μεταξύ τους, αυξάνεται η τοπική αποτυχία της τμηματοποίησης,



άρα εισάγεται μεγαλύτερο σφάλμα.

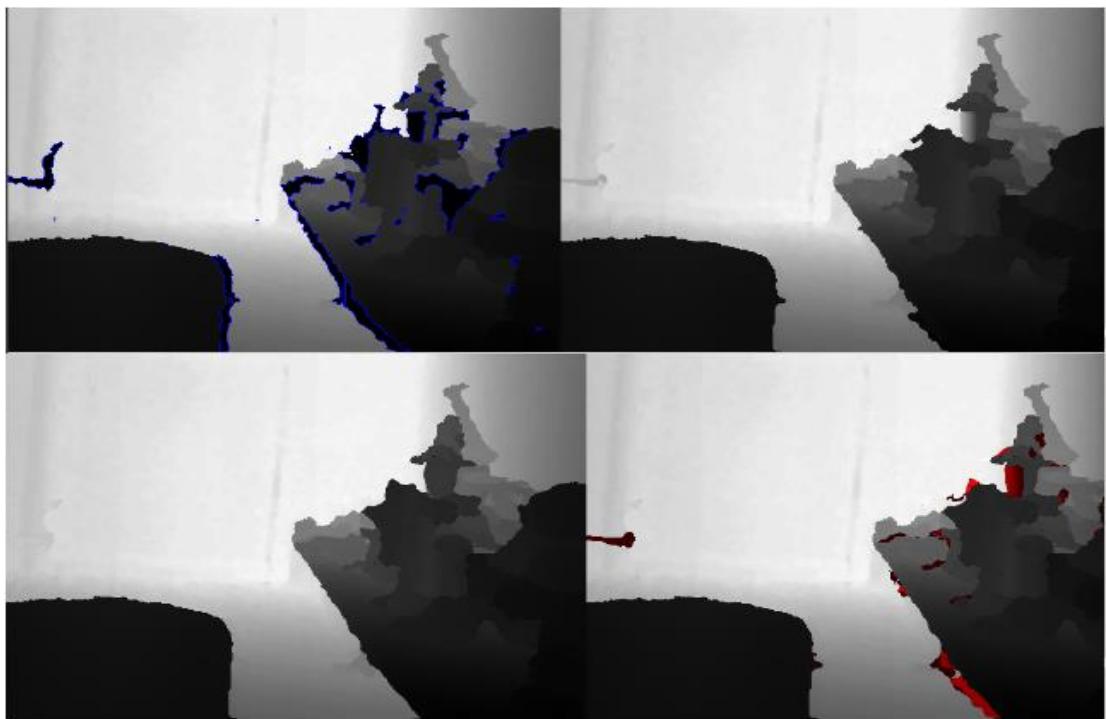


Εικόνα 6.3.2.7: Δείγμα Kinect 21, MSE: 4.5185, ΛΟΒΠ: 0.8333



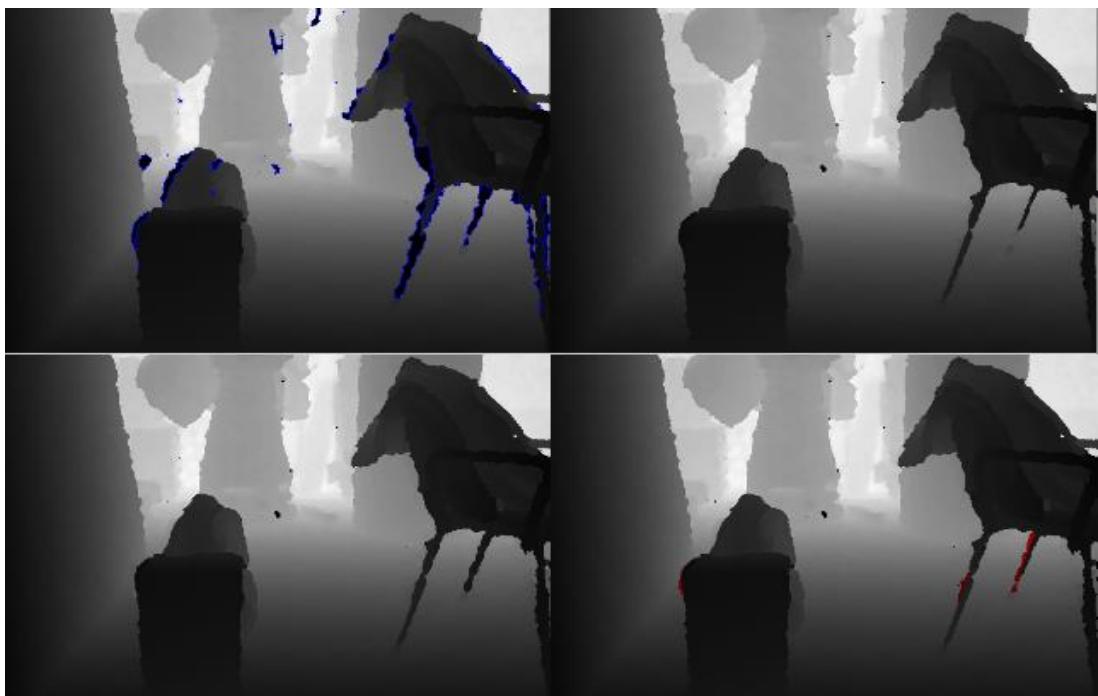
Εικόνα 6.3.2.8: Δείγμα Xtion 12, MSE: 2.5263, ΛΟΒΠ: 0.9189





Εικόνα 6.3.2.9: Δείγμα Xtion 10, MSE: 18.9914, ΛΟΒΠ: 0.8926

- **Περίπτωση 4:** Ύπαρξη συγχωνευμένου θορύβου σκιάς – απορροφητικής επιφάνειας – ακραίας απόστασης σε σταθερό / κατευθυνόμενο επίπεδο



Εικόνα 6.3.2.10: Δείγμα Xtion 16, MSE: 4.6627, ΛΟΒΠ: 0.9753

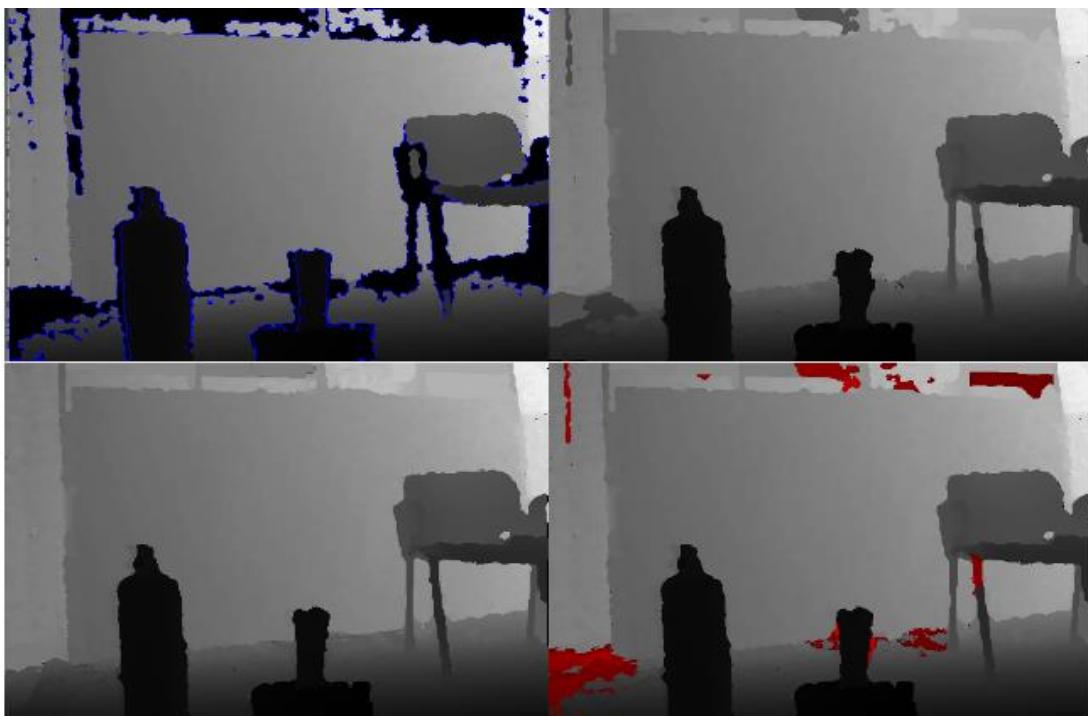
Παρότι δεν υπήρχε απόδοση πληροφορίας βάθους στα πόδια της καρέκλας εντός του αρχικού χάρτη, βλέπουμε ότι έχει επιτευχθεί η ορθή συμπλήρωσή τους.





Εικόνα 6.3.2.11: Δείγμα Xtion 20, MSE: 6.6650, ΛΟΒΠ: 0.9487

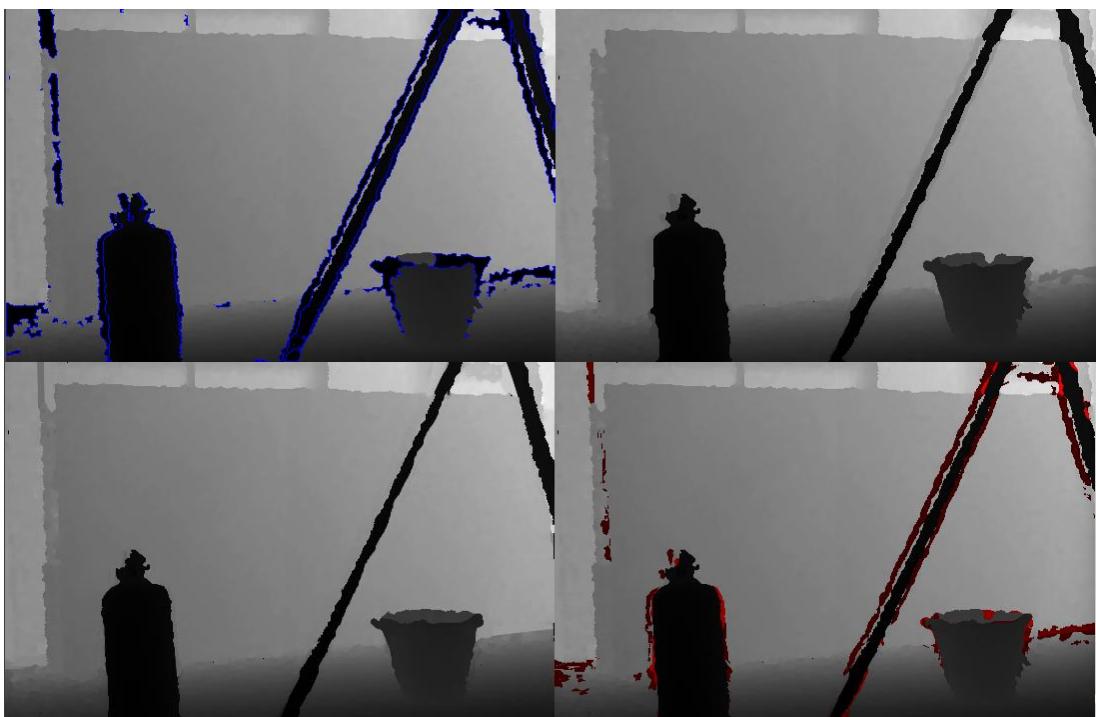
Παρουσιάζεται αποτυχία λόγω ομαδοποίησης όλης της απορροφητικής επιφάνειας σε μία ομάδα του προσκηνίου.



Εικόνα 6.3.2.12: Δείγμα Kinect 23, MSE: 6.74344, ΛΟΒΠ: 0.7865

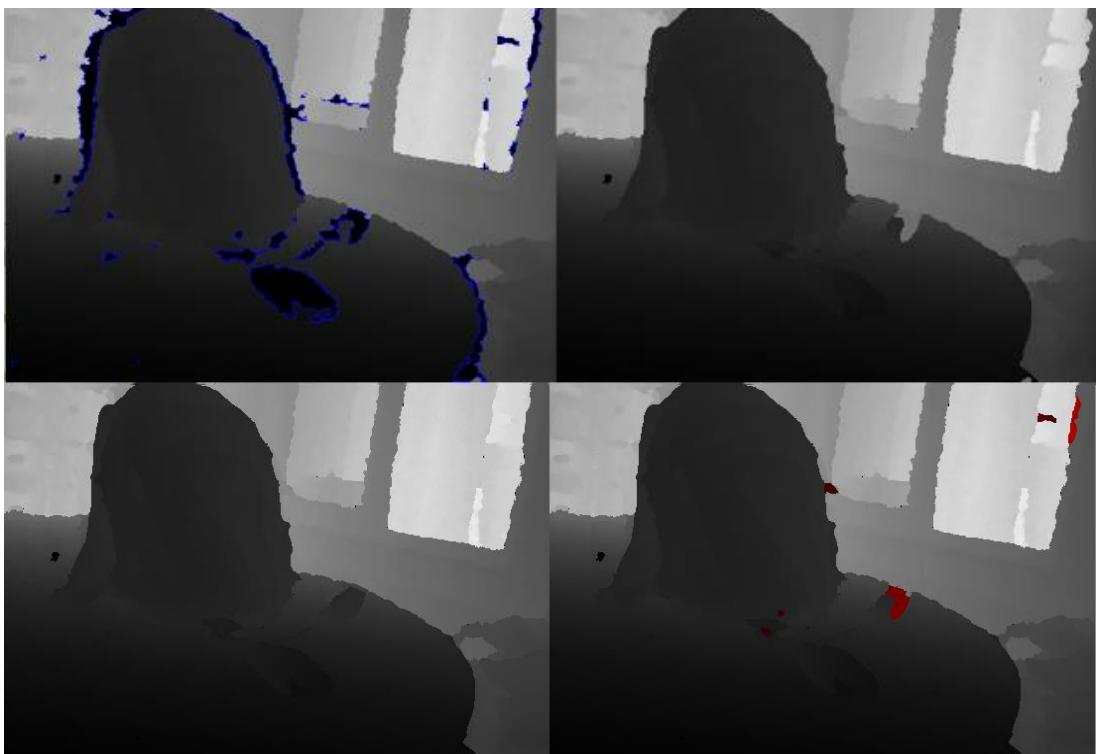
Παρουσιάζεται πολύς και συγχωνευμένος θόρυβος, οπότε και μεγάλα σφάλματα. Παρόλα αυτά, έχουμε σχετικά καλή απεικόνιση παρά το μικρό ΛΟΒΠ.





Εικόνα 6.3.2.13: Δείγμα Kinect 22, MSE: 16.5990, ΛΟΒΠ: 0.3962

Οι περιοχές θορύβου είναι ιδιαίτερα εκτενής, με αποτέλεσμα την εσφαλμένη λειτουργία της διόρθωσης λόγω λανθασμένου υπολογισμού ΔΚΒ.



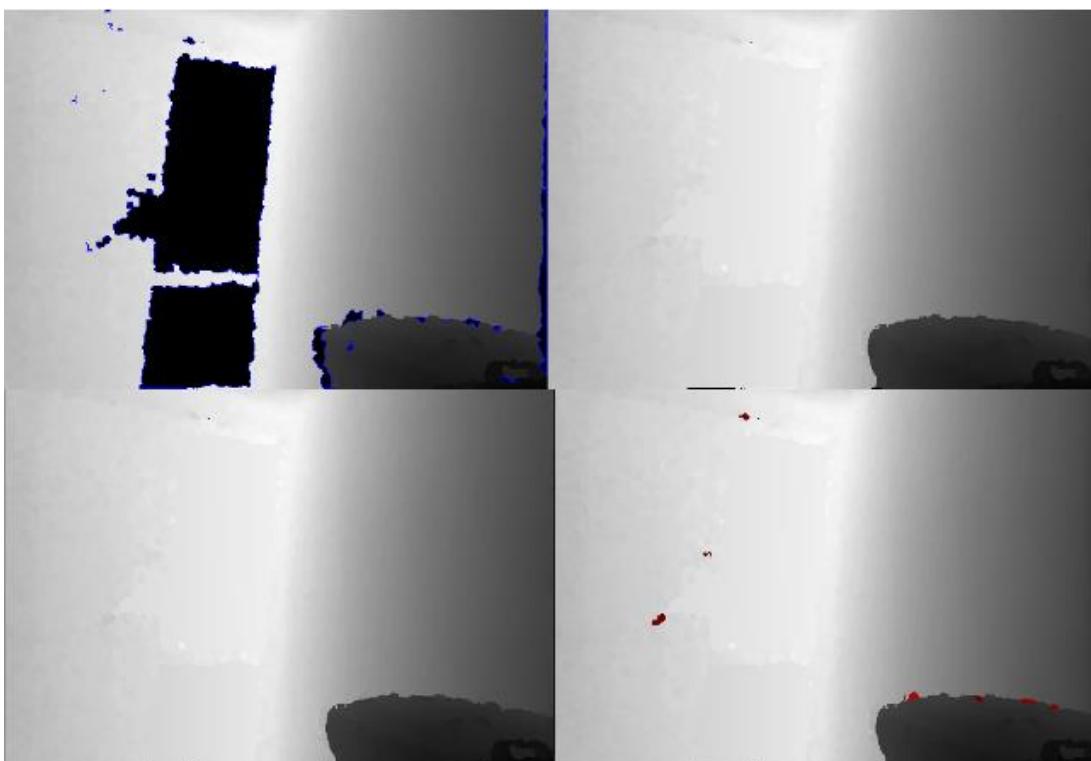
Εικόνα 6.3.2.14: Δείγμα Kinect 22, MSE: 5.8581, ΛΟΒΠ: 0.9032



Έχουμε αποτυχία απόδοσης τμήματος απορροφητικής επιφάνεια, λόγω τοπικά κακής τμηματοποίησης.

