דוח מיני פרוייקט 2

השיפורים בחלק זה:

1. MultiThreading – ריבוי תהליכונים.
2. Bounding volume hierarchy – היררכיית הקופסאות התוחמות.

מטרת שיפורים אלו היא לקצר באופן משמעותי את זמן הריצה הדרוש ליצירת סצנות מורכבות.

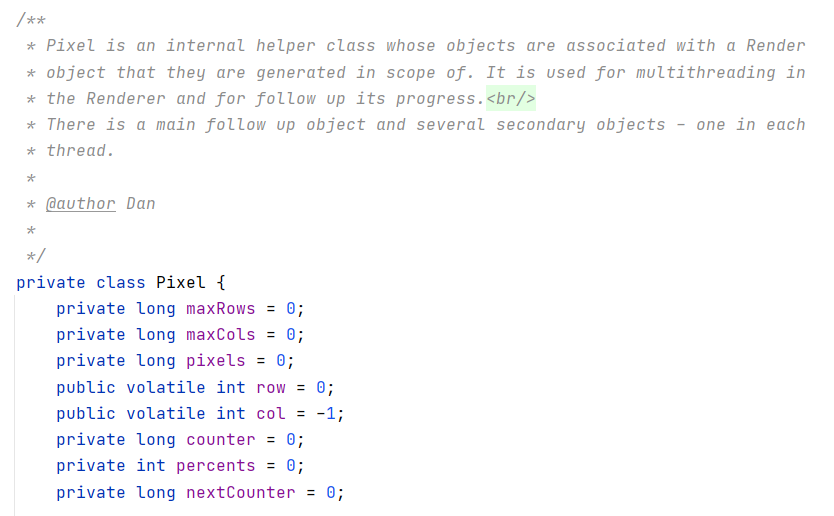
הסבר שיפור ראשון – MultiThreading:

הרעיון הוא לקצר את משך פעולת הרינדור ולחלק פעולה זו לכמה תהליכונים שיעבדו במקביל (למעשה בפרוייקט שלנו השתמשנו בשלושה תהליכונים כיוון שביותר מכך אין כבר שיפור רציני).

כל תהליכון יקבל פיקסל ויבדוק חיתוך של הקרניים עם הפיקסל וכאשר הוא יגמור הוא יקבל את הפיקסל הבא וחוזר חלילה.

השינויים שנעשו במחלקה render על מנת להתאימה לעבודה עם תהליכונים:

נוספה מחלקה פנימית בשם pixel שכשמה כן היא – מייצגת פיקסל:



(maxRows, maxCols הם מספר עמודות ושורות של פיקסלים, בהתאמה

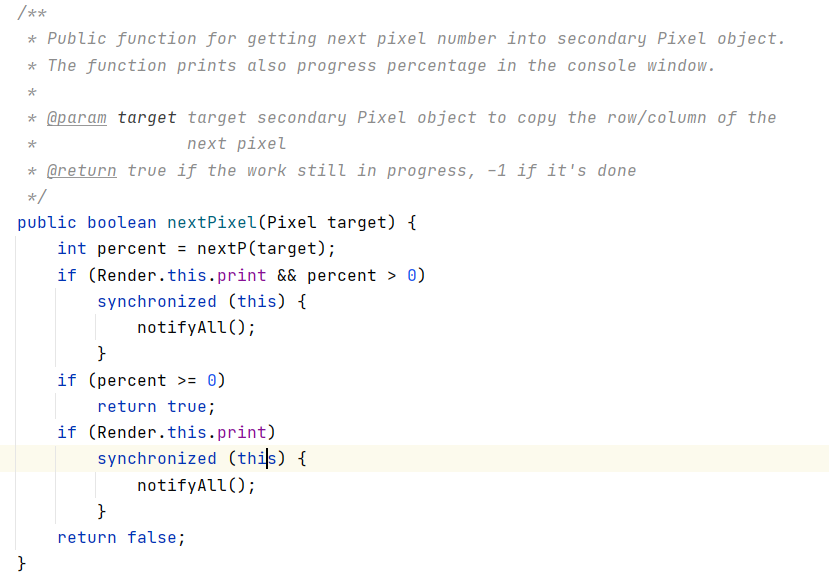
row, col הם מספר העמודה והשורה של הפיקסל, בהתאמה

percents הוא אחוזי התקדמות הרינדור – מתעדכן בכל התקדמות מסוימת ברינדור).

הפונקציות במחלקה החדשה:

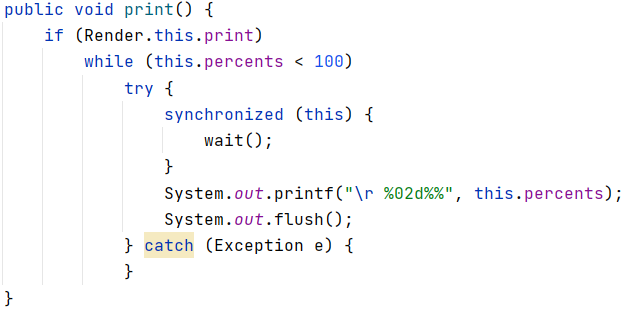


הפונקציה הזו אחראית להעביר את הפיקסל הבא לרינדור.

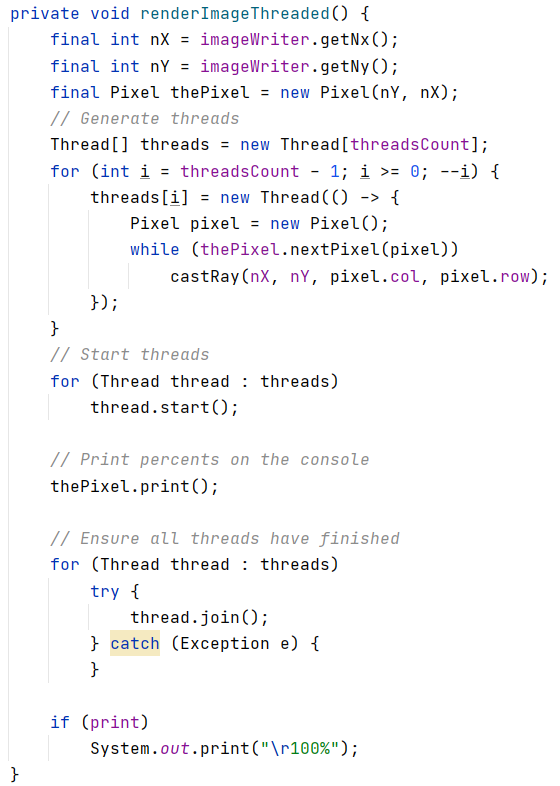


פועלת בצורה דומה לקודמת וקוראת לפונקציית הדפסת האחוזים במקרה והמשתנה הבוליאני print שווה ל – true (מה שמעיד על כך שאנחנו מעוניינים בהדפסת קצת בהתקדמות הרינדור באחוזים).

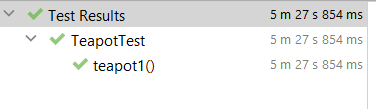
הפונקציה המדפיסה בפועל את קצב ההתקדמות:



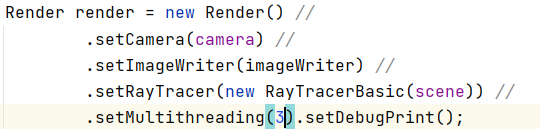
הפונקציה במחלקה render היוצרת את התהליכונים ומבצעת את הרינדור:



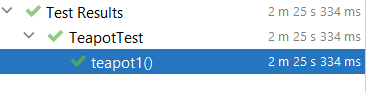
הרצנו את המבחן של ה- teapot (הקומקום) עם תהליך 1, ועם השיפורים שעשינו בשלב 1 לקח לנו:



לאחר מכן הרצנו את אותו מבחן, הפעם עם 3 תהליכונים



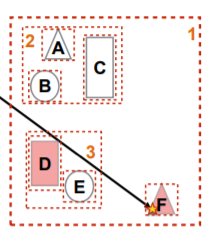
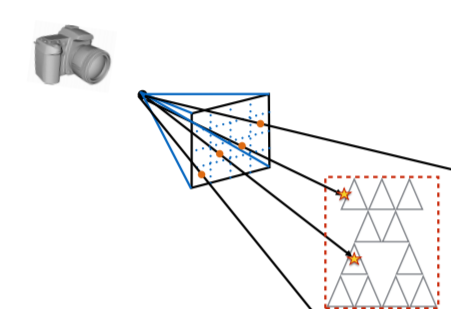
וקיבלנו שיפור של פי 2 בזמן:



הסבר השיפור השני Bounding volume hierarchy –

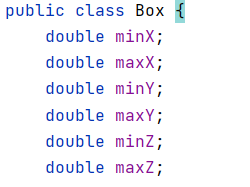
הרעיון הוא לחסוך עוד בזמן ריצה על ידי הימנעות מבדיקת חיתוכים מיותרים.

כאשר נרצה לבדוק האם קרן חותכת גוף גיאומטרי כלשהוא, נבדוק קודם האם היא חותכת את הקופסא העוטפת את כלל הגופים שבסצנה, במידה וכן נבדוק האם הקרן חותכת את הקופסא העוטפת את הגיאומטריה הספציפית ורק אם כן נמשיך ונבדוק חיתוך בין הגיאומטריה לסצנה.



את המימוש למעשה ביצענו כך:

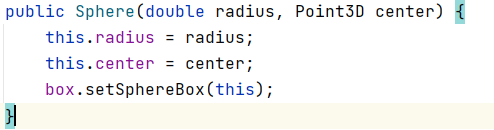
יצרנו מחלקה חדשה בשם Box שמייצגת קופסא חוסמת.



הוספנו לכל גוף גיאומטרי בו אנחנו משתמשים, שדה של קופסא החוסמת אותו, וכן ב – geometries (קופסא החוסמת מספר גיאומטריות).



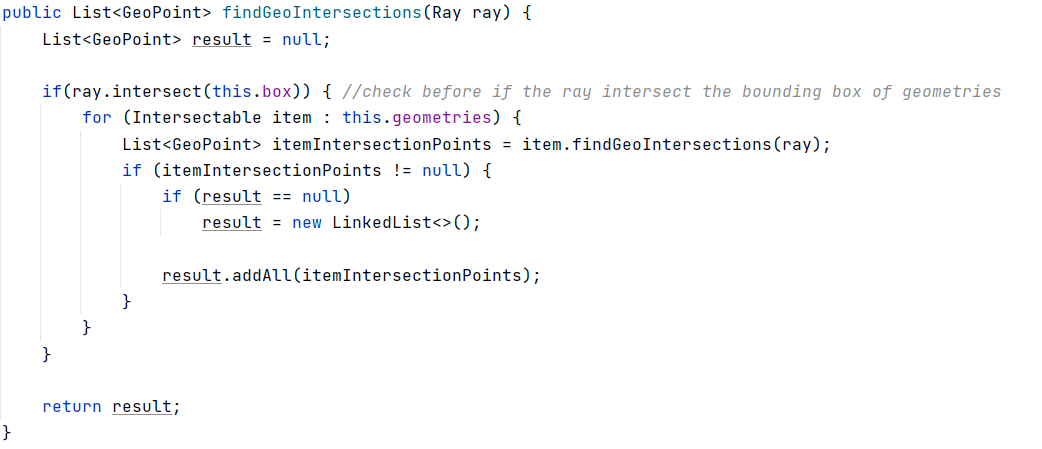
בבנאי של כל גיאומטריה וכן בבנאי של geometries (וכן בפונקציה add של geometries אחרי הוספה של גיאומטריות צריך לבנות קופסא חדשה) קראנו לפונקציה setBox המתאימה על מנת לבנות את הקופסאות סביב הגופים.



