Υπολογιστικές διαδικασίες εύρεσης βιομηχανικών προϊόντων σε αποθήκες

Τσακιρίδης Σωτήριος,
Πληροφορικός - Γεωπόνος, sotiris@serres.gr

Τσιμπίρης Αλκιβιάδης,
Αναπληρωτής Καθηγητής, atsimpiris@ihu.gr

Βαρσάμης Δημήτριος,
Καθηγητής, dvarsam@ihu.gr
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών, ΔΙΠΑΕ

Περίληψη

Η καταμέτρηση προϊόντων σε έναν υπαίθριο ή στεγασμένο αποθηκευτικό χώρο βασίζεται στην εκ των προτέρων γνώση των διαστάσεων της παλέτας. Τα δεδομένα εισόδου είναι ένα νέφος τρισδιάστατων σημείων τα οποία δημιουργούνται από τρισδιάστατους σαρωτές (LIDAR) προσαρμοσμένους σε εναέρια μέσα (drones). Για τους σκοπούς της έρευνας έχει υλοποιηθεί ένας προσομοιωτής αποθήκης στην γλώσσα python (έκδοση 3.9). Στην πρώτη φάση της έρευνας εξετάζεται η ιδανική περίπτωση διασποράς των σημείων στο χώρο με ακέραιες τιμές συντεταγμένων και με μία μονάδα μήκους να αντιστοιχεί στην απόσταση μεταξύ δύο γειτονικών pixels κατά την οριζόντια ή κατακόρυφη διεύθυνση. Σε επόμενο στάδιο της έρευνας θα τροποποιηθεί ο γεννήτορας τρισδιάστατων σημείων ώστε να παράγονται πιο ρεαλιστικά μοντέλα αποθήκης και θα προταθούν βελτιωμένες εκδόσεις των υφιστάμενων αλγορίθμων που να λαμβάνουν υπόψη τις διακυμάνσεις του ύψους.

Λέξεις κλειδιά: Νέφος σημείων, Τρισδιάστατοι σαρωτές, Μοντελοποίηση αποθήκης

Abstract

Counting products in an outdoor or indoor warehouse is based on prior knowledge of pallet dimensions. The input data is a cloud of 3D points generated by 3D scanners (LIDAR) attached to aerial vehicles (drones). For the purposes of this research, a warehouse simulator has been implemented in the python language. In the first phase of the research we have examined the ideal case of dispersion of the points in space having integer coordinate values and with one length unit corresponding to the distance between two adjacent pixels in the horizontal or vertical direction. In a later stage of the research, the 3D point generator will be modified to produce more realistic warehouse models and improved versions of the existing algorithms will be proposed that take into account the height variations.

Keywords: Point cloud, 3D scanners, Warehouse modeling

1. Εισαγωγή

Η εισαγωγή των τεχνολογιών αυτοματισμού στους χώρους αποθήκευσης διαφόρων προϊόντων κρίνεται σήμερα απαραίτητη, προκειμένου η λειτουργία και οργάνωση των χώρων αποθήκευσης να είναι εύρυθμη και αποτελεσματική. Η σημασία της ενσωμάτωσης των τεχνολογιών αυτοματισμού είναι ιδιαίτερα μεγάλη, ειδικά στη σύγχρονη εποχή της παγκοσμιοποίησης, καθώς υπάρχουν πολυάριθμες πολυεθνικές εταιρίες με ευρεία προϊόντων κάθε μορφής, οι οποίες απαιτείται να ελέγχουν και να οργανώνουν τις αποθήκες διανομής τους με τον πλέον σύγχρονο και επωφελή για αυτές τρόπο, ο οποίος επιτυγχάνεται με την αξιοποίηση της τεχνολογίας στο έπακρο.

Η αυτοματοποιημένη δημιουργία δεδομένων και εξαγωγή γνώσης από χώρους αποθήκευσης απασχολεί εκτενώς την επιστημονική κοινότητα τα τελευταία χρόνια. Η ραγδαία ανάπτυξη των εναέριων μέσων (drones) σε συνδυασμό με το σχετικά χαμηλό κόστος αγοράς και λειτουργίας τους συμβάλει σημαντικά προς αυτή την κατεύθυνση. Στην εργασία των Vergouw B. et. al. [1] περιγράφονται οι διαφορετικοί τύποι των εναέριων μέσων και γίνεται μία κατηγοριοποίησή τους με βάση το μέγεθός τους, το βάρος τους και την κατανάλωση ενέργειας ώστε να γίνει η σωστή επιλογή σε κάθε περίπτωση χρήσης.

