Ένας απλός και γρήγορος αλγόριθμος για την αποκοπή γραμμών στο Scratch

Δημήτριος Ματθές[1], Κωνσταντίνος Κάππας [2]

[1]Εκπαιδευτικός Πληροφορικής ΠΕ20, dimmat@gmail.com
[2]Εκπαιδευτικός Πληροφορικής ΠΕ19, kostas@kappas.eu

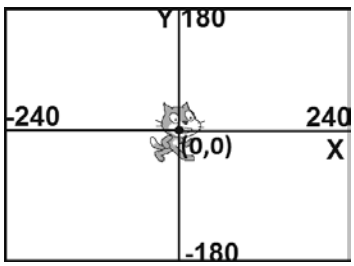
Περίληψη

Στην εργασία αυτή προτείνεται ένας αλγόριθμος για την αποκοπή γραμμών (line clipping) στο προγραμματιστικό περιβάλλον Scratch. Οι περισσότεροι γνωστοί αλγόριθμοι αποκοπής γραμμών βασίζονται στους δύο πιο δημοφιλείς: α) των Cohen-Sutherland και β) των Liang-Bansky. Και οι δύο αυτοί αλγόριθμοι κατά την εφαρμογή τους στο Scratch, απαιτούν πολλές συγκρίσεις και πραγματοποιούν έναν μεγάλο αριθμό υπολογισμών για το επιθυμητό αποτέλεσμα και, εν γένει, δεν είναι τόσο απλοί στην υλοποίησή τους. Ο προτεινόμενος αλγόριθμος δείχνει σημαντικά γρήγορος, είναι απλός οπότε μπορεί πολύ εύκολα να ενταχθεί και στην εκπαιδευτική διαδικασία.

Λέξεις κλειδιά: αποκοπή γραμμής, αλγόριθμος, γρήγορος, απλός, αποδοτικός.

1. Εισαγωγή

Η εύρεση του αλγορίθμου προέκυψε από την ανάγκη για αποκοπή γραμμών κατά την ενασχόληση των συγγραφέων με το Scratch και τη σχεδίαση γραφικών μέσα από αυτό. Η οθόνη του Scratch είναι συγκεκριμένων διαστάσεων (480 εικονοστοιχεία πλάτος x 360 εικονοστοιχεία ύψος) στην οποία ο χρήστης μπορεί να κινεί τις διάφορες μορφές ή να σχεδιάζει σχήματα με τη βοήθεια της πένας. Το Scratch όπως και πολλές άλλες γλώσσες προγραμματισμού, δεν επιτρέπει στις μορφές ή στην πένα να «βγουν» εκτός των ορίων της οθόνης του. Αν για παράδειγμα, ο προγραμματιστής θελήσει να σχεδιάσει μία πάρα πολύ μεγάλη γραμμή η οποία εκτείνεται εκτός των ορίων, αυτό δεν είναι εφικτό με τις ήδη υπάρχουσες επιλογές (Scratch Wiki, 2017).

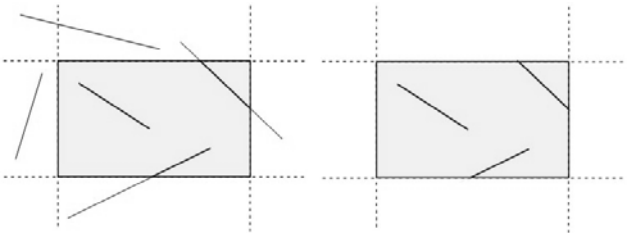


Εικόνα 1. *Διαστάσεις οθόνης του Scratch*

ISBN: 978-960-578-032-6

Τη λύση στο πρόβλημα της σχεδίασης μιας μεγάλης γραμμής ή γενικότερα ενός μεγάλου σχήματος έρχεται να δώσει η τεχνική που ονομάζεται «αποκοπή γραμμής» (line clipping).

Ως αποκοπή γραμμής ορίζεται η διαδικασία αφαίρεσης των τμημάτων της γραμμής τα οποία βρίσκονται εκτός μιας επιθυμητής περιοχής (Hearn, 1997).

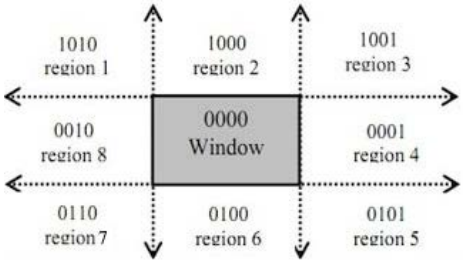


Εικόνα 2. *Περιοχή πριν και μετά την αποκοπή γραμμών*

Η διαδικασία της αφαίρεσης των περιττών τμημάτων γίνεται χρησιμοποιώντας τα μαθηματικά. Ο προγραμματιστής σχεδιάζει το τμήμα της γραμμής που βρίσκεται εντός των ορίων χρησιμοποιώντας είτε την εξίσωση της ευθείας y=a*x+b είτε κάνοντας χρήση διανυσμάτων.