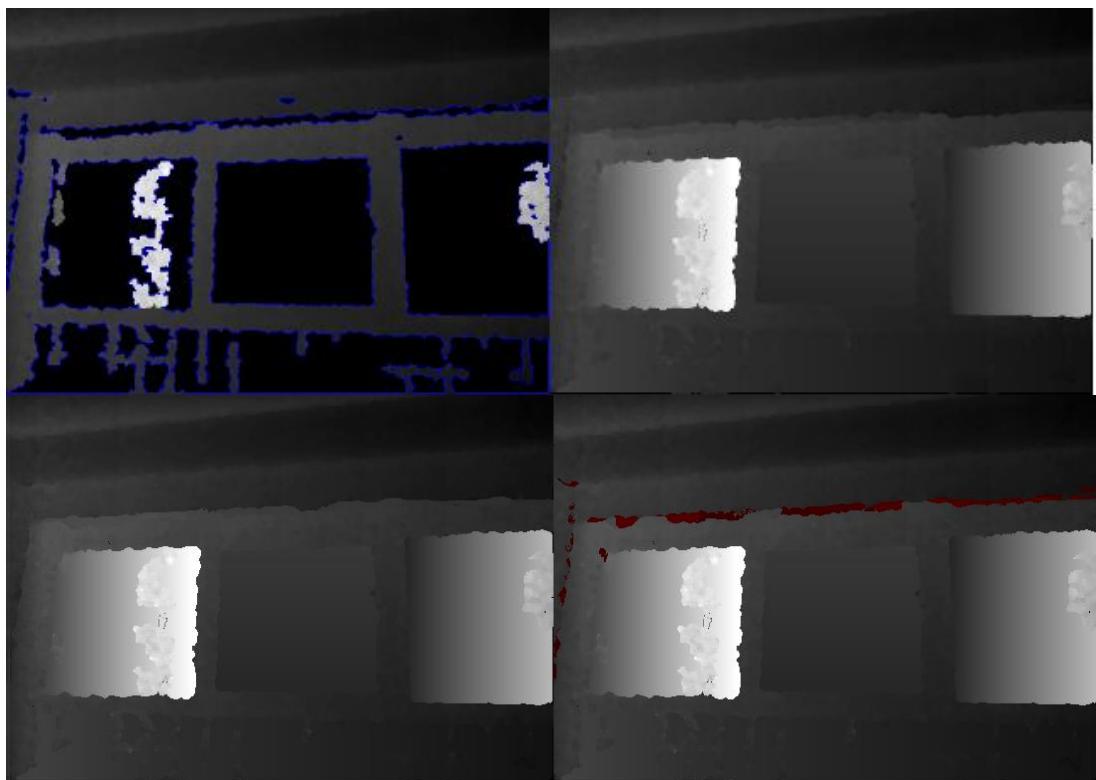
- Περίπτωση 5: Ύπαρξη μόνο απλής φυσικής σκιάς και ανακλαστικών επιφανειών σε κατευθυνόμενο επίπεδο.

Οι ανακλαστικές επιφάνειες αποδίδονται ικανοποιητικά, με βάση τον τρόπο που ορίσαμε εμείς ορθό ιδιαίτερα για εκείνες με ύπαρξη πληροφορία του παρασκηνίου στο εσωτερικό τους. Όσο αφορά τις φυσικές σκιές, η συμπεριφορά είναι ανάλογα με αυτή που περιγράφηκε και σε προηγούμενες εικόνες (τοπικά λάθη λόγω τμηματοποίησης).

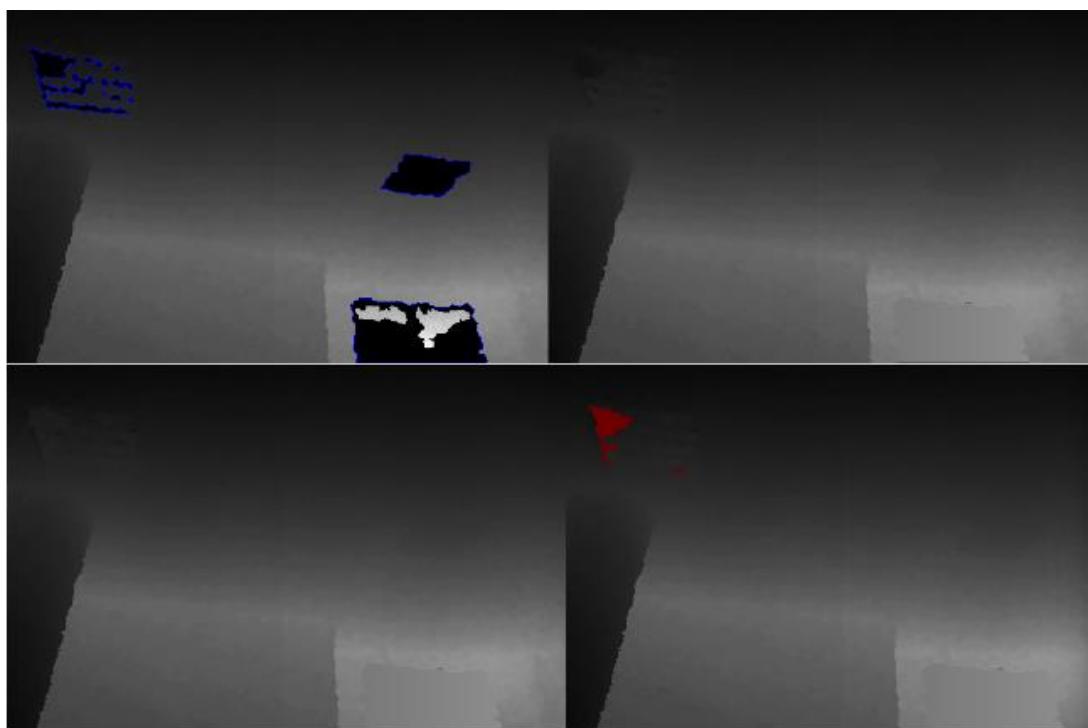


Εικόνα 6.3.2.15: Δείγμα Xtion 9, MSE: 0.1871, ΛΟΒΠ: 0.6818





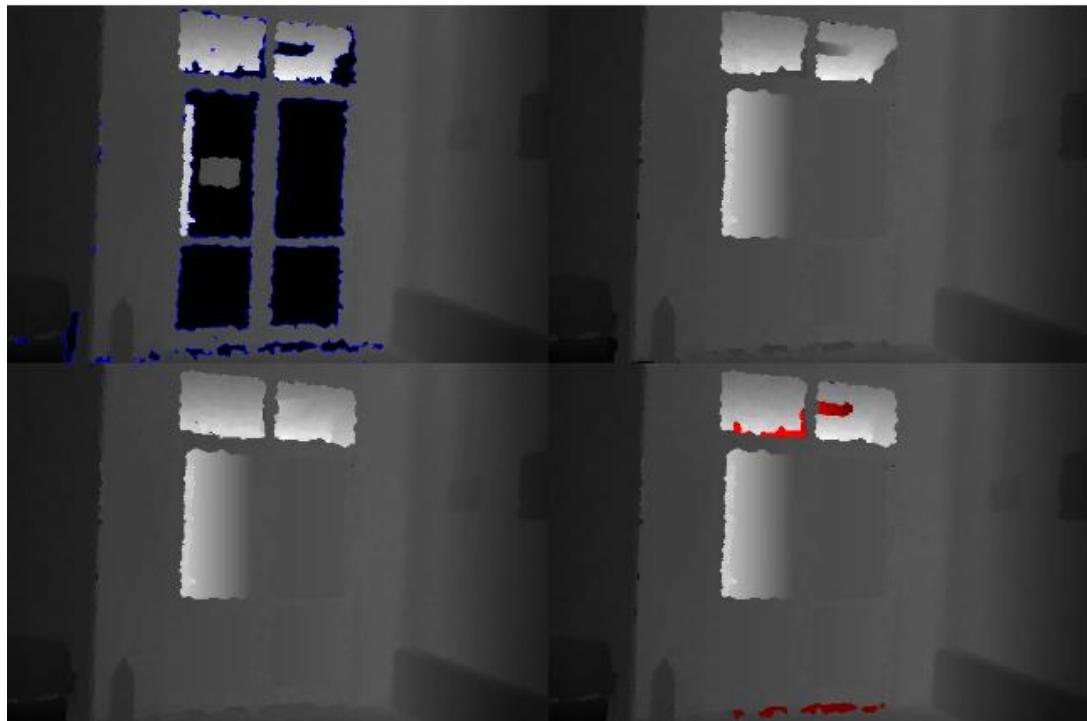
Εικόνα 6.3.2.16: Δείγμα Kinect 3, MSE: 0.1523, ΛΟΒΠ: 0.7741



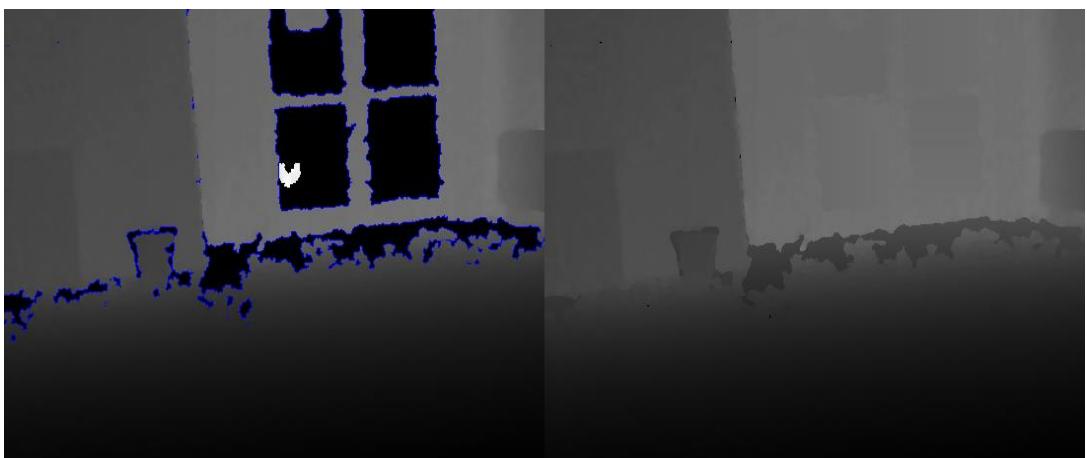
Εικόνα 6.3.2.17: Δείγμα Kinect 3, MSE: 0.066, ΛΟΒΠ: 0.7143



- Περίπτωση 6: Ύπαρξη σύνθετου θορύβου ανακλαστικών επιφανειών – σκιάς – ακραίας απόστασης σε κατευθυνόμενο επίπεδο.

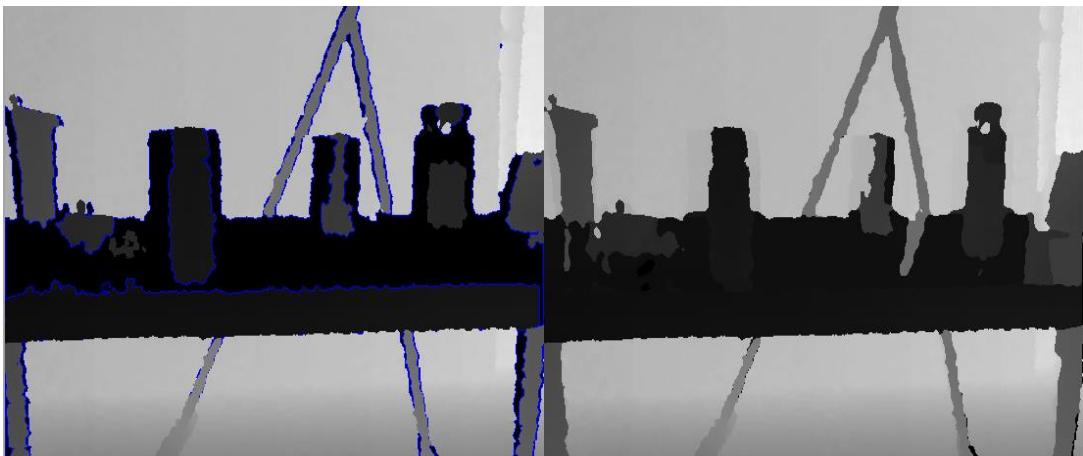


Εικόνα 6.3.2.18: Δείγμα Kinect 10, MSE: 6.9360, ΛΟΒΠ: 0.7755



Εικόνα 6.3.2.19: Εικόνες βάθους του δείγματος 17 της Kinect πριν και μετά τη διόρθωση





Εικόνα 6.3.2.20: Εικόνες βάθους του δείγματος 31 της Kinect πριν και μετά τη διόρθωση

Στις παραπάνω σύνθετες περιπτώσεις, βλέπουμε και εποπτικά ότι σε περιπτώσεις ακραίας απόστασης από την κάμερα, αλλά και σε εκείνες κακού φωτισμού τοπικά (βλ. Εικόνα 6.3.2.18) ή γενικά (βλ. Εικόνα 6.3.2.19), η τρηματοποίηση αποτυγχάνει σημαντικά. Το τελευταίο συμβαίνει και σε περιπτώσεις μεγάλων ΕΠΑΜ συγχωνευμένου θορύβου πολλών αντικειμένων και επιφανειών (βλ. Εικόνα 6.3.2.20).

6.4 Αξιολόγηση Αποτελεσμάτων

Στην παράγραφο αυτή, θα εξετάσουμε τα αριθμητικά αποτελέσματα των πειραμάτων σε σχέση με εποπτικές παρατηρήσεις στις διορθωμένες εικόνες, θα ξεχωρίσουμε περιπτώσεις και θα σχολιάσουμε την ακρίβεια του αλγορίθμου, τόσο στο ζήτημα του εντοπισμού ανακλαστικών επιφανειών όσο και στην εξουδετέρωση θορύβου.

Εξετάζοντας αρχικά τον αλγόριθμο εντοπισμού ανακλαστικών επιφανειών, όπως παρατηρούμε από τα αποτελέσματα των πινάκων της Παραγράφου 6.2 βγάζουμε τα εξής συμπεράσματα:

1. Τα ποσοστά εύρεσης ανακλαστικών επιφανειών εμφανίζονται ιδιαίτερα υψηλά, τόσο ανά εικόνα, πλην ελάχιστων εξαιρέσεων (dataset Kinect, δείγμα 30), όσο και στο σύνολο των δύο datasets. Οι μετρικές precision και recall για το σύνολο των δυο datasets ξεπερνούν το 90%, αποδεικνύοντας ότι, με σωστή ρύθμιση των παραμέτρων, το συγκεκριμένο κομμάτι του αλγορίθμου είναι εξαιρετικά ακριβές (λίγα false negatives) και επιλεκτικό (λίγα false positives) στο χαρακτηρισμό των επιφανειών ως ανακλαστικές.
2. Οι υψηλές τιμές των παραπάνω σε συνδυασμό με τη μικρή τους διαφορά αποδίδει εξίσου μεγάλη τιμή και στη μετρική f – measure, η οποία αποτελεί τον σταθμισμένο μέσο των δύο, επιβεβαιώνοντας το συμπέρασμα μας. Πιο συγκεκριμένα, όπως αποδίδεται και από τις διαφορετικές κατηγορίες παρουσιαζόμενων εικόνων στην παράγραφο 6.3, στο dataset της Kinect, όπου εμφανίζεται μεγαλύτερη ποικιλία



τέτοιων επιφανειών, την πλειοψηφία των false positives αποτελούν μικρά σε μέγεθος τμήματα (τέσσερα) ανακλαστικών επιφανειών (δείγμα 29), με τα υπόλοιπα (τρία) να οφείλονται στη απόσταση της κάμερας από την επιφάνεια (δείγμα 26) ή στο χαμηλό φωτισμό (δείγμα 10). Ο τελευταίος επηρεάζει, όπως είναι φυσικό, και τον αριθμό των false negatives (δείγμα 17).

3. Οι ΕΠΑΜ που δεν αναγνωρίζονται ως ανακλαστικές περιλαμβάνουν ακόμη περιπτώσεις στις οποίες υπάρχει προσκολλημένος ανομοιόμορφος θόρυβος άλλης επιφάνειας (δείγμα 30), καθιστώντας την προκαταβολικά μη πιθανή ανακλαστική από άποψη μορφολογίας.

Τα παραπάνω συμπεράσματα εφαρμόζουν και στις εικόνες του dataset της Xtion, με τις περισσότερες από αυτές να συγκεντρώνονται στο δείγμα 11, όπου εμφανίζονται οι περισσότερες από τις παρουσιαζόμενες περιπτώσεις διαφορετικών ΕΠΑΜ για εικόνες με ικανοποιητικό φωτισμό.

Συγκεντρώνοντας επιπλέον όλα τα αποτελέσματα των πειραμάτων από τους πίνακες της Παραγράφου 6.3, διακρίνουμε τρεις κατηγορίες αποτελεσμάτων για κάθε εικόνα:

1. Εικόνες με μέσο τετραγωνικό σφάλμα μικρότερο του υποφερτού ορίου ($MSE < 3.2 \text{ cm}^2$): Οι εικόνες αυτές αποτελούν περισσότερο από το 42 % του συνόλου των δειγμάτων. Ουσιαστικά, αυτές είναι οι περιπτώσεις που ο αλγόριθμος αφαίρεσης θορύβου έχει δουλέψει ικανοποιητικά, παρέχοντας για κάθε ΕΠΑΜ εκτιμώμενο βάθος με ασήμαντα σφάλματα και αποδίδοντας ορθά το βάθος του συνόλου της σκηνής. Τέτοιες περιπτώσεις αποτελούν η ύπαρξη μόνο ανακλαστικής ή μόνο φυσικής σκιάς σε σταθερό ή κατευθυνόμενο επίπεδο βάθους (π.χ. δείγμα 13 της Kinect, 9 της Xtion), καθώς και η πλειοψηφία ύπαρξης συγχωνευμένου θορύβου φυσικής σκιάς σε σταθερό επίπεδο βάθους.
2. Εικόνες με μέσο τετραγωνικό σφάλμα που υπερβαίνει το υποφερτό όριο, αλλά με πολύ μεγάλο ΛΟΒΠ (>0.8): Οι εικόνες αυτές αποτελούν περίπου το 71.5 % του συνόλου των δειγμάτων, με αρκετές από τις υπόλοιπες να κυμαίνονται άνω του 0.75 ΛΟΠΒ. Αυτές είναι οι περιπτώσεις που η μαζική πλειοψηφία των ΕΠΑΜ της εικόνας έχουν διορθωθεί ικανοποιητικά, αλλά υπάρχουν κάποιες ΕΠΑΜ που εισάγουν πολύ μεγάλο σφάλμα. Ο αλγόριθμος δηλαδή έχει δουλέψει με ακρίβεια για το μεγαλύτερο μέρος της εικόνας, αλλά έχει αποτύχει σημαντικά στην απόδοση μιας ΕΠΑΜ ή πιο συχνά ενός τμήματος ΕΠΑΜ (π.χ. δείγμα 28 της Kinect, 10 της Xtion).

