

Pic2Peak

הפיכת תמונת נוף דו-מימדית לסביבת משחק תלת-מימדית



מאי יחזקאל

טל חדד

מנחה: תמר שרוט

תוכן עניינים

4.....	הקדמה
5.....	מסגרת תיאורטית וסקירת ספרות
6.....	שיטות וארכיטקטורה
7.....	דיאגרמת UML
8.....	מסמך דרישות
11.....	תהליך הבניה
31.....	תוצאה
38.....	מסמך בדיקות
49.....	ביבליוגרפיה

רשימת איורים

7	איור 1 - דיאגרמת UML
12	איור 2 - מבנה כללי של רשת נוירונים
13	איור 3 - הדגמת פילטרים של שכבות קונבולציוניות
14	איור 4 - קטע קוד מנסיון הרצה ראשוני
15	איור 5 - גרפים המתארים את דיוק המודל
16	איור 6 - המחשת פילוח תמונה
17	איור 7 - המחשת מודל זיהוי אובייקטים
18	איור 8 - השוואה בין מודל פילוח תמונה למודל זיהוי אובייקטים
18	איור 9 - השוואה בין כלל השיטות שנידונו עד כה
20	איור 10 - דוגמא לתהליך תיוג
21	איור 11 - האובייקטים המוכרים למודל YOLOV3
22	איור 12 - לפני ואחרי הרצת המודל (דוגמה 1)
23	איור 13 - לפני ואחרי הרצת המודל (דוגמה 2)
29	איור 14 - מדריך להגדרות ששונ
29	איור 15 - תוצאת תהליך השינוי בתוכנה
30	איור 16 - חלון המידע בתוכנה
30	איור 17 - הדגמה לכתיבת קוד בתוכנה
31	איור 18 - ממשק המשתמש
32	איור 19 - הממשק לאחר העלאת תמונה
33	איור 20 - השלבים לאחר לחיצת START
34	איור 21 - חלון ההודעות
34	איור 22 - האובייקטים שהמודל זיהה
35	איור 23 - פלט המודל בטבלת CSV
36	איור 24 - הסקריפט בבלנדר
37	איור 25 - הסביבה שנבנתה בבלנדר

הקדמה

כידוע, העולם הטכנולוגי והוירטואלי בפרט נמצא בכל תחומי החיים השונים ועל כן הוא רלוונטי יותר מתמיד ומהווה חלק בלתי נפרד מחיינו. אותה רלוונטיות, מביאה לידי ביטוי טשטוש גבולות שבין המציאות, החיים, לבין העולם הוירטואלי. ניתן לראות זאת בפיתוחים אחרונים שמנסים להציג הדמיית מציאות. הדמיית המציאות התפתחה רבות, כאשר מהפיתוחים האחרונים והדומיננטים שנראו נמצאים משקפי 'מציאות מדומה' (VR), שאף משמשים לא רק את שחקני המחשב [1]. בנוסף לכך, אחד מהפיתוחים שנמצאים כעת בשיח הטכנולוגי הוא על ידי פיתוח מבית היוצר של הרשת החברתית 'פייסבוק', הנקרא 'Metaverse' שבהתאם לפייסבוק, גם פיתוח זה בא במטרה לחבר בין אנשים, אך בשל קצב ההתקדמות הטכנולוגית וההליך הוירטואליזציה שנעשה בכל העולם, ה-'Metaverse' שואפת להכניס את העולם הוירטואלי אל האינטרנט וליצור רשת חברתית במציאות מדומה [2].

על כן, עולה הצורך להתאים את תחום המשחקים גם כן. במסגרת סמינר זה, נציג את התוכנה Pic2Peak שמביאה ליד ביטוי שלב אחד קדימה בתחום הוירטואלי והגיימינג ומספקת עבור המשתמש את האפשרות לבחור באופן פשוט וחופשי את סביבת המשחק שלו. מהצורך שהוצג לעיל, אנו צופים שפיתוח זה יפיק תועלת בתחומים אחרים גם כן. מטרת הפרויקט היא לחקור ולפתח טכנולוגיה חדשנית, אשר תרחיב את האפשרויות העומדות כיום לרשותם של משחקי המחשב. וזאת בהתאם לעליית הביקוש בוירטואליזציה תלת מימדית.

לפיתוח זה מגוון יתרונות, כאשר הראשון הוא עבור היצרן ומאפשר לו לחסוך בזמן תכנות על סביבת המשחק לשחקן. יתרון שני הוא העובדה שפיתוח זה מהווה כגשר בין הדמיון למציאות, צורך שקיים כיום אצל השחקנים (לראייה: משקפי VR). יתרון שלישי קורה כאשר המערכת שלנו היא חלק ממשחק. סביבת המשחק הופכת לסביבה דינאמית ולפעמים מוכרת, אשר מקנה ערך מוסף לשחקן. יתרון נוסף הוא כאמור לעיל, הרחבת הפיתוח לתוכנות נוספות בחיינו כיום, והם הדמיות בתחום האדריכלות, הדמיות בתחום ההסברה והחינוך, הדמיות בתחום התרבות ועוד...

Pic2Peak היא מערכת שבעזרתה ניתן לבנות סביבת משחק תלת מימדית מתמונה אחת דו ממדית (RGB). מערכת זו מבוססת על שימוש בטכנולוגיית בינה מלאכותית (AI).

הבעיה שאנו מנסים לפתור היא המרת מישור דו-מימדי (תמונת נוף), למרחב תלת מימדי (סביבה גרפית).

מסגרת תיאורטית וסקירת ספרות

הבנה של סביבה מנקודת מבט אחת, כמו הערכת עומק וזיהוי אובייקטים, היא בעיה עתיקה שמנסים לפתח. מחקר שנעשה בשנת 2016 מציג שיטת הערכת עומק של תמונה מתצולם RGB יחיד. השיטה פותחה על בסיס רשת נוירונים, ויחסית הייתה מדויקת בשל העובדה שנבנתה על סט אימונים קטן באופן יחסי לשיטות הערכת עומק דומות [3]. לרשת נוירונים יש ארכיטקטורות שונות, כאשר זו שמשמשת בעיקר למשימות של זיהוי דפוסים בתמונות, נקראת CNN [4].

מחקר נוסף שנעשה בשנת 2019 מראה ניסיון להפיק את אותן מפות עומק מתמונות RGB רגילות וזאת גם כן על ידי שימוש בשיטת הבינה של רשת נוירונים. שיטה זו מוצגת בהשוואה לשיטות נוספות שמבצעות את אותה פעולה, וזאת במטרה להביא לידי ביטוי תוצאות איכותיות ומדויקות יותר. אחת הבעיות המוצגות בשימוש בשיטה זו היא בעיית שחזור הסצנות בתלת מימד, שכן אי שלמות של צילום המרחב או לחילופין אי בהירות או חומרים שקופים ומראות, עשויים לשבש את ההערכה שמבצעת התוכנה ולגרום לאי-דיוק של השחזור. עם זאת, החוקרים מציגים לכך פתרון וטוענים שניתן לשחזר תמונה כאשר משתמשים בחיישנים או במספר תמונות עם מצלמות איכותיות [5].

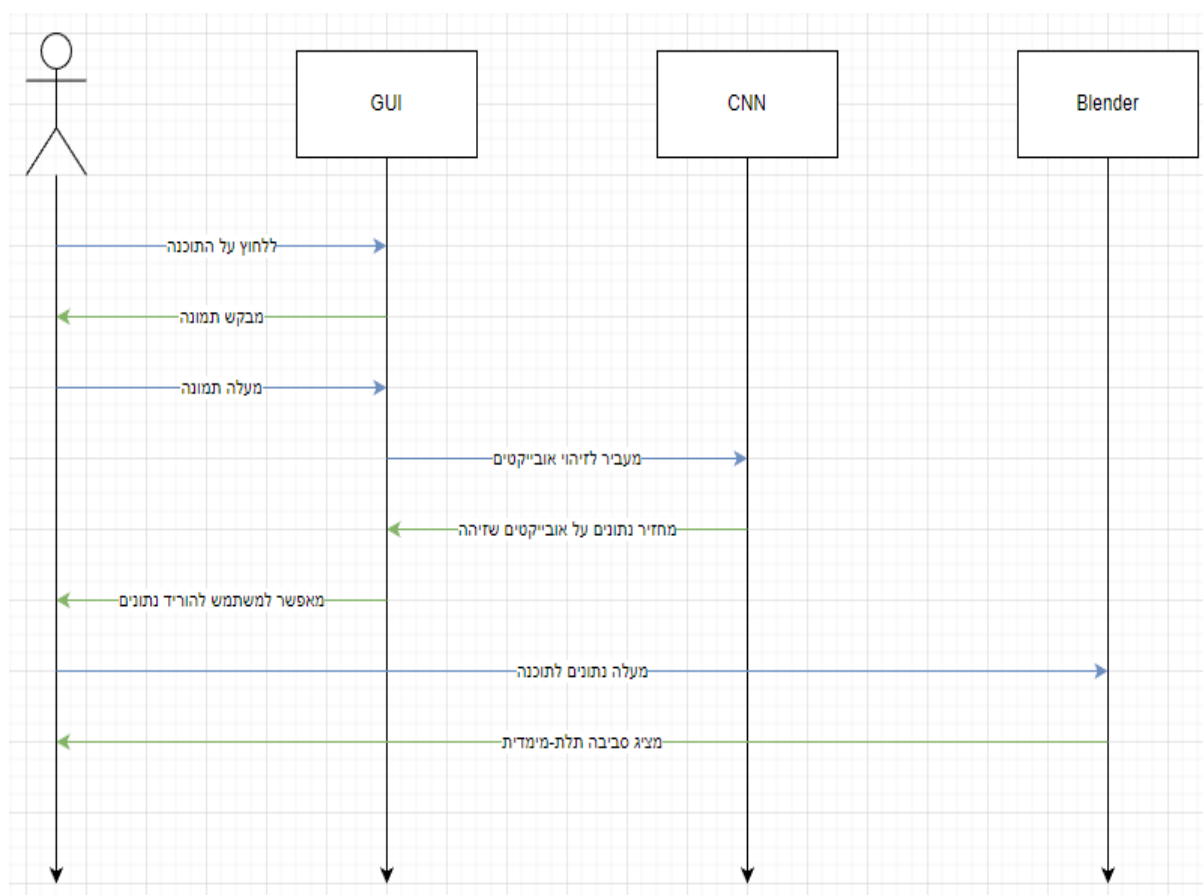
כשמדובר על שחזור סביבות מתמונה, קיימות מגוון תוכנות שמשתמשות בחיישנים בכדי ליצור סביבות שאותן מנסים לבנות. אחת מהם היא שיטה שמסתמכת על בקשת תמונה ומפת עומק (RGB-D), כאשר הנתונים שזו מוציאה יכולים לאמן מודל AI שתפקידו הוא למדוד עומקים בתמונה ולהעריך את תנחות האובייקטים בה. הדרך של שיטה זו להתמודד עם חפצים מוסתרים היא צילום כמה תמונות שונות [6].

תוכנה נוספת שמשתמשת בפורמט RGB-D מצליחה לתת לנו סביבה פנימית שלמה (חדר), שנוכל להסתכל עליה מזוויות שונות כרצוננו [7]. במאמר נוסף הציגו שיטה שגם כן מסתמכת על RGB-D שמזהה חפצים במרחב, מפעילה יד רובוטית שמצליחה לתפוס חפצים ולהזיזם ממקום למקום [8]. אחת מהבעיות הקשות ביותר של הבנת הסביבה היא תיחום אזורי התמונה (מה נמצא איפה). בדרך כלל אלגוריתם לתיחום תמונה מתבסס על Clustering ועל מידע נוסף של קווי מתאר וקצוות. לדוגמא, תיחום תמונה יכול להתבצע בהצלחה על ידי קיבוץ פיקסלים דומים הממוקמים סמוכים. אך קיים אלגוריתם נוסף שתוחם תמונה בעזרת רשת נוירונים. הוא עושה זאת בעזרת זיהוי האובייקטים שבתמונה [9].