Οι πιο γνωστοί αλγόριθμοι για την αποκοπή γραμμών είναι δύο: α) Ο αλγόριθμος των Cohen-Sutherland και β) ο αλγόριθμος των Liang-Barsky. Πάνω σε αυτούς βασίστηκαν κι άλλοι αλγόριθμοι, όπως οι Cyrus-Beck, Nicholl-Lee-Nicholl, οι οποίοι όμως θεωρούνται παραλλαγές των πρώτων.

Ο αλγόριθμος των Danny Cohen και Ivan Sutherland αναπτύχθηκε το 1967 κατά τη δημιουργία ενός εξομοιωτή πτήσεων. Θεωρείται ένας από τους πρώτους αλγόριθμους αποκοπής γραμμής στην ιστορία της σχεδίασης γραφικών. Σύμφωνα με αυτόν, ο δισδιάστατος χώρος στον οποίο βρίσκεται η προς αποκοπή γραμμή χωρίζεται σε εννέα τμήματα προκειμένου να εντοπιστούν σε ποια από αυτά βρίσκονται τα δύο σημεία που ορίζουν την γραμμή. Ανάλογα με τις περιοχές αυτές, ο αλγόριθμος πραγματοποιεί πλήρη, μερική ή καθόλου σχεδίασή της (Foley, 1996).

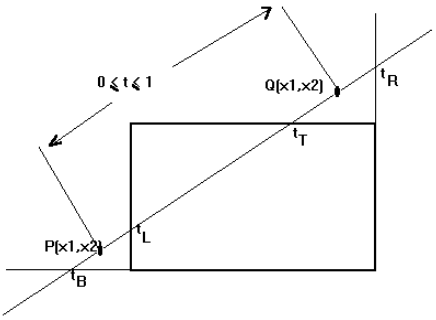


Εικόνα 3. *Τα εννέα τμήματα που χωρίζει τον δισδιάστατο χώρο ο αλγόριθμος Cohen-Sutherland*

ISBN: 978-960-578-032-6

Η υλοποίηση του συγκεκριμένου αλγορίθμου στο Scratch απαιτεί έναν σχετικά μεγάλο αριθμό συγκρίσεων που πρέπει να γίνουν προκειμένου να προσδιοριστούν οι περιοχές που βρίσκονται τα δύο σημεία που ορίζουν τη γραμμή. Επιπλέον, η εύρεση των περιοχών αυτών απαιτεί πράξεις λογικού ΚΑΙ (bitwise AND) κάτι το οποίο δεν είναι άμεσα εφικτό και για τον λόγο αυτό θα πρέπει να δημιουργηθούν οι αντίστοιχες υπορουτίνες. Η κατασκευή όμως μιας τέτοιας υπορουτίνας επιβαρύνει κατά πολύ σε ταχύτητα τον αλγόριθμο.

Ο αλγόριθμος των You-Dong Liang και Brian Barsky χρησιμοποιεί την εξίσωση της ευθείας καθώς και κάποιες ανισότητες για να βρει την περιοχής αποκοπής και να προσδιορίσει τα σημεία τομής της με την προς σχεδίαση γραμμή (Liang & Barsky, 1984). Όμως, κι αυτός ο αλγόριθμος στην υλοποίησή του στο Scratch απαιτεί έναν σχετικά μεγάλο αριθμό συγκρίσεων προκειμένου να σχεδιαστεί η γραμμή με αποτέλεσμα να μην είναι ιδιαίτερα αποδοτικός.



Εικόνα 4. *Προσδιορισμός της προς αποκοπή γραμμής με τον αλγόριθμο Liang-Barsky*

Τις παραπάνω δυσκολίες των δύο γνωστών αλγορίθμων αποκοπής γραμμής στο Scratch φαίνεται να ξεπερνά ο προτεινόμενος αλγόριθμος. Στοχεύει στην απλότητα και την ταχύτητα και πραγματοποιεί μόνο τις απολύτως απαραίτητες συγκρίσεις προκειμένου να προσδιορίσει αν η αρχή και το τέλος της γραμμής βρίσκονται εντός της περιοχής αποκοπής. Επιπλέον, κάνει χρήση μόνο των απολύτως απαραίτητων μεταβλητών.