Οι H.R. Everett et. al. [2] περιγράφουν με επιτυχία τον ταυτόχρονο έλεγχο πολλών ρομπότ αυτόνομης πλοήγησης σε περιβάλλον βιομηχανικής αποθήκης.

Οι Arango C, Morales CA. [3] προτείνουν μία μέθοδο υπολογισμού του όγκου ενός αποθηκευμένου υλικού το οποίο είναι ατάκτως διασκορπισμένο στον χώρο της αποθήκης.

Οι Mokroš M. et. al. [4] χρησιμοποιούν μία συσκευή GNSS προσαρμοσμένη σε μη επανδρωμένο εναέριο όχημα για την εκτίμηση του όγκου τεσσάρων πασσάλων από ροκανίδια με την χρήση αλγορίθμων επεξεργασίας εικόνας.

Οι συγκεκριμένες προσεγγίσεις απαιτούν αρχικά την λήψη των δεδομένων και σε δεύτερο χρόνο την επεξεργασία τους για την εξαγωγή των αποτελεσμάτων. Στην παρούσα εργασία επικεντρωνόμαστε στην απλοποίηση των αλγοριθμικών μεθόδων, όχι όμως σε βάρος της ακρίβειας των αποτελεσμάτων, ώστε να μειωθεί η υπολογιστική τους πολυπλοκότητα και να είναι κατάλληλοι για εφαρμογή σε πραγματικό χρόνο. Η συλλογή των δεδομένων γίνεται με αισθητές LIDAR (**LI**ght **D**etection **A**nd **R**anging) προσαρμοσμένους σε εναέρια μέσα (UAV - **U**nmanned **A**erial **V**ehicle).
Στην συγκεκριμένη εργασία παρουσιάζεται σε θεωρητικό επίπεδο η μεθοδολογία επεξεργασίας των δεδομένων και όχι η μεθοδολογία συλλογής και μοντελοποίησης αυτών. Θέματα σχετικά με τις τεχνικές προδιαγραφές των αισθητήρων και των εναέριων μέσων και της εύρεσης των βέλτιστων συχνοτήτων δειγματοληψίας και ταχύτητας πτήσης θα μας απασχολήσουν σε μελλοντική έρευνα.

2. Μέθοδος ωμής βίας (brute force)

Η μέθοδος της ωμής βίας [5] εξαντλητικά όλα τα σημεία του χώρου. Απαραίτητη προϋπόθεση είναι η διάταξη αυτών κατά γεωγραφικό μήκος και πλάτος. Έτσι γίνεται αναγωγή των σημείων του χώρου σε ένα είδος δισδιάστατου πλέγματος στο οποίο ο αριθμός γραμμής ενός κελιού αντιστοιχεί στην x συνιστώσα του σημείου, ο αριθμός στήλης στην y συνιστώσα και η τιμή του κελιού στην z συνιστώσα. Σε συνθήκες προσομοίωσης οι τιμές των συνιστωσών είναι ακέραιοι αριθμοί.

Έστω $P = \{\hat{p}_1, \hat{p}_2 \dots \hat{p}_n\}$ ένα πεπερασμένο σύνολο τρισδιάστατων διανυσμάτων, όπου κάθε διάνυσμα αντιστοιχεί σε ένα σημείο του νέφους με $\hat{p}_i = (x_i, y_i, z_i)$. Αρχικά δημιουργούνται δύο διατεταγμένα σύνολα X, Y των διακριτών τιμών των συνιστωσών x και y αντίστοιχα. Η δημιουργία τους γίνεται σε δύο βήματα. Στο πρώτο βήμα καταγράφονται σε δύο σύνολα οι μοναδικές τιμές των x και y για όλα τα σημεία του νέφους και στο επόμενο βήμα ταξινομούνται τα σύνολα αυτά κατά αύξουσα σειρά. Η κατάταξη των

συνιστωσών στα δύο σύνολα γίνεται με ένα πέρασμα των σημείων του νέφους, δηλαδή έχει πολυπλοκότητα $O(n)$. Για την ταξινόμηση των συνόλων έχει επιλεγεί ο αλγόριθμος merge sort με χρονική πολυπλοκότητα $O(nlogn)$. Έτσι, η συνολική πολυπλοκότητα της φάσης δημιουργίας των διατεταγμένων συνόλων για τις συνιστώσες x, y είναι $O(nlogn)$.