Η βασική αιτία για την εμφάνιση αυτών των δειγμάτων αποτελεί η αποτυχημένη τημηματοποίηση χρώματος τοπικά για την συγκεκριμένη περιοχή, με αποτέλεσμα κομμάτι της ΕΠΑΜ να ομαδοποιηθεί ως ίδιο χρωματικό τμήμα με άλλο αντικείμενο ή με το παρασκήνιο. Η αποτυχία αυτή μεταφράζεται ως διαφορά βάθους που μπορεί να υπερβαίνει τις 100 μονάδες απόχρωσης (δηλ. σφάλμα $\sim 1.255\text{m}$). Το σφάλμα αυτό είναι εμφανώς πολύ μεγαλύτερο από το επιτρεπτό όριο και γι' αυτό επηρεάζει δραματικό το MSE . Ο υψηλός ΛΟΒΠ όμως μας δείχνει ότι το σφάλμα αυτό είναι συγκεντρωμένο σε λίγες περιοχές, ενώ ο υπόλοιπος χάρτης έχει αποδοθεί με σχετική ακρίβεια.

3. Εικόνες με μέσο τετραγωνικό σφάλμα που υπερβαίνει το υποφερτό όριο και με όχι κυρίαρχο ΛΟΒΠ (<0.8). Οι εικόνες αυτές αποτελούν το 29.5 % του συνόλου των



δειγμάτων. Αυτές είναι οι περιπτώσεις που πολλές διαφορετικές ΕΠΑΜ εντός της εικόνας έχουν διορθωθεί με εσφαλμένο βάθος, λόγω αποτυχίας ομαδοποίησης και εξαγωγής MEB που αντιστοιχεί σε άσχετη περιοχή. Οι περιπτώσεις αυτές αποτελούν ακραίες περιπτώσεις που είτε ο θόρυβος αποτελεί πολύ μεγάλο κομμάτι της εικόνας ($>50\%$), είτε είναι ενιαίος σε μια περιοχή, είτε οι συνθήκες φωτισμού στην έγχρωμη εικόνα είναι τέτοιες που οδηγούν σε ολοκληρωτική αποτυχία της τμηματοποίησης (π.χ. δείγμα 22 της Kinect),

Όσον αφορά τη διαδικασία της διόρθωσης, όπως αναφέρθηκε και νωρίτερα, για επιφάνειες που αναγνωρίζονται ορθά ως ανακλαστικές (true positives) και έχουν στο εσωτερικό τους πληροφορία του παρασκηνίου, επιλέξαμε την απόδοση μίας βαθμιαίας μεταβολής των τιμών τους, ξεκινώντας από τη μικρότερη τιμή (προσκήνιο) και φτάνοντας στη μεγαλύτερη (παρασκήνιο). Από τις εικόνες της παραγράφου 6.4, η παραπάνω επιλογή αποτυπώνεται πλήρως, με τα μέγιστα και τα ελάχιστα να εντοπίζονται στις λογικά ορθές θέσεις. Σχετικά με εκείνες που δεν περιέχουν πληροφορία παρασκηνίου στο εσωτερικό τους, η διαδικασία διόρθωσης δείχνει να δουλεύει εξαιρετικά, με τις υπολογιζόμενες τιμές να μην ξεπερνούν ποτέ το ελάχιστο περιθώριο θεωρούμενου λάθους που έχουμε ορίσει. Για όλες τις υπόλοιπες ΕΠΑΜ, στις οποίες αποδίδονται τιμές με βάση τη χρωματική πληροφορία, όπως αυτή προκύπτει από τη διαδικασία της τμηματοποίησης, παρατηρούμε ότι η ορθή ρύθμιση των παραμέτρων, στην πλειοψηφία των εικόνων, αποδίδει ικανοποιητικά αποτελέσματα με το λάθος να περιορίζεται μόλις σε μερικά εκατοστά. Σε περιπτώσεις που η τμηματοποίηση δεν αποδίδει ορθά τις χρωματικές περιοχές, η διαδικασία bins που περιγράψαμε στο κεφάλαιο 5, περιορίζει, σε ένα βαθμό, την πιθανή αρνητική επίδρασή της στην πλειοψηφία τέτοιων ΕΠΑΜ. Ωστόσο, σε περιπτώσεις που εντός των τελευταίων εμφανίζεται πληθώρα διαφορετικών χρωματικών τμημάτων, προκύπτουν τοπικά σημαντικές αποκλίσεις μεταξύ των πραγματικών και των υπολογιζόμενων τιμών, ιδιαίτερα όταν εμπλεκόμενες επιφάνειες του παρασκηνίου αποδίδονται στο προσκήνιο και αντίστροφα (dataset Xtion, δείγμα 6). Η κατάσταση αυτή, παρότι περιορίζεται σε μικρό ποσοστό των ΕΠΑΜ και μόλις σε ένα περιορισμένο αριθμό των τμημάτων τους, είναι δυνατό να εκτοξεύσει το μέσο τετραγωνικό λάθος. Ο δείκτης ΛΟΠΒ, σε συνδυασμό με την τιμή του τελευταίου, υποδεικνύουν το παραπάνω συμπέρασμα στην πλειοψηφία τέτοιων περιπτώσεων. Σημαντική επίδραση στην τμηματοποίηση έχει, όπως είναι φυσικό, και η ύπαρξη γειτονικών επιφανειών ίδιας χρωματικής απόχρωσης, καθώς είναι αρκετά πιθανό να αποδοθούν ως ένα χρωματικό τμήμα. Όπως και πριν, η διαδικασία bins καταφέρνει να περιορίσει, συχνά, τους λανθασμένους υπολογισμούς, ωστόσο υπάρχουν περιπτώσεις που αυτό είναι αναπόφευκτο (dataset Kinect, δείγμα 23). Τέλος, παρόμοια με την εύρεση ανακλαστικών επιφανειών, ο φωτισμός ενός χώρου, τόσο τοπικά, όσο και στο σύνολό του, παρουσιάζει σημαντική επίδραση στην τμηματοποίηση (dataset Kinect, δείγμα 17). Χαρακτηριστικό παράδειγμα αποτελούν οι περιπτώσεις όπου δύο κάθετες επιφάνειες (τοίχος και πάτωμα) συναντίονται, με τη διεύθυνση λήψης να είναι σχεδόν παράλληλη ως προς τη μία (πάτωμα), με τον εμφανιζόμενο θόρυβο κοντά στο τέλος της έκτασης της παράλληλης στη διεύθυνση λήψης επιφάνειας, να λαμβάνει τοπικά χαμηλότερες από τις αναμενόμενες τιμές (dataset Kinect, δείγμα 10).

Παρατηρώντας τους πίνακες της παραγράφου 6.4 που σχετίζονται με τον χρόνο, συμπεραίνουμε ότι η μέση ταχύτητα επεξεργασίας είναι λίγο χαμηλότερη από ένα πλαίσιο ανά δευτερόλεπτο, με καλύτερη εμφανιζόμενη επίδοση κοντά στα τρία. Σε ορισμένες περιπτώσεις, ωστόσο, όπου εμφανίζονται ΕΠΑΜ σχετικά μεγάλου μεγέθους με πλήθος διαφορετικών χρωματικών τμημάτων, η καθυστέρηση που εισάγει κυρίαρχα η διαδικασία



της τμηματοποίησης είναι καταλυτική σε αυτή της συνολικής επεξεργασίας, με χρόνους που, ακόμα και με τη χρήση threading, ξεπερνούν τα 1.5 δευτερόλεπτα. Σημαντική επίδραση στο χρόνο μπορεί να έχει, επίσης, η επαναληπτική διαίρεση μίας ΕΠΑΜ, λόγω μορφολογίας, που συμβαίνει με τη διαδικασία solidify noise, προκαλώντας την πολλαπλή κλήση της πιο αργής από τις δημιουργούμενες από εμάς διαδικασίες, αυτή της εύρεσης του μέτρου και της διεύθυνσης των μεταβολών του βάθους (grad). Ως αποτέλεσμα, συγκρίνοντας αυτούς, με εκείνους των ευρύτερων εφαρμογών στις οποίες θα μπορούσε να είναι χρήσιμη η λειτουργικότητα μας, συμπεραίνουμε ότι η ενσωμάτωσή της, στην παρούσα μορφή, θα μπορούσε να είναι εφικτή μόνο σε μια offline διαδικασία.

Σημειώνουμε ωστόσο στο σημείο αυτό, ότι τα παρουσιασμέμα αποτελέσματα προέκυψαν από πειράματα που διεξήχθησαν στον φορητό υπολογιστή, οι τεχνικές προδιαγραφές του οποίου παρουσιάστηκαν στην Παράγραφο 6.1, ο οποίος είναι σχετικά παλιάς τεχνολογίας (επεξεργαστής Intel Core i5, 1.7 GHz). Σε ένα σύγχρονο υπολογιστικό σύστημα με αισθητά μεγαλύτερη υπολογιστική ισχύ ο χρόνος εκτέλεσης μπορεί να μειωθεί σημαντικά, καθιστώντας την απόδοση του συστήματος ικανή για την ενσωμάτωση του σε εφαρμογές.



7. Συμπεράσματα – Ανοιχτά Θέματα

Στο κεφάλαιο αυτό διατυπώνουμε συνοπτικά τα συμπεράσματα τα οποία εξήχθησαν κατά την ολοκλήρωση της διπλωματικής εργασίας, συνοψίζοντας τα κεντρικά σημεία και τις αδυναμίες του αλγορίθμου αφαίρεσής θορύβου που υλοποιήθηκε. Επίσης, προτείνουμε μια σειρά από μελλοντικές επεκτάσεις της εργασίας για την αντιμετώπιση των αδυναμιών αυτών και τη γενική βελτίωση της λειτουργικότητας του συστήματος.

7.1 Συμπεράσματα

Τα συμπεράσματα που εξήχθησαν κατά την ολοκλήρωση της διπλωματικής εργασίας, τόσο σε ότι αφορά την ποιότητα του τελικού υλοποιημένου συστήματος, όσο και την απόπειρα ενσωμάτωσής του σε low – latency ρομποτικές εφαρμογές είναι τα εξής:

1. Ο αλγόριθμος που υλοποιήθηκε έχει στη μαζική πλειοψηφία του dataset μεγάλη ακρίβεια στη διόρθωση του θορύβου, επιτυγχάνοντας μέσο τετραγωνικό σφάλμα ~ 6 cm² και διορθώνοντας με μέσο σφάλμα κάτω από το υποφερτό όριο των 3.5 cm κατά μέσο όρο ποσοστό μεγαλύτερο του 85% των περιοχών θορύβου που υπάρχουν στην εικόνα.
2. Η ακρίβεια του συστήματος βασίζεται σχεδόν αποκλειστικά στην ποιότητα της τμηματοποίησης χρώματος, καθιστώντας την τη βασικότερη διαδικασία του αλγορίθμου. Η τμηματοποίηση χρώματος είναι αυτή που θα καθορίσει τις ομάδες τμημάτων ΕΠΑΜ – περιοχών έγκυρου βάθους, διασπώντας έτσι κάθε περιοχή θορύβου σε περιοχές και ταξινομώντας σε αντίστοιχα αντικείμενα ίδιου χρώματος στην RGB εικόνα. Ύπαρξη μεγάλου ποσοστού θορύβου στην εικόνα, ύπαρξη συγχωνευμένου θορύβου που οφείλεται σε διαφορετικές πηγές, κακές συνθήκες φωτισμού της σκηνής και γενικά εκφυλισμένη έγχρωμη πληροφορία, οδηγεί σε λανθασμένες τμηματοποιήσεις χρώματος, η οποία με τη σειρά της οδηγεί στην εισαγωγή μεγάλων σφαλμάτων.
3. Η προσέγγιση μας στην αναγνώριση ΕΠΑΜ ως ανακλαστικές επιφάνειες, παρότι παρουσιάζει εκπληκτική ακρίβεια για την απλότητα και το χαμηλό χρόνο διεκπεραίωσής της, επηρεάζεται σημαντικά από μία σειρά παραμέτρων, όπως αυτές παρουσιάστηκαν στο κεφάλαιο 4. Δεδομένου ότι η λήψη των datasets πραγματοποιήθηκε σε συγκεκριμένους εσωτερικούς χώρους, είναι κατανοητό ότι η προκαταβολική ρύθμισή τους για ένα τυχαίο περιβάλλον θα αποτελούσε σημαντική πρόκληση.
4. Η δοκιμή του αλγορίθμου σε πρακτικές εφαρμογές, δηλαδή σε εικόνες δοσμένες από υλικό που αποτυπώνουν πραγματικές σκηνές σε πραγματικό χρόνο, σε αντίθεση με τη δοκιμή σε έτοιμα προεπεξεργασμένα datasets, οδηγεί συχνά σε απρόβλεπτες και προβληματικές συμπεριφορές, ανάλογα με τις συνθήκες διεξαγωγής του πειράματος. Άλλαζοντας το περιβάλλον που αποτυπώνεται, τη σχετική θέση και ταχύτητα στροφής της κάμερας (η οποία εφαρμόζεται πάνω σε



κάποιο πράκτορα) και κυρίως τις συνθήκες φωτισμού της σκηνής, υπάρχουν περιπτώσεις που ο αλγόριθμος καθυστερεί σημαντικά ή (και) οδηγεί σε λανθασμένες εκτιμήσεις βάθους. Είναι σημαντικό λοιπόν, προτού εφαρμοστεί το σύστημα αφαίρεσης θορύβου σε κάποια πρακτική εφαρμογή, η διεξαγωγή πειραμάτων και η διόρθωση παραμέτρων στις εκάστοτε συνθήκες.

5. Στην εργασία μας το ενδιαφέρον επικεντρώθηκε στην επίτευξη υψηλής ακρίβειας σε κόστος της απόδοσης του υλοποιημένου συστήματος. Έτσι, βλέπουμε ότι σε περιπτώσεις ασθενούς θορύβου ή θορύβου που οφείλεται αποκλειστικά σε ανακλαστικές επιφάνειες, το σύστημα μπορεί να λειτουργήσει σε frame rate εώς και 3 fps, στην πλειοψηφία περιπτώσεων με διαφορετικές πηγές θορύβου στα ~ 1 fps, ενώ σε περιπτώσεις με ισχυρό ή υψηλά συγχωνευμένο θόρυβο μπορεί να ξεπεράσει σε χρόνο εκτέλεσης το 1 sec, καθιστώντας το μη πρακτικό για την ενσωμάτωση του σε κάποια διανεμημένη αρχιτεκτονική που χρησιμοποιεί κάμερα βάθους. Ωστόσο, σε εφαρμογές που ο πράκτορας κινείται σε περιβάλλοντα με υψηλά ποσοστά θορύβου (εσωτερικά περιβάλλοντα) και σε χαμηλές ταχύτητες κίνησής του, το σύστημα μπορεί να αξιοποιηθεί καθώς παρότι καθυστερεί σημαντικά για την εξαγωγή της ορθής εικόνας βάθους, αυτή είναι πολύ πιο αξιοποιήσιμη σε σχέση με την αρχική. Συνεπώς, χρόνος που ενδεχομένως απαιτείται για τη σωστή απεικόνιση του περιβάλλοντος από τον πράκτορα με χρήση θορυβώδους εικόνας βάθους, προκειμένου να διεξαχθεί π.χ. ένα έγκυρο μονοπάτι, εξοικονομείται με την χρήση του συστήματος αφαίρεσης θορύβου. Χαρακτηριστικά το σημείο αυτό φαίνεται στην Εικόνα 7.1.1, όπου με χρήση του πακέτου depth_image_proc (βλ. Παράγραφο 3.4.4) έχουμε εξάγει το τρισδιάστατο μοντέλο του περιβάλλοντος της σκηνής που θα χρησιμοποιήσει ο πράκτορας συγκριτικά μεταξύ των δυο εικόνων βάθους.



Εικόνα 7.1.1: Επίδειξη τρισδιάστατου μοντέλου σκηνής πριν (αριστερά) και μετά (δεξιά) τη διόρθωση

7.2 Ανοιχτά Θέματα

Στην παράγραφο αυτή θα συγκεντρώσουμε τα βασικά ζητήματα που αντιμετωπίζει το σύστημα μας, θα επικεντρωθούμε στην πηγή του προβλήματος και θα προτείνουμε ενδεχόμενες μελλοντικές λύσεις των ζητημάτων. Όπως φαίνεται από τα συμπεράσματα που βγήκαν από την εξέταση των αποτελεσμάτων των πειραμάτων στην προηγούμενη



παράγραφο, τα βασικότερα ζητήματα της υλοποίησής μας σχετίζονται γύρω από την αποτυχία εκτίμησης σωστού βάθους για περιοχή θορύβου εξαιτίας σφάλματος τμηματοποίησης, καθώς και από το σημαντικά χαμηλό frame rate του συστήματος, κάτι που τον καθιστά μη εύχρηστο για τις περισσότερες ρομποτικές εφαρμογές.

(α) Η ευστάθεια παραμέτρων

Για διαφορετικές συνθήκες φωτισμού, έμφραξη πολλαπλών αντικειμένων, παρεμβολή αντικειμένου από μεγαλύτερη επιφάνεια, γειτονικές περιοχές που ανήκουν σε διαφορετικά αντικείμενα αλλά έχουν παρόμοια απόχρωση στο RGB κ.α., η τμηματοποίηση χρώματος για τις default παραμέτρους του συστήματος της παραγράφου 5 δεν δίνουν πάντα το καλύτερο δυνατό αποτέλεσμα. Ο χρήστης πρέπει να διερευνήσει τις συνθήκες που επικρατούν στην έγχρωμη εικόνα της σκηνής και να ρυθμίσει κατάλληλα τις παραμέτρους για να πάρει το καλύτερο δυνατό αποτέλεσμα τμηματοποίησης, άρα και τελικής αφαίρεσης θορύβου. Η ακρίβεια του αλγορίθμου εξαρτάται σημαντικά από την ακρίβεια της τμηματοποίησης, καθώς λανθασμένες ταξινομήσεις περιοχών σε διαφορετικά αντικείμενα κοστίζουν, εισάγοντας πολύ μεγάλα τετραγωνικά σφάλματα.

Οι παράμετροι αυτές, μπορούν να εισαχθούν ως παράμετροι εντός της πλατφόρμας ROS και να ενημερώνονται σε πραγματικό χρόνο κατά τη διάρκεια της εκτέλεσης του προγράμματος από τον χρήστη, εφαρμόζοντας έτσι το λεγόμενο *tuning*. Έτσι, ο κάθε χρήστης του πακέτου, έχοντας δεδομένο περιβάλλον στο οποίο θα πλοηγείται ο ρομποτικός πράκτορας που επιθυμεί να προγραμματίσει, θα μπορεί να τραβήξει δείγματα εικόνων, να εφαρμόσει τον αλγόριθμο και να ρυθμίσει τις παραμέτρους κατάλληλα ώστε να δίνουν τη βέλτιστη συμπεριφορά για το συγκεκριμένο περιβάλλον. Η διαδικασία αυτή είναι δημοφιλής για διάφορες εφαρμογές εντός του ROS και αποκαλείται *dynamic reconfiguration*.

