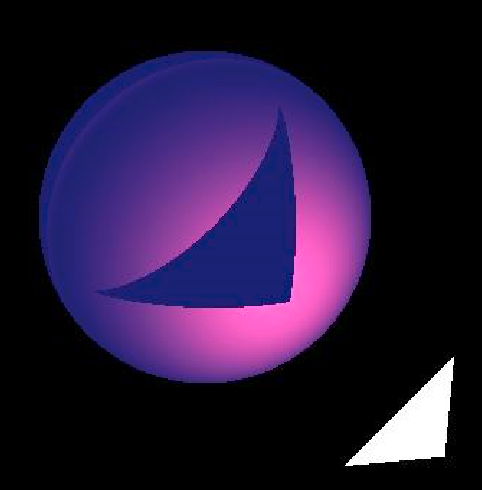
בס"ד

**Supersampling**

***תיאור הבעיה:***



כפי שניתן לראות בתמונה הנ"ל, הסצנה שנוצרה נראית כלא אמתית,

מכוון שהקצוות מזוגזגים.

***הסבר למקור הבעיה:***

מכוון שאנחנו שולחים קרן יחיד על כל פיקסל, וכל קרן דוגם משטח גדול כך שנקודה אחת משפיע על כל השטח

לדוגמא באיור:

כל הפיקסל, מושפע מהצבע הספציפי של הנקודה שמרכזו ללא תלות בשאר הנקודות של השטח של הפיקסל.

***פתרון הבעיה: distribution of rays***

במקום להפיץ קרן אחת לכל פיקסל, אנחנו נשלח מספר קרניים ע"פ דרישת דיוק של לקוח.

***תיאור האלגוריתם:***

כמובן שאנחנו לא משנים את הפונקציונליות של יצירת קרן, כי המטרה היא שאנחנו מעבדים תמונה מסצנה שמורכבת מגיאומטריות, תאורות, מצלמה ומשטח צפייה, והמטרה שלנו זה ליעיל את התמונה ע"פ דרישת הבעיה שלעיל, ועל כן אנחנו צריכים לשפר את הפונקציה renderImage שנמצאת ב- Render שמעבדת ויוצרת קרניים למשטח צפייה פר פיקסל בשביל לחשב את הצבע בפיקסל.

אנחנו נשפר את הפונקציה כך שתיצור מספר קרניים והצבע של הפיקסל ייקבע ע"י ממוצע של חישוב צבעים של כל קרן. כך שאני יקבל יותר אמינות על צבע בסביבה מסוימת.

1. יצרנו שדה בשם level כך שמייצג את מספר הקרניים לפי הדרישה, כמובן שמאותחל באופן דיפולטיבי למס' מסוים.



1. בתחילת הפונקציה אתחלנו רשימה color במטרה לאסוף את הצבעים של כל קרני הפיקסל.

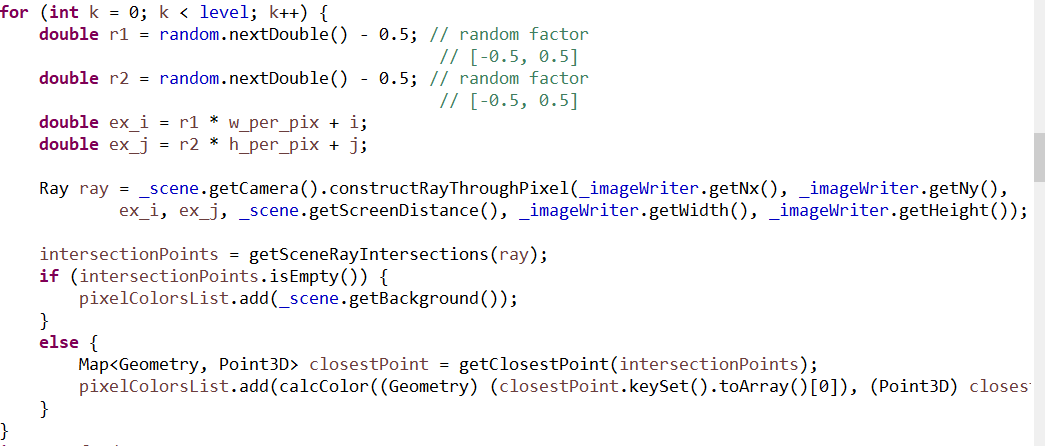


1. איך יצרנו יותר קרניים לאותו פיקסל?