Εν συνεχεία δημιουργείται ένας πίνακας κατακερματισμού (hash table) για την κατάταξη της τιμής της z συνιστώσας των σημείων του νέφους. Ως κλειδί χρησιμοποιείται μία κωδικοποίηση των συνιστωσών x,y του σημείου σε ένα αλφαριθμητικό της μορφής «(x,y)». Δηλαδή, η συνάρτηση κατακερματισμού ορίζεται ως εξής:

$$f: "(x_i,y_i)" \rightarrow z_i, \forall \hat{p}_i = (x_i, y_i, z_i) \in P$$

Ο χρονική πολυπλοκότητα δημιουργίας του πίνακα κατακερματισμού είναι O(n) ενώ η πολυπλοκότητα ανάκτηση στην γενική περίπτωση είναι O(1).

Τα διατεταγμένα σύνολα X, Y μαζί με τον πίνακα κατακερματισμού της z συνιστώσας δημιουργούν την επιθυμητή διάταξη των σημείων του νέφους σε ένα πλέγμα. Δημιουργείται δηλαδή ένα ορθογώνιο σύστημα καρτεσιανών συντεταγμένων με άξονες τις τιμές των συνόλων X και Y όπου στη θέση (x, y) υπάρχει η τιμή ύψους z.

Βασική παραδοχή του συγκεκριμένου αλγορίθμου που αποτυπώνεται τυπικά στο Σχήμα 1 είναι ότι όλες οι στοίβες είναι στοιχισμένες στο χώρο και η σάρωση γίνεται κατά μήκος της μίας πλευράς τους. Αν το σημείο $P_i = (x_i, y_i, z_i)$ ανήκει στην κάτω αριστερή γωνία της στοίβας και θεωρήσουμε ότι το πλάτος και το μήκος αυτής είναι w και l μονάδες μήκους αντίστοιχα, τότε όλα τα σημεία που βρίσκονται στην ορθογώνια περιοχή που οριοθετείται από τις συντεταγμένες των σημείων (x_i, y_i) και $(x_i + w, y_i + l)$ θα ανήκουν επίσης στην στοίβα. Θα έχουν δηλαδή την ίδια τιμή z.

Αλγόριθμος καταμέτρησης πλήθους παλετών σε νέφος σημείων

```
1.  procedure COUNT_PALLETS(P, w, l, h)
2.     X ← []
3.     Y ← []
4.     HASH ← {}
```

```
 5.    PALLETS ← 0
 6.    for p in P do
 7.      if p.x not in X then
 8.        X.append(p.x)
 9.      end if
10.      if p.y not in Y then
11.        Y.append(p.y)
12.      end if
13.      key ← "(p.x, p.y)"
14.      HASH[key] ← p.z
15.    end for
16.    merge_sort(X)
17.    merge_sort(Y)
18.    for i←1 to LENGTH(X) do
19.      for j←1 to LENGTH(Y) do
20.        EXISTS, HEIGHT ← CHECK_STACK(i, j, X, Y, HASH,
     w, l)
21.        if EXISTS then
22.          PALLETS ← PALLETS + HEIGHT/h
23.          REMOVE_STACK(i, j, X, Y, HASH, w, l)
24.        end if
25.      end for
26.    end for
27.    return PALLETS
28.  end procedure
```

ΣΧΗΜΑ 1: Αλγόριθμος καταμέτρησης πλήθους παλετών

Ο αλγόριθμος ελέγχου ύπαρξης στοίβας στην θέση (i, j) (CHECK_STACK) παρουσιάζεται στο Σχήμα 2.

```
Αλγόριθμος ελέγχου ύπαρξης στοίβας στη θέση i,j
 1.  procedure CHECK_STACK(i, j, X, Y, HASH, w, l, dx, dy)
 2.    KEY ← "(X[i],Y[j])"
 3.    z ← HASH[KEY]
 4.    if z = NULL or z = 0 then
 5.      return FALSE, 0
 6.    end if
 7.
 8.    k ← i + 1
 9.    width ← 0
10.    while X[k] - X[i] <= w do
11.      KEY ← "(X[k],Y[j])"
12.      if HASH[KEY] = z then
13.        width ← X[k] - X[i]
14.        k ← k + 1
```