(β) Η ταχύτητα εκτέλεσης

Όπως φαίνεται στην Παράγραφο 6.3, ο μέσος χρόνος λειτουργίας του συστήματος για την παραγωγή μιας καθαρής εικόνας βάθους από ένα ζεύγος δεδομένων είναι περίπου στα 0.8 seconds. Αυτό σημαίνει ότι κατά την εφαρμογή του συστήματος ο κόμβος αφαίρεσης θορύβου σκιάς μπορεί να λειτουργεί σε μέγιστο frame rate 1 fps. Οι περισσότερες ρομποτικές εφαρμογές χρησιμοποιούν ελεγκτές που λειτουργούν σε μεγαλύτερες συχνότητες, για να γίνονται εγκαίρως οι απαραίτητες διαδικασίες χαρτογράφησης, εντοπισμού θέσης και σχεδιασμού πλάνου. Η Python σαν γλώσσα προγραμματισμού εμφανίζει χειρότερο γενικό performance από τη C++, με αποτέλεσμα μία υλοποίηση των διαδικασιών μας στη δεύτερη να είναι αναμενόμενο να επιφέρει βελτιώσεις στη ταχύτητα επεξεργασίας. Επίσης, Όπως αναφέραμε και στην παράγραφο περιγραφής των καμερών, συχνά παρατηρείται το φαινόμενο εναλλαγής μεταξύ τιμών έγκυρου βάθους και μηδενικών στα ίδια pixels διαδοχικών frames. Το πρόβλημα αυτό οφείλεται συχνά είτε στη γωνία λήψης, είτε στο φωτισμό επάνω στη συγκεκριμένη επιφάνεια. Ακόμη, κυρίως σε νεότερες κάμερες, όπως η RealSense της Intel, παρουσιάζονται διακυμάνσεις σε έγκυρες τιμές βάθους από frame σε frame. Μία λογική αντιμετώπισης θα ήταν η ανάπτυξη και εισαγωγή έξυπνου ανιχνευτή θορύβου που αφαιρεί από διαδοχικές εικόνες παροδικό θόρυβο ή θόρυβο που έχει ήδη διορθώσει. Τέλος, μιας και τη σημαντικότερη πηγή καθυστέρησης αποτέλεσε η τμηματοποίηση, προτείνουμε την υλοποίηση διαφορετικού αλγορίθμου τμηματοποίησης, που δίνει σταθερό άνω όριο από τμήματα για τις ίδιες



διαστάσεις εικόνας και ανεξάρτητα από την υφή (*texture*). Η ακρίβεια, σε μία τέτοια περίπτωση, θα μειωθεί, ωστόσο η απόδοση θα αυξηθεί δραματικά. Ούτως ή άλλως, σε ένα πραγματικό πρόβλημα, όπου κάποιο ρομποτικό όχημα είναι σε κίνηση με την κάμερα πάνω του, εμφανίζονται συχνά αντικείμενα σε κοντινά επίπεδα βάθους, με αποτέλεσμα μία τέτοια τμηματοποίηση να μην αποφέρει εξαιρετικά μεγάλο σφάλμα.

Παράρτημα Α: OpenCV



A1) Χρωματικοί χώροι – Colorspace

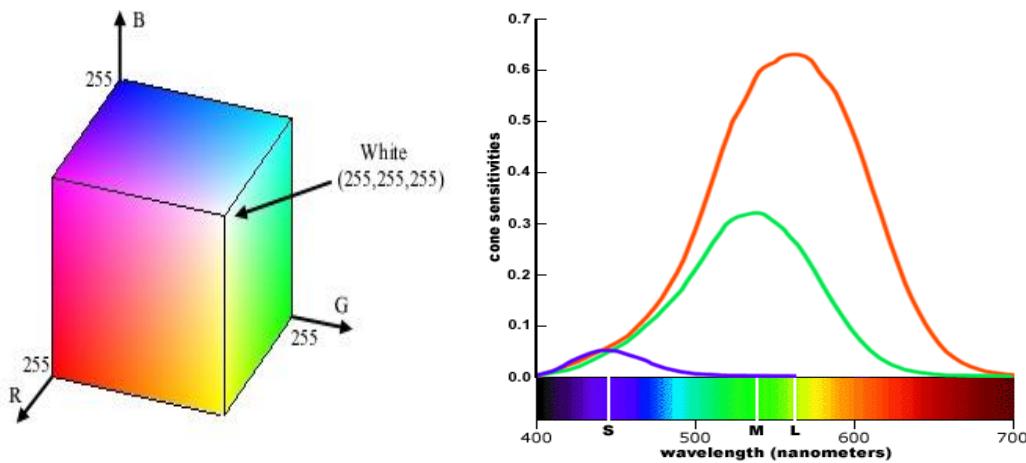
Τα color models αποτελούν αφηρημένα μαθηματικά μοντέλα, που περιγράφουν τον τρόπο με τον οποίο μπορούμε να αναπαραστήσουμε τα χρώματα με η – άδες αριθμών (συνήθως τριάδες), γνωστές ως χρωματικές συνιστώσες. Όταν ένα τέτοιο μοντέλο συνδέεται με μία συγκεκριμένη περιγραφή για τον τρόπο ερμηνείας των συνιστωσών του, το προκύπτον σύνολο χρωμάτων συνιστά ένα colorspace. Όπως είναι κατανοητό, η ύπαρξη ενός color model χωρίς την ύπαρξη καθολικών περιγραφών που το καθιστούν κατανοητό και ευρέως αποδεκτό ως colorspace είναι αυθαίρετη και απουσιάζει χρησιμότητας. Αντιθέτως, η ύπαρξη μίας συνάρτησης αντιστοίχισης μεταξύ τους προσδιορίζει το colorspace με μοναδικά αποδεκτό και αναγνωρισμένο τρόπο. Το γνωστότερο από τα color models που χρησιμοποιείται σαν μία τέτοια κλίμακα είναι το RGB, ενώ συχνά μπορεί να γίνεται αναφορά στα CIEXYZ και CIELAB, καθώς τα τελευταία σχεδιάστηκαν ειδικά με στόχο την απόδοση όλων των διαφορετικών χρωμάτων που μπορεί να διακρίνει ένα μέσο ανθρώπινο μάτι. Παρακάτω, θα περιγράψουμε συνοπτικά μερικά από τα σημαντικότερα color models και μερικά από τα colorspaces τους, εστιάζοντας περισσότερο σε εκείνα που χρησιμοποιήθηκαν ή δοκιμάστηκαν στα πλαίσια της παρούσας διπλωματικής εργασίας.

Το RGB model απορρέει από την προσπάθεια αποτύπωσης της χρωματικής αντίληψης του ανθρώπινου ματιού. Συγκεκριμένα, στο τελευταίο εμφανίζονται μία σειρά από κύτταρα που ονομάζονται φωτοϋποδοχείς, οι οποίοι χωρίζονται σε δύο βασικές κατηγορίες: τα ραβδία και τα κωνία. Τα πρώτα είναι υπεύθυνα για την αντίληψη του αμυδρού φωτός, ενώ τα δεύτερα για την αντίληψη των χρωμάτων. Υπάρχουν τρία είδη κωνικών κυττάρων:

- **S – κωνία:** είναι ευαίσθητα σε φωτόνια μικρού μήκους κύματος και παρουσιάζουν μέγιστη ευαισθησία σε μήκος κύματος περίπου 4.200 Å (420 nm), δηλαδή στο μπλε φως.
- **M – κωνία:** είναι ευαίσθητα σε φωτόνια μεσαίου μήκους κύματος και παρουσιάζουν μέγιστη ευαισθησία σε μήκος κύματος περίπου 5.300 Å (530 nm), δηλαδή στο πράσινο φως.
- **L – κωνία:** είναι ευαίσθητα σε φωτόνια μεγάλου μήκους κύματος και παρουσιάζουν μέγιστη ευαισθησία σε μήκος κύματος περίπου 5.600 Å (560 nm), δηλαδή στο κόκκινο φως.

Με βάση τα παραπάνω, προκύπτουν και οι καμπύλες απορρόφησης του φωτός από κάθε κατηγορία κωνίων, οι οποίες παρουσιάζονται παρακάτω.

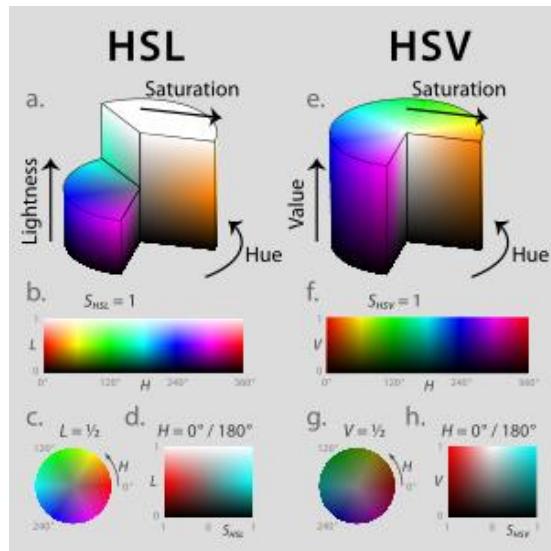




Εικόνα A.1.1: Χρωματικός κύβος RGB – Καμπύλες χρωματικής απορρόφησης κωνών

Στο πλαίσιο αυτό, η δημιουργία των μέσων αναπαραγωγής εικόνων, όπως οι τηλεοράσεις, χρησιμοποιούν χρωματικές μίξεις με τα χρώματα κόκκινο, πράσινο και μπλε ως βασικά. Το ευρύ φάσμα των τελευταίων καλύπτει το σημαντικότερο μέρος του λεγόμενου ανθρώπινου colorspace και παράγει την πλειοψηφία των χρωματικών μας αντιλήψεων. Δυστυχώς, δεν υπάρχει επικρατούσα άποψη ως προς τον ακριβή προσδιορισμό μίας καθαρής μορφής των τριών παραπάνω χρωμάτων, με αποτέλεσμα για τις ίδιες τιμές στα R, G και B κανάλια να αντιλαμβανόμαστε συχνά διαφορετική απόχρωση σε ένα σύνολο συσκευών απεικόνισης. Ακόμη, για δύο φωτεινές πηγές που συνίστανται από ένα εύρος διαφορετικών μηκών κύματος, είναι πιθανό να αντιληφθούμε το ίδιο χρώμα, γνωστό και ως φαινόμενο του μεταμερισμού. Λόγω της εν γένει προβληματικής αυτής γεωμετρίας και αστάθειας της χρωματικής αντίληψης, την δεκαετία του 1970 αναπτύχθηκαν από ερευνητές του κλάδου της γραφικής υπολογιστών δύο διαφορετικά color models, τα HSV (Hue, Saturation, Value) και HSL (Hue, Saturation, Lightness), που αποτελούν μη γραμμικές παραμορφώσεις του RGB. Το πρώτο κανάλι (Hue) αντιστοιχεί στο όρο απόχρωση και προσδιορίζεται από ένα χρωματικό κύλινδρο γύρω από έναν άξονα ουδέτερων χρωμάτων, όπως φαίνεται στις παρακάτω εικόνες. Τα πλήρως κορεσμένα (saturated) χρώματα κάθε απόχρωσης (εκείνα με τη μεγαλύτερη ένταση) βρίσκονται στην περιφέρεια ενός κύκλου μίας οποιασδήποτε τομής του κυλίνδρου, με τον κορεσμό να μειώνεται με κατεύθυνση προς το κέντρο. Στο HSV, το ύψος στο οποίο πραγματοποιείται μία οποιαδήποτε τομή προσδιορίζει το κανάλι Value, το οποίο αποτυπώνει την απόσταση μίας απόχρωσης από το μαύρο χρώμα που της αντιστοιχεί, ενώ η απόχρωση του λευκού τείνει να βρεθεί προς το κέντρο του κυλίνδρου. Αντίθετα, το HSL, σε μία προσπάθεια καλύτερης απόδοσης της διαισθητικής έννοιας των χρωμάτων, αντιστοιχεί το ύψος στη φωτεινότητα (Lightness), με τη βάση του κυλίνδρου αναφοράς να αποτελεί το μαύρο και την κορυφή το λευκό. Η περιγραφή αυτή απεικονίζεται στην παρακάτω εικόνα.

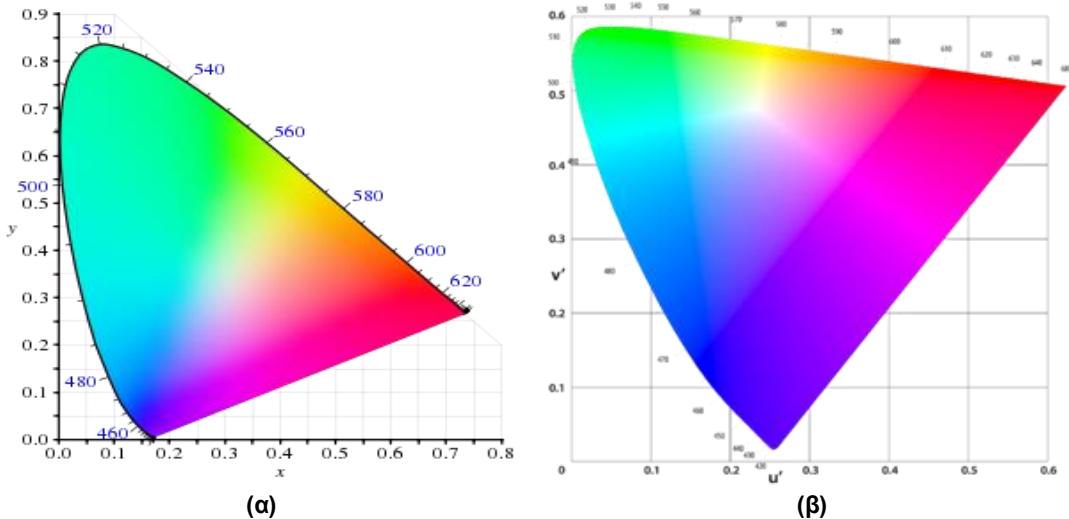




Εικόνα A.1.2: Σχηματική αναπαράσταση των HSL και HSV

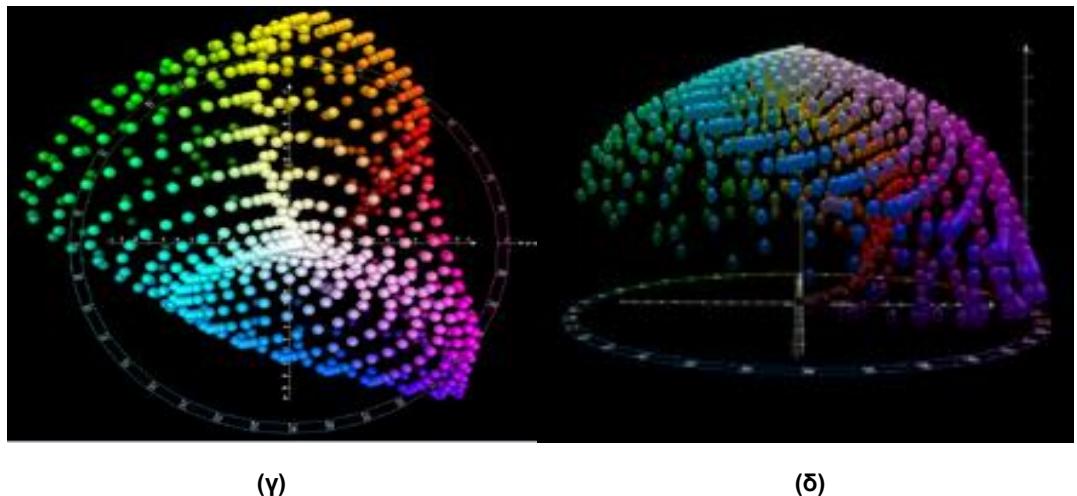
Η πρώτη από τις κατηγορίες colorspaces, που σύνδεσαν την κατανομή των μηκών κύματος με το ορατό φάσμα και την ανθρώπινη αντίληψη αυτού ήταν η CIE 1931. Η συγκεκριμένη δημιουργήθηκε, όπως φαίνεται από την ονομασία, το 1931 και βασίστηκε, όπως μετέπειτα και το RGB, στο τρόπο με τον οποίο το ανθρώπινο μάτι αντιλαμβάνεται τη χρωματική εναλλαγή, ενώ αποτελεί και τη βάση των περισσότερων σύγχρονων μοντέλων. Μερικά από αυτά, δύο από τα οποία αναφέραμε και νωρίτερα, είναι τα CIE XYZ, CIELUV και CIELAB. Όσο αφορά το πρώτο, η αντιστοίχιση στο σύστημα XYZ οφείλεται στο γεγονός ότι η αίσθηση της φωτεινότητας γίνεται κυρίως αντιληπτή από τα M – κωνία, δηλαδή μέσω του πράσινου φωτός. Ως αποτέλεσμα, η φωτεινότητα (luminance) αντιστοιχήθηκε στο κανάλι Y. Αντίστοιχα, το Z παρουσιάζει μεγάλη ομοιότητα με τη διέγερση των S – κωνίων (μπλε χρώμα), ενώ το X αποτελεί ένα μη αρνητικό γραμμικό συνδυασμό των καμπυλών των κωνίων. Για δεδομένο Y, το επίπεδο XZ περιέχει όλες τις δυνατές αποχρώσεις τις συγκεκριμένης φωτεινότητας. Το CIELUV τώρα αποτελεί έναν εύκολο στον υπολογισμό μετασχηματισμό του προηγούμενου, με κατεύθυνση προς την απόκτηση της διαίσθησης χρωματικής ομοιομορφίας. Βρίσκει εφαρμογή στα γραφικά υπολογιστών και κυρίως σε εικόνες με μίξεις φωτός από πολλαπλές πηγές, λόγω της γραμμικότητάς του. Τέλος, το CIELAB επιχειρεί να δημιουργήσει μία πιο γραμμική αίσθηση στην αντίληψη του χρώματος, με στόχο η αλλαγή σε ένα από τα κανάλια τιμών (L: Lightness, A: συνδυασμός κόκκινου – πράσινου, B: συνδυασμός κίτρινου – μπλε) να αναλογεί σε αντίστοιχη οπτική σημασία. Με τον τρόπο αυτό, προσεγγίζει με τον καλύτερο δυνατό τρόπο την αντίληψη του ανθρώπινου ματιού, σε αντίθεση με τα RGB και CMYK colorspace, τα οποία αποτελούν υποσύνολα του CIELAB ως προς το εύρος δυνατών χρωματικών αντιστοιχίσεων. Πιο συγκεκριμένα, το CIELAB επεκτείνεται πέρα από τις δυνατότητες των σύγχρονων συσκευών αναπαραγωγής εικόνας και της ανθρώπινης χρωματικής αντίληψης. Τέλος, η δομή των συνιστωσών του επιτρέπει διορθώσεις στις ισορροπίες των χρωμάτων και της φωτεινότητας, καθιστώντας το colorspace αυτό ιδανικό για διαδικασίες προεπεξεργασίας τυχαίων εικόνων και ιδανικό για το δικό μας προς επίλυση πρόβλημα.