שיטות וארכיטקטורה

- התמונות שעליהם נבסס את המערכת יהיו בפורמט [.JPEG](#).
- ארכיטקטורת המערכת Pic2Peak תהיה מובנת תכנות עצמים OOP.
- מערכת ה-Desktop (GUI) תיבנה בשפת Python
- מודל הבינה המלאכותי יהיה מסוג [.CNN](#)
- מודל ה-CNN ייובא מ-[TensorFlow](#).
- מאגר הנתונים (תמונות) יהיה [.Open Image](#).
- הסביבה תיבנה בתוכנה ליצוג גרפי [.Blender](#).

דיאגרמת UML



איור 1 - דיאגרמת UML

מסמך דרישות

ממשק משתמש:

1. המערכת תאפשר תצוגה תלת ממדית של נתונים על מנת להציג אותה למשתמש.
2. המערכת תכיל סטנדרט תצוגות ותפעול אחיד למערכות ההפעלה לינוקס וחלונות.
3. המערכת תציג חלון ראשי אשר הינו חלון תצוגה יחיד וקבוע המכסה את כל שטח המסך ומרכיביו הם:
 - 3.1. שורת תפריט – התפריט הראשי יכלול את האופציות העלאה, איפוס, יציאה ועזרה.
 - 3.2. שטח תצוגה – שטח עיקרי בחלון עליו תוצג התצוגה התלת ממדית.
 - 3.3. שורת הודעות – שורה הכוללת הודעה המודיעה על הצלחה או על שגיאה.
4. החלון הראשי יבנה על בסיס תבנית אחידה, כאשר יוקצו מקומות קבועים עבור הצגת הרכיבים הייחודיים למסך.
 5. שורת תפריט:
 - 5.1. שורת התפריט תופיע לכל אורך חלקו העליון של המסך.
 - 5.2. המשתמש יכול להעלות תמונה על ידי פעולת הקשה אחת על כפתור "העלאה".
 - 5.3. כפתור "העלאה" יפתח את סייר הקבצים על מנת לאפשר העלאת תמונה מהמחשב.
 - 5.4. המשתמש יוכל לאפס את מסך התצוגה בכפתור "איפוס" במידה ומוצגת בו סביבה, על מנת להעלות תמונה חדשה.
 - 5.5. המשתמש יוכל לצאת מהמערכת על ידי פעולת הקשה אחת על כפתור ה"יציאה".
 - 5.6. המשתמש יכול לקבל מדריך לתוכנה על ידי פעולת הקשה אחת על כפתור "עזרה".
 6. שטח תצוגה:
 - 6.1. החלון יהיה חלון של Windows עם האפשרויות שלו.
 7. שורת הודעות:
 - 7.1. שורת ההודעות תופיע לכל אורך חלקו התחתון של המסך.
 8. המסך יחולק לאזורים ובהם מרכיביו המפורטים בדרישה מספר 2, ניתן יהיה לבצע שינוי גודל למרכיבים בכפוף לשליטת המשתמש על סביבת העבודה.
 9. תצורת החלונות תבנה כך שינוצל מלוא שטח התצוגה.
 10. הודעות המערכת תוצגנה ברכיב שורת ההודעות הנמצא החלון הראשי.

11. המערכת תספק למשתמש התראה ברורה על שגיאה והנחיה ספציפית מה עליו לעשות לגבי סוג השגיאה.

11.1. הודעת השגיאה תלווה בהדגשה ויזואלית על מנת להדגישה.

12. המערכת תתמוך בשפה האנגלית כשפה הטבעית של המערכת בשביל להיות רלוונטית לאוכלוסייה גדולה יותר של משתמשים.

13. לחצני פעולה ושדות יהיו זמינים רק כאשר הפעילות המבוצעות באמצעותם אפשרית.

14. המערכת תאפשר מעבר יעיל ומושכל בין רכיביה השונים, כאשר המשתמש יכול להיות מנותב מיידית לכל רכיבי התפריט הראשי הרלוונטיים לו.

תכונות מערכת:

1. כמערכת ארצה לזהות אובייקטים בתמונת הקלט על מנת להמירם לאובייקטים תלת מימדים

1.1. כמערכת אצטרך אלגוריתם למידת מכונה על מנת לדעת לזהות אובייקטים בתמונה

1.1.1. כמערכת אצטרך מאגר תמונות המחולק לסט אימון וסט בדיקה על מנת לאמן את

המערכת שלי למצוא אובייקטים בתמונה.

1.1.2. כמערכת אצטרך שסט הבדיקה יגיע כזוג המורכב מאובייקט קלט והפלט הרצוי

עבורו על מנת ללמד את האלגוריתם.

2. המערכת תדע לזהות את מיקום האובייקטים בתמונה על מנת לשמור את וקטור המיקום שלו.

3. המערכת תשמור את שמות האובייקטים שזוהו יחד עם וקטור המיקום שלהם על מנת לשחזרם בסביבה התלת מימדית.

4. כמערכת ארצה לדעת להתמודד עם אובייקטים שאינם מוכרים על מנת לקבל סביבה נקייה.

5. כמערכת ארצה להחזיק מאגר נתונים על מנת לשלוף ממנו מידע.

5.1. כמערכת ארצה שבמאגר הנתונים יהיו אובייקטים תלת מימד על מנת שאוכל להתאימם לסביבה התלת מימדית.

6. המערכת תוכל להציג את הסביבה התלת מימדית שנוצרה על מנת שהמשתמש יוכל לראות לפני שמירה.

7. המערכת תוכל לייצא קובץ FBX על מנת שהמשתמש יכול להמשיך לעבוד איתו.

8. כמשתמש ארצה את היכולת להכניס תמונה כקלט למערכת על מנת לקבל סביבה תלת מימדית כפלט.

8.1. כמשתמש ארצה כפתור הורדת קובץ.

9. כמערכת ארצה לזהות את סביבת התמונה על מנת לדעת האם התאורה טבעית.

9.1. כמערכת ארצה לזהות את תאורת התמונה על מנת להבחין בין אור לחושך.

9.1.1. כמערכת ארצה להוסיף אובייקט תאורה במצב של אור על מנת להתאימה לתמונה

מוארת.

9.2. כמערכת ארצה לזהות את סביבת התמונה על מנת לדעת אם צולמה במקום סגור

9.2.1. כמערכת ארצה לזהות את גבולות הקירות הנמצאים בתמונה על מנת ליצור

פרספקטיבה.

9.2.2. כמערכת ארצה לדעת להוסיף תקרה בצמוד לגבול עליון על מנת ליצור מבנה סגור.

9.2.3. כמערכת ארצה לדעת להוסיף קרקע בצמוד לגבול תחתון על מנת ליצור בנה סגור.

9.2.4. כמערכת ארצה לדעת להוסיף קירות בצמוד לגבולות צדדים ב90 מעלות על מנת

ליצור מבנה סגור.

9.2.5. כמערכת ארצה לדעת את סוג הקרקע בתמונה על מנת להתאימה בסביבה התלת

מימדית.

9.2.6. כמערכת ארצה לדעת את סוג הקרקע בתמונה על מנת להתאימה בסביבה התלת

מימדית.

9.3. כמערכת ארצה לזהות את סביבת התמונה על מנת לדעת אם צולמה במקום פתוח.

9.3.1. כמערכת ארצה לזהות את אופק התמונה על מנת להתאים את אופק הסביבת התלת

מימדית.

9.3.2. כמערכת ארצה לזהות את סוג הקרקע בתמונה על מנת להתאימה בסביבה התלת

מימדית.

תהליך הבניה

שפת תכנות:

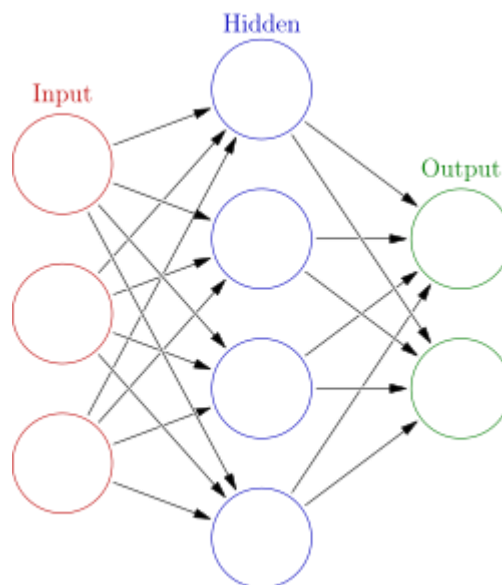
כדי לבחור שפת תכנות אנחנו צריכים להתחשב בכל הפונקציונליות שאנו רוצים בפרוייקט שלנו. כהתחלה רצינו שכל הפרוייקט יהיה כתוב בשפה אחת בכל היבטיו, אם זה ממשק המשתמש, מערכת הבינה המלאכותית והקוד ביצרית הגרפיקה. לכן השפה שמצאנו שמתאימה ביותר לדרישה זו היא שפת Python, ונסביר כעת מדוע. Python היא שפה נוחה מאוד למשתמש, שפה עם דגש על קריאות (Readability) ולכן היא מאוד נפוצה. Python היא שפת מפרש, מבצעת את הקוד שורה אחר שורה. במקרה של שגיאה כלשהי, המפרש מפסיק את הביצוע ומדווח על השגיאה שהתרחשה. המפרש מציג שגיאה אחת בלבד גם אם לתוכנית יש מספר שגיאות. דבר זה מקל מאוד על איתור הבאגים.

השפה היא שפה מבוססת ארכיטקטורת מונחה עצמים. כמעט כל דבר ב-Python הוא אובייקט, עם המאפיינים והפונקציות שלו. ומכאן אנחנו מגיעים למימושים הרבים של סוגים שונים של בינה מלאכותית בשפה שקיימים ברשת. במהלך החקר שלנו הומלץ לנו להשתמש ב-Python כאשר אנחנו רוצים לפתח מערכת שמשתמשת בבינה מלאכותית. בנוסף, במהלך הסקירה שעשינו באינטרנט, של מודלים מוכנים של בינה הנמצאים האינטרנט נתקלנו במימושים רק בשפה הזו (יש גם בשפות אחרות, אך לא נתקלנו). מצאנו המון מימושים קלים לתפעול של מודלים כמו רשתות נוירונים, עצי החלטות ועוד... חברות גדולות כמו גוגל מפתחות או לפחות מנגישות את מודלי הבינה שלהם בשפה הזו. וכן לאחר סיום הפרוייקט אנחנו יכולים להגיד שהשפה חסכה לנו זמן רב על פיתוח מתודות מורכבות. שכן אם היינו בוחרים שפה אחרת היינו גם כנראה לא מגיעים לתוצאות בזמן שהגענו אליהם. שימוש ב-Python מאפשר לקוד שנכתב בשפה לפעול בפלטפורמות שונות (Portability). מה שמועיל לפרוייקט שלנו שפותח גם על מערכת Windows וגם על מערכת Linux ונבדק על שניהם. לפי כל מה שהוצג כאן בחרנו לכתוב את כל הפרוייקט שלנו בשפת Python.

בניית מודל בינה מלאכותית שמזהה אובייקטים מתמונות:

השלב הראשון של המערכת שלנו הוא לדעת לזהות מה יש בתמונות הקלט. מסקירת הספרות עולה כי עלינו להשתמש במודל של רשת נוירונים כדי לזהות את האובייקטים מהתמונות. כדי שלאחר מכן נוכל לשחזר את הסביבה שצולמה בתלת-מימד.