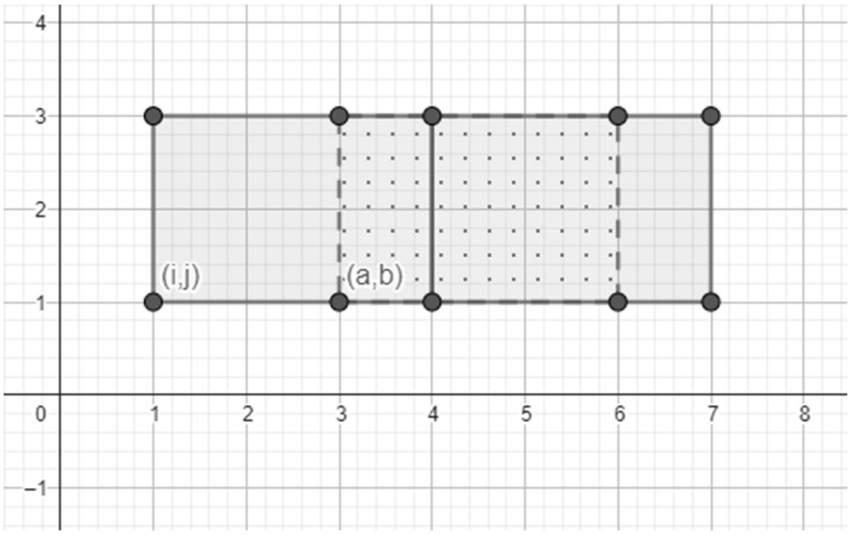
```
15.      else
16.        break
17.      end if
18.    end while
19.
20.    k ← j + 1
21.    length ← 0
22.    while Y[k] - Y[j] <= l do
23.      KEY ← "(X[i],Y[k])"
24.      if HASH[KEY] = z then
25.        length ← Y[k] - Y[j]
26.        k ← k + 1
27.      else
28.        break
29.      end if
30.    end while
31.
32.    if width >= w – 2*dx and length >= l – 2dy then
33.      return TRUE, z
34.    else
35.      return FALSE, 0
36.    end if
37.  end procedure
```

ΣΧΗΜΑ 2: Αλγόριθμος καταμέτρησης πλήθους παλετών

Επειδή η σάρωση του χώρου από τους αισθητήρες LIDAR γίνεται βηματικά με μία συγκεκριμένη και προκαθορισμένη συχνότητα είναι απίθανο να ανιχνευθούν σημεία ακριβώς στα όρια της στοίβας.

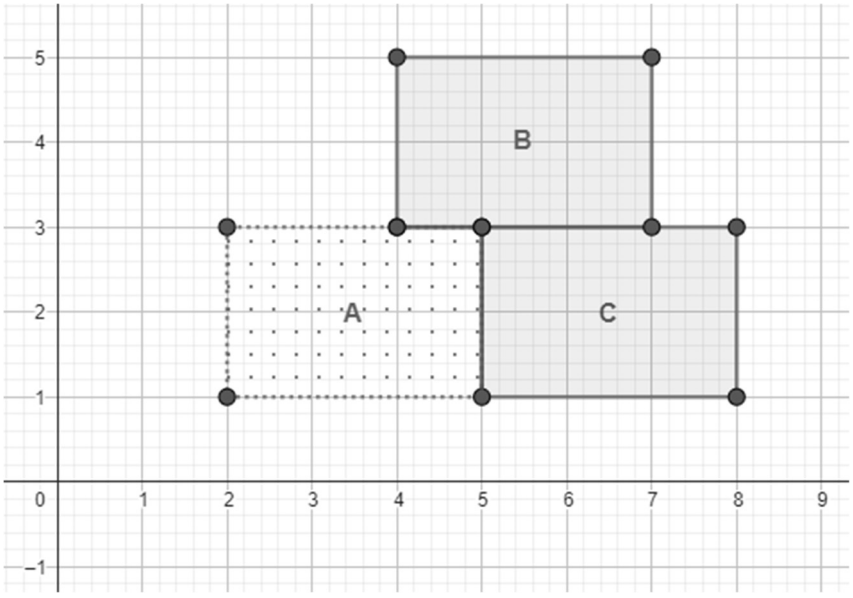
Έστω dx η ελάχιστη απόσταση μεταξύ δύο διαδοχικών σαρώσεων κατά πλάτος η οποία εξαρτάται από την μέγιστη συχνότητα δειγματοληψίας του LIDAR και την ταχύτητα πτήσης του UAR. Αν το αναμενόμενο πλάτος της στοίβας είναι W τότε το χειρότερο σενάριο που οδηγεί στην μέγιστη απόκλιση είναι να ανιχνευθεί ένα σημείο λίγο πριν ή αμέσως μετά την αρχή και το τέλος της στοίβας αντίστοιχα. Έτσι, η μέγιστη απόκλιση που μπορεί να παρατηρηθεί είναι $2dx$. Τα ίδια ακριβώς ισχύουν και για την σάρωση κατά μήκος. Αυτές οι παράμετροι λαμβάνονται υπόψη στον αλγόριθμο CHECK_STACK ώστε να ανιχνεύονται στοίβες με πλάτος και μήκος ελαφρώς μικρότερα του αναμενόμενου με βάση την συχνότητα σάρωσης.