(α)

(β)



(γ)

(δ)

Εικόνες A.1.3: Χρωματικά διαγράμματα των CIE XYZ (α), CIELUV (β) και CIELAB (άνω και πλάγια όψη – (γ) και (δ))

Στη ίδια λογική με το RGB, σύγχρονες συσκευές εκτύπωσης χρησιμοποιούν ως βάση το σύστημα CMY (Cyan, Magenta, Yellow). Το Cyan αντιστοιχεί στο κυανό, το οποίο απορροφά το κόκκινο, επιτρέποντας τη μετάδοση του πράσινου και μπλε φωτός. Τα Magenta (απόχρωση του κόκκινου) και Yellow (κίτρινο) απορροφούν το πράσινο και μπλε φως αντίστοιχα, επιτρέποντας τα εναπομείναντα. Η YCbCr αποτελεί επίσης σημαντική οικογένεια colorspace, κυρίως στο χώρο της αναπαραγωγής βίντεο και της ψηφιακής φωτογραφίας, καθώς χρησιμοποιείται συχνά σε αλγόριθμους συμπίεσης. Η πρώτη συνιστώσα Y αναπαριστά την φωτεινότητα, ενώ οι Cb και Cr εκφράζουν τη χρωματική διαφορά ως προς το μπλε και κόκκινο αντίστοιχα. Τα δύο color models που μόλις περιγράψαμε δεν χρησιμοποιήθηκαν στα πλαίσια της διπλωματικής, αλλά αναφέρονται λόγω της ευρείας εφαρμογής τους σε σύγχρονες συσκευές.

Τέλος, οι εικόνες που αποδίδονται με αποχρώσεις του γκρι αντιστοιχούν στο gray



colorspace. Η OpenCV επιτρέπει τους μετασχηματισμούς εικόνων από ένα colorspace σε άλλο, με τη διαδικασία cvtColor. Παρακάτω, παρουσιάζουμε την ίδια εικόνα σε μία σειρά από colorspaces, τα οποία περιγράψαμε παραπάνω, όπως αυτή αποδίδεται με χρήση της OpenCV.



Εικόνα A.1.4: Μετασχηματισμός εικόνας σε μία σειρά από colorspaces

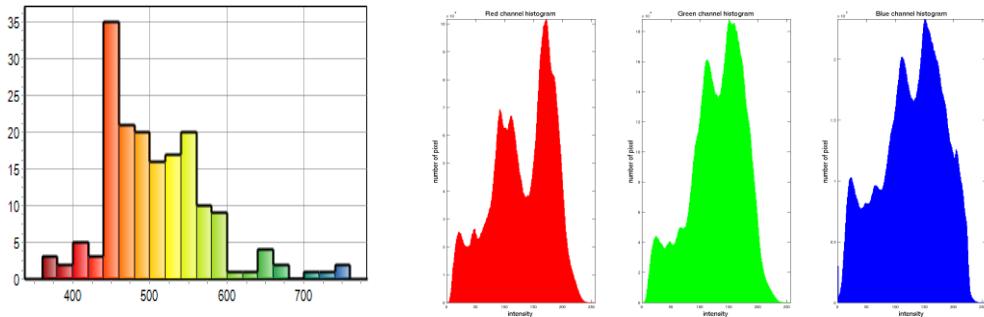
A2) Ιστόγραμμα και Εξισορρόπηση

Το ιστόγραμμα, που προτάθηκε πρώτη φορά από τον Karl Pearson, αποτελεί μία ακριβή αναπαράσταση της διανομής αριθμητικών δεδομένων και μία εκτίμηση της κατανομής πιθανότητας συνεχών μεταβλητών. Κατηγοριοποιείται ως διάγραμμα ράβδων, με την κάθε μία να περιέχει τον αριθμό των εμφανίσεων των τιμών εντός ενός συγκεκριμένου εύρους που ονομάζεται κουβάς (bin). Τα εύρη των bins είναι συχνά ίδια και πάντοτε προκαθορισμένα, χωρίζοντας το συνολικό εύρος των τιμών ενδιαφέροντος σε μη επικαλυπτόμενες περιοχές. Πέρα από την απόδοση της απόλυτης συχνότητας εμφάνισης τιμών εντός ενός εύρους, το ιστόγραμμα μπορεί να αναπαραστήσει τη σχετική συχνότητα με τη μορφή ενός αριθμού μεταξύ του μηδέν και της μονάδας. Η χρήση τους είναι διαδεδομένη σε ποικιλία τεχνολογικών τομέων επεξεργασίας πληροφορίας. Ένας από αυτούς είναι και η επεξεργασία εικόνας, με τα σχετικά εργαλεία, όπως η OpenCV, να παρέχουν τη δυνατότητα παραγωγής και επεξεργασίας τους, μέσω της calcHist.

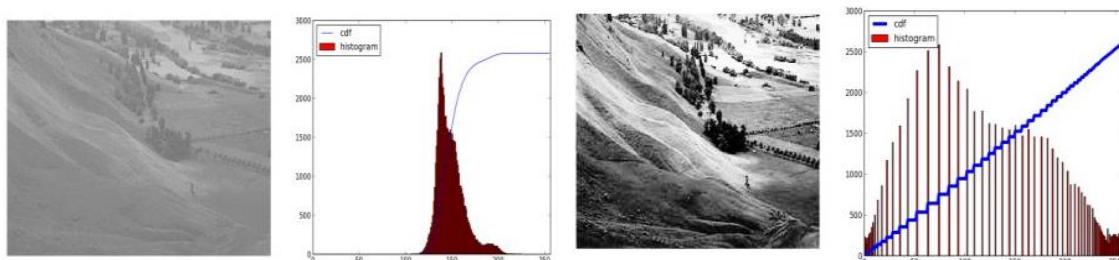
Μία από τις σημαντικότερες μεθόδους επεξεργασίας ιστογραμμάτων είναι η λεγόμενη εξισορρόπηση. Η τελευταία πετυχαίνει μία καλύτερη διανομή των τιμών του ιστογράμματος στο συνολικό εύρος τιμών, βασιζόμενη στην συχνότητα εμφάνισης τους σε μία περιοχή, με αποτέλεσμα μία πιο ομοιόμορφη κατανομή. Επεξεργαζόμενοι το ιστόγραμμα των τιμών καναλιών πληροφορίας σε μία εικόνα, πετυχαίνουμε καλύτερη αντίθεση (contrast), καθώς τονίζονται περισσότερο τα bins με μικρότερες συχνότητες εμφάνισης (περιοχές χαμηλής φωτεινότητας) και όχι μόνο εκείνα με υψηλές συχνότητες (υψηλή φωτεινότητα). Η OpenCV επιτρέπει την παραπάνω επεξεργασία με τη διαδικασία equalizeHist. Συχνά, στη



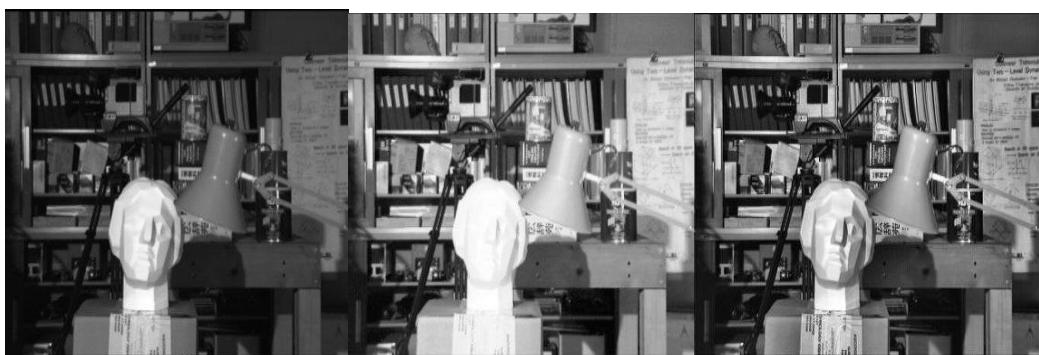
προσπάθειά μας να έχουμε όσο λιγότερη απώλεια πληροφορίας ως προς το texture της εικόνας μετά από μία τέτοια διαδικασία, θα θέλαμε η εξισορρόπηση να λαμβάνει υπόψην κάθε φορά μόνο την τοπική πληροφορία των εντάσεων. Μία λογική είναι αυτή του διαχωρισμού της εικόνας σε μικρότερα pixel blocks μέσα στα οποία πραγματοποιείται η εξισορρόπηση, η οποία στην OpenCV επιτυγχάνεται με τη χρήση του αντικειμένου clahe και της εφαρμογής του σε μία εικόνα. Για απόδοση της ομοιομορφίας λόγω του ενισχυμένου contrast, στο τέλος μεταξύ των γειτονικών pixel blocks πραγματοποιείται διωνυμική παρεμβολή (bilinear interpolation). Στη συνέχεια, παρουσιάζουμε τη δομή ενός ιστογράμματος, καθώς και το αποτέλεσμα της εξισορρόπησης σε μία εικόνα.



Εικόνα A.2.1: Ιστόγραμμα διακριτής (αριστερά) και συνεχών (δεξιά) μεταβλητών



Εικόνα A.2.2: Πρότυπο εικόνας με το ιστόγραμμα της πριν (αριστερά) και μετά (δεξιά) την εξισορρόπηση



Εικόνα A.2.3: Επίδειξη της απλής εξισορρόπησης και της προσαρμοσμένης με clahe



A3) Thresholding Εικόνας

Το thresholding αποτελεί την απλούστερη μέθοδο τμηματοποίησης μίας εικόνας. Η βασική της λειτουργία επιτυγχάνει τη δημιουργία μίας εικόνας ίδιων διαστάσεων με την αρχική, όπου η τιμή κάθε pixel αντικαθίσταται από τη μηδενική (μαύρο pixel), όταν η ένταση του αντίστοιχου της αρχικής εικόνας είναι μικρότερη από ένα συγκεκριμένο όριο (threshold) και από τη μέγιστη δυνατή (άσπρο pixel) στην αντίθετη περίπτωση. Η παραπάνω αποτελεί μόνο μία γενική περιγραφή της διαδικασίας. Στην πραγματικότητα, υπάρχει μία ευρεία σειρά τέτοιων μεθόδων, όπως αυτές ορίστηκαν από τους T. Senzgin και Sankur (2004), ανάλογα με την πληροφορία, την οποία ο αλγόριθμος καλείται να διαχειριστεί:

Α) Μέθοδοι βασισμένες στη μορφή του ιστογράμματος της εικόνας, όπου αναλύονται οι μορφολογικές ιδιότητες (καμπύλες, κορυφές και άλλα) του ιστογράμματος των εντάσεων των pixels της εικόνας

Β) Μέθοδοι βασισμένες στην τμηματοποίηση της εικόνας, όπου επιτυγχάνεται ο διαχωρισμός της εικόνας σε περιοχές που ανήκουν στο προσκήνιο (foreground) και στο παρασκήνιο (background) ή η μοντελοποίησή τους ως μία μίξη δύο γκαουσιανών καμπυλών

Γ) Μέθοδοι βασισμένες στην εντροπία, όπου μπορεί να χρησιμοποιηθούν οι εντροπίες του προσκηνίου (foreground) και του παρασκηνίου (background), η συνδυαστική εντροπία της αρχικής εικόνας και της προκύπτουσας δυαδικού περιεχομένου (binary image).

Δ) Μέθοδοι βασισμένες σε χαρακτηριστικά αντικειμένων εντός του πλαισίου, όπου αναζητείται μία μετρική ομοιότητας μεταξύ των εντάσεων της εικόνας αποχρώσεων του γκρι (gray – scale) και εκείνης δυαδικού περιεχομένου (binary image), όπως οριακή ομοιότητα διαμορφούμενων σχημάτων, διαμόρφωση ακμών και άλλα

Ε) Χωρικές μέθοδοι, οι οποίοι χρησιμοποιούν μεγαλύτερης τάξης πιθανοτικές κατανομές και, συχνά, συσχέτιση μεταξύ των pixels

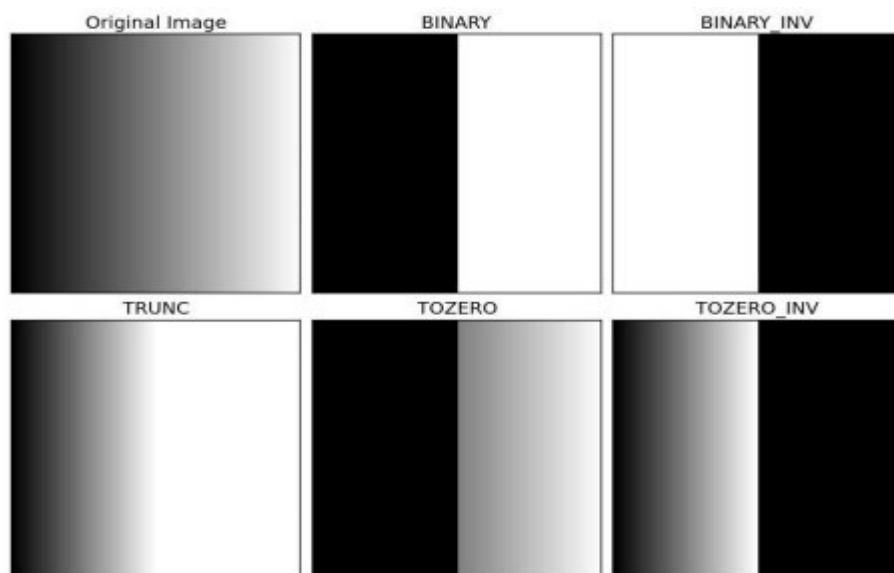
ΣΤ) Μέθοδοι οι οποίες βασίζονται σε τοπικά χαρακτηριστικά στις διάφορες περιοχές ενός πλαισίου, χρησιμοποιώντας κάθε φορά ένα διαφορετικό φίλτρο

Όταν η διαδικασία του thresholding εφαρμόζεται σε εικόνες αποχρώσεων του γκρι (gray – scale), προκύπτει εικόνα δυαδικού περιχειομένου (binary image). Ωστόσο, η διαδικασία μπορεί να εφαρμοστεί ακόμη και σε έγχρωμες εικόνες. Μία λογική αποτελεί η εφαρμογή σε κάθε ένα από τα κανάλια RGB χρωματικής πληροφορίας και η μετέπειτα σύνθεσή τους με τη μέθοδο λογικού AND. Η προκύπτουσα εικόνα αντικατοπτρίζει τον τρόπο που λειτουργεί η κάμερα και αποθηκεύονται τα δεδομένα στον υπολογιστή, όμως, όπως αναφέραμε νωρίτερα, δεν αποδίδει ορθά την αντίληψη του ανθρώπινου ματιού για το χρώμα. Για το λόγο αυτό, χρησιμοποιούμε άλλα χρωματικά μοντέλα, όπως το HSL ή το HSV, με τη μόνη διαφορά ότι το πρώτο κανάλι (Hue), λόγω της κυκλικής του κλίμακας, υφίσταται κυκλικό (circular) thresholding. Ένα ακόμα χρωματικό μοντέλο το οποίο μπορεί να χρησιμοποιηθεί είναι το CMYK.

Για τις ανάγκες της παρούσας διπλωματικής εργασίας, εφαρμόστηκε συχνά η διαδικασία του thresholding με τη βοήθεια εργαλείων της OpenCV. Τα εργαλεία αυτά χρησιμοποιούν



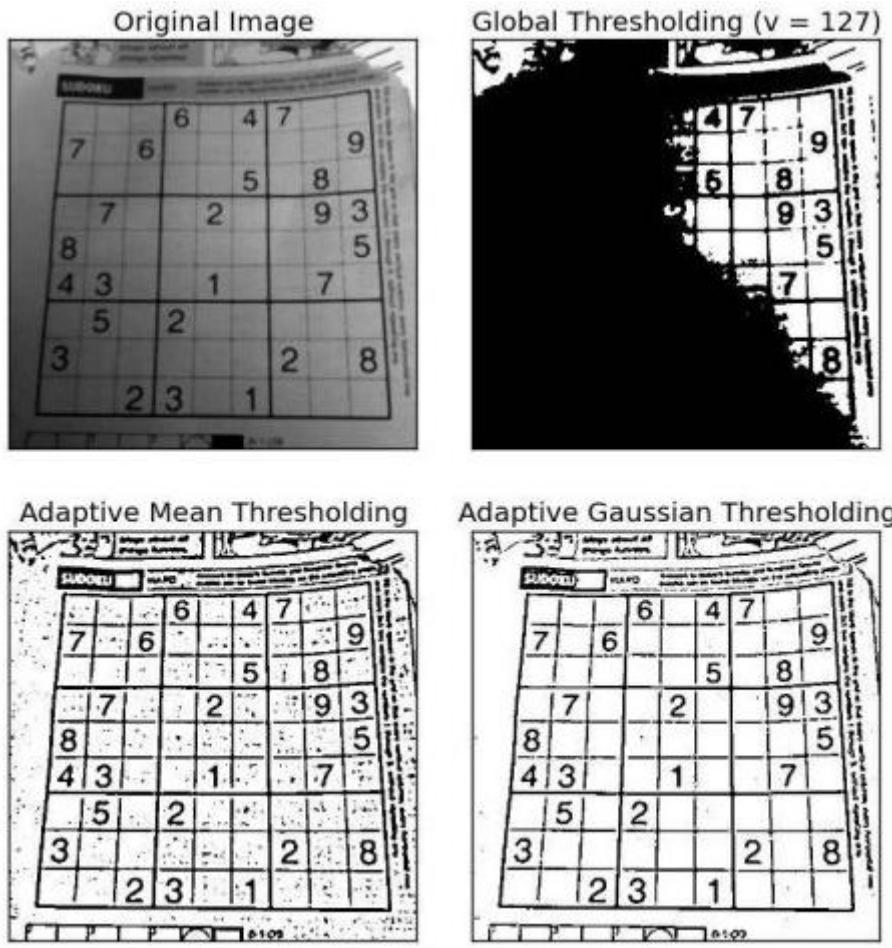
την εικόνα αποχρώσεων του γκρι (gray – scale) και δύο τιμές, με τη μία να καθορίζει το όριο πριν και μετά το οποίο αλλάζουν οι αποδιδόμενες τιμές έντασης στη νέα εικόνα και την άλλη να προσδιορίζει τη μέγιστη δυνατή εμφανιζόμενη τιμή της κλίμακας του γκρι στην τελευταία, ενώ παρέχουν μία σειρά από διαφορετικές λογικές προς εφαρμογή. Πιο συγκεκριμένα, η binary αποτελεί την απλούστερη λογική, αντιστοιχίζοντας τη μέγιστη τιμή έντασης σε όποια pixels έχουν τιμή μεγαλύτερη του αποδιδόμενου ορίου και το μαύρο στα υπόλοιπα. Το αντίστροφο ακριβώς αποτέλεσμα, όπως φαίνεται στις παρακάτω εικόνες, εππιτυγχάνεται με τη binary_inv λογική. Περνώντας σε πιο σύνθετες διαδικασίες, η trunc αντιστοιχεί στη μεγαλύτερη δυνατή ένταση τα pixels με τιμή ανώτερη του ορίου, ενώ αποδίδει στα υπόλοιπα μία νέα τιμή έντασης ανάλογα με την απόσταση της αρχικής τους τιμής από αυτή του τελευταίου. Ακόμη, δίνεται η δυνατότητα χρήσης της tozero λογικής, κατά την οποία τα pixel με ένταση μικρότερη του ορίου αντιστοιχίζονται στο χρώμα μαύρο, ενώ εκείνα με μεγαλύτερη, αποκτούν τιμή στην κλίμακα του γκρι (gray scale), η οποία αυξάνεται ανάλογα με την απόσταση από το αυτό, φτάνοντας μέχρι τη δεδομένη μέγιστη. Τέλος, αντίστροφη ακριβώς λογική από την τελευταία έχει η tozero_inv, όπως φαίνεται παρακάτω.