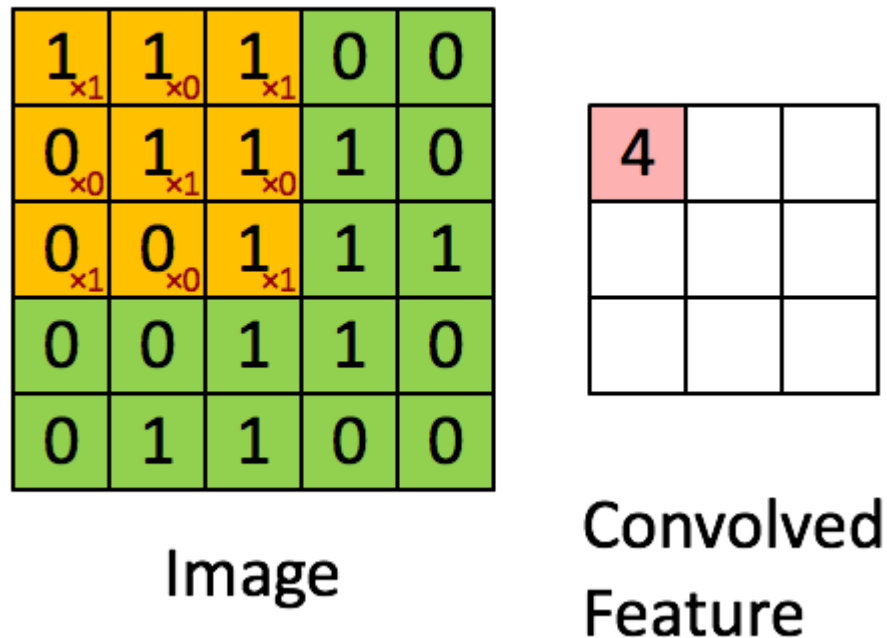
רשת נוירונים הוא מודל בינה מלאכותית שפותח בהשראת המוח האנושי, ולפי כך גם שמו. המוח האנושי מורכב ממיליארדי נוירונים שבעזרתם אנו בני האדם מתפקדים. לכל נוירון תפקיד אחד פשוט שבעזרת רשת גדולה מאוד שלהם אנו מצליחים לבצע פעולות מורכבות במוח שלנו. ברשת עצבית מלאכותית הדבר קורה בצורה זהה. אנו בונים נוירונים שלכל אחד תפקיד פשוט. כאשר אנו בונים רשת שלמה של הרבה נוירונים המחשב יכול לבצע פעולות מורכבות כמו זיהוי תבניות. במחשב, מבנה הרשת בנוי בצורה כזו שיש שכבת קלט שתפקידה לטעון את הקלט. שכבת הקלט מעבירה את המידע שנכנס אליה לשכבות הנסתרות, שתפקידם לקבל החלטות מהשכבות הקודמות להם שהם שכבות הפלט או שכבות נסתרות קודמות. תהליך זה נקרא תהליך הלמידה ובדרך כלל הוא תהליך ללא פיקוח והתערבות בן אדם. כאשר יש הרבה שכבות נסתרות תהליך זה נקרא "למידה עמוקה". ולבסוף, מגיע המידע לשכבת הפלט, שזה בעצם תוצאות המודל. בשכבה זו המודל קיבל החלטה כמו כמה אחוז המודל חושב שאובייקט מסויים נמצא בתמונה.



איור 2 - מבנה כללי של רשת נוירונים

קיימים מימושים שונים וארכיטקטורות שונות של מודלים של רשתות נוירונים. הארכיטקטורה המתאימה והפופולרית ביותר שיש כיום לעבוד עם נתוני תמונות היא ארכיטקטורת "רשת עצבית קונבולוציונית" (CNN). רשתות הללו הם כלי חזק בעיבוד תמונה, המשתמש בלמידה עמוקה לביצוע משימותיו. כלי זה גם

שמיש מאוד בעבודה עם סרטונים. ל-CNN יש שכבה קונבולוציונית אחד או יותר המשמשת לחילוץ מידע מתמונות הקלט. שכבה זו היא עיקר רשת ה-CNN. היא מכילה "פילטרים" שמחפשים תבניות בתמונות הקלט. כל פילטר רץ על התמונה ויוצר מפת פעולה, שבעזרתה ניתן לראות אילו אזורים בתמונה רלוונטים למחלקה (אובייקט) שאנו מחפשים.



איור 3 - הדגמת פילטרים של שכבות קונבולוציוניות

כדי שרשת הנוירונים תזהה אובייקטים צריכים להכניס לה כקלט סט תמונות שיהיה סט האימון של המודל. שעליו המודל ילמד וישפר את עצמו בזיהוי כל דבר שנרצה. אחד החסרונות ברשת נוירונים הוא כדי שהמודל יזהה אובייקט מסויים, צריך סט אימון גדול שיכול להגיע לאלפי תמונות.

כעת נפרט על הגישות השונות שחקרנו וניסינו במהלך בניית הפרויקט. כל השיטות שנציג משתמשות ברשת נוירונים כמודל שמסווג את התוצאות, אך שונות במה שהמודל צריך לזהות [10]. נציג את מהלך המחקר והניסיונות של הגישות השונות ולבסוף נציג את הגישה הנבחרת לפרויקט זה. חשוב לציין שאנחנו מחפשים את הגישה שתביא לנו את התוצאות ואחוזי הדיוק הטובים ביותר ובכך נוכל לשחזר את הסביבה שהתמונה צולמה בה באופן הטוב ביותר. לכל השיטות הללו הוספנו הדגמות ואיורים של השוואות ביניהם כדי להבהיר את השוני ואת הדימיון ביניהם.

:Image classification

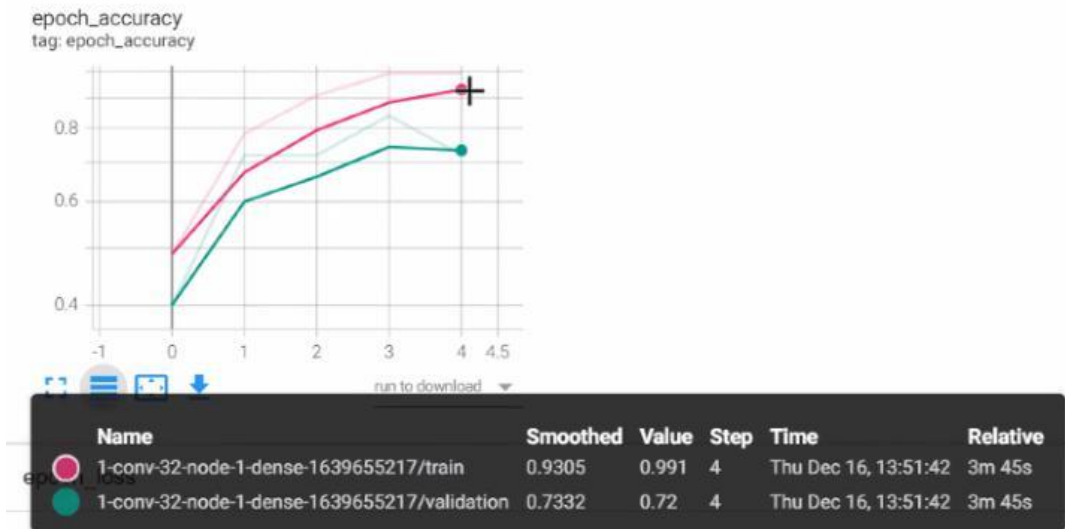
אחד מהדרכים שניסינו כלמידת מכונה, הייתה "קלסיפיקציה" של התמונה כולה לסביבה שנגדיר מראש. דרך זאת נבעה מהצורך שלנו בלהתחיל לבנות את הסביבה הרצויה בסביבה התלת-מימדית. ואכן הייתה הדרך הראשונה שניסינו עם מודל רשת נוירונים. רצינו להגיע למצב שיש לנו בסיס כלשהו לבניית הסביבה. והחלטנו לזהות מהתמונות האם הם בסביבת דשא או בסביבה חולית. בשביל כך ייבאנו מודל רשת נוירונים מ-TensorFlow (על TensorFlow נדבר בהרחבה בהמשך) שממייין תמונות בין שני סוגי הסביבות. אימנו את המודל כמה פעמים עם שינויים בין אימון לאימון בערכי שכבות הרשת כדי לנסות וליעל את תוצאות הזהוי. להלן קטע קוד שבו מוצג ערכי השכבות שאותם שינינו כדי להפיק את התוצאות הטובות ביותר מבחינת סיווג התמונות לסביבות שהגדרנו.

```
dense_layers=[1,2]
layer_size=[16,32,64]
conv_layers=[1,2]

for dense in dense_layers:
    for layer in layer_size:
        for conv in conv_layers:
            NAME = '{}-conv-{}-node-{}-dense-{}'.format(conv,layer,dense,int(time.time()))
            tensorboard = TensorBoard(log_dir='log/{}'.format(NAME))
```

איור 4 - קטע קוד מנסיון הרצה ראשוני

קטע הקוד שהוצג מראה את הערכים שאותם היינו צריכים לשנות כדי להגדיל את אחוזי הדיוק. נאמר גם שערכים הללו הם כמו הגרלה בשבילנו כי המודל שניסינו איתו היה "קופסא שחורה" בשבילנו. ולא באמת ידענו מה כל ערך באמת משנה. כעת נציג גם את אחוזי ההצלחה של המודל הזה כאשר הצלחנו להגיע לתוצאה הטובה ביותר.



איור 5 - גרפים המתארים את דיוק המודל

הגרף שהוצג מראה כי תוצאות המסווג צודקות ב-93% מהמקרים על הבדיקות שנעשו על סט האימון. אבל בסט האימות המסווג צודק רק ב-73% מהמקרים. במילים אחרות אם ניתן למסווג לאחר האימון שביצענו תמונת נוף כלשהי. המסווג ידע לסווג את התמונה לסביבה עם דשא או סביבה עם חול. והוא יעשה את זה כאשר אחוזי הדיוק שלו הם 73%. זאת אומרת שהסיכוי שלו לסווג נכון הוא 73%, והסיכוי לטעות הוא 27%.

ובכן תהליך זה היה מאין התחלה לפרוייקט שלנו, אבל סיווג זה אינו משרת אותנו טוב. מכיוון שלא מספיק לנו לסווג לאיזה סביבה התמונה משתייכת ולכן המשכנו בגישות אחרות.

Image segmentation:

שיטה הנקראת פילוח תמונה (לפעמים נתקל בשמה גם כ-Instance Segmentation). תחילה נאמר שכאשר רשת נוירונים מופעלת על סט נתונים של תמונות המודל תמיד עושה פילוח ("סגמנטציה"). וגם כן כל מודל עושה סיווג ("קלסיפיקציה"). אבל נסביר מה ההבדל של שיטה זו מהשיטות האחרות שאותן ניסינו. שיטה זו מחלקת את כל התמונה לקבוצות של פיקסלים, שכל פיקסל מייצג חלק מאובייקט. במילים אחרות, לכל אובייקט שנמצא בתמונה, נדע איזה פקסלים שייכים אליו.



איור 6 - המחשת פילוח תמונה

שיטה זו נראתה לנו בשלב מסויים כדרך הטובה ביותר לפתרון. וזאת כי אם נדע לגבי כל פיקסל בתמונה לאיזה אובייקט הוא שייך ואיפה הוא נמצא. נדע למקם בצורה הטובה ביותר את האובייקטים בסביבה התלת-מימדית שתבנה. אבל לאחר חקר מעמיק יותר, להשתמש בשיטה זאת יצור לנו עבודה נוספת לאחר הסיווג של התמונה. אם זה לפרק את קבוצות הפיקסלים שנוצרו למיקומי אובייקטים. למשל אם נתייחס לתמונה שהוצגה למעלה. לדעת שכל מה שמסומן בצורה זאת עצים מקרב אותי לפתרון אבל לא מספיק. אנו נצטרך איכשהו לדעת איפה כל עץ נמצא. או לדעת איזה עץ נמצא קרוב יותר מהעצים שנתפסו כולם כמקבץ פיקסלים אחד גדול. בנוסף לא מצאנו מודל שיועד לזהות מספיק אובייקטים מהסוג שאנחנו רוצים לזהות שמשמש בשיטה זו כשיטת הסיווג שלו. ולכן המשכנו לשיטה האחרונה שאותה חקרנו ואותה גם מימשנו.

:Object detection

זיהוי אובייקטים היא בן השיטות הראשונות שפגשנו שהתחלנו לחקור על מודל רשת נוירונים שמוזהה אובייקטים. שיטה זאת עובדת בצורה כזו שהיא מקבלת תמונת כקלט, ומזהה את האובייקטים המוכרים לה בתמונה. שיטה זו גם מסמנת את מיקום האובייקט בתמונה, זאת אומרת שיהיה אפשר לדעת את מיקום האובייקטים הרצויים מהתמונות. שיטה זו יודעת גם להגיד בכמה היא בטוחה שהאובייקט שהיא זיהתה

הוא באמת אותו אובייקט. המטרה העיקרית שלשמה פותחה שיטה זו היא כדי שמחשב יוכל לדעת איזה אובייקטים הוא רואה ואיפה [11].



איור 7 - המחשת מודל זיהוי אובייקטים

באיור הבא נציג את ההבדלים בין השיטה השניה שהצגנו שהיא Image segmentation לבין השיטה שלבסוף בחרנו שהיא Object detection. הבדלים הלו נראים כהבדלים בפלט המודלים השונים. כמובן שיש גם שוני במודלי עצמם אבל יש גם דימיון רב. כמו שהסברנו, כל המודלים משתמשים בחלקים אחד של השני. למשל כל מודל מבצע פילוח של תמונה. אם זה מודל שעובד בשיטה של סיווג, זיהוי או חלוקת התמונה למקטעים.

Object Detection

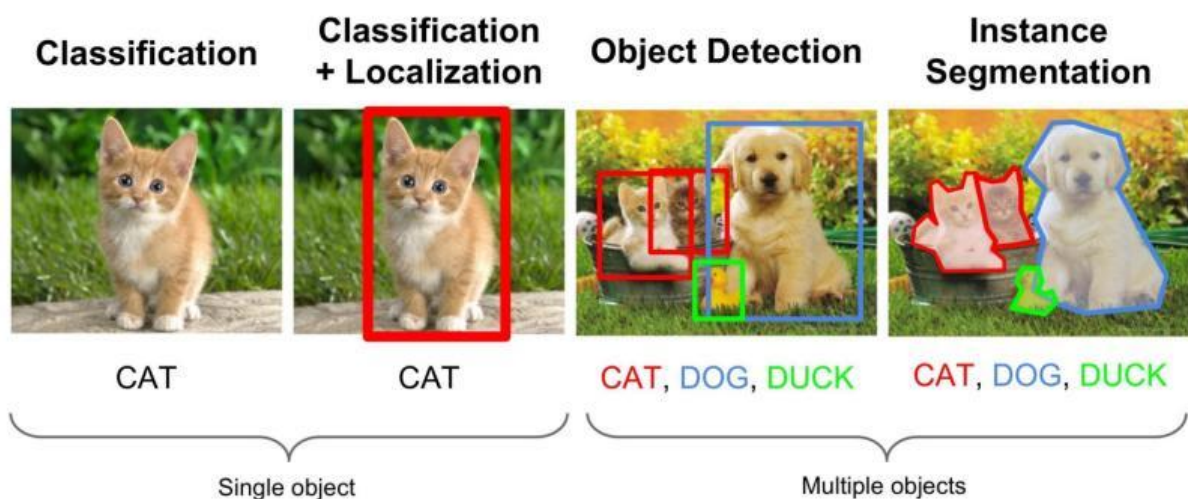


Instance Segmentation



איור 8 - השוואה בין מודל פילוח תמונה למודל זיהוי אובייקטים

נציג איור נוסף שמראה באופן טוב את ההבדל בין השיטות שהוצגו בפתרון לפרוייקט זה. כמו שכבר נאמר קודם Instance segmentation מתייחס ל-Image segmentation.



איור 9 - השוואה בין כלל השיטות שנידונו עד כה

בשלב זה נרחיב על החקר שעשינו על השיטה שנבחרה לפרוייקט שלנו שהיא שיטת Object detection. ונראה את השלבים שעברנו עד שהגענו למודל שמשמש היום את הפרוייקט שלנו בצורה הטובה ביותר. שיטה זו נבחרה מכיוון שכמו שנאמר, לדעת איזה אובייקט נראה בתמונה ואיפה זאת מטרתה. בנוסף השיטות האחרות לא היו מספיק טובות לתוצאה למרות שבהתחלה נראו מאוד מבטיחות.

את המודל שלנו רצינו לאמן לזהות אובייקטים מסויימים מתמונות. הרי אנו רוצים לזהות ולשחזר תמונות נוף, תמונות המכילות אובייקטים מסויימים. אנחנו לא מחפשים לזהות כל אובייקט קיים אלא רק את השכיחים ביותר לפרוייקט שלנו. לכן תחילה כהצבת מטרה התחלתית, דרשנו שהמודל שלנו יזהה אובייקטים ספציפים שחשבנו עליהם בעצמנו והם: הר, גבעה, אבן, מים, עץ ירוק, עץ צהוב, קקטוס, דשא, עשבים, שיח, חול, שביל, בית, גדר.

לאחר שהגדרנו לעצמנו כמה סוגי אובייקטים שמבחינתנו אנחנו רוצים לזהות. נשאר לנו למצוא מודל שיודע לזהות את אותם האובייקטים כדי שנוכל להמשיך לשלבים הבאים בבניית המערכת. אבל בשלב זה גם כן נתקלנו המכשולים רבים מכיוון שאין מודל שמזהה את אותם האובייקטים. רוב המודלים הקיימים יודעים לזהות דברים יותר שכיחים בחיינו. וככל הנראה מי שפיתח אותם פיתח למטרה מסויימת. נציג בהמשך כמה מהמודלים שחקרנו במסגרת הפרוייקט. אבל תחילה נציג חלק מתהליך פיתוח מודל שנקרא Labeling, תהליך חובה שקורה על ידי בני אדם כדי שמחשב ידע לזהות אובייקטים בעצמו. חקרנו תהליך זה מכיוון שכמו שאמרנו, לא מצאנו שום מודל שיודע לזהות את כל אותם האובייקטים שאנו רוצים לזהות. והגענו להחלטה שכנראה והמצב הוא שאנחנו צריכים לתייג (Labeling) תמונות בעצמנו. מה שבעצם קורה בתהליך התייג הוא שעל מנת שמחשב ידע לזהות אובייקט. אנחנו צריכים לקחת תמונה שבה האובייקט מופיע ולסמן למחשב איפה נמצא האובייקט הספציפי הזה. בנוסף לסימון הזה צריך לתת שם (תייג). למשל אם יש לי תמונה שבה יש עץ, אני אסמן את העץ בצורה ידנית וירשום על סימון זה "עץ". תהליך זה צריך לקרות המון פעמים כדי שמודל ידע לזהות אובייקט מסויים. זאת אומרת שצריך לתייג אלפי תמונות כדי שמודל ידע לסווג אובייקט יחיד.

כדי שנוכל להתחיל בתהליך התייג עלינו למצוא סט נתונים שמכיל הרבה מאוד תמונות נוף. וזאת כי אלו התמונות שאנחנו מצפים מהמודל שלנו לזהות. מצאנו סט תמונות גדול של אלפי תמונות נוף באתר בשם [Kaggle](https://www.kaggle.com/datasets). אתר המתעסק ב"מדע הנתונים" ומכיל אלפי סטים של נתונים שונים. לאחר מכן היינו צריכים למצוא תוכנה שתאפשר לנו לתייג את התמונות שהשגנו. מצאנו תוכנה בשם [LabelMe](https://labelme.ai/), תוכנה עם ממשק ידידותי למשתמש שמאפשרת לסמן פוליגונים על תמונות ולתת להם תייג. כדי להתחיל באופן התייג חילקנו את סט התמונות שלנו לסט אימון שאותו התכוונו לתייג, ואז לאמן בעזרתו את המודל שלנו. וסט בדיקה שבעזרתו רצינו לבדוק את אחוזי ההצלחה של המודל.



איור 10 - דוגמא לתהליך תיוג

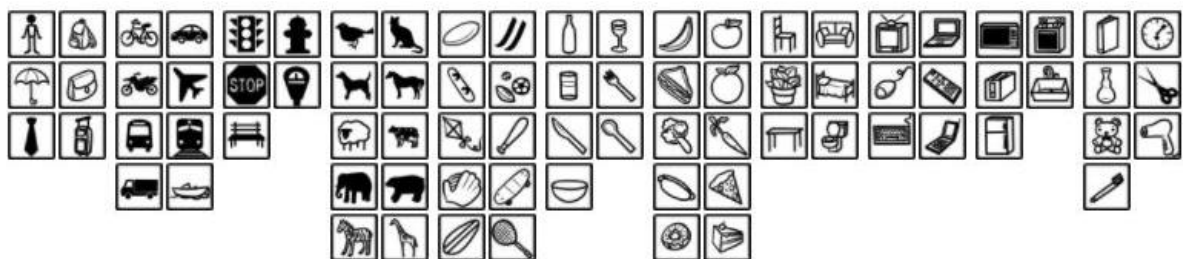
את התיוג בתמונה למעלה אנחנו עשינו. כמו שניתן לראות אפשר לחלץ מהתמונה עוד הרבה מידע. אבל כדי לבנות רשת נוירונים אנחנו צריכים אלפי תמונות מכל אובייקט שנרצה לזהות. ולנו לוקח 4 דקות בממוצע לתמונה, שבדרך כלל לא מכילה את כל אותם אובייקטים שנרצה לזהות. בכל מקרה מדובר בתהליך שיקח במקרה הטוב ביותר 700 שעות לשני אנשים.

ולכן מכיוון שהתהליך הזה ארוך מדי, אין ביכולתנו לתייג מספר רב כזה של תמונות. עלינו למצוא אופציה חלופית שתביא אותנו לתוצאות טובות. אחת מהאופציות שעלתה במהלך החקירה לפתרון שלנו היא לקחת תהליך אימון של מכונה קיימת ולשלב אותו במכונה שאנחנו נבנה. מה שבאמת קורה זה שאנחנו לוקחים מודל רשת נוירונים מוכן וממומש ומשנים כמה מהשכבות שלו כדי שיתאים למודל שלנו. המודל שממנו ניקח את תהליך האימון צריך להיות מודל דומה כמה שיותר למודל שאנחנו רוצים לבנות כדי שאופציה זו תעבוד. אופציה זו אמורה לחסוך לנו הרבה זמן, מכיוון שלא נצטרך לתייג אלפי תמונות אלא פחות. ואת התמונות הללו נכניס למודל המוכן ונאמן אותו שוב עם התמונות שלנו בנוסף לנתונים הקודמים של המודל עצמו. לבסוף את אופציה זאת לא ניסינו מכיוון שלא היה בידינו מספיק מידע איך תהליך זה קורה. וגם אם נצליח לא ידענו כמה באמת תיוגים תהליך זה יחסוך לנו. החלטנו ללכת על אופציה בטוחה יותר שהיא להשתמש במודל מוכן לגמרי עם יכולת זיהוי אובייקטים טובה יחסית לאובייקטים שאותם נרצה לזהות.

בשלב זה נציג שניים מהמודלים שאותם ניסינו כמודל Object detection. המטרה הייתה למצוא את המודל שיודע לזהות כמה שיותר אובייקטים שיכולים להופיע בתמונות נוף רנדומליות. נאמר שקיימים המון מודלים רשתות נוירונים שתפקידם הוא לבצע זיהוי אובייקטים. ולכן סקרנו המון מהם כדי למצוא את

המודל הטוב ביותר. כל המודלים שיוצגו שייכים לספריית [TensorFlow](#) שהיא ספריית קוד פתוח ללמידת מכונה של חברת Google. ספרייה זאת מאוד נוחה, עם הרבה מאוד מסמכים המפרטים את החלקים שבונים אותה והאפשרויות שהיא מציעה. בנוסף יש המון מדריכים באינטרנט שמדגימים שימוש בכל מני מודלים שהחברה מציעה. קיימים גם מדריכים מצולמים רבים ב-Youtube של אנשי החברה עצמה אשר מסבירים את אופן השימושים השונים. וכמובן שקיימים עוד יותר סרטונים של אנשים שלא עובדים בגוגל אשר מדגימים שימוש ותוצאות בספרייה הזאת. ספריית TensorFlow מאוד רחבה במה שיש לה להציע לאנשים שרוצים ללמוד על בינה מלאכותית ובעיקר על רשתות נוירונים. קיים גם שרת אינטרנטי של גוגל שעליו אפשר להריץ את המודלים השונים של הספרייה שאותו ניתן להפעיל מכל מכשיר. זאת אומרת שבכל מקום ניתן לגשת אל המידע של הספרייה ולהריץ אותו. שרת זה נקרא [Colab](#), ועליו ניסינו את רוב המודלים שהפעלנו מהסיפרייה הזאת.

אחד מהמודלים שבדקנו נקרא [YoloV3](#). מודל רשת עצבית קונבולוציונית שיועד לזהות אובייקטים ספציפיים בזמן אמת מסרטונים ותמונות. מודל זה הוכן בשנת 2018 אבל הגרסאות הראשונות שלו נכתבו בשנת 2016. הוא מאומן על סט הנתונים [Coco](#) שמכיל 80 סוגי אובייקטים. ומאומן על יותר מ-200 אלף תמונות מתוויגות ב-1.5 מליון מופעים של האובייקטים שהוא מכיר.



איור 11 - האובייקטים המוכרים למודל YOLOV3

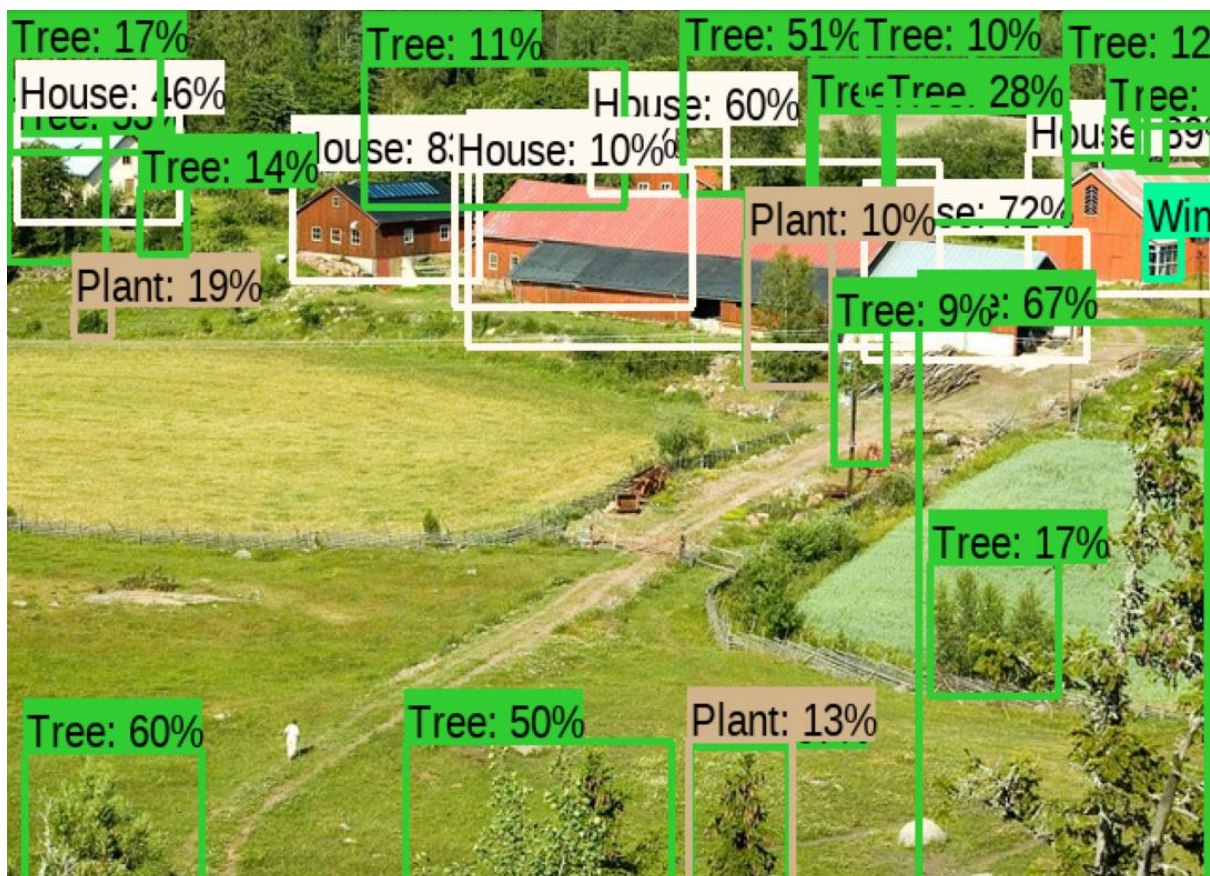
מודל זה לא מכיל את האובייקטים שאותם רצינו לזהות ולכן היינו צריכים לחפש מודל אחר ויותר רלוונטי לפרויקט שלנו.

המודל שבו השתמשנו בסופו של דבר הוא [Resnet V2](#). המאומן על סט הנתונים של [OpenImageV4](#). סט המכיל [600 סוגים](#) של אובייקטים, המאומן על מעל 1.7 מליון תמונות. מודל זה לא עונה על כל הדרישה ההתחלתית של זיהוי האובייקטים שלנו. אבל הוא המודל שעושה את העבודה הכי טובה שמצאנו עד כה, ולכן המשכנו איתו לשלבים הבאים. מודל זה מצליח לזהות לא מעט אובייקטים שכיחים מסביבה חיצונית (תמונות נוף). ובעזרתו הצלחנו להגיע לתוצאות יפות של שחזורי סביבה שמהם התמונה נלקחה. קיים לו מדריך מפורט באתר של TensorFlow עם אפשרות הרצה מהירה בשרת.

בתמונות הבאות שנציג נראה תמונות נוף רנדומליות שלקחנו מהאינטרנט ולאחר כל תמונה נראה איך המודל שבחרנו מסווג את האובייקטים בה. ההרצות הללו הם הרצות שאנחנו עשינו.



איור 12 - לפני ואחרי הרצת המודל (דוגמה 1)



איור 13 - לפני ואחרי הרצת המודל (דוגמה 2)

כמו שניתן לראות מהדוגמאות הקודמות, המודל יודע לסווג אובייקטים עיקריים בסביבות חיצוניים. למשל עצים, זהו אובייקט מאוד חשוב לפרוייקט שלנו. וזה כי כמעט כל סביבה חיצונית שאנחנו מכירים יש בה עצים. אם נתייחס לבתים, זה לא משהו שרצינו בהתחלה בתוך המודל שלנו. אבל גם בתים יכולים להופיע בתמונות נוף, לכן זה שיפור למודל שלנו. כמובן שיש עוד הרבה אובייקטים שהיינו רוצים שהמודל יזהה מהתמונות נוף. כמו דשא, סלעים, הרים ועוד. אבל אנחנו יודעים להגיד בוודאות שלא קיים מודל שיוודע לזהות את כל האובייקטים הקיימים. ולכן זה בא לידי ביטוי בתמונות שנראו למעלה. המודל יודע לזהות אך ורק דברים שלימדו אותו לזהות. לכן כעת ניתן לומר שיש לנו מודל מספיק טוב שאיתו נתחיל לבנות את הסביבה התלת מימדית. אנחנו נצטרך לחלץ את המידע שהמודל נותן לנו על התמונות שנכנסות אליו כקלט. ועם המידע הזה נשתמש כדי לבנות את הסביבה התלת-מימדית הדומה ביותר לסביבה שצולמה.

מיקום האובייקטים בסביבת הפלט:

השאלה איך נמקם את האובייקטים בסביבה שנבנה העסיקה אותנו מאוד. לא מצאנו שום מודל בינה מלאכותית שעושה הערכת מרחקים מתמונות חיצוניות. ואף על פי שאנחנו משתמשים בתמונה אחד בלבד, דבר זה הופך לקשה אפילו יותר. אבל קיימים מודלים שמעריכים מרחקים בסביבות פנימיות (בתוך מבנה). בדרך כלל המודלים האלה דורשים חיישן שיודע להוציא "מפת עומק" או שהמודל דורש כמה תמונות כדי לדעת להעריך את המרחקים. יש מקרים שבהם צריך גם מספר תמונות וגם חיישן כדי להעריך את מפת העומק. את המודלים הללו סקרנו בשלב סקירת הספרות של פרוייקט זה.

כבר בתחילת הפרוייקט החלטנו שאנחנו לא צריכים להגיע להערכת עומקים בתמונה מדוייקת. מספיק שנדע להעריך בקירוב מספיק נכון שהסביבה המשוחזרת תהיה דומה לתמונה שממנה היא צולמה. ואם נחשוב על זה, גם אנחנו כבני אדם לא נדע להעריך מרחקים מדוייקים בתמונות. לכן שחשבנו על פתרון לבעיה הזו רצינו שהמערכת שלנו תדע לפחות להעריך איזה אובייקט נמצא לפני איזה אובייקט, ולתת הערכה של מרחקים אבל לא חייב לדייק. כמו שאנחנו בני האדם היינו משחזרים תמונה, לא מדוייק אלא בקירוב מספיק טוב. לכן הפתרון שהגענו אליו הוא פתרון אריתמטי בלבד, ללא כל צורך בבינה מלאכותית.

כדי למקם את האובייקטים שזיהינו מהתמונה בסביבה אנחנו צריכים רק להבין איזה אובייקטים נמצא לפני אובייקט אחר. במילים אחרות עם נעמוד במיקום שממנו התמונה נלקחה. איזה אובייקט יהיה קרוב אלינו יותר. השיטה שלנו היא סריקת התמונה מלמטה למעלה. זאת כי אובייקט שממוקם בתמונה נמוך יותר, מבחינת מיקום פיקסלים בתמונה, יהיה קרוב יותר. כמובן שיש חריגים, במקרה של אדמה לא ישרה או זוויות צילום נמוכות. אבל כמו שכבר הזכרנו אנחנו מחפשים להגיע לקירוב הכי טוב. נדגים ונסביר את הכוונה שלנו שמה שיותר נמוך בתמונה נמצא יותר קרוב אלינו בעזרת תמונה. נסתכל על התמונה הבאה:



אפשר להבין מהתמונה הזו שמה שנמצא נמוך יותר מבחינת פיקסלים בתמונה הוא מה שקרוב יותר. אם נתחיל לסרוק את התמונה מלטה כלפי מעלה, שורת פיקסלית אחת אחרי השניה. נראה שהעץ המרכזי בתמונה שממוקם במרכז התמונה עם נטיה לצד ימין. הוא העץ שהכי קרוב אל הצלם שצילם את התמונה. נחדד ונאמר שאם נסרוק מלמטה, הנקודה שבה אנחנו פוגשים את התחלת הגזע של העץ היא נקודת התחלת האובייקט. וכמו כן אם נשים לב שליד העץ שדיברנו עליו עכשיו, נמצא מעין שטיל עץ קטן, שבערך הוא והעץ המרכזי נמצאים על אותו שורת פיקסלים. לכן בסיסה שלנו הם יהיו באותו מרחק, ואכן גם אם ישאלו אותנו נגיד שאנחנו רואים מהתמונה ששניהם נמצאים בערך באותו מרחק מצלם התמונה. העץ שממוקם בצד שמאל של התמונה הוא רחוק יותר מהעץ המרכזי. אנחנו יכולים להבין את זה בקלות, ואם נחשוב על זה. העץ הזה נמצא על כמה שורות פיקסלים מעל העץ הראשי שבתמונה. ולכן העץ הזה יהיה מסודר רחוק יותר בסביבה התלת-מימדית. כמו שכבר אמרנו יכולים להיות דברים שהדרך שבה אנחנו משתמשים אינה תעבוד אבל זה עובד ברוב המקרים ולכן דרך זאת היא פתרון לבעיה של הערכת מרחקים מהתמונות שאותם נשחזר.

יצירת הסביבה התלת מימדית:

את הסביבה התלת-מימדית אנו רוצים לבנות במנוע משחק. כדי שמפתחים אחרים יוכלו לקחת את התוצר של התוכנה ולהכניס אותו ישירות למשחקים. לכן חקרנו את מנועי המשחק הקיימים היום. חשוב לציין כאבן דרך כי עלינו להבין במנוע המשחק שבו נשתמש מכיוון שאנחנו צריכים לבנות אוטומציה מלאה הבונה סביבה מהנתונים שרשת הנוירוניים תזוהה. בשלב זה נפרט על מנועי המשחק שסקרנו במסגרת פרוייקט זה. ולבסוף נציג את התוכנה שאיתה בנינו את המערכת שלנו.

תחילה נסביר מהו מנוע משחק בכלל ומה המשמעות שלו. מנוע משחק הוא תוכנה שבעזרתה בונים משחקים. לרוב במנוע משחק יסופק כמעט כל הכלים שבעזרתם אדם כלשהו יוכל לפתח משחק שלם. לא צריך להיות מפתח תוכנות כדי לפתח משחקים בעזרת מנוע משחק אבל לרוב זה עוזר. מנוע משחק יכול לספק את הגרפיקה של המשחק שיבנה. את הפעולות שהמפתח רוצה שהמשחק יכיל. יש במנוע המון שבלונות של משחקים שאותם אפשר לשנות או להרחיב כרצוננו. בגדול הרעיון הוא לבנות תוכנה שתתן מהין בסיס התחלתי בפיתוח משחקים.

:Unreal Engine

כהתחלה הלכנו על מנוע המשחק המפורסם [Unreal Engine](#). מנוע משחק זה הוא אחד מהפופולרים בתחום המשחקים. יש לו אלפי מדריכים באינטרנט שמסבירים איך בונים משחקים איתו בצורה מלאה. המנוע יצא לראשונה בשנת 1998 כחלק ממשחק שפיתחו ששמו היה Unreal. מנוע משחק זה נכתב בשפת ++C והוא ניתן להורדה חינם באינטרנט. כדי ללמוד את הפונקציונליות של מנוע זה צפינו בסרטונים המסבירים על הפעולות הבסיסיות של התוכנה.

אבל לאחר התייעצות עם המנחה שלנו, הגענו להחלטה שכדי לנו להשתמש בשפת תכנות אחת שהיא Python בכל הפרוייקט. לכן עברנו למנוע משחק אחר בשם [Unity](#), שגם כן הוא מאוד מפורסם. ההחלטה לעבור למנוע המשחק Unity הייתה על פי סקירה של כתבות לגבי מנוע משחקים [12]. ראינו שכאשר שפת התכנות של המתכנת היא Python אז Unity הוא מבין המנועי משחק המומלצים. שני מנועים אלה הם המובילים בתחום והם המתחרים העיקריים אחד של השני. בנוסף שני מנועים אלה הם מנועים עם גרפיקה תלת-מימדית. מה שהרבה מאוד מנועים פשוטים בתחום אינם יודעים לעשות.

:Unity

את מנוע המשחק Unity חקרנו רבות מכיוון שהיינו בטוחים שהוא הבחירה הטובה ביותר לפרוייקט שלנו. Unity נכתב גם הוא בשפת ++C בדומה למנוע המשחק Unreal engine. הוא יצא לקהל הרחב בשנת 2005, ועשה מהפכה בשוק מנועי המשחקים מכיוון שהוא היה הראשון שסופק חינם למפתחי המשחקים. הדגש הראשוני והעיקרי ששמו עליו מפתחי המנוע הם ממשק המשתמש. עד לפני Unity היה מאוד מסובך לעבוד עם מנועי המשחקים שבשוק.

כמו שכבר נאמר, את מנוע משחק זה ניסינו רבות. בנינו איתו משחק קטן כחלק מתהליך הלמידה שלו בעזרת מדריכים באתר [Udemy](https://www.udemy.com/). אבל באמצע תהליך הלמידה ועוד לפני שהספקנו לעשות עם התוכנה הזאת משהו שקשור לפרוייקט שלנו. נתקלנו בתוכנה הרבה יותר נוחה למערכת שאנחנו רוצים לבנות. למרות שהתוכנה הבאה שאותה נציג היא אינה מנוע משחק. יש לה אפשרות להציג את כל מה שנבנה בתוכה כקבצים של Unity או Unreal engine. במילים אחרות, התוכנה הבאה מאפשר לנו לבנות את הסביבה שלנו בתוכה וממשק כל מה שרק נרצה מתוכה אל מנועי המשחק שהצגנו כאן.

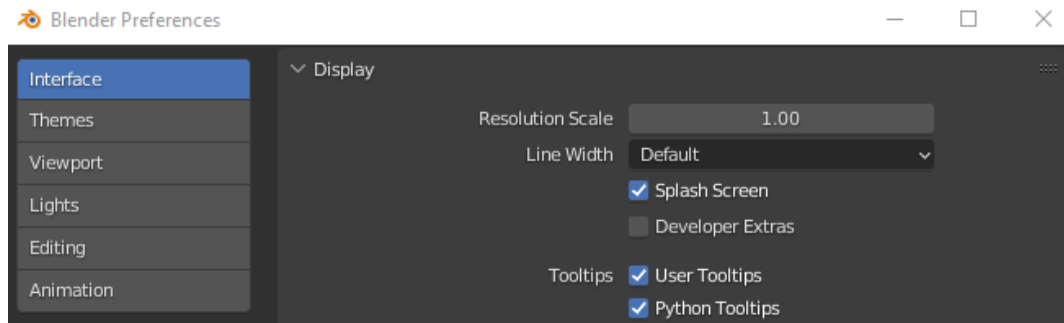
:Blender

[Blender](https://www.blender.org/) היא תוכנת קוד פתוח לגרפיקה ממוחשבת הכתובה בשפת Python. יש לה ממשק עם קוד של Python מאוד נוח וקל לתפעול. התוכנה נועדה לשם יצרית יישומי תלת-ממד אינטראקטיביים, אמנות, דגמים מודפסים בתלת-ממד, גרפיקה בתנועה, מציאות מדומה ועוד..

ל-Blender יש ממשק קל גם עם Unity ו-Unreal engine כמו שכבר נאמר. אפשר להכניס את הקבצים שנוצרו מ-Blender למנועי המשחק שהוצגו ולהשתמש בהם בפיתוח משחקים. אפשרות זאת מאפשרת לנו לקחת את הסביבה שפיתחנו בתוכנה ולהמשיך לפתח עליה משחק של ממש.

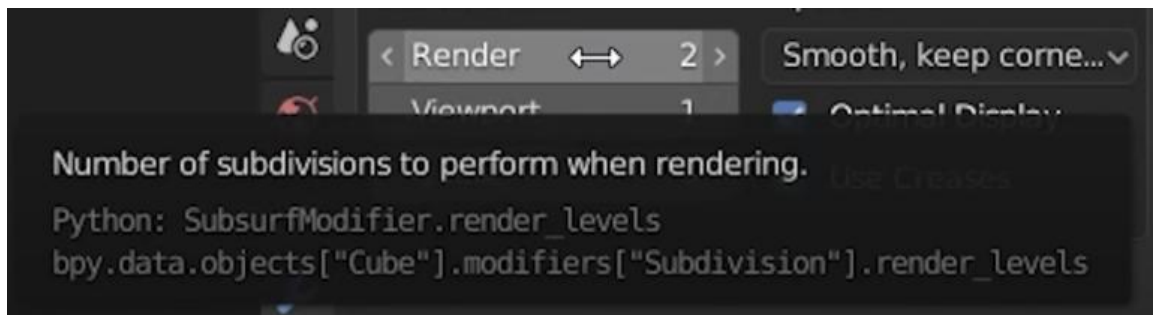
עקב כל מה שכתבנו על התוכנה ומה שנראה בדפים הבאים, אנחנו יכולים להגיד שהתוכנה Blender היא התוכנה הטובה ביותר לבסס את המערכת שלנו עליה. אם זה בגלל שהיא קוד פתוח וכתובה בפיתון מה שמאפשר לנו לפתור בעיות או להרחיב כל פיתוח כלשהו על התוכנה. אם זה כי מהות התוכנה הוא לפתח גרפיקות תלת-מימדיות. מה שבעצם גם כן מטרת המערכת שלנו, לשחזר את הסביבה בתלת-מימד. אבל מה שהכי עניין אותנו זה הממשק של התוכנה עם Python. ממשק משתמש נוח מאוד שאותו נציג בדפים הבאים. התמונות הבאות נלקחו מתוך התוכנה ללא קשר למערכת שאותה בנינו. במטרה להדגים מדוע השתמשנו בתוכנה.

כאשר מאפשרים בתוכנה צפיה בקוד ה-Python של האפשרויות השונות בה (Python Tooltips).



איור 14 - מדריך להגדרות ששנו

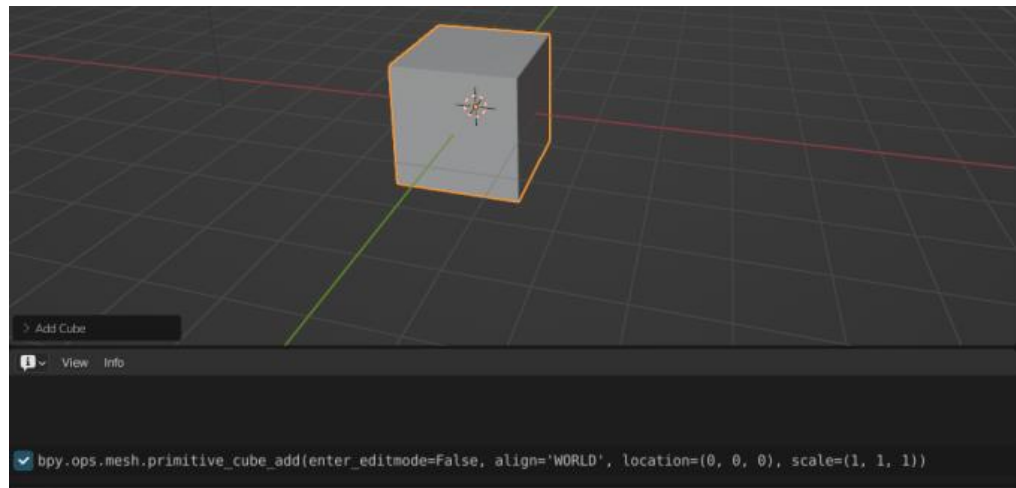
ניתן לראות לכל אפשרות שהיא בתוכנה איך להגיע אליה מתוך הקוד.



איור 15 - תוצאת תהליך השינוי בתוכנה

התמונה שהוצגה מראה אפשרות רנדומלית בתוכנה. כאשר אנחנו מצביעים עם העכבר על מאפיין של אובייקט כלשהו שנקרא "Render". אפשר לראות בלבן את התיאור של מאפיין זה, בעצם מה המשמעות שלו. ובאפור ניתן לראות איך לגשת אליו בקוד Python. השתמשנו בכך כאשר רשמנו את הקוד שלנו כאשר רצינו לסדר את האובייקטים בסביבה התלת-מימדית.