Το παρόν άρθρο έχει την εξής διάρθρωση: Στην Ενότητα 2 γίνεται παρουσίαση του προτεινόμενου αλγόριθμου αποκοπής γραμμής, στην Ενότητα 3 παρουσιάζονται τα αποτελέσματα που προέκυψαν μετά τη σύγκριση του αλγορίθμου αυτού με τους δύο πιο δημοφιλείς αντίστοιχους αλγόριθμους (Cohen-Sutherland και Liang-Barsky) στο Scratch και στην Ενότητα 4 αναφέρονται συμπεράσματα που προέκυψαν από την μελέτη και την χρήση του αλγορίθμου στην πράξη καθώς και γίνονται

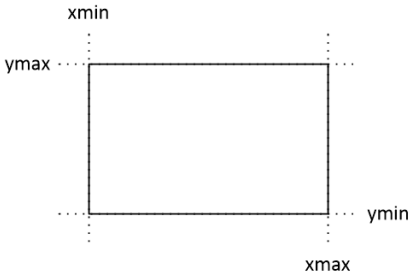
ISBN: 978-960-578-032-6

προτάσεις για βελτίωση καθώς και προτάσεις για ένταξή του στην εκπαιδευτική διαδικασία.

2. Ο προτεινόμενος αλγόριθμος αποκοπής γραμμών

2.1 Θεωρητικό υπόβαθρο

Θεωρούμε ότι θέλουμε να αποκόψουμε μια γραμμή εντός μιας ορθογώνιας περιοχής η οποία προσδιορίζεται από τα εξής σημεία: (xmin, ymax) και (xmax, ymin). Η περιοχή αυτή φαίνεται στην Εικόνα 5.



Εικόνα 5. *Περιοχή αποκοπής*

Έστω (x1, y1) και (x2,y2) δύο γνωστά σημεία που προσδιορίζουν τη γραμμή που θέλουμε να σχεδιάσουμε. Σύμφωνα με τα μαθηματικά, η κλίση m της γραμμής είναι πάντα σταθερή και προσδιορίζεται από τον λόγο:

$$m = \frac{(y2 - y1)}{(x2 - x1)} \Rightarrow$$

$$\Rightarrow y2 - y1 = m * (x2 - x1) \Rightarrow$$

$$\Rightarrow y2 = m * (x2 - x1) + y1$$

Για ένα οποιοδήποτε σημείο (x,y) της γραμμής, ο παραπάνω τύπος μπορεί να γραφεί υπό τη μορφή εξίσωσης ως εξής:

$$y = m * (x - x1) + y1 \Rightarrow$$

$$\Rightarrow y = \frac{(y2 - y1)}{(x2 - x1)} \cdot (x - x1) + y1 \qquad (A)$$

Ομοίως, αν λύσουμε ως προς x, η παραπάνω εξίσωση γίνεται:

$$x = \frac{(x2 - x1)}{(y2 - y1)} \cdot (y - y1) + x1 \qquad (B)$$

ISBN: 978-960-578-032-6

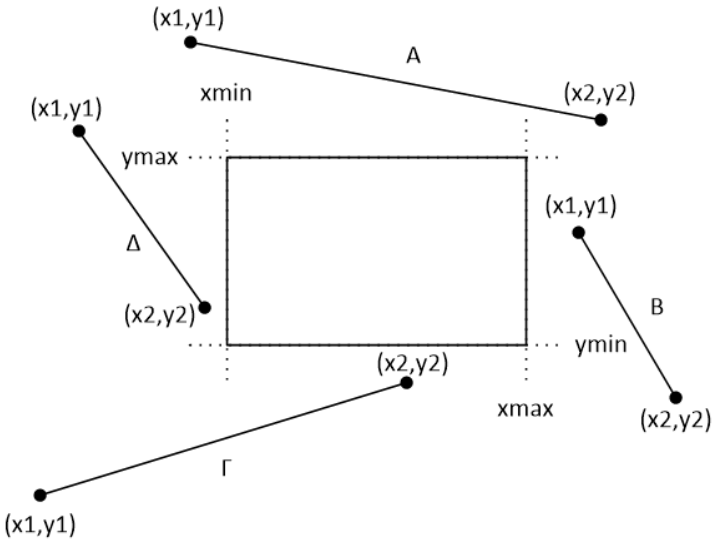
Οι δύο παραπάνω εξισώσεις (Α) και (Β) αποτελούν την γνωστή από τα μαθηματικά εξίσωση της ευθείας y=m*x+b και θα χρησιμοποιηθούν στον αλγόριθμο για τον προσδιορισμό του τμήματος της γραμμής εντός της περιοχής αποκοπής.

2.2 Ανάλυση των βημάτων του αλγορίθμου

Έστω ότι η προς αποκοπή γραμμή προσδιορίζεται από τα σημεία (x1,y1) και (x2,y2).