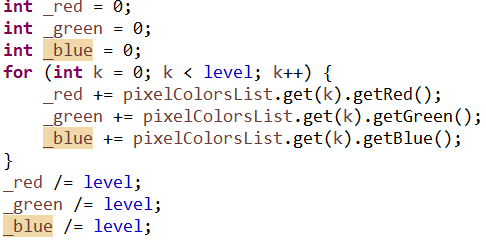
הקצנו אובייקט ממחלקה Random שתעזור לנו עם הפונקציות שלו לתת מס' רנדומלי בין 0 ל-1, מכוון שאנחנו נמצאים במרכז הפיקסל ע"י הנוסחה, אז אנחנו צריכים לדאוג שבכל פיקסל נתון אנחנו נזוז רק בתחום לכן אפשר לזוז עם ציר ה-x רק עד חצי של רוחב הפיקסל וכנ"ל בציר ה- y , לכן, החסרנו את המספר הרנדומלי בחצי כדי שנקבל מס' רנדומלי בטווח מינוס 0.5 לפלוס 0.5

והכפלנו בגודל של הרוחב/גובה כך שלא נחרוג.

יצרנו קרן לפי החישובים הרנדומלים, וחישבנו את צבעו והוספנו לרשימה- כל התהליך נעשה בלולאה כך שלכל פיקסל חישבנו את כל הצבעים של כל הקרניים.



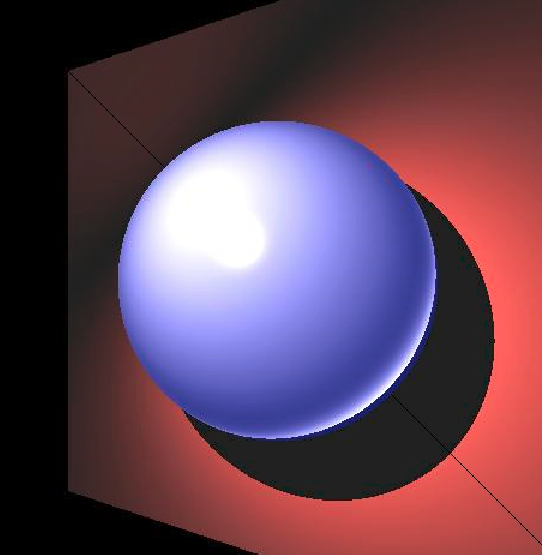
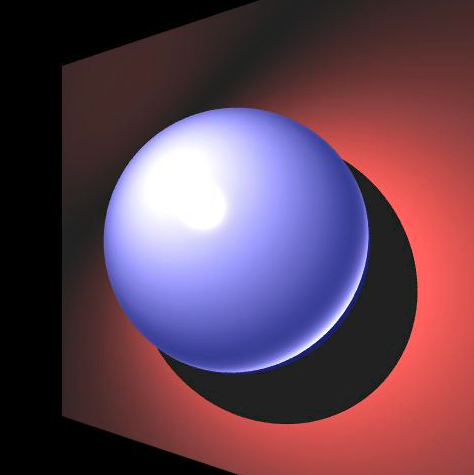
1. בסוף לכל פיקסל חישבנו את הממוצע של כל הצבעים (רשימה)



1. צבענו את הפיקסל בחישוב 4 מחקנו את הרשימה בשביל הפיקסל הבא. 

***Unit tests:***

לפני הייעול: אחרי הייעול:

לפני הייעול: אחרי הייעול:

