ב"ה

פרוייקט רובוט חשאי-פגישה 6#-שירה עוזרי

מנחה: פרופ' רועי פורן

עידכון על התקדמות:

משימות בהנחיית מר פורן לשבוע הקרוב:

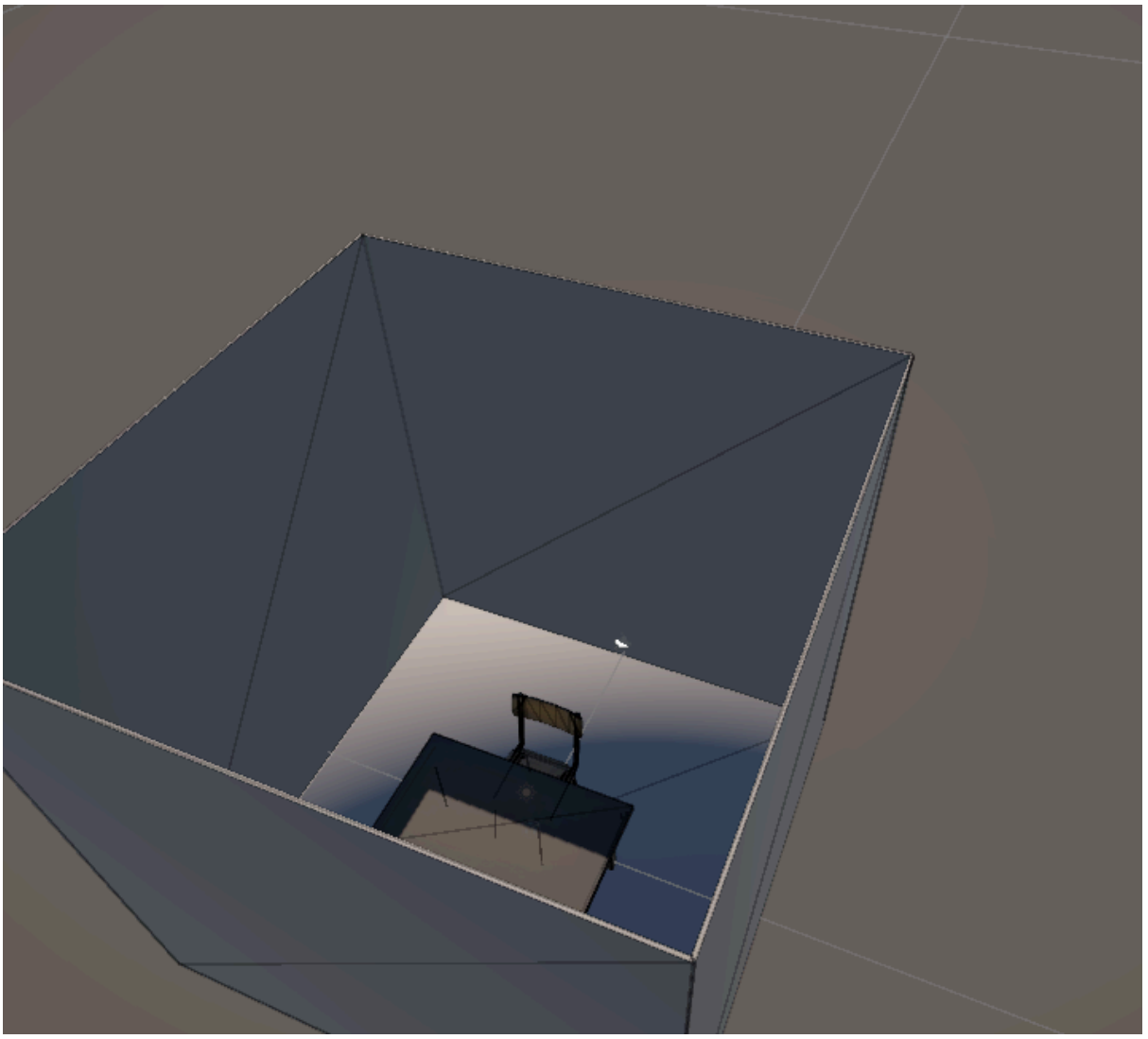
1) יצירת סצנת משרד ריאליסטית:
לבנות חדר שדומה למשרד אמיתי (עם קירות, שולחן, כיסא וכו').
להשתמש ב־placeholder-ים פשוטים – בלי מודלים מורכבים.
לוודא פרופורציות אמינות בין אובייקטים.