Ιδιαίτερη περίπτωση αποτελεί η ύπαρξης γειτονικών στοιβών, όπως φαίνεται ενδεικτικά στο Σχήμα 3.



ΣΧΗΜΑ 3: Ανίχνευση δύο γειτονικών στοιβών

Αν το τρέχον σημείο ελέγχου είναι το (a, b) είναι προφανές ότι θα ανιχνευθεί μία μη υπάρχουσα στοίβα από την επικάλυψη των σημείων των δύο γειτονικών στοιβών. Για να αντιμετωπιστεί αυτή η περίπτωση και με δεδομένο ότι ο αλγόριθμος θα εξετάσει το σημείο (i, j) πριν από το (a, b) λόγω της διάταξης των συνόλων X και Y, εφόσον ανιχνευθεί στοίβα στη θέση (i, j) αυτή θα αφαιρείται από τον πίνακα κατακερματισμού αφαιρώντας όλα τα κλειδιά των σημείων που ανήκουν σ' αυτήν εκτός από τα σημεία της πάνω και δεξιάς πλευράς που ενδέχεται να ανήκουν στην αρχή μιας γειτονικής στοίβας. Έτσι, το σημείο (a, b) δεν θα ελεγχθεί ξανά. Στο Σχήμα 4 παρουσιάζονται τρεις γειτνιάζουσες στοίβες (A, B, και C) και αποτυπώνονται τα σημεία που έχουν αφαιρεθεί από τον πίνακα κατακερματισμού μετά την ανίχνευσης της στοίβας A.



ΣΧΗΜΑ 4: Ανίχνευση τριών γειτονικών στοιβών

Ο αλγόριθμος αφαίρεσης στοίβας (REMOVE_STACK) παρουσιάζεται στο Σχήμα 5.

Αλγόριθμος αφαίρεσης στοίβας από τη θέση i,j (κάτω δεξιά)

```
 1.  procedure REMOVE_STACK(i, j, X, Y, HASH, w, l)
 2.    m ← i
 3.    n ← j
 4.    while X[m] - X[i] < w do
 5.      while Y[n] + Y[j] < l do
 6.        KEY ← "(X[m],Y[n])"
 7.        HASH.remove(KEY)
 8.        n ← n + 1
 9.      end while
10.      m ← m + 1
11.      n ← j
12.    end while
13.  end procedure
```

ΣΧΗΜΑ 5: Αλγόριθμος αφαίρεσης στοίβας

3. Μελέτη χρονικής πολυπλοκότητας

Η ελάχιστη χρονική πολυπλοκότητα του αλγορίθμου καταμέτρησης είναι προφανώς $O(n)$ αφού καθένα από τα n σημεία του νέφους θα πρέπει να εξεταστεί τουλάχιστον μία φορά. Οι δύο βρόγχοι while στον αλγόριθμο CHECK_STACK ανεβάζουν την χρονική πολυπλοκότητα θεωρητικά σε $O(n^2)$. Αυτό όμως πρακτικά δεν υφίσταται αφού αφενός οι δύο βρόγχοι ελέγχουν μόνο τις δύο πλευρές της στοίβας και όχι τα ενδιάμεσα σημεία και αφετέρου οι διαστάσεις της στοίβας είναι κατά πολύ μικρότερες αυτών της αποθήκης. Έτσι, με προκαθορισμένο και σταθερό μέγεθος στοιβών ο αλγόριθμος CHECK_STACK έχει πρακτικά σταθερή πολυπλοκότητα $O(1)$!

Κάτι ανάλογο ισχύει και για τον αλγόριθμο REMOVE_STACK. Οι δύο εμφωλευμένοι βρόγχοι while στις γραμμές 4 και 5 μπορεί θεωρητικά να εκτελεστούν n φορές καθιστώντας έτσι την συνολική πολυπλοκότητα του αλγορίθμου σε $O(n^2)$. Το ακραίο θεωρητικό σενάριο εμφανίζεται στην περίπτωση ύπαρξης μίας στοίβας στο μέγεθος της αποθήκης, το οποίο προφανώς δεν ισχύει στην πραγματικότητα. Έτσι, οι πολύ μικρότερες διαστάσεις της στοίβας σε σχέση με την αποθήκη και το γεγονός ότι ο REMOVE_STACK εκτελείται μόνο στην περίπτωση ανίχνευσης στοίβας, καθιστούν την χρονική του πολυπλοκότητα πρακτικά σταθερή, δηλαδή O(1)!. Δηλαδή, μπορούμε να θεωρήσουμε ότι πρακτικά η χρονική πολυπλοκότητα του αλγορίθμου καταμέτρησης είναι $O(n)$ η οποία συνδυαστικά με την πολυπλοκότητα ταξινόμησης των σημείων του νέφους καθιστούν την συνολική χρονική πολυπλοκότητα της προτεινόμενης μεθοδολογίας σε $O(nlogn)$!

4. Συμπεράσματα - Μελλοντική έρευνα

Η απαίτηση για την παράλληλη στοίχιση των στοιβών με την διεύθυνση σάρωσης είναι αρκετά δεσμευτική. Στην περίπτωση άτακτης τοποθέτησης αυτών στον χώρο της αποθήκης η προτεινόμενη μεθοδολογία αποτυγχάνει. Αν το τρέχον εξεταζόμενο σημείο έχει ύψος μεγαλύτερο του σημείου αναφοράς (δεν ανήκει δηλαδή στο δάπεδο της αποθήκης) δεν μπορούμε να το θεωρήσουμε ως το κάτω δεξιό άκρο μιας ενδεχόμενης στοίβας. Οι βρόγχοι ελέγχου while του CHECK_STACK θα πρέπει να τροποποιηθούν ώστε να ανιχνεύουν την διεύθυνση των πλευρών. Μία προφανής μέθοδος αντιμετώπισης θα μπορούσε να είναι η αναδρομική εξέταση των γειτόνων

του υπό εξέταση σημείου προκειμένου να βρεθεί ο προσανατολισμός της στοίβας. Αυτό όμως αυξάνει δραματικά την χρονική πολυπλοκότητα και καθιστά τον αλγόριθμο πρακτικά μη εφαρμόσιμο οπότε θα πρέπει να ακολουθηθεί μία διαφορετική προσέγγιση.

Στην συνέχεια της έρευνας θα μελετηθούν και θα αξιολογηθούν αλγόριθμοι αναγωγής του νέφους σημείων σε εικόνες με τιμές φωτεινότητας την τιμή της z-συνιστώσας ώστε να εφαρμοστούν τεχνικές τμηματοποίησης από το επιστημονικό πεδίο της ψηφιακής επεξεργασίας εικόνων για τον εντοπισμό συνεκτικών περιοχών η οποίας πιθανόν να ανήκουν σε θέσεις στοίβας.

Για την εφαρμογή του αλγορίθμου σε πραγματικές συνθήκες θα πρέπει να ληφθούν υπόψη οι μικρές διακυμάνσεις στις τιμές των συνιστωσών των σημείων. Στην συγκεκριμένη ιδανική περίπτωση που εξετάστηκε, οι τιμές των συνιστωσών ήταν ακέραιοι αριθμοί. Στην πράξη αναμένονται μικρές αποκλίσεις στις τιμές αυτών για σημεία που θα πρέπει να κατηγοριοποιηθούν στο ίδιο ύψος, πλάτος ή μήκος. Ο αλγόριθμος προσομοίωσης της αποθήκης θα τροποποιηθεί κατάλληλα με την εισαγωγή παραμέτρων σφάλματος για τις τρεις συνιστώσες. Συνιστώσες σημείων με τιμές εντός των ορίων του σφάλματος θα θεωρούνται ίσες.

Τέλος, θα δοκιμαστούν τεχνικές παραλληλοποίησης των αλγορίθμων ώστε να μειωθούν οι χρόνοι εκτέλεσης και να επιτραπεί η εφαρμογή τους σε πραγματικό χρόνο. Οι συγκεκριμένες υλοποιήσεις των αλγορίθμων CHECK_STACK και REMOVE_STACK προσφέρονται για παράλληλη εκτέλεση αφού δεν υπάρχουν γραμμικές εξαρτήσεις. Η παραλληλοποίηση θα μπορούσε να γίνει, ιδανικά, στην κάρτα γραφικών (GPU). Οι σύγχρονες GPU διαθέτουν εκατοντάδες πυρήνες που μπορούν να εκτελέσουν παράλληλα στοιχειώδης υπολογισμούς. Ωστόσο, ο αντικειμενικός στόχος της εργασίας είναι η εφαρμογή των αλγορίθμων «on-the-fly». Η συλλογή και επεξεργασία των δεδομένων θα γίνεται από μικροελεγκτές ή μικροϋπολογιστές (π.χ. Raspberry Pi) προσαρμοσμένους στα εναέρια μέσα. Αυτοί οι μικρουπολογιστές έχουν εγγενώς περιορισμένη υπολογιστική ισχύ και κάρτες γραφικών με λίγους πυρήνες. Έτσι, πρωταρχικά ο παραλληλισμός θα γίνει σε επίπεδο CPU. Προφανώς σε δεύτερο στάδιο μπορεί να γίνει επεξεργασία των συνολικών δεδομένων «off-the-fly» σε

υπολογιστικά συστήματα που προσφέρουν δυνατότητες παραλληλοποίησης στην GPU.

Η παρούσα εργασία υλοποιήθηκε στο πλαίσιο του έργου με τίτλο «Βελτιστοποίηση τοποθέτησης και καταμέτρησης εμπορευμάτων σε μεγάλους βιομηχανικούς χώρους με χρήση μη επανδρωμένων αεροσκαφών» (Κωδικός έργου: ΚΜΡ6-0083129) στη Δράση «Επενδυτικά Σχέδια Καινοτομίας», του Επιχειρησιακού Προγράμματος «Κεντρική Μακεδονία 2014-2020» της Περιφέρειας Κεντρικής Μακεδονίας και συγχρηματοδοτήθηκε από το Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης (ΕΤΠΑ) της Ευρωπαϊκής Ένωσης (ΕΕ) και Εθνικούς πόρους στο πλαίσιο του Επιχειρησιακού Προγράμματος «Κεντρική Μακεδονία» 2014-2020»’’

Αναφορές

1. Vergouw B, Nagel H, Bondt G, Custers B. Drone technology: types, payloads, applications, frequency spectrum issues and future developments. The Future of Drone Use. New York, NY: Springer; 2016:21-45.

2. Everett HR, Gage DW, Gilbreath GA, Laird RT, Smurlo RP. Real-world issues in warehouse navigation. In: Wolfe WJ, Chun WH, eds. Mobile Robots IX. Vol 2352. Bellingham,Washington: International Society for Optics and Photonics, SPIE; 1995:249-259.

3. Arango C, Morales CA. Comparison between multicopter uav and total station for estimating stockpile volumes. ISPRS Int Arch Photogramm Remote Sens Spatial Inf Sci. 2015;XL-1(W4):131-135.

4. Mokroš M, Tabacˇák M, Lieskovsky` M, Fabrika M. Unmanned aerial vehicle use for wood chips pile volume estimation. Int Arch Photogramm Remote Sens Spat Inf Sci. 2016;41:953.

5. HEULE, Marijn JH; KULLMANN, Oliver. The science of brute force. *Communications of the ACM*, 2017, 60.8: 70-79.