Εικόνα A.3.1: Εφαρμογή διαφορετικών λογικών thresholding σε εικόνα της κλίμακας του γκρι

Στις λογικές εφαρμογής της διαδικασίας thresholding που μόλις περιγράψαμε, το όριο καθορισμού της τιμής στην τελική εικόνα ήταν σταθερό για όλο το υπό επεξεργασία πλαίσιο. Ωστόσο, η OpenCV παρέχει εργαλεία με τα οποία το συγκεκριμένο όριο μπορεί να αλλάζει αυτόματα ανάλογα με την περιοχή του πλαισίου στην οποία η διαδικασία εφαρμόζεται. Πιο συγκεκριμένα, μπορούμε να λάβουμε αυτόματα μία εικόνα, όπου το όριο προσδιορίζεται τοπικά είτε από τη μέση τιμή των γειτονικών pixels (Adaptive Mean Thresholding), είτε από ένα σταθμισμένο άθροισμα των γειτονικών τιμών, με τα βάρη να προσδιορίζονται από ένα παράθυρο γκαουσιανής κατανομής με κέντρο το pixel ενδιαφέροντος (Adaptive Gaussian Thresholding). Παρακάτω, βλέπουμε παραδείγματα εφαρμογής αυτών των διαδικασιών.





Εικόνα A.3.2: Εφαρμογή διαδικασίας thresholding με τοπικά προσδιοριζόμενο όριο

Μία τελευταία από τις διαθέσιμες λογικές εφαρμογής thresholding με εργαλεία της OpenCV αποτελεί αυτή του Otsu thresholding. Η διαδικασία προσφέρεται για περιπτώσεις εικόνων με ιστόγραμμα που εμφανίζει δύο κορυφές, ενώ στις υπόλοιπες η προκύπτουσα τμηματοποίηση δεν είναι η αναμενόμενη. Για τη συγκεκριμένη λογική, το thresholding πραγματοποιείται με όριο μία τιμή έντασης μεταξύ αυτών των δύο κορυφών του ιστογράμματος, ενώ, συχνά, για να προκύψει μία τέτοια μορφή ιστογράμματος, καλούμαστε να εφαρμόσουμε ένα γκαουσιανό φίλτρο στην εικόνα για να αποκτηθεί μία με λιγότερο θόρυβο. Πιο συγκεκριμένα, σκοπός του αλγορίθμου Otsu είναι να ελαχιστοποιηθεί η εντός κλάσης σταθμισμένη διακύμανση (weighted within – class variance), που αποδίδεται από την παρακάτω σχέση:

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

όπου

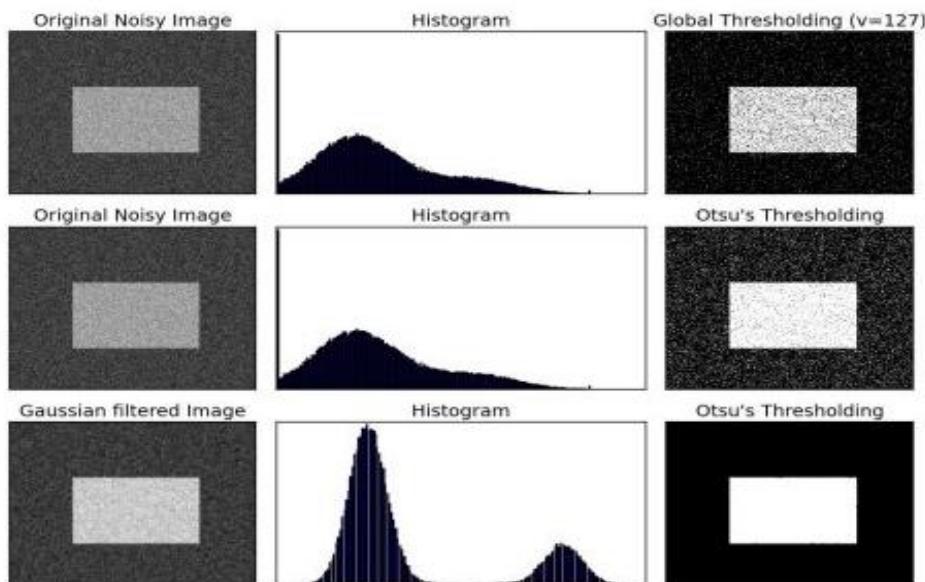
$$q_1(t) = \sum_{i=1}^t P(i) \quad \& \quad q_2(t) = \sum_{i=t+1}^I P(i)$$



$$\mu_1(t) = \sum_{i=1}^t \frac{iP(i)}{q_1(t)} \quad \& \quad \mu_2(t) = \sum_{i=t+1}^I \frac{iP(i)}{q_2(t)}$$

$$\sigma_1^2(t) = \sum_{i=1}^t [i - \mu_1(t)]^2 \frac{P(i)}{q_1(t)} \quad \& \quad \sigma_2^2(t) = \sum_{i=t+1}^I [i - \mu_1(t)]^2 \frac{P(i)}{q_2(t)}$$

Ως αποτέλεσμα, θα προκύψει μία τιμή t μεταξύ των δύο κορυφών του ιστογράμματος της εικόνας, τέτοια ώστε η διακύμανση να ελαχιστοποιείται και για τις δύο κλάσεις (1 και 2). Παρακάτω, παρατηρούμε τις προκύπτουσες εικόνες από την εφαρμογή της παραπάνω λογικής.



Εικόνα A.3.3: Εφαρμογή *thresholding* με χρήση της λογικής Otsu

A4) Βασικά Μορφολογικά Φίλτρα

Τα μορφολογικά φίλτρα αποτελούν μία οικογένεια μετασχηματισμών, συνήθως δυαδικών (binary), εικόνων που προκύπτουν ως αποτέλεσμα της δισδιάστατης συνέλιξης της εικόνας ενδιαφέροντος με ένα πυρήνα (kernel), ο οποίος καθορίζει τη φύση της διαδικασίας. Τα περισσότερα εργαλεία επεξεργασίας εικόνας, όπως και η OpenCV, διαθέτουν μία σειρά από αλγορίθμους που διεκπεραιώνουν τις συγκεκριμένες διαδικασίες. Παρακάτω, θα περιγράψουμε μερικές από τις σημαντικότερες, κάποιες από τις οποίες χρησιμοποιήθηκαν ευρέως κατά την επίλυση του προβλήματος που πραγματεύομαστε.

Αρχικά, με τον όρο erosion, αναφερόμαστε σε μία διαδικασία διάβρωσης. Στα πλαίσια της επεξεργασίας εικόνας, αυτό αποτυπώνεται με την μείωση των ορίων που καθορίζουν ένα αντικείμενο ενδιαφέροντος (προσδιοριζόμενο από άσπρα pixels), με αποτέλεσμα και τη μείωση της επιφάνειάς του. Το αποτέλεσμα αυτό προκύπτει, καθώς ο πυρήνας σαρώνει



την εικόνα και ως αποτέλεσμα ένα pixel θα διατηρήσει το άσπρο του χρώμα, μόνο εάν η καθοριζόμενη από το μέγεθος του πυρήνα γειτονιά του έχει επίσης άσπρο χρώμα. Αντίθετα, με τον όρο dilation, προσδιορίζουμε μία διαδικασία επέκτασης, όπου καθώς ο πυρήνας σαρώνει μία εικόνα, ένα pixel θα λαμβάνει το άσπρο χρώμα όταν στην γειτονιά του υπάρχει έστω και ένα άσπρο pixel. Όπως είναι κατανοητό, η επιφάνεια του αντικειμένου που καθορίζεται από τα άσπρα pixels θα αυξηθεί. Παρακάτω, παρουσιάζουμε και οπτικά το αποτέλεσμα των διαδικασιών.



Εικόνα A.4.1: Πρότυπη εικόνα και αποτελέσματα erosion και dilation (αριστερά προς δεξιά)

Τα παραπάνω φίλτρα χρησιμοποιούνται συχνά συνδυαστικά. Συγκεκριμένα, σε περιπτώσεις που θέλουμε να πετύχουμε αφαίρεση θορύβου, επιθυμούμε να εξαλείψουμε μικρές περιοχές που πιθανώς δημιουργήθηκαν λόγω του τελευταίου και στη συνέχεια να επαναφέρουμε τις βασικές επιφάνειες αντικειμένων ενδιαφέροντος στην πρότερη κατάστασή τους. Για το σκοπό αυτό, πραγματοποιούμε πρωτίστως erosion και κατόπιν dilation, μία διαδικασία που ονομάζεται μορφολογικό άνοιγμα (opening). Αντίθετα, στις περιπτώσεις που επιθυμούμε να ενώσουμε σπασμένα κομμάτια του ίδιου αντικειμένου ή να συμπληρώσουμε απούσες περιοχές από τη συνολική επιφάνεια, που εμφανίζονται με τη μορφή τρυπών, χρησιμοποιούμε τις διαδικασίες με την αντίστροφη σειρά. Η μορφολογική αυτή διεργασία καλείται κλείσιμο (closing). Τα αποτελέσματα των δύο τελευταίων φαίνονται στη συνέχεια.



Εικόνα A.4.2: Αποτέλεσμα του opening (αριστερά) και του closing (δεξιά)

Άλλος συχνά χρήσιμος μετασχηματισμός είναι το λεγόμενο gradient, το οποίο αποτελεί τη διαφορά μεταξύ dilation και erosion και πρακτικά αποδίδει μία μορφή του περιγράμματος του αντικειμένου ενδιαφέροντος. Τέλος, υπάρχουν οι μετασχηματισμοί top hat και black



hat, που αποτελούν τη διαφορά του opening και του closing αντίστοιχα με την εικόνα εισόδου. Κατόπιν, παρουσιάζουμε παραδείγματα των τελευταίων.



Εικόνα A.4.3: Αποτελέσματα για gradient, top hat και black hat (αριστερά προς δεξιά)

A5) Smoothing Εικόνας

Στους τομείς της στατιστικής και της επεξεργασίας εικόνας, με τον όρο smoothing αναφερόμαστε στη διαδικασία επεξεργασίας ενός σετ δεδομένων με στόχο την ανίχνευση και διατήρηση ενιαίων σημαντικών μοτίβων, καθώς και την αφαίρεση πιθανού θορύβου και υψησυχνών φαινομένων. Ως αποτέλεσμα, μοτίβα με μικρή πυκνότητα τοπικά τείνουν να μειωθούν ή να εξαφανιστούν, οδηγώντας σε ένα πιο ομαλό σετ. Με τη διαδικασία αυτή, πετυχαίνουμε την εξαγωγή περισσότερων συμπερασμάτων και αποκτούμε περισσότερη ευελιξία και ακρίβεια στη διαχείριση των δεδομένων. Στην περίπτωση που οι νέες τιμές μπορούν να γραφούν ως γραμμικός συνδυασμών των παλιών, τότε πρόκειται για διαδικασία γραμμικού smoothing ή τη γνωστή σε όλους συνέλιξη. Μία σειρά τέτοιων διαδικασιών, οι οποίες παρέχονται προς χρήση από την OpenCV και χρησιμοποιήσαμε στην υλοποίηση μας, περιγράφονται παρακάτω.

Σε μία εικόνα, η απλούστερη λογική smoothing είναι αυτή του κινούμενου μέσου σταθερών βαρών, όπου κάθε pixel λαμβάνει ως νέα τιμή έντασης τον μέσο όρο της γειτονιάς του, όπως αυτή ορίζεται από τον αντίστοιχο πυρήνα, με το κάθε pixel – γείτονα να συμμετέχει ισότιμα στην τελική τιμή. Στην OpenCV, η παραπάνω διαδικασία πραγματώνεται μέσω της filter2D, η οποία αποτελεί μία συνέλιξη με ένα δεδομένο πυρήνα. Ένα παράδειγμα τέτοιου φαίνεται παρακάτω.

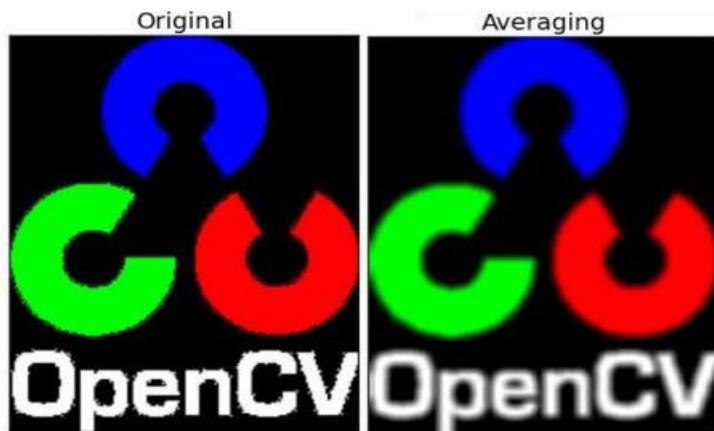
$$K = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Εικόνα A.5.1: Πυρήνας συνέλιξης διαστάσεων 5x5

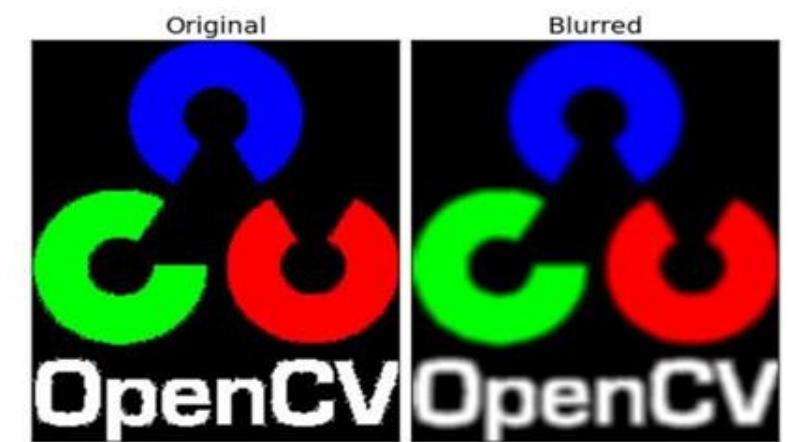
Η διαδικασία λειτουργεί αντίστοιχα με την εφαρμογή ενός χαμηλοπερατού φίλτρου και μειώνει τον πιθανό θόρυβο από την εικόνα, ενώ ταυτόχρονα θολώνει τις περιοχές όπου εμφανίζονται απότομες αλλαγές στην ένταση των pixels, αφαιρώντας, ως αποτέλεσμα, την



περιεχόμενη πληροφορία που αφορά τις ακμές της εικόνας. Παρόμοια επίδραση σαν εργαλεία της OpenCV έχουν οι διαδικασίες blur και boxFilter. Στην συνέχεια, παραθέτουμε τα αποτελέσματα της εφαρμογής των διαδικασιών που μόλις περιγράψαμε.



Εικόνα A.5.2: Εφαρμογή της *filter2D*



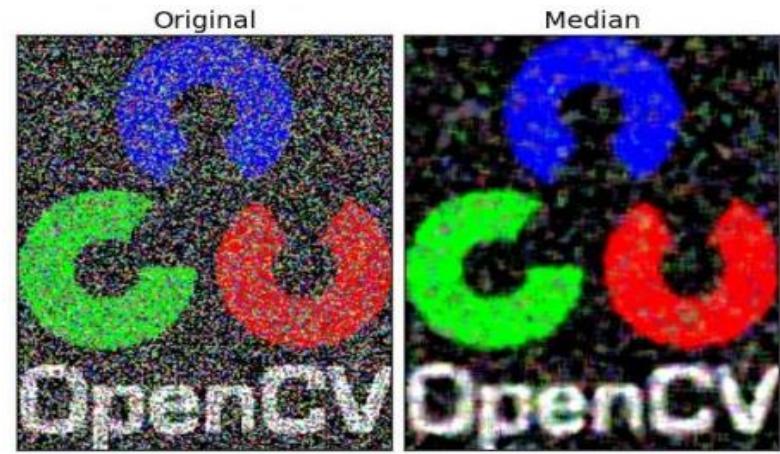
Εικόνα A.5.3: Εφαρμογή της *blur – boxFilter*

Με ανάλογο τρόπο, λειτουργεί και η GaussianBlur, με τον πυρήνα γύρω από το pixel ενδιαφέροντος να σχηματίζει γκαουσιανή κατανομή, προσδιορίζοντας αντίστοιχα τα βάρη συμμετοχής των γειτονικών pixels στον καθορισμό της τελικής τιμής. Για την ορθότερη λειτουργία της λογικής αυτής, είναι σημαντικό να οριστούν οι τυπικές αποκλίσεις στον οριζόντιο και κατακόρυφο άξονα. Επιπλέον, υπάρχει η medianBlur, όπου η νέα τιμή ενός pixel ορίζεται ως η διάμεσος των τιμών της γειτονιάς του, όπως αυτή προσδιορίζεται από τις τιμές των pixels του πυρήνα. Η medianBlur λογική είναι ιδιαίτερα αποτελεσματική στις περιπτώσεις οξύ και απότομου θορύβου (salt and pepper noise), όπως φαίνεται παρακάτω.





Εικόνα A.5.4: Εφαρμογή της GaussianBlur



Εικόνα A.5.5: Εφαρμογή της medianBlur

Τέλος, μία ιδιαίτερα σημαντική στην κατηγορία των smoothing λογικών αποτελεί αυτή του `bilateralFilter`. Η τελευταία, πέρα από την αφαίρεση του θορύβου, πετυχαίνει και τη διατήρηση των ακμών της εικόνας, απαιτώντας, ωστόσο, περισσότερη υπολογιστική ισχύ από όλες τις παραπάνω. Η εφαρμογή της περιλαμβάνει ένα γκαουσιανό φίλτρο, όπως αυτό που περιγράψαμε μόλις, που θα καθορίζει την περιοχή επιρροής των γειτονικών pixels στην τελική τιμή, καθώς και ένα που θα αποκλείει pixels με σημαντικά διαφορετική ένταση από το pixel ενδιαφέροντος, διατηρώντας, ταυτόχρονα μόνο εκείνα που η τελευταία είναι παρόμοια, για τον καθορισμό της νέας τιμής του τελευταίου. Η τελική εικόνα περιγράφεται μαθηματικά από τον παρακάτω τύπο:

$$I^{\text{filtered}}(x) = \frac{1}{W_p} \sum_{x_i \in \Omega} I(x_i) f_r(\|I(x_i) - I(x)\|) g_s(\|x_i - x\|),$$

όπου ο όρος κανονικοποίησης δίνεται ως

$$W_p = \sum_{x_i \in \Omega} f_r(\|I(x_i) - I(x)\|) g_s(\|x_i - x\|)$$

ενώ I είναι η εικόνα πριν την εφαρμογή του φίλτρου, x οι συντεταγμένες του pixel



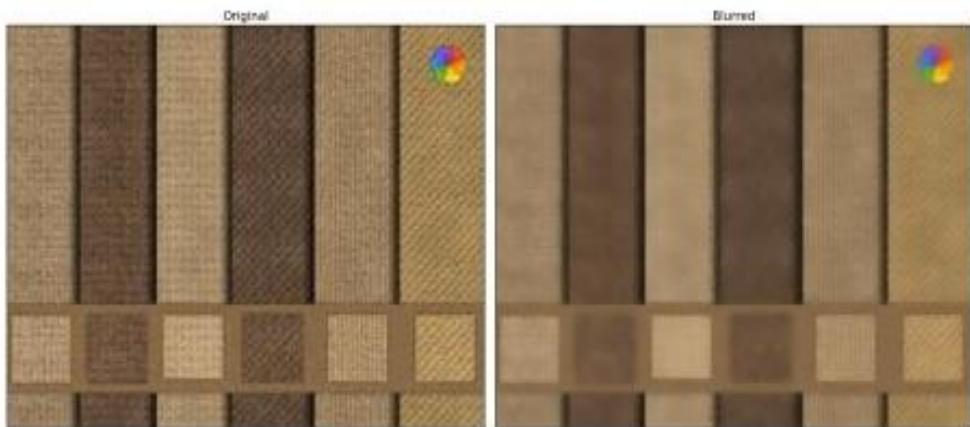
ενδιαφέροντος, Ω η περιοχή του πυρήνα, fr ο πυρήνας που αφορά τις εντάσεις και gs ο πυρήνας που αφορά τις συντεταγμένες. Ως αποτέλεσμα, ο συντελεστής συμμετοχής ενός pixel (k,l) στον καθορισμό της έντασης για ένα pixel (i,j) είναι

$$w(i, j, k, l) = \exp\left(-\frac{(i - k)^2 + (j - l)^2}{2\sigma_d^2} - \frac{\|I(i, j) - I(k, l)\|^2}{2\sigma_r^2}\right)$$

ενώ, μετά τον υπολογισμό όλων των βαρών των γειτόνων, η έντασή του προκύπτει ως

$$I_D(i, j) = \frac{\sum_{k,l} I(k, l)w(i, j, k, l)}{\sum_{k,l} w(i, j, k, l)}$$

Παρακάτω, βλέπουμε το αποτέλεσμα της εφαρμογής ενός τέτοιου φίλτρου. Είναι εμφανές ότι η υφή πάνω σε χρωματικά όμοιες επιφάνειες εξαφανίζεται, ενώ οι ακμές διατηρούνται ακέραιες.



Εικόνα A.5.6: Εφαρμογή της *bilateralFilter*

A6) Ανίχνευση Ακμών Εικόνας

Η ανίχνευση ακμών περιλαμβάνει μία σειρά από μαθηματικές μεθόδους που στοχεύουν στην εύρεση σημείων ψηφιακών εικόνων στα οποία η φωτεινότητα αλλάζει απότομα, δηλαδή παρουσιάζει ασυνέχειες. Τα σημεία στα οποία συμβαίνει αυτή η αλλαγή είναι συχνά οργανωμένα σε καμπύλες με το όνομα ακμές (edges). Ο επίσημος όρος ασυνέχεια είναι ο λόγος για τον οποίο συχνά οι μέθοδοι αυτοί αναφέρονται ως *image gradients*, που αντιστοιχεί στην ανίχνευση αλλαγών σε μία σειρά συνεχόμενων τιμών. Τα συγκεκριμένα εργαλεία αποτελούν αναπόσπαστο κομμάτι της επεξεργασίας εικόνας και βοηθούν ιδιαίτερα στο τομέα της εξαγωγής χαρακτηριστικών εικόνων (*feature detection*), όπως αντικείμενα, μέλη ενός κινούμενου ανθρώπινου σώματος και άλλα. Όπως είναι φυσικό, τέτοιες διαδικασίες έχουν ήδη κατασκευαστεί για βιβλιοθήκες, όπως η OpenCV. Στη συνέχεια θα περιγράψουμε μερικές από τις σημαντικότερες.

Το φίλτρο Sobel – Feldman εισήχθη από τους Irwin Sobel και Gary Feldman στο



εργαστήριο τεχνητής νοημοσύνης του Stanford. Βασίζεται στις διαφορές από pixel σε pixel, προσεγγίζοντας τη λογική των γνωστών μαθηματικών gradients. Το αποτέλεσμα είναι ένα διάνυσμα διαφορών των συνεχόμενων τιμών ή η νόρμα του τελευταίου και προκύπτει από τη συνέλιξη με ένα μικρό φίλτρο – πυρήνα ακεραίων, διατηρώντας τον υπολογιστικό φόρτο χαμηλό. Ωστόσο, λόγω της απλότητας και, κατ' επέκταση, ταχύτητάς του, η εκτίμηση δεν είναι ιδιαίτερα ακριβής για υψηλής συχνότητας αλλαγές. Ο πυρήνας λαμβάνει την παρακάτω μορφή για τις οριζόντιες και κάθετες μεταβολές αντίστοιχα, με την εκτίμηση τους να προσδιορίζεται ως ένα διάνυσμα.

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A} \quad \mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2} \quad \Theta = \text{atan}\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$

Εικόνα A.6.1: Πυρήνες για οριζόντια και κάθετη συνέλιξη – Υπολογισμός gradient σε φίλτρα Sobel

Βελτίωση των παραπάνω αποτελούν τα φίλτρα Scharr, που ήρθαν να λύσουν το πρόβλημα της περιστροφικής συμμετρίας που προέκυπτε από τα Sobel –Feldman. Συγκεκριμένα, οι μέθοδοι αυτοί μείωσαν το σταθμισμένο τετραγωνικό σφάλμα γωνίας μέσα από ένα μετασχηματισμό Fourier, χρησιμοποιώντας τους παρακάτω πυρήνες.

$$\begin{bmatrix} +3 & 0 & -3 \\ +10 & 0 & -10 \\ +3 & 0 & -3 \end{bmatrix} \quad \begin{bmatrix} +3 & +10 & +3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$$

Εικόνα A.6.2: Πυρήνες για οριζόντια (αριστερά) και κάθετη (δεξιά) συνέλιξη σε φίλτρα Scharr

Με την ίδια ακριβώς λογική με τα προηγούμενα, συμπεριφέρεται και η μέθοδος Laplacian of Gaussian (LoG). Βασίζεται στον υπολογισμό της λεγόμενης λαπλασιανής (laplacian), ενός ισοτροπικού δισδιάστατου μέτρου δεύτερης τάξης που δίνεται από τον τύπο:

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

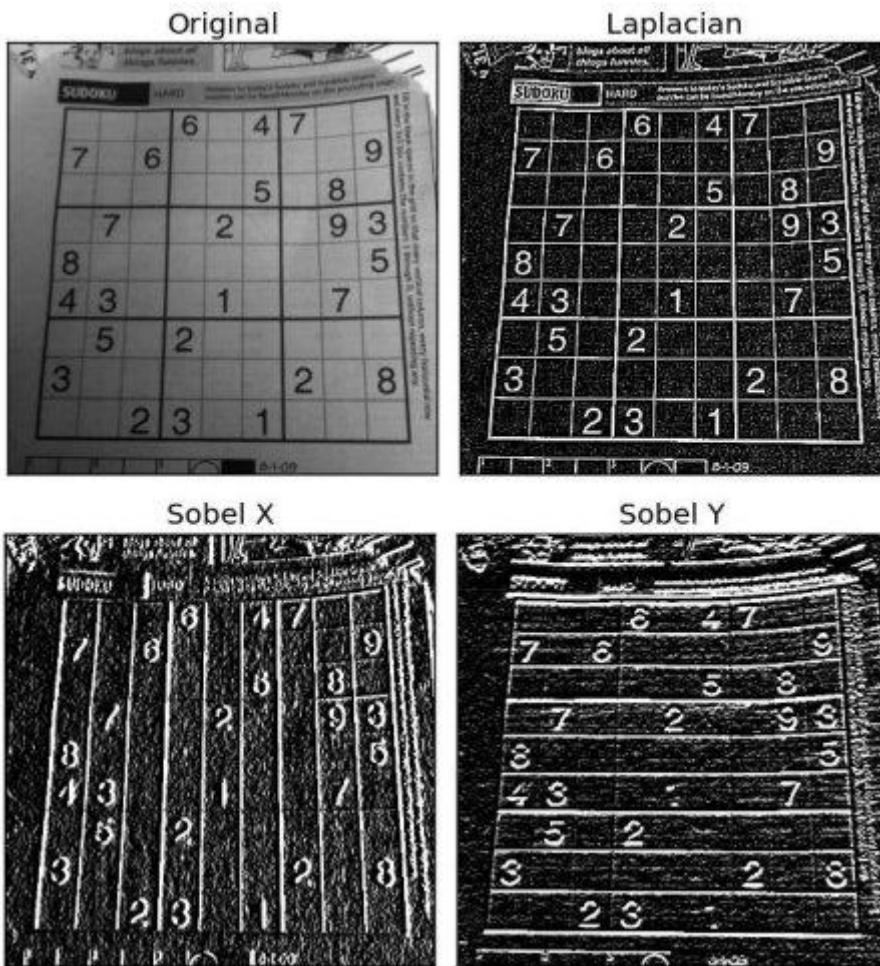
Η κάθε μία από τις παραπάνω παραγώγους δεύτερης τάξης υπολογίζεται μέσω των αντίστοιχων παραγώγων των Sobel – Feldman φίλτρων. Η συγκεκριμένη μέθοδος έχει την ιδιότητα ακριβή υπολογισμού των εναλλαγών υψηλής συχνότητας και ως αποτέλεσμα αποδίδει περισσότερη πληροφορία ως προς αυτές. Συχνά, λοιπόν, για να αποφύγουμε την ύπαρξη θορύβου και λανθασμένα ανιχνευμένων ακμών, χρησιμοποιούμε κάποιο φίλτρο αφαίρεσής του τελευταίου, όπως το γκαουσιανό. Οι πυρήνες που χρησιμοποιούνται συνήθως σε τέτοιες περιπτώσεις είναι οι παρακάτω.



$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Εικόνα A.6.2: Πυρήνες για εφαρμογή της μεθόδου Laplacian of Gaussian (LoG)



Εικόνα A.6.3: Εφαρμογή των Laplacian και Sobel (στη x και y κατεύθυνση)

Παραπάνω βλέπουμε το αποτέλεσμα του τελευταίου ανιχνευτή ακμών που θα παρουσιάσουμε, του Canny Edge Detector. Ο τελευταίος αναπτύχθηκε από τον John F. Canny το 1986 και έχει καθιερωθεί ως ένας από τους κυριότερους αλγορίθμους εύρεσης ακμών στον ευρύτερο χώρο της επεξεργασίας εικόνας. Μπορούμε να περιγράψουμε τις βασικές διαδικασίες του αλγορίθμου σε πέντε βήματα:

- A) Smoothing της εικόνας μέσω συνέλιξης με ένα γκαουσιανό φίλτρο για να αφαιρεθεί ο θόρυβος υψηλών συχνοτήτων



Β) Υπολογισμός των κλίσεων (gradients) των εντάσεων της εικόνας ως

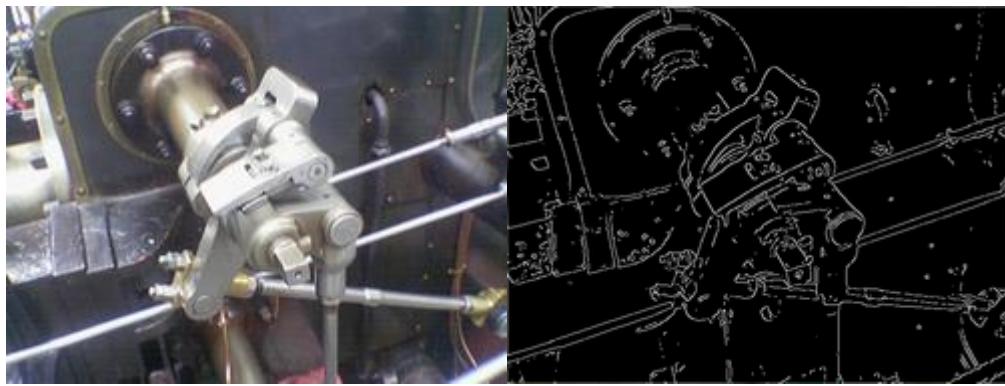
$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2}$$
$$\Theta = \text{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

με τα \mathbf{G}_x και \mathbf{G}_y να αποτελούν τις μεταβολές της έντασης στον κατακόρυφο και οριζόντιο άξονα, ενώ τα \mathbf{G} και Θ το συνολικό μέτρο και την κατεύθυνση της μεταβολής για κάθε pixel, με την τελευταία να στρογγυλοποιείται σε μία εκ των τεσσάρων βασικών κατευθύνσεων

Γ) Αφαίρεση των λανθασμένα ανιχνευμένων ακμών με non-maximum suppression, όπου κάθε pixel συγκρίνεται με εκείνα στην θετική και αρνητική κατεύθυνση που ορίζει η Θ και απαλείφεται από τη binary εικόνα αν δεν έχει τη μέγιστη μεταξύ αυτών ένταση

Δ) Εφαρμογή διπλού thresholding με ένα άνω και ένα κάτω όριο για τον καθορισμό των gradients, όπου όσα pixels έχουν ένταση μεγαλύτερη από το άνω ορίζονται ως pixels ισχυρών ακμών, όσα έχουν ανάμεσα στα δύο όρια αδύναμων και όσα έχουν μικρότερη από το κάτω απαλείφονται

Ε) Ανίχνευση των ακμών με χρήση του φαινομένου της υστέρησης (*hysteresis*), ώστε αδύναμες ακμές που δεν τείνουν να ενωθούν με άλλες ισχυρά καθορισμένες να παραλείπονται



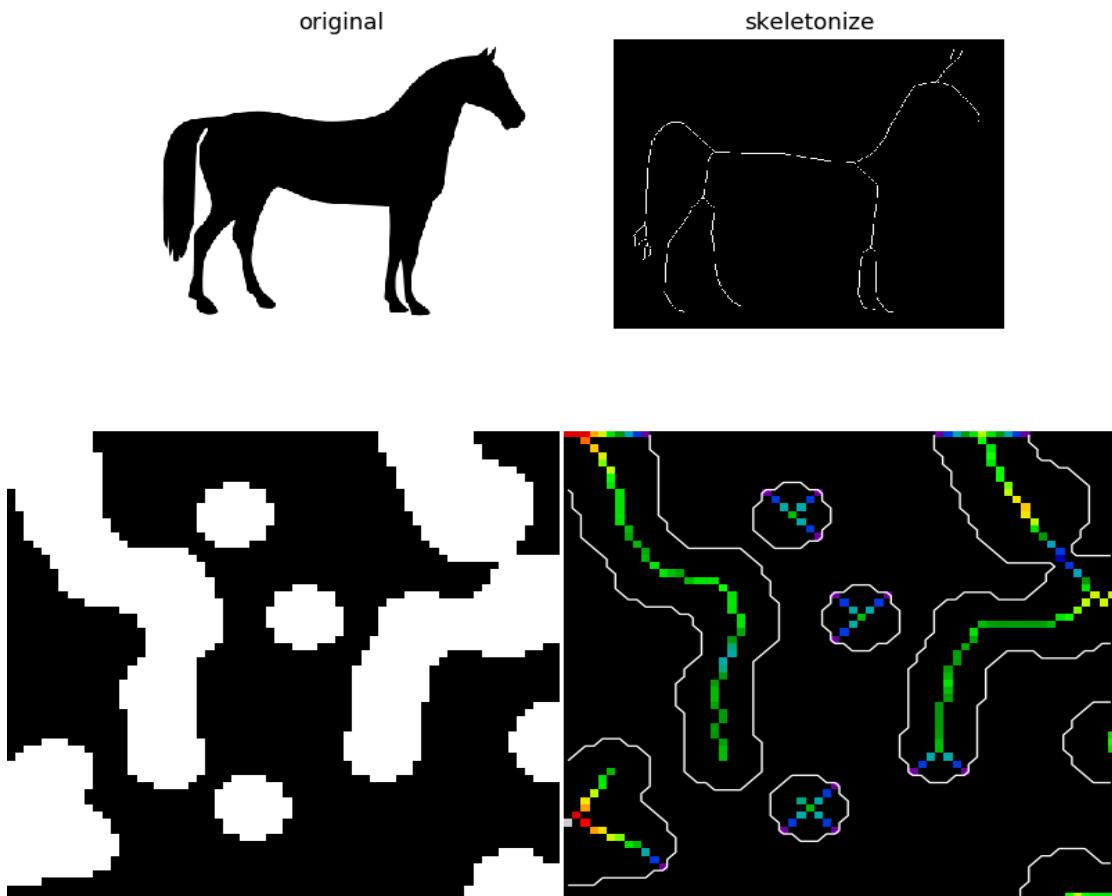
Εικόνα A.6.4: Εφαρμογή του Canny Edge Detector

A7) Αλγόριθμος Skeletonization

Συχνά, σε μία εικόνα, είναι πιθανό να μας ενδιαφέρει η εύρεση του σκελετού ενός αντικειμένου εντός ενός πλαισίου. Ακόμη, όπως γίνεται αντιληπτό, η επίδραση των παραπάνω εργαλείων σε μία binary εικόνα προκαλεί μεταβολές στην τελευταία. Πιο συγκεκριμένα, η διαδικασία του dilation σε μία εικόνα μπορεί να οδηγήσει σε ακμές με



πάχος μεγαλύτερο του επιθυμητού ενός pixel. Για να ληφθεί, λοιπόν, μία εικόνα με τέτοιες ακμές, χρησιμοποιείται η λογική της skeletonization. Η τελευταία υπάγεται στο πακέτο skimage, το οποίο λειτουργεί ως επέκταση της OpenCV για πιο ειδικές διαδικασίες στην επεξεργασία εικόνας. Η λειτουργία του βασίζεται σε μία επαναληπτική διαδικασία, όπου κάθε pixel αφαιρείται (λαμβάνει μαύρο χρώμα στην binary εικόνα) εφόσον δεν προκαλείται διακοπή της συνδεσιμότητας από την απομάκρυνσή του. Παρακάτω παρουσιάζουμε δύο παραδείγματα εφαρμογής της διαδικασίας αυτής.