עוד תכונה של התוכנה זה חלון מידע שמאפשר לנו לראות את כל מה שעשינו גם כן כקוד Python.



איור 16 - חלון המידע בתוכנה

בדוגמא שהראנו למעלה, הוספנו קובייה לסביבה. בחלון הקטן למלטה נותן לראות שורת קוד. שורת הקוד הזאת היא השורה שנצטרך לרשום כדי להוסיף את אותה הקובייה בדיוק לאותו המקום עם נרצה לעשות את זה בצורה אוטומטית.

ובנוסף לכל אלה התוכנה מאפשרת לנו להריץ סריפטים של Python בתוכה. זאת אומרת שיכלנו להתנסות בכתיבת קוד עם התוכנה ללא צורך בסביבה חיצונית של מפתחים, הכל נעשה בתוך התוכנה.

```
View Text Edit Select Format Templates operator_simple.py
1 import bpy
2 from math import radians
3
4 class MyOperator(bpy.types.Operator):
5     bl_idname = "object.my_operator"
6     bl_label = "My Operator"
7
8     def execute(self, context):
9
10        # Create cube and store it in selectedObject
11        bpy.ops.mesh.primitive_cube_add()
12        selectedObject = bpy.context.active_object # Hold the object
13        # Blender set any object that just added as the selected obje
14
```

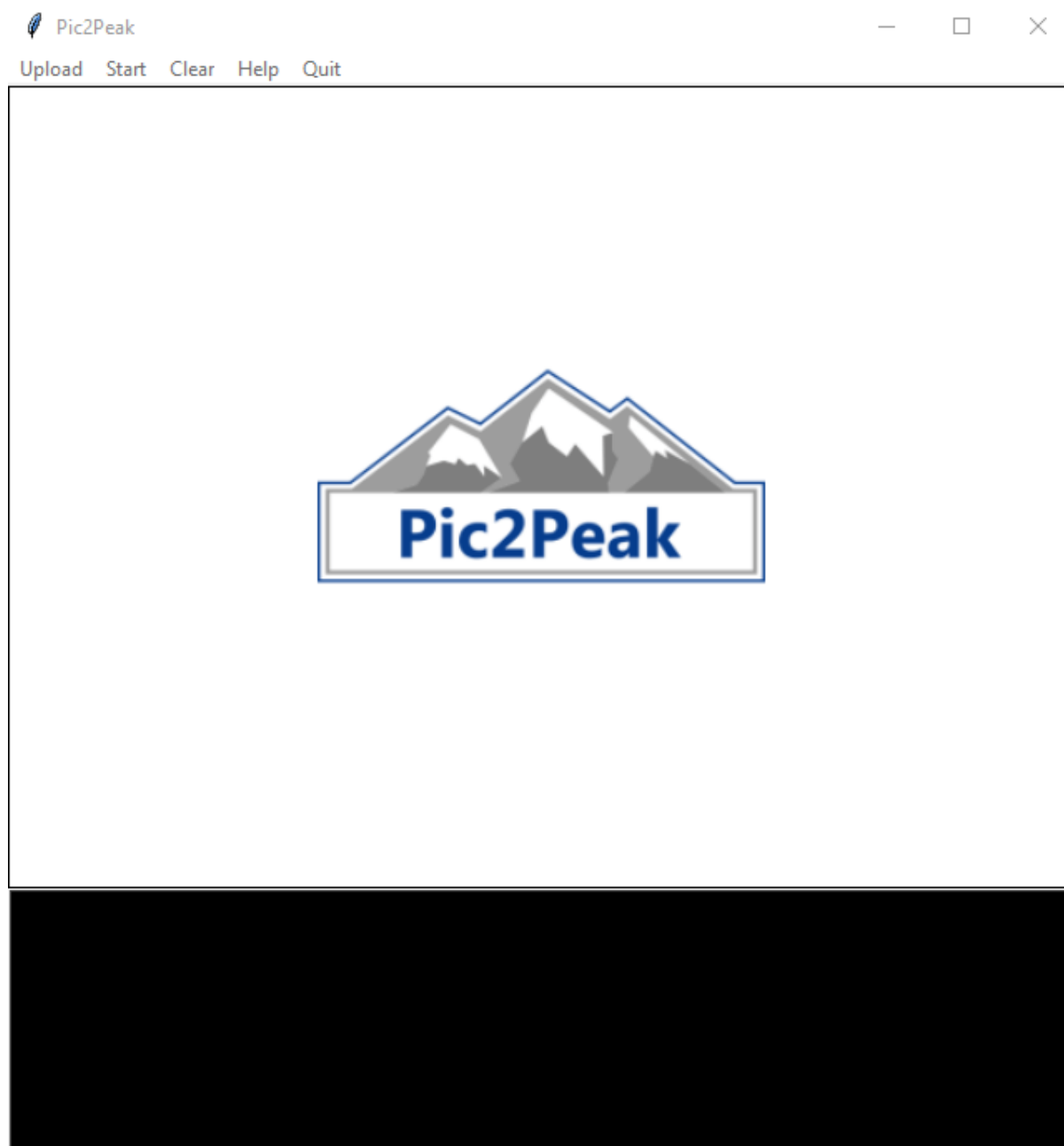
איור 17 - הדגמה לכתיבת קוד בתוכנה

כל סקריפט שרושמים אפשר לעלות כהרחבה לתוכנה ובכך להעביר אותו בין משתמשים. זאת אומרת שאת מה שרשמנו אנחנו יכולים לתת לאחרים להשתמש כרצונם רק עליהם להוריד את הקוד והוסיף אותו כהרחבה לתוכנה.

תוצאה

ממשק המשתמש:

למערכת קיים ממשק משתמש שכתבנו ב-Python.



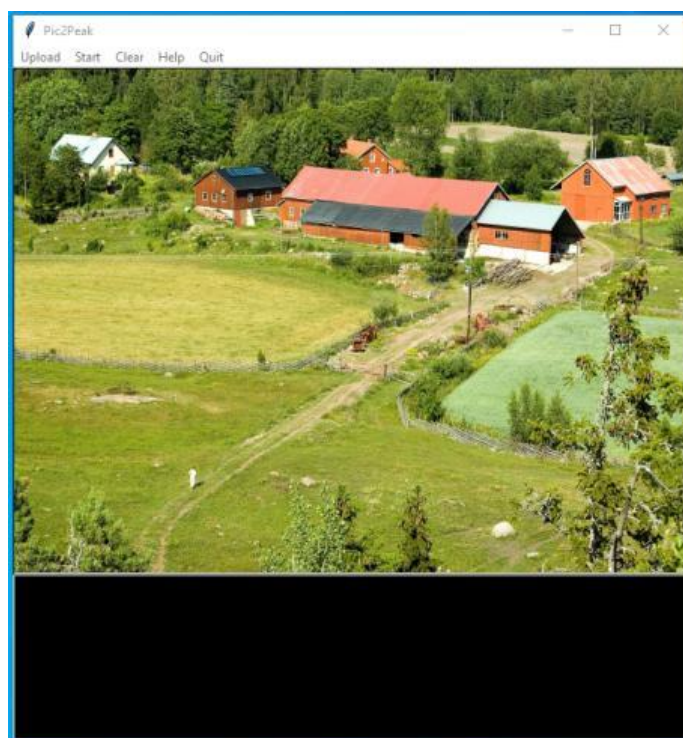
איור 18 - ממשק המשתמש

לממשק האפשרויות הבאות :

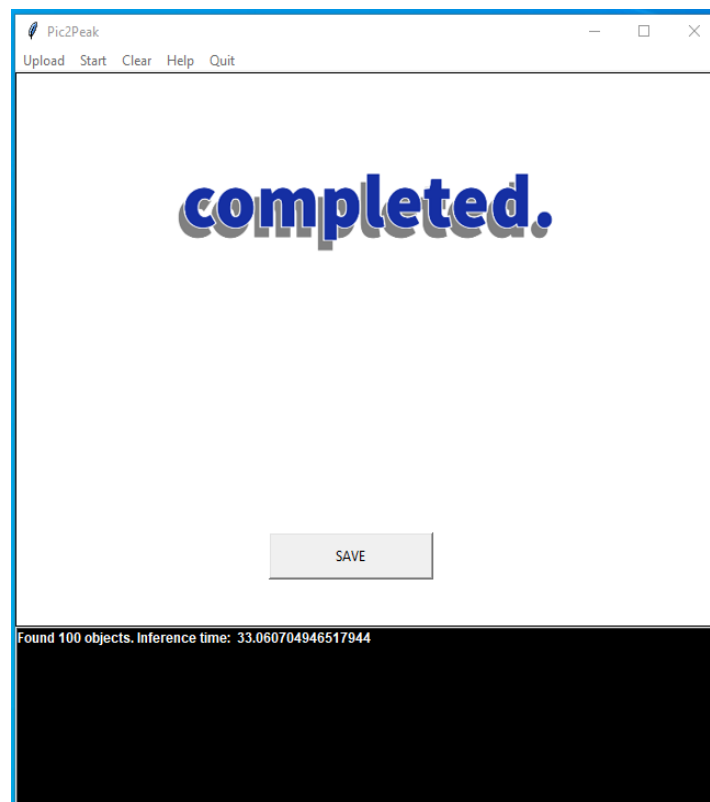
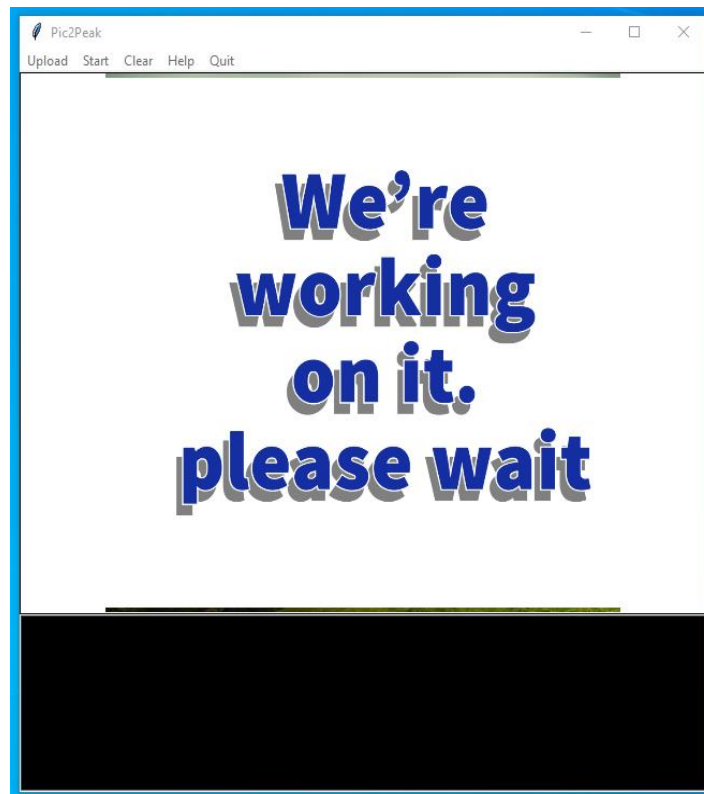
1. לעלות תמונה "Upload" – אפשרות זאת מאפשרת לעלות תמונה כלשהי לממשק שאותה הוא יעביר לשאר רכיבי המערכת
2. להתחיל בפעולה "Start" – בלחיצה על כפתור זה המערכת תתחיל בביצוע סיווג התמונה. מה שקורה זה שהתמונה שהמשתמש העלה עוברת סיווג על ידי המודל שלנו.
3. ניקוי התמונה "Clear" – אפשרות זאת מאפשרת לאפס את הבחירה של התמונה. אם העלתי תמונה, והחלטתי שאני רוצה תמונה אחרת. אני לוחץ על כפתור זה ומעלה תמונה אחרת.
4. כפתור עזרה "Help" – כפתור זה מספק מידע למשתמש במקרה והוא לא מתמצא בממשק.
5. כפתור יציאה "Quit" – כפתור שסוגר את הממשק.