2) שיפור חישוב ה־Heatmap ל־Online:
לחשב את המפה באופן דינמי תוך כדי ריצה, לא מראש.
להריץ את חישובי הנראות ב־Thread נפרד או Coroutine.
לעדכן את המפה בהדרגה בזמן שהרובוט מחכה או עומד.

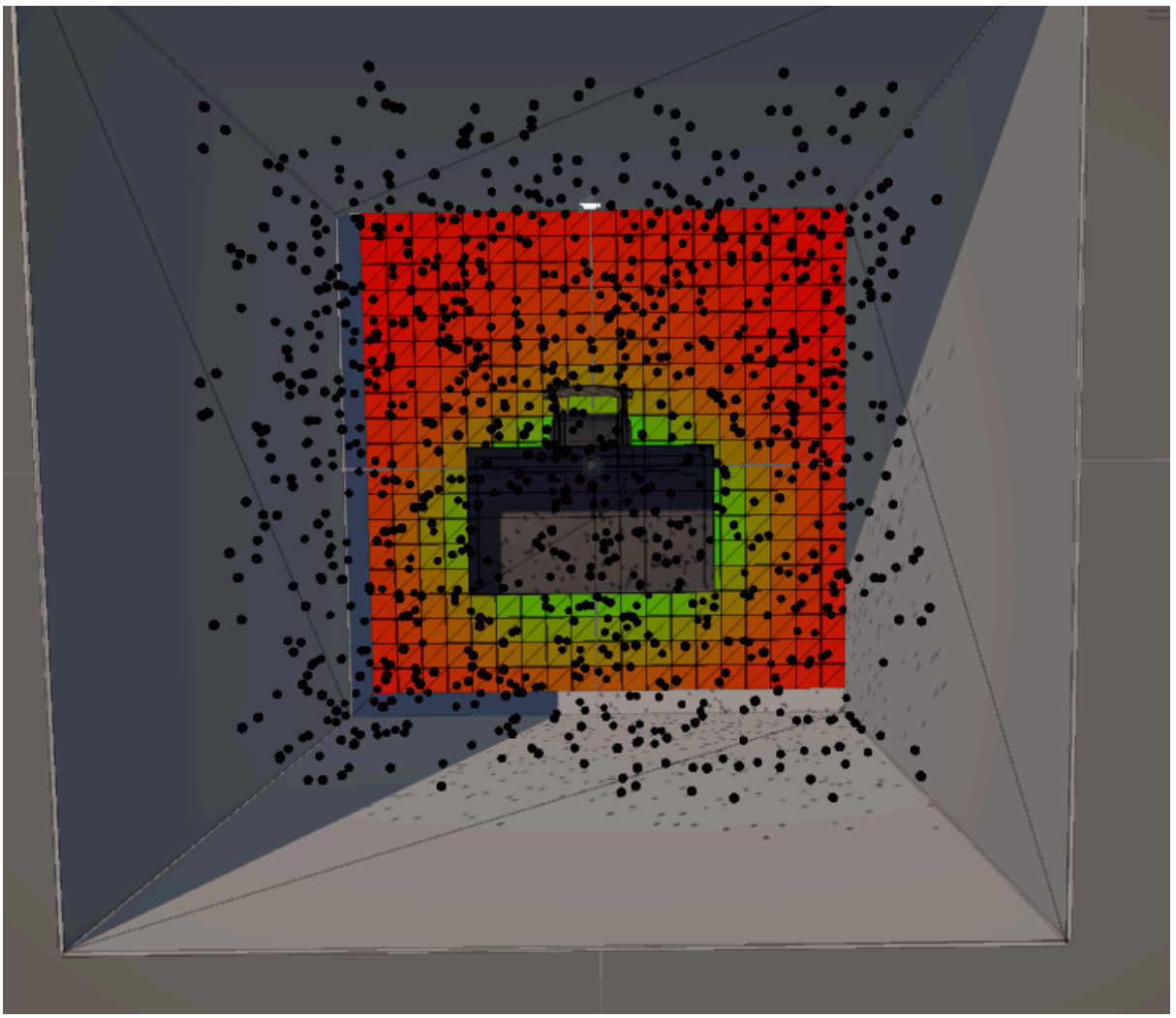
3) שיפור חישוב המסלול:
לשפר את שיטת חישוב המסלול (כרגע Dijkstra), לעבור ל־שיטה מתקדמת כמו A* או Gradient-Based, לפי המאמר ולהשוות את התוצאות מול Dijkstra (בדיוק, זמן, חשיפה).