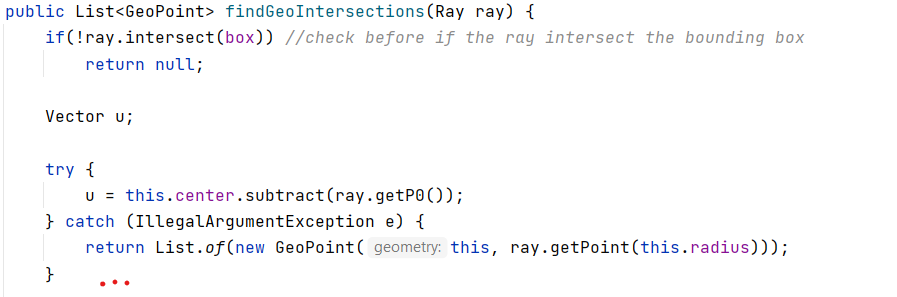
לכן עכשיו בכל פונקציה הבודקת חיתוך בין קרן לגיאומטריות או לגיאומטריה ספציפית, הוספנו בדיקה טרם אנחנו מתחילים לבדוק חיתוכים.

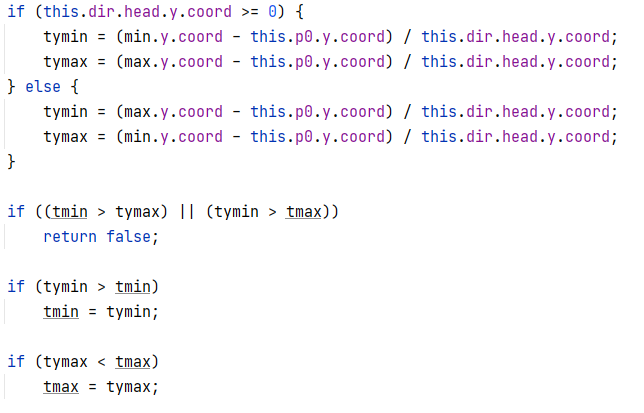
בבדיקת חיתוך בין קרן לגיאומטריות שבסצנה, הוספנו בדיקה הבודקת אם הקרן בכלל חותכת את הקופסא הגדולה העוטפת את הגיאומטריות, ואם כן עבור כל גיאומטריה קודם שבודקים חיתוך שלה עם הקרן נבדוק האם הקרן חותכת את הקופסא העוטפת את הקרן , ואם כן נמשיך ונבדוק חיתוך ספציפי יותר של הקרן עם הגיאומטריה.

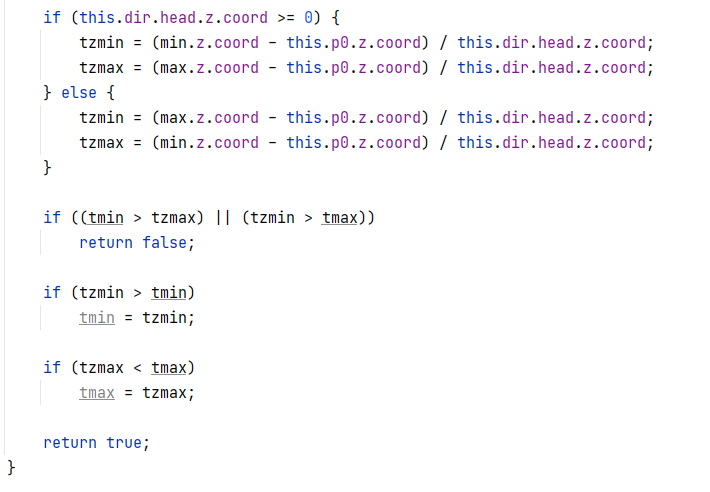
בדיקת חיתוך של הקרן עם הקופסא העוטפת את geometries:



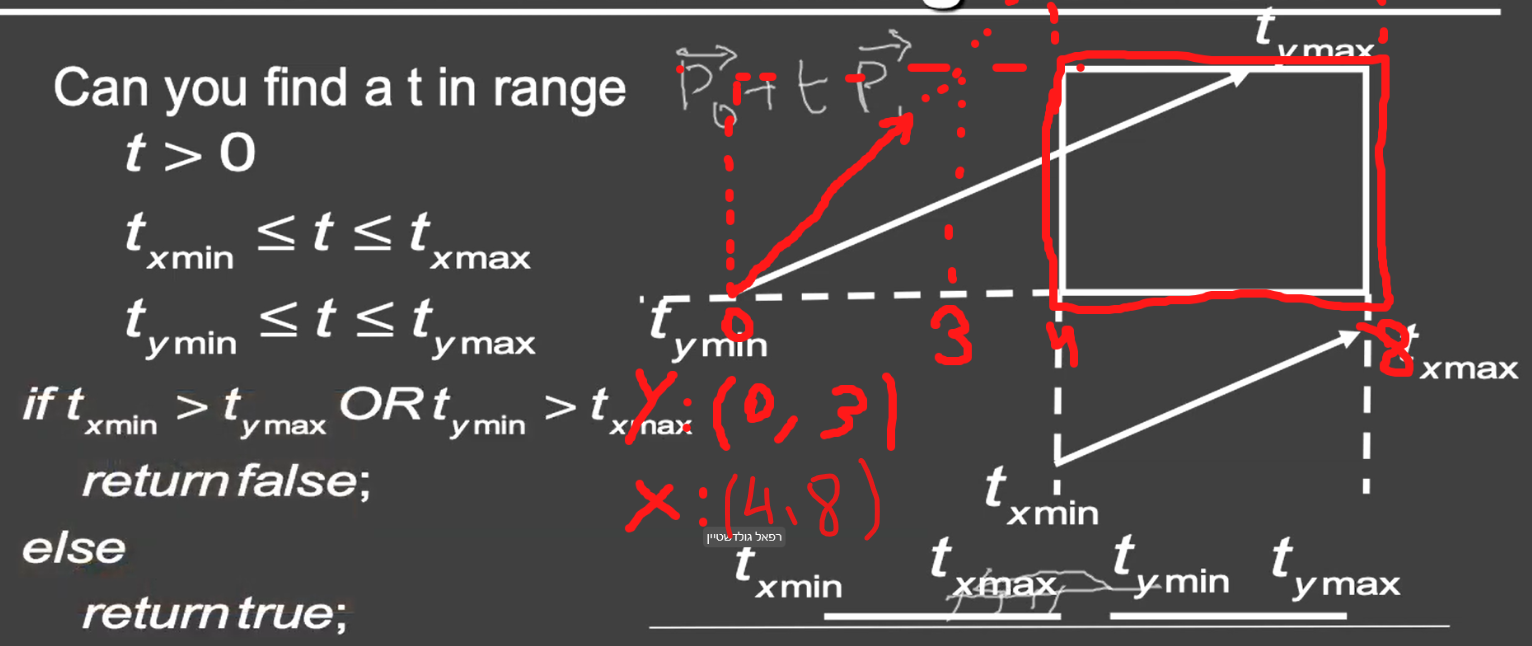
בדיקת חיתוך של הקרן עם הקופסא העוטפת גוף מסויים (לדוגמא sphere):



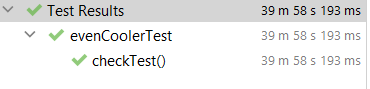
האלגוריתם למציאת חיתוך בין קרן לקופסא:



הסבר:



לאחר השיפורים שעשינו התמונה רצה במחשב שלי בזמן: (כולל את השיפור הראשון גם)



כשאצל יונתן לפני השיפורים זה לקח שבע שעות בערך.