Κατά το πρώτο βήμα του αλγορίθμου ελέγχεται αν και τα δύο σημεία της γραμμής βρίσκονται εκτός της περιοχής αποκοπής και ταυτόχρονα στο ίδιο τμήμα (πάνω, κάτω, δεξιά, αριστερά). Αν κάτι από τα παρακάτω συμβαίνει τότε ολόκληρη η γραμμή είναι εκτός της περιοχής αποκοπής και ο αλγόριθμος δεν την σχεδιάζει (βλ. Εικ. 6):

x1<xmin ΚΑΙ x2<xmin (η γραμμή βρίσκεται αριστερά της περιοχής αποκοπής)
x1>xmax ΚΑΙ x2>xmax (η γραμμή βρίσκεται δεξιά της περιοχής αποκοπής)
y1<ymin ΚΑΙ y2<ymin (η γραμμή βρίσκεται κάτω από την περιοχή αποκοπής)
y1>ymax ΚΑΙ y2>ymax (η γραμμή βρίσκεται πάνω από την περιοχή αποκοπής)

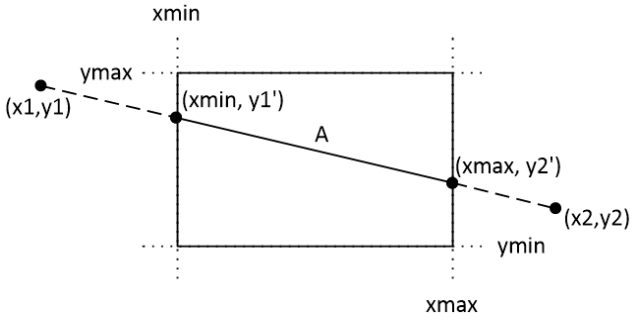


Εικόνα 6. *Οι γραμμές Α, Β, Γ, Δ δεν σχεδιάζονται σύμφωνα με το 1ο βήμα του αλγορίθμου*

Στο δεύτερο βήμα, ο αλγόριθμος συγκρίνει τις συντεταγμένες των δύο σημείων με τα όρια της περιοχής αποκοπής. Δηλαδή συγκρίνει κάθε ένα από τα x1 και x2 με τα όρια xmin και xmax αντίστοιχα καθώς και κάθε ένα από τα y1,y2 με τα όρια ymin και ymax. Αν κάποια από τις παραπάνω συντεταγμένες είναι εκτός ορίων

ISBN: 978-960-578-032-6

τότε το συγκεκριμένο όριο χρησιμοποιείται στην εξίσωση που προσδιορίζει την γραμμή μας προκειμένου αλλάξουν τα σημεία της γραμμής και να επιτευχθεί αποκοπή (βλ. Εικ. 7).



Εικόνα 7. *Αλλαγή των σημείων που προσδιορίζουν τη γραμμή βάσει των ορίων της περιοχής αποκοπής*

Για κάθε μία από τις συντεταγμένες των δύο σημείων και σύμφωνα με τις εξισώσεις (A) και (B) της προηγούμενης παραγράφου, οι συγκρίσεις και οι αλλαγές που γίνονται είναι:

- Αν το $x_i < xmin$ τότε

  $x_i = xmin$

  $$y_i = \frac{(y2 - y1)}{(x2 - x1)} \cdot (x\,min - x1) + y1$$

- Αν το $x_i > xmax$ τότε

  $x_i = xmax$

  $$y_i = \frac{(y2 - y1)}{(x2 - x1)} \cdot (x\,max - x1) + y1$$

- Αν το $y_i < ymin$ τότε

  $y_i = ymin$

  $$x_i = \frac{(x2 - x1)}{(y2 - y1)} \cdot (y\,min - y1) + x1$$

- Αν το $y_i > ymax$ τότε

  $y_i = ymax$

ISBN: 978-960-578-032-6

$$x_i = \frac{(x2 - x1)}{(y2 - y1)} \cdot (y\,max - y1) + x1$$

όπου i: από 1 έως 2.

Το τρίτο και τελευταίο βήμα του αλγορίθμου συγκρίνει τα νέα σημεία που προέκυψαν μετά τις αλλαγές κι αν κι αυτά είναι εντός των ορίων της περιοχής αποκοπής τότε σχεδιάζει τη γραμμή που προσδιορίζεται από αυτά.