משימה 1: יצירת סצנת משרד ריאליסטית:

יצרתי חדר עם כיסא ושולחן משרדיים:



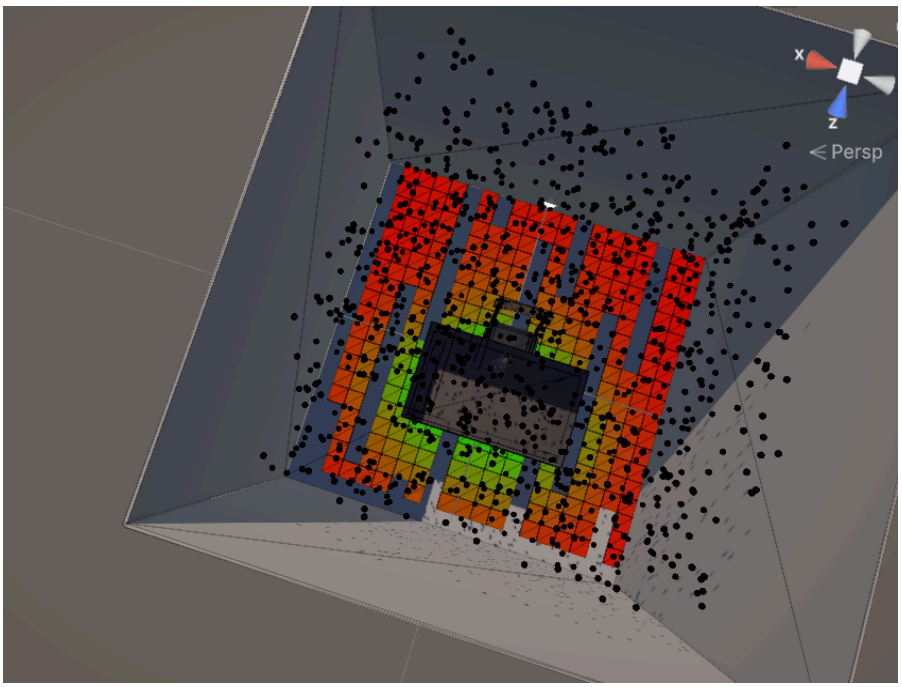
והפעלתי עליו את האלגוריתם:

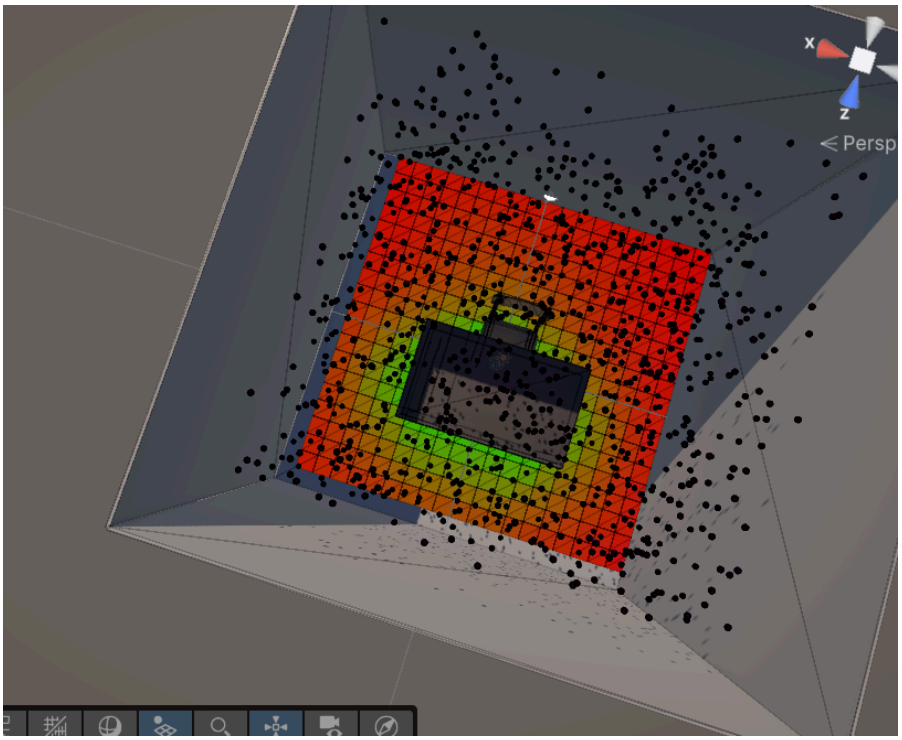


התמונה ממחישה את נכונות האלגוריתם בכך שהיא מציגה הבדל מובהק ברמות החשיפה בין אזורים פתוחים לאזורים מוצלים: תאים החשופים ישירות לנקודות תצפית קיבלו ערכי חשיפה גבוהים (צבועים באדום), בעוד שהתאים שמאחורי הרהיטים (כיסא ושולחן) קיבלו חשיפה נמוכה יותר (צבועים בירוק), בהתאם לחסימה האופטית שנוצרת. פיזור הקרניים והשפעת החסמים ניכרים בבירור, ומעידים על חישוב נראות תקין ואפקטיבי.

משימה 2: שיפור חישוב ה־Heatmap ל־Online:

בשלב זה מימשתי מערכת Heatmap אינטראקטיבית שמחשבת את רמת החשיפה של כל משבצת ברצפת הכיתה לנקודות תצפית שונות, תוך התחשבות במכשולים כמו שולחנות וכיסאות. כדי להבטיח חישוב מהיר ויעיל גם בסצנות מורכבות, חילקתי את המשימה למספר תהליכים מקבילים (Coroutines), כאשר מספר ה־Workers נקבע דינמית לפי מספר ליבות המעבד של המשתמש (SystemInfo.processorCount). כך ניתן לנצל בצורה מיטבית את המשאבים הזמינים בכל מחשב, ולבצע את החישוב בצורה סקיילבילית שמגיבה לגודל המפה ולמורכבות הסביבה. המימוש מבטיח גם דיוק בהתחשבות בהסתרה של אובייקטים וגם ביצועים טובים יותר.




במימוש הנוכחי שילבתי חישוב מקבילי ודינמי של ערכי ה־heatmap כדי לשפר ביצועים ולהתמודד עם סצנות מורכבות יותר. החישוב המקבילי מתבצע ע"י חלוקת הגריד לחלקים, כאשר כל חלק נשלח לחישוב עצמאי באמצעות Task.Run, וכך נוצרים מספר threads שעובדים במקביל – מה שמאפשר לנצל את המעבד בצורה יעילה יותר. החישוב עצמו אינו תלוי בין חלק לחלק, ולכן אין התנגשויות או צורך בסנכרון מורכב. לעומת זאת, יצירת האובייקטים עצמם (כמו הקוביות של התאים והנקודות במרחב) מתבצעת בלולאה רגילה על ה-main thread, מאחר ו־Unity אינה תומכת ביצירת אובייקטים מתוך תהליכים מקביליים. מהבחינה הדינמית – הקוד גמיש לגמרי לגודל הגריד או כמות נקודות התצפית, ומתעדכן אוטומטית לפי פרמטרים שהמשתמש קובע מראש, כך שכל שינוי בקלט מתקבל מיידית בהתאמה בחישוב.

**משימה 3 ותכנון לתקדמות:**

השלב הבא במימוש הוא לאפשר עדכון דינמי ורציף של המפה בזמן ריצה, כך שהרובוט יוכל להמשיך לנוע תוך כדי שהוא מגלה אזורים חדשים בחדר שלא היו חשופים לו קודם. כלומר, החישוב של ה־heatmap

יתבצע במקביל לתנועת הרובוט, בהתאם לשדה הראייה שלו בכל רגע נתון. במצב כזה, המסלול אל היעד לא יוכל להיות קבוע מראש, אלא יהיה עליי לחשב בכל רגע את המסלול החלקי הנוכחי הטוב ביותר ליעד, על בסיס המידע שזמין לרובוט בזמן אמת. מדובר באתגר שמשלב תכנון נתיב חלקי (partial path planning) עם תגובתיות סביבתית – דבר שדורש מעבר לאסטרטגיות מסלול חכמות יותר כמו A* או gradient-based navigation.