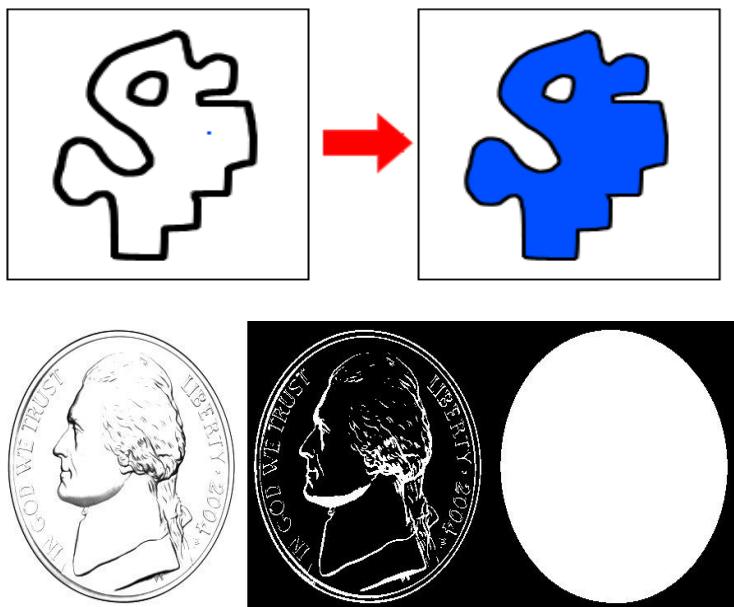
Εικόνα A.7.1: Εφαρμογή της διαδικασίας skeletonization

A8) Αλγόριθμος Floodfill

Ο αλγόριθμος floodfill χρησιμοποιείται για τον καθορισμό μίας σειράς από συνδεδεμένα στοιχεία (στην περίπτωση μας pixels), με στόχο αυτά στο τέλος να διαχωρίζονται με ένα διαφορετικό χρώμα. Με δεδομένα ένα αρχικό σημείο μίας εικόνας και ένα επιθυμητό χρώμα, ο floodfill εξετάζει τα γειτονικά προς τέσσερις κατευθύνσεις pixels, ελέγχοντας το χρώμα τους και συνεχίζει τη διαδικασία αναδρομικά έως ότου τα εναπομείναντα pixels να μην παρουσιάζουν σύνδεση με τα προηγούμενα. Η OpenCV παρέχει τη δυνατότητα εφαρμογής με την ομώνυμη ονομασία. Στην επεξεργασία εικόνας, ο αλγόριθμος αυτός



μπορεί να χρησιμοποιηθεί σε μία σειρά διαδικασιών, με στόχο το γέμισμα κλειστών περιγραμμάτων σε binary εικόνες, ώστε είτε να καθοριστούν ως επιφάνειες, είτε να απαλειφθούν τα μικρά σε έκταση μη κλειστά τμήματα στο εσωτερικό τους. Η τελευταία περίπτωση είναι ο λόγος για τον οποίο χρησιμοποιήθηκε ο floodfill στην υλοποίηση μας και παραδείγματα εφαρμογής του θα παρουσιαστούν στο επόμενο κεφάλαιο. Παρακάτω βλέπουμε μερικές εικόνες από την εφαρμογή του με στόχο το γέμισμα του εσωτερικού περιγραμμάτων με συγκεκριμένο χρώμα.

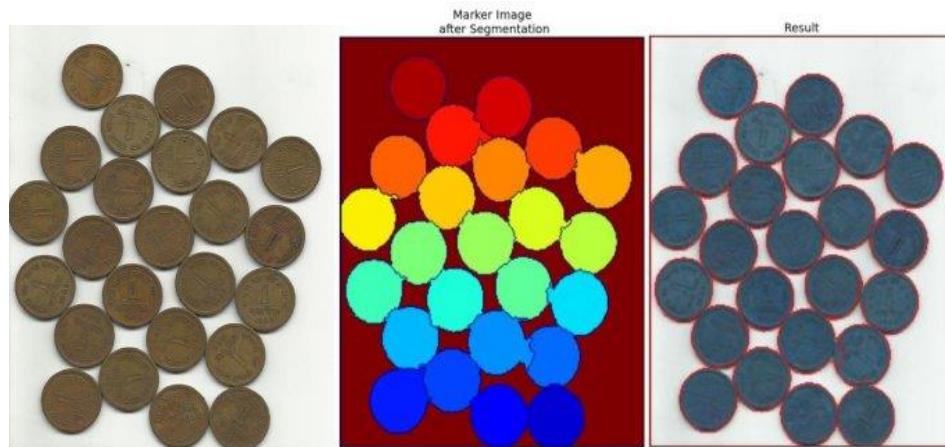
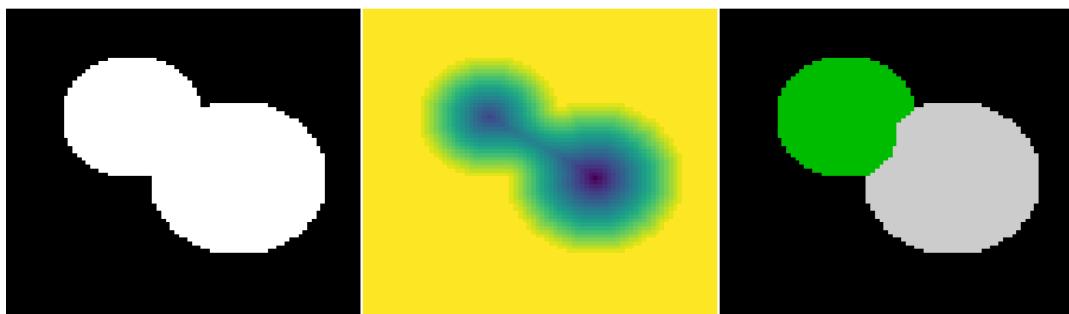


Εικόνα A.8.1: Εφαρμογή αλγορίθμου floodfill

A9) Αλγόριθμος Watershed

Στον τομέα της επεξεργασίας εικόνας, με τον όρο watershed αναφερόμαστε σε ένα μετασχηματισμό μίας gray scale εικόνας, όπου, όπως δηλώνει και η μεταφορική του ονομασία (πλημμύρα), επιχειρούμε να λάβουμε ως αποτέλεσμα μία εικόνα σε μορφή τοπογραφικού χάρτη, με την ένταση κάθε pixel να αντιπροσωπεύει το ύψος του στον τελευταίο. Με τον τρόπο αυτό, εμφανίζεται μία σειρά από κορυφές, οι οποίες αντιπροσωπεύουν τα φράγματα μετάβασης μεταξύ ενιαίων περιοχών. Η λογική αυτή αποδεικνύεται ιδιαίτερα χρήσιμη σε διαδικασίες τμηματοποίησης εικόνων, όπου μία ενιαία περιοχή συνιστά ένα χρωματικό cluster, ενώ τα pixels υψηλής έντασης συνθέτουν τις ακμές (edges) της εικόνας. Η OpenCV διαθέτει μία υλοποίηση της παραπάνω λογικής βασισμένες σε δείκτες (markers), οι οποίοι προσδιορίζονται ως μία σειρά από συνδεδεμένα pixels, έχοντας καθορίσει, πρωτίστως, τα στοιχεία της εικόνας τα οποία αποτελούν σύγουρα τμήματα του προσκηνίου (foreground) και του παρασκηνίου (background). Περισσότερες λεπτομέρειες θα δωθούν στο κεφάλαιο 4, όπου θα περιγράψουμε την χρήση του watershed στην υλοποίηση μας. Παρακάτω, παρουσιάζουμε δύο ενδεικτικά παραδείγματα εφαρμογής του.



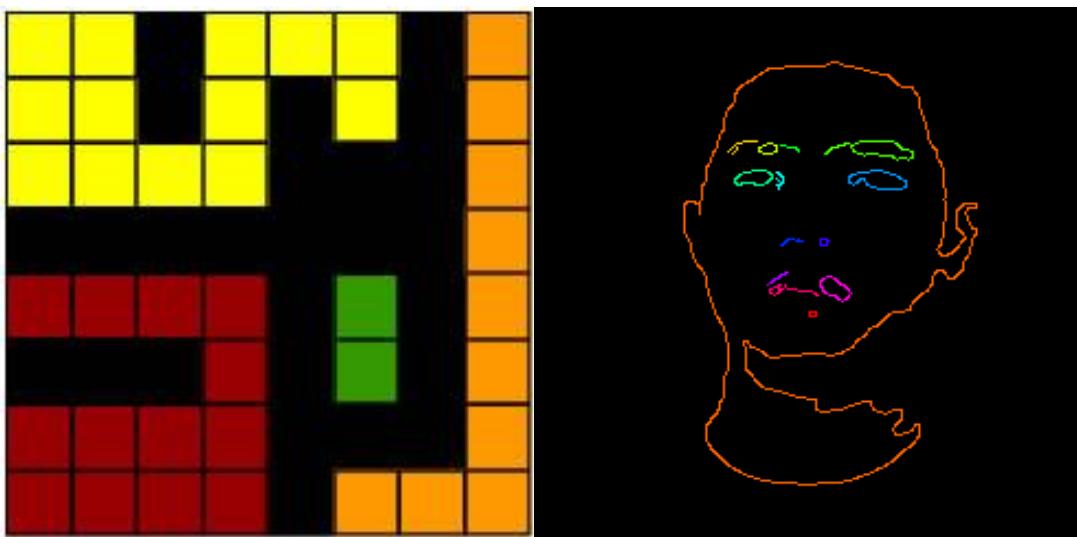


Εικόνα A.9.1: Εφαμοργή του watershed

A10) Απόδοση ετικέτας σε συνδεδεμένα στοιχεία (connected components)

Κατά το χειρισμό binary εικόνων, μας ενδιαφέρει συχνά να επεξεργαστούμε rixel προς pixel κάποια περιγράμματα. Η δυνατότητα αυτή μας δίνεται μέσω του αλγορίθμου connected components. Πιο συγκεκριμένα, ο τελευταίος εξετάζει την εικόνα, ξεκινώντας από ένα pixel και ελέγχοντας το κατά πόσο υπάρχουν γείτονες του με κοινό χρώμα (συνήθως λευκό για τις binary) προς τις τέσσερις βασικές ή οχτώ κατευθύνσεις, ώστε να αποδοθεί σε αυτά ένας κοινός αριθμός ή χρώμα που προσδιορίζει τη συνδεδεμένη ομάδα. Η διαδικασία, κατόπιν, συνεχίζεται για τους γείτονες, έως ότου να μην υπάρχει άλλο rixel μέλος της ομάδας αυτής και τερματίζει όταν κάθε rixel της εικόνας έχει χαρακτηριστεί με έναν τέτοιο αριθμό. Η OpenCV προσφέρει τη δυνατότητα εφαρμογής της παραπάνω διαδικασίας, αποτελέσματα της οποίας βλέπουμε παρακάτω.





Εικόνα A.10.1: Εφαρμογή του αλγορίθμου `connectedComponents`

A11) Λοιπές Διαδικασίες της OpenCV

Πέρα από τα παραπάνω εργαλεία, σε μεγάλο εύρος χρησιμοποιήθηκε μία σειρά άλλων διαδικασιών. Πολλές από αυτές συντελούν στην εύρεση στατιστικών στοιχείων μίας εικόνας, όπως η μέση τιμή με τη `mean`, οι θέσεις και οι τιμές του μέγιστου και του ελάχιστου με τη `minMaxLoc` και άλλα. Ακόμα, σημαντικές αποτέλεσαν εκείνες που αποδίδουν μία σειρά στοιχείων και γεωμετριών για ένα περίγραμμα. Συγκεκριμένα, χρησιμοποιήθηκαν οι εξής:

- A) `contourArea` για την εύρεση της έκτασης της καλυπτόμενης από κλειστό περίγραμμα περιοχής
- B) `minAreaRect` για το ελάχιστο σε έκταση τετράγωνο ενός περιγράμματος
- C) `boundingRect` για το ελάχιστο σε έκταση μη περιστραμμένο τετράγωνο που περιβάλλει ένα περίγραμμα
- D) `boxPoints` για την εύρεση των σημείων δεδομένου παραλληλογράμμου
- E) `arcLength` για την εύρεση του μήκους ενός περιγράμματος
- F) `approxPoly` για την εύρεση του πολυγώνου που προσεγγίζει όσο το δυνατόν καλύτερα ένα περίγραμμα

Ακόμη, ιδιαίτερα σημαντικές αποδείχτηκαν οι `pixel` to `pixel` λογικές πράξεις μεταξύ μασκών (`bitwise_and`, `bitwise_or`, `bitwise_not`) για την απομόνωση περιοχών ενδιαφέροντος. Τέλος, η δυνατότητα σχεδιασμού (`drawContours`, `rectangle`, `circle`) σε πίνακες και η οπτικοποίησή τους (`imshow`) σε εικόνες συνεισέφεραν στην επαλήθευση αποτελεσμάτων.



Βιβλιογραφία



Αφαίρεση Θορύβου Σκιάς από Αισθητήρες Depth Cameras
Μπούτης Πρόδρομος, Τζιαφάς Γεώργιος

Σελίδα 126

- [1] Miles Hansard, Seungkyu Lee, Ouk Choi, Radu Horaud, "Time of Flight Cameras: Principles, Methods, and Applications", pp.95, 2012, SpringerBriefs in Computer Science, ISBN 978-1-4471-4658-2
- [2] Tanwi Mallick; Partha Pratim Das; Arun Kumar Majumdar, "Characterizations of Noise in Kinect Depth Images: A Review", IEEE Sensors Journal, 2014
- [3] Gabriel Danciu; Simona Maria Banu; Alexandru Căliman, "Shadow removal in depth images morphology-based for Kinect cameras", 2012 16th International Conference on System Theory, Control and Computing (ICSTCC), 2012
- [4] Teng Deng; Hui Li; Jianfei Cai; Tat-Jen Cham; Henry Fuchs, "Kinect Shadow Detection and Classification", 2013 IEEE International Conference on Computer Vision Workshops, 2013
- [5] Yu Y., Song Y., Zhang Y., Wen S. (2013) A Shadow Repair Approach for Kinect Depth Maps. In: Lee K.M., Matsushita Y., Rehg J.M., Hu Z. (eds) Computer Vision – ACCV 2012. ACCV 2012. Lecture Notes in Computer Science, vol 7727. Springer, Berlin, Heidelberg
- [6] Yatong Xu; Xin Jin; Qionghai Dai, "Spatial-temporal depth de-noising for Kinect based on texture edge-assisted depth classification", 2014 19th International Conference on Digital Signal Processing, 2014
- [7] Zhaojie Ju; Yuehui Wang; Wei Zeng; Shengyong Chen; Honghai Liu, "Depth and RGB image alignment for hand gesture segmentation using Kinect", 2013 International Conference on Machine Learning and Cybernetics, 2013
- [8] Saumik Bhattacharya; Sumana Gupta; K. S. Venkatesh, "High accuracy depth filtering for Kinect using edge guided inpainting", 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014
- [9] Aarti Ghatkesar; Saumik Bhattacharya; K. S. Venkatesh, "Edge distortion removal in depth map using alpha matting ", 2015 Third International Conference on Image Information Processing (ICIIP), 2015
- [10] Nidhi Chahal; Santanu Chaudhury, "High quality depth map estimation by kinect upsampling and hole filling using RGB features and mutual information", 2013 Fourth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2013
- [11] Haiyang Du; Zhenjiang Miao, "Kinect depth maps preprocessing based on RGB-D data clustering and bilateral filtering", 2015 Chinese Automation Congress (CAC), 2015
- [12] Haikun Li; Hui Chen; Changhe Tu; Hanxiao Wang, "A Recovery Method for Kinect-Like Depth Map Based on Color Image Segmentation", 2015 14th International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015
- [13] Kang Xu, Jun Zhou, Zhen Wang, "A Method of Hole-filling for the Depth Map Generated by Kinect with Moving Objects Detection", IEEE international Symposium on Broadband Multimedia Systems and Broadcasting, 2012
- [14] Soumik Sarkar, Vivek Venugopalan, Kishore Reddy, Michael Giering, Julian Ryde, Navdeep Jaitly, "Using Deep Convolutional Networks for Occlusion Edge Detection in RGB-D



Frames”, pp., 2014

- [15] Litong Feng, Lai-Man Po, Xuyuan Xu, Ka-Ho Ng, Chun-Ho Cheung *, Kwok-Wai Cheung. “AN ADAPTIVE BACKGROUND BIASED DEPTH MAP HOLE-FILLING METHOD FOR KINECT”, IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society, 2013
- [16] A. Criminisi, P. Perez, K. Toyama, “Region filling and object removal by exemplar-based image inpainting,” IEEE Transactions on Image Processing, vol. 13(9), pp.1200-1212, 2004
- [17] Karan, Branko, Calibration of Kinect-type RGB-D sensors for robotic applications, FME Transactions, 43. 47-54, 10.5937/fmet1501047K, 2015
- [18] Davudinasab Ahmad, Kinect Sensor, 10.13140/2.1.1068.5124, 2014