2.3 Ο αλγόριθμος σε ψευδογλώσσα

Για λόγους ευκολίας και πρακτικότητας, παρατίθεται ο αλγόριθμος της αποκοπής γραμμών σε ψευδογλώσσα:

```
Αλγόριθμος  Scratch_Line_Clipping
Δεδομένα // x[2], y[2], xmin, ymax, xmax, ymin //
Αν  όχι(x[1]<xmin και x[2]<xmin) και όχι(x[1]>xmax και x[2]>xmax) τότε
  Αν  όχι(y[1]<ymin και y[2]<ymin) και όχι(y[1]>ymax και y[2]>ymax) τότε
    i<-1
    Αρχή_επανάληψης
      Αν x[i] < xmin ΤΟΤΕ
         x[i]<-xmin
         y[i]<- ((y[2]-y[1])/(x[2]-x[1])*(xmin-x[1])+y[1]
      Τέλος_αν
      Αν x[i] > xmax ΤΟΤΕ
         x[i]<-xmax
         y[i]<- ((y[2]-y[1])/(x[2]-x[1])*(xmax-x[1])+y[1]
      Τέλος_αν
      Αν y[i] < ymin ΤΟΤΕ
         y[i]<-ymin
         x[i]<- ((x[2]-x[1])/(y[2]-y[1])*(ymin-y[1])+x[1]
      Τέλος_αν
      Αν y[i] > ymax ΤΟΤΕ
         y[i]<-ymax
         x[i]<- ((x[2]-x[1])/(y[2]-y[1])*(ymax-y[1])+x[1]
      Τέλος_αν
      i<-i+1
    Μέχρις_ότου i>2
    Αν  όχι(x[1]<xmin και x[2]<xmin) και όχι(x[1]>xmax και x[2]>xmax) τότε
      Αν  όχι(y[1]<ymin και y[2]<ymin) και όχι(y[1]>ymax και y[2]>ymax) τότε
       Σχεδίασε_γραμμή(x[1],y[1],x[2],y[2])
      Τέλος_αν
```

ISBN: 978-960-578-032-6

```
    Τέλος_αν
  Τέλος_αν
Τέλος_αν Τέλος Scratch_Line_Clipping
```

3. Σύγκριση με άλλους αλγόριθμους αποκοπής γραμμών

3.1 Προετοιμασία

Για να προσδιορίσουμε την αποδοτικότητα του προτεινόμενου αλγορίθμου αποφασίσαμε να τον συγκρίναμε με τους δύο πιο γνωστούς αλγόριθμους αποκοπής γραμμών: τον αλγόριθμο των Cohen-Sutherland και τον αλγόριθμο των Liang-Barsky. Όλοι οι αλγόριθμοι υλοποιήθηκαν έχοντας ως βασικούς άξονες την απλότητα και την αποδοτικότητα και καταβλήθηκε προσπάθεια προκειμένου να έχουν την ίδια περίπου δομή.

Το προγραμματιστικό περιβάλλον του Scratch πλεονεκτεί σε τέτοιου είδους συγκρίσεις σε σχέση με άλλα περιβάλλοντα για τους ακόλουθους λόγους: α) το Scratch διαθέτει ενσωματωμένη οθόνη/περιοχή σχεδίασης όπου μπορεί να δει κανείς άμεσα το οπτικό αποτέλεσμα του αλγορίθμου, β) διαθέτει εντολές χρονομέτρησης με συνέπεια την εύκολη μέτρηση του απαιτούμενου χρόνου εκτέλεσης, γ) ο αλγόριθμος είναι προσβάσιμος από όλους μέσω του διαδικτύου, δ) επιτρέπει την προσωρινή παρέμβαση των χρηστών στις εντολές του αλγορίθμου για πειραματισμό.

Το πείραμα που διεξήχθη ήταν το εξής: Κάθε ένας από τους αλγορίθμους θα έπρεπε να δημιουργεί 10.000 τυχαίες γραμμές σε ένα δισδιάστατο χώρο τετραπλάσιο από την βασική οθόνη σχεδίασης του Scratch. Ένας τέτοιος χώρος προσδιορίζεται από τα σημεία (-960, 720) και (960,-720). Η περιοχή αποκοπής θα έπρεπε να βρίσκεται στο κέντρο της οθόνης σχεδίασης του Scratch και ορίζεται από τα σημεία (-100, 75) και (100,-75), δηλαδή έχει πλάτος 200 εικονοστοιχεία, ύψος 150 εικονοστοιχεία. Η γραμμές θα έπρεπε να εκτείνονται τυχαία, οπουδήποτε μέσα στον μεγάλο δισδιάστατο χώρο, να αποκόπτονται τα περιττά τμήματά τους και να σχεδιάζονται μόνο τμήματα εκείνα που βρίσκονται εντός της περιοχής αποκοπής (βλ. Εικ. 8).

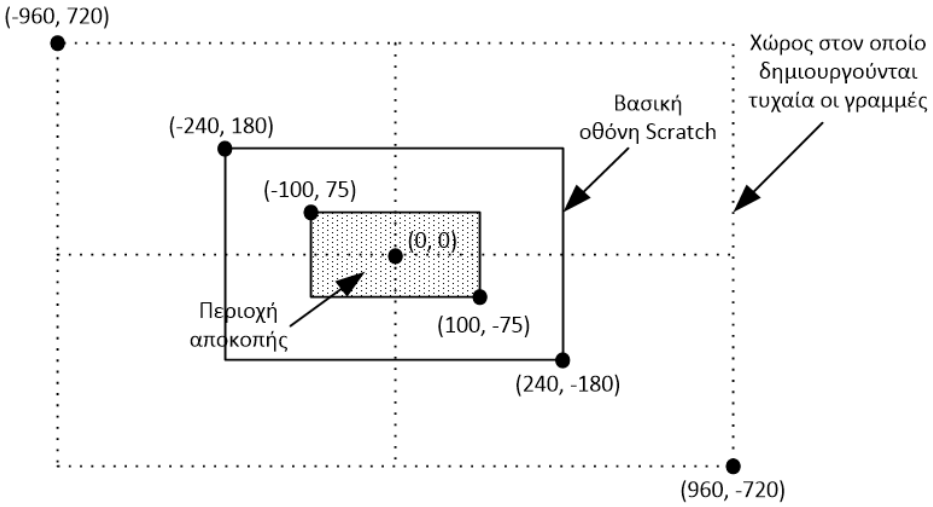
Ο χρόνος που απαιτείται για τη σχεδίαση και των 10.000 γραμμών καταγράφεται για κάθε έναν από τους αλγορίθμους μετά το πέρας τους. Η διαδικασία επαναλαμβάνεται 10 φορές και στο τέλος υπολογίζεται ο μέσος χρόνος εκτέλεσης.

Το υπολογιστικό σύστημα που χρησιμοποιήθηκε για τη διεξαγωγή των μετρήσεων διέθετε τα εξής χαρακτηριστικά: α) Επεξεργαστή AMD FX 4300 Quad Core στα 3.80GHz , β) Μνήμη RAM 8GB, γ) Λ/Σ Windows 10 Professional, δ) Scratch 2.0

ISBN: 978-960-578-032-6

Πίνακας 1. *Διευθύνσεις URL με τις υλοποιήσεις των αλγορίθμων στο Scratch*

Αλγόριθμος	Διεύθυνση URL
Cohen-Sutherland	https://scratch.mit.edu/projects/166917422
Liang-Barsky	https://scratch.mit.edu/projects/166820820
Προτεινόμενος	https://scratch.mit.edu/projects/166877443



Εικόνα 8. *Προσδιορισμός του δισδιάστατου χώρου για τη δημιουργία των γραμμών καθώς και προσδιορισμός της περιοχής αποκοπής*

3.2 Αποτελέσματα

Στον πίνακα που ακολουθεί φαίνονται οι χρόνοι εκτέλεσης των συγκρινόμενων αλγορίθμων:

Πίνακας 2. *Χρόνοι εκτέλεσης των αλγορίθμων για 10000 γραμμές*

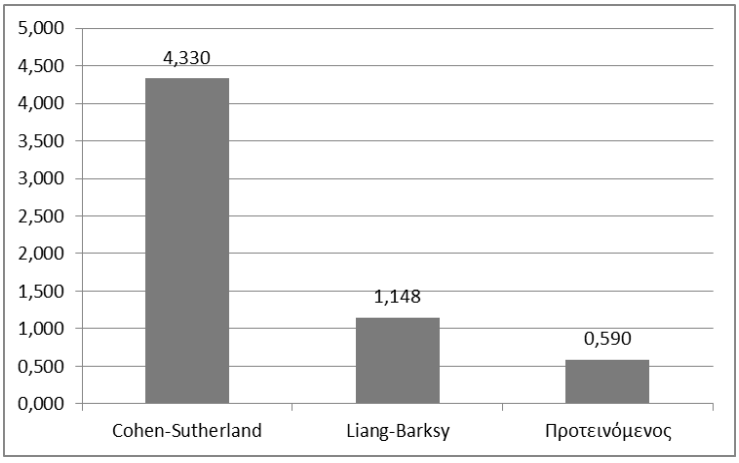
Εκτέλεση	Cohen-Sutherland (sec)	Liang-Barsky (sec)	Προτεινόμενος Αλγόριθμος (sec)
1	4,245	1,159	0,625
2	4,279	1,149	0,632
3	4,456	1,159	0,592
4	4,364	1,135	0,618
5	4,275	1,132	0,592
6	4,352	1,142	0,427
7	4,347	1,167	0,630

ISBN: 978-960-578-032-6

8	4,356	1,132	0,620
9	4,295	1,143	0,577
10	4,326	1,159	0,587
Μέσος χρόνος :	**4,330**	**1,148**	**0,590**

4. Συμπεράσματα

Μελετώντας τα αποτελέσματα και συγκρίνοντας τους χρόνους εκτέλεσης μεταξύ τους παρατηρούμε ότι ο προτεινόμενος αλγόριθμος είναι περίπου 10 φορές πιο γρήγορος από τον αλγόριθμο των Cohen-Sutherland και σχεδόν 2 φορές πιο γρήγορος από τον αλγόριθμο των Liang-Barsky στο περιβάλλον του Scratch.



Εικόνα 9. *Γράφημα με τους χρόνους σύγκρισης μεταξύ των αλγορίθμων (μικρότερη τιμή → καλύτερο)*

Παρατηρούμε επίσης ότι ο αλγόριθμος των Cohen-Sutherland είναι πιο αργός στην υλοποίησή του στο Scratch σε σχέση με τους άλλους δύο διότι οι πράξεις του λογικού ΚΑΙ (bitwise AND) που απαιτούνται σε κάθε επανάληψη καθυστερούν σημαντικά την εκτέλεση του. Όπως προαναφέρθηκε, το Scratch δεν διαθέτει εγγενή πράξη λογικού ΚΑΙ μεταξύ αριθμών και συνεπώς ο προγραμματιστής θα πρέπει να βρει έμμεσα το ζητούμενο αποτέλεσμα κάνοντας διαδοχικά υπολογισμούς με το υπόλοιπο της διαίρεσης των προς σύγκριση αριθμών.

Ο αλγόριθμος των Liang-Barsky δείχνει αρκετά πιο γρήγορος σε σχέση με τον Cohen-Sutherland και μοιάζει να πλησιάζει τον προτεινόμενο αλγόριθμο σε ταχύτητα αλλά θεωρείται πιο πολύπλοκος στην υλοποίησή του και ελαφρώς πιο δυσνόητος διότι εμπεριέχει δυσκολότερες μαθηματικές έννοιες.

ISBN: 978-960-578-032-6

Αν το παραπάνω πείραμα επαναληφθεί με διαφορετικές παραμέτρους (π.χ. σχεδίαση περισσότερων γραμμών, διαφορετικός δισδιάστατος χώρος δημιουργίας των γραμμών, διαφορετική περιοχή αποκοπής, άλλο υπολογιστικό σύστημα) τα αποτελέσματα είναι ανάλογα.

Ενδιαφέρον επίσης παρουσιάζει και η πιθανή υλοποίηση του προτεινόμενου αλγορίθμου στις τρεις διαστάσεις. Με τις κατάλληλες τροποποιήσεις και με μια σειρά επιπλέον συγκρίσεων ο αλγόριθμος μπορεί εύκολα να αποκόπτει 3Δ γραμμές στον χώρο, ενδεχομένως το ίδιο γρήγορα και αποδοτικά με τις δύο διαστάσεις.

Όσον αφορά την εκπαιδευτική διαδικασία, ο αλγόριθμος μπορεί εύκολα να διδαχθεί σε οποιαδήποτε βαθμίδα της δευτεροβάθμιας εκπαίδευσης και άνω. Προαπαιτούμενες γνώσεις είναι η εξίσωση της ευθείας από την Άλγεβρα και τα Μαθηματικά ενώ επιθυμητές γνώσεις είναι τα βασικά στοιχεία του δομημένου προγραμματισμού καθώς και η έννοια της μεταβλητής. Μία προτεινόμενη προσέγγιση για τη διδασκαλία της αποκοπής γραμμής με τη χρήση του συγκεκριμένου αλγορίθμου είναι η εξής: Αρχικά, ζητείται από τους μαθητές να δημιουργήσουν μία νέα μορφή της οποίας οι διαστάσεις είναι 1x1 εικονοστοιχεία. Στη συνέχεια ζητείται από αυτούς, γράφοντας σενάριο και κάνοντας χρήση της πένας, να προσδιορίσουν τρία σημεία εντός των ορίων της περιοχής σχεδίασης του Scratch και να σχεδιάσουν το τρίγωνο που σχηματίζεται από τα σημεία αυτά. Μετά, ζητείται από αυτούς να προσδιορίσουν τρία νέα διαφορετικά σημεία που βρίσκονται εκτός της περιοχής σχεδίασης και να σχεδιάσουν πάλι το αντίστοιχο τρίγωνο. Αποτέλεσμα της όποιας προσπάθειας σχεδίασης εκτός των ορίων είναι ότι το τρίγωνο παραμένει σταθερό όσο κι αν προσπαθήσει να ξεπεράσει κανείς τα όρια της οθόνης του Scratch. Στο επόμενο βήμα, ο διδάσκων εξηγεί τους περιορισμούς της σχεδίασης που έχει το Scratch καθώς και κάνει μια μικρή εισαγωγή στην εξίσωση της γραμμής από τα μαθηματικά και πως αυτή εφαρμόζεται στις δύο διαστάσεις (2Δ). Τέλος, κάνει την παρουσίαση του προτεινόμενου αλγόριθμου αποκοπής γραμμών ως έναν τρόπο επίλυσης του προβλήματος στην σχεδίαση του τριγώνου αφού οι γραμμές του τριγώνου μπορούν απλά να αποκοπούν και αυτό να σχεδιάζεται σωστά εντός των ορίων της οθόνης.

Εν κατακλείδι, ο αλγόριθμος αποκοπής γραμμών που παρουσιάστηκε, φαίνεται να είναι αρκετά γρήγορος στην υλοποίησή του στο Scratch αφού δύναται να σχεδιάσει 10.000 γραμμές σε χρόνο κάτω του 1 δευτερολέπτου και είναι αρκετά απλός. Δίνει τη δυνατότητα σε όσους προγραμματίζουν στο Scratch και θέλουν να σχεδιάσουν πάρα πολύ γρήγορα οτιδήποτε εκτείνεται πέρα από τα όρια να άρουν τους περιορισμούς του συγκεκριμένου περιβάλλοντος. Τέλος, δίνει τη δυνατότητα σε εκπαιδευτικούς να διδάξουν την έννοια της αποκοπής γραμμής πολύ απλά και με πολύ γρήγορα αποτελέσματα.

ISBN: 978-960-578-032-6

Αναφορές

Foley, J., & Van Dam, A., & Feiner, S., & Hughes, J. (1996). Computer Graphics: Principles and Practice. Addison-Wesley Publishing Company.

Godse, A. P. (2009): Computer Graphics. Technical Publications Pune.

Hearn, D., & Baker, M., P. (1997). Computer Graphics C Version. Prentice Hall.

Iraji, Mazandarami & Motameni, An efficient line clipping algorithm based on Cohen-Sutherland line clipping algorithm. American journal of Scientific Research, 14, 2011.

Kodituwakku, Wijeweera & Chamikara. An efficient algorithm for line clipping in computer graphics programming.

Liang, Y. D., & Barsky, B. (1984). A New Concept and Method for Line Clipping. ACM Transactions on Graphics, 3(1):1–22.

Lu, G., & Wu, X., & Peng, Q. (2001). An efficient line clipping algorithm basedon adaptive line rejection. Best Papers of CAD and CB 2001.

Nicholl, T., M., & Lee, D., T., & Nicholl, R., A. (1987). An Efficient New Algorithm for 2-D Line Clipping: Its Development and Analysis, Computers and Graphics, Vol. 21, No. 4, pp. 253-262.

Pachghare, V. K. (2011). Comprehensive Computer Graphics. Laxmi Publications LTD.

Pandey & Jain. (2013). Comparison of various line clipping algorithms for Improvement. International Journal of Modern Engineering Research.

Scratch Wiki. (2017). Stage. Ανακτήθηκε 29 Απριλίου, 2017, από Wiki: https://wiki.scratch.mit.edu/wiki/Stage

Wikipedia. (2017). Cohen–Sutherland algorithm. Ανακτήθηκε 30 Απριλίου, 2017, από Wiki: https://en.wikipedia.org/wiki/Cohen-Sutherland_algorithm

Wikipedia. (2017). Liang–Barsky algorithm. Ανακτήθηκε 30 Απριλίου, 2017, από Wiki: https://en.wikipedia.org/wiki/ Liang–Barsky_algorithm

Wikipedia. (2017). Line clipping. Ανακτήθηκε 30 Απριλίου, 2017, από Wiki: https://en.wikipedia.org/wiki/Line_clipping

ISBN: 978-960-578-032-6

Abstract

In this paper an algorithm for line clipping is being presented. Most algorithms for line clipping are based on the two popular algoritmhs: a) Cohen-Sutherland and b) Liang-Barksy. Both of them are widely used but in the Scratch environment they execute many comparisons, do a lot of calculations and they are not easily applicable. The proposed algorithm is simpler, significantly faster and can be applied easily in the education process.

Λέξεις κλειδιά: line clipping, algorithm, fast, simple, efficient.

ISBN: 978-960-578-032-6