לממשק קיימים 2 חלונות עיקריים :

1. חלון תצוגה – חלון שעליו תופיע התמונה שנעלה ולאחר מכן הודעות למשתמש כאשר התמונה תעבור לביצוע של המודל בלחיצה על "Start".



איור 19 - הממשק לאחר העלאת תמונה

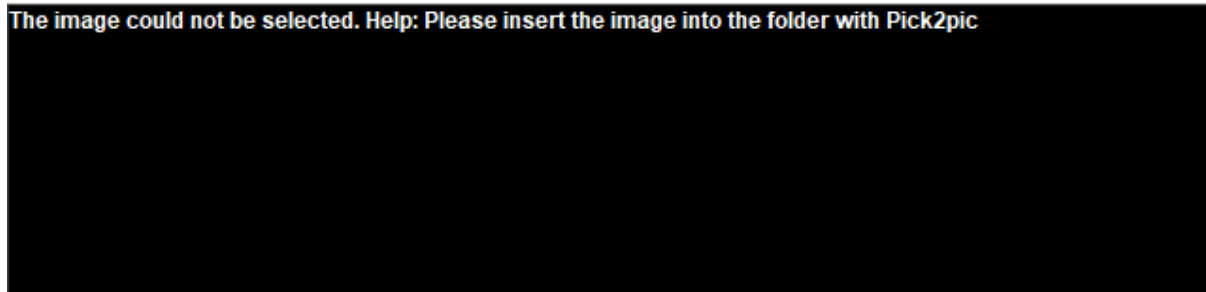


איור 20 - השלבים לאחר לחיצת *START*

לאחר השלמת הסיווג הממשק יתן אפשרות למשתש להוריד כקובץ CSV את המידע שהמודל מוציא כפלט.

2. חלון שגיאות והודעות – חלון שעליו יופיעו כל השגיאות שהמערכת יודעת להתמודד איתם. בחלון

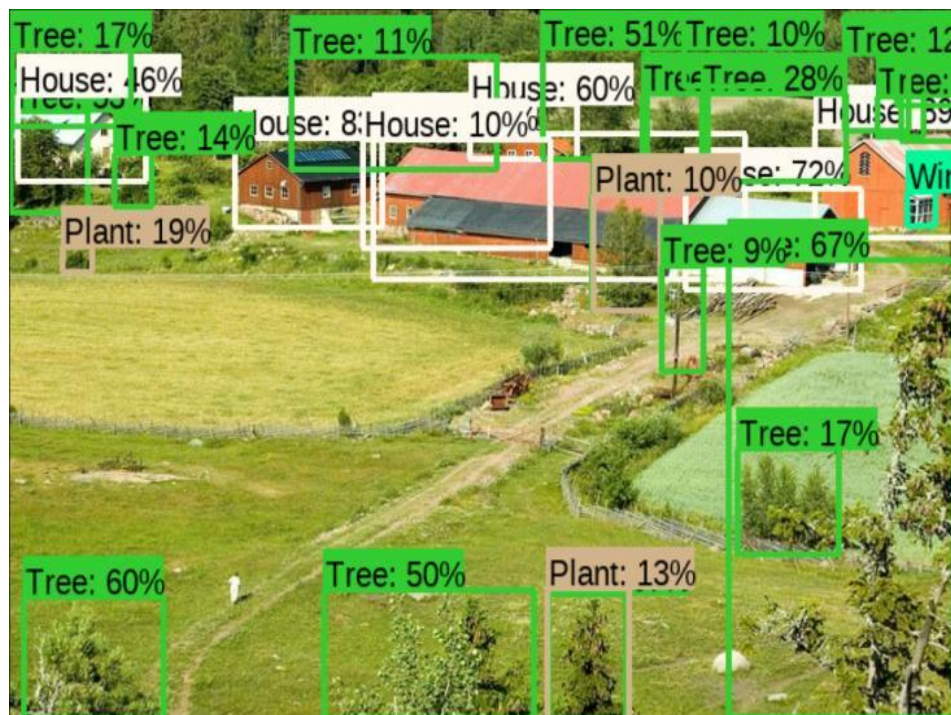
זה גם יופיעו הודעות שחשבנו שהמשתמש ירצה לדעת.



איור 21 - חלון ההודעות

מודל הבינה המלאכותית:

כחלק מהמערכת קיים מודל בינה מלאכותית של רשת נוירונים אשר יודע לעבוד עם תמונות. המודל יודע למצוא אובייקטים שהוא מכיר בכל תמונה שניתן לו.



איור 22 - האובייקטים שהמודל זיהה

לאחר הפעולה של המודל. אנחנו מוציאים את כל הנתונים שאנחנו צריכים ממנו כקובץ CSV.

	A	B	C
	detection_class_entities	detection_scores	detection_boxes
1			
2	b'House'	0.8978849	[0.9227758646011353, 0.6806906461715698]
3	b'House'	0.8371412	[0.31453877687454224, 0.6956505477428436]
4	b'House'	0.8189821	[0.5788055062294006, 0.621443122625351]
5	b'House'	0.7225058	[0.8034965991973877, 0.6075653731822968]
6	b'Tree'	0.67874503	[0.8751996755599976, 0.006403088569641113]
7	b'Tree'	0.6779676	[0.607780933380127, 0.0041779279708862305]
8	b'Tree'	0.60651785	[0.08936427533626556, 0.005098879337310791]
9	b'House'	0.60634565	[0.5400846004486084, 0.7916330099105835]
10	b'Tree'	0.5388915	[0.04224643483757973, 0.7151646614074707]
11	b'Tree'	0.5183819	[0.6162204742431641, 0.7913278341293335]
12	b'Tree'	0.50684536	[0.44148164987564087, 0.004419207572937012]
13	b'Tree'	0.49014533	[0.6481947898864746, 0.5788791179656982]
14	b'House'	0.46459144	[0.07663275301456451, 0.7585079967975616]
15	b'Tree'	0.43652365	[0.6997880339622498, 0.7768758833408356]
16	b'Window'	0.32313538	[0.9583925008773804, 0.6972405314445496]
17	b'Tree'	0.28371686	[0.8039751052856445, 0.7589514553546906]
18	b'Plant'	0.1954319	[0.07155741751194, 0.6331574022769928]
19	b'Tree'	0.17866224	[0.8189020156860352, 0.23580843210220337]
20	b'Tree'	0.17292134	[0.06616292148828506, 0.8371074497699738]
21	b'Tree'	0.15643059	[0.9682067632675171, 0.8163221180438995]
22	b'Tree'	0.14643276	[0.1304406225681305, 0.7245932221412659]
23	b'Tree'	0.14588062	[0.9369971752166748, 0.8211230486631393]
24	b'Plant'	0.13304996	[0.6085229516029358, 0.0029441118240356445]
25	b'Tree'	0.12737305	[0.9657387137413025, 0.8653010576963425]
26	b'Tree'	0.121416315	[0.9118005037307739, 0.8299691677093506]

איור 23 - פלט המודל בטבלת CSV

הנתונים שחילצנו הם :

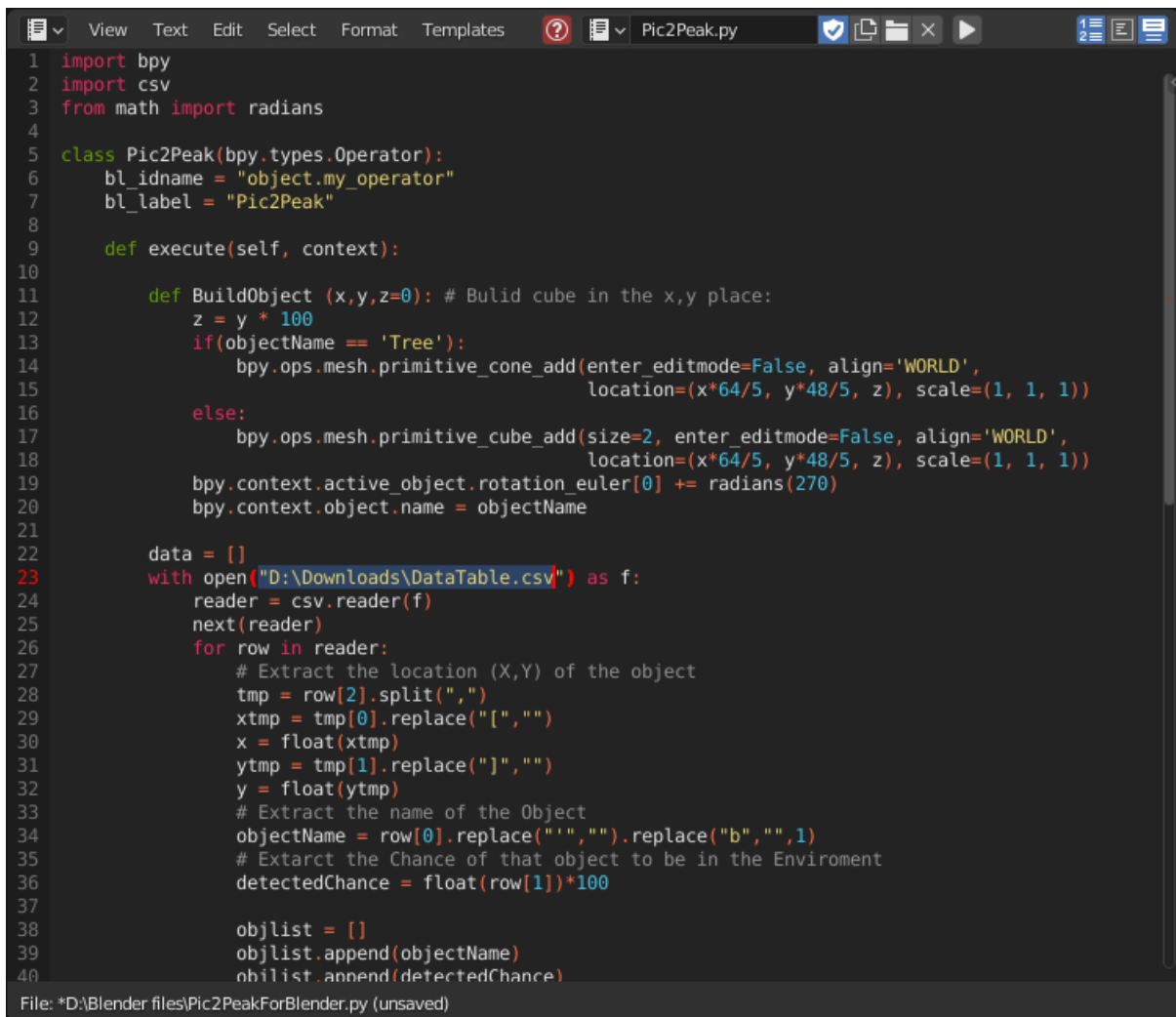
1. שם האובייקט (detection_class_entities).
2. כמה המודל בטוח בזה שזה באמת האובייקט שהוא זיהה (detection_scores).
3. מיקום האובייקט בתמונה (detection_boxes).

תצוגה תלת-מימדית:

החלק האחרון של המערכת שלנו היא התוכנה Blender. כדי שהמערכת תעבוד עם הנתונים שאותם מצאנו מהתמונה עלינו לדאוג שבתוכנה נמצא הקוד שכתבנו. צריך לעלות את הקוד כהרחבה לתוכנה. תהליך זה צריך לקרות פעם אחת בלבד.

כאשר התוכנה Blender פתוחה נלחץ בסרגל הראשי למעלה על "Edit" ואז על "Preferences...". לאחר מכן יפתח לנו חלון בשם "Blender Preferences". בחלון זה יש בצד שמאל אפשרות "Add-ons", נלחץ עליה ואז על "Install..." בצד ימין למעלה. ונעלה את הקובץ בשם "Pic2PeakForBlender.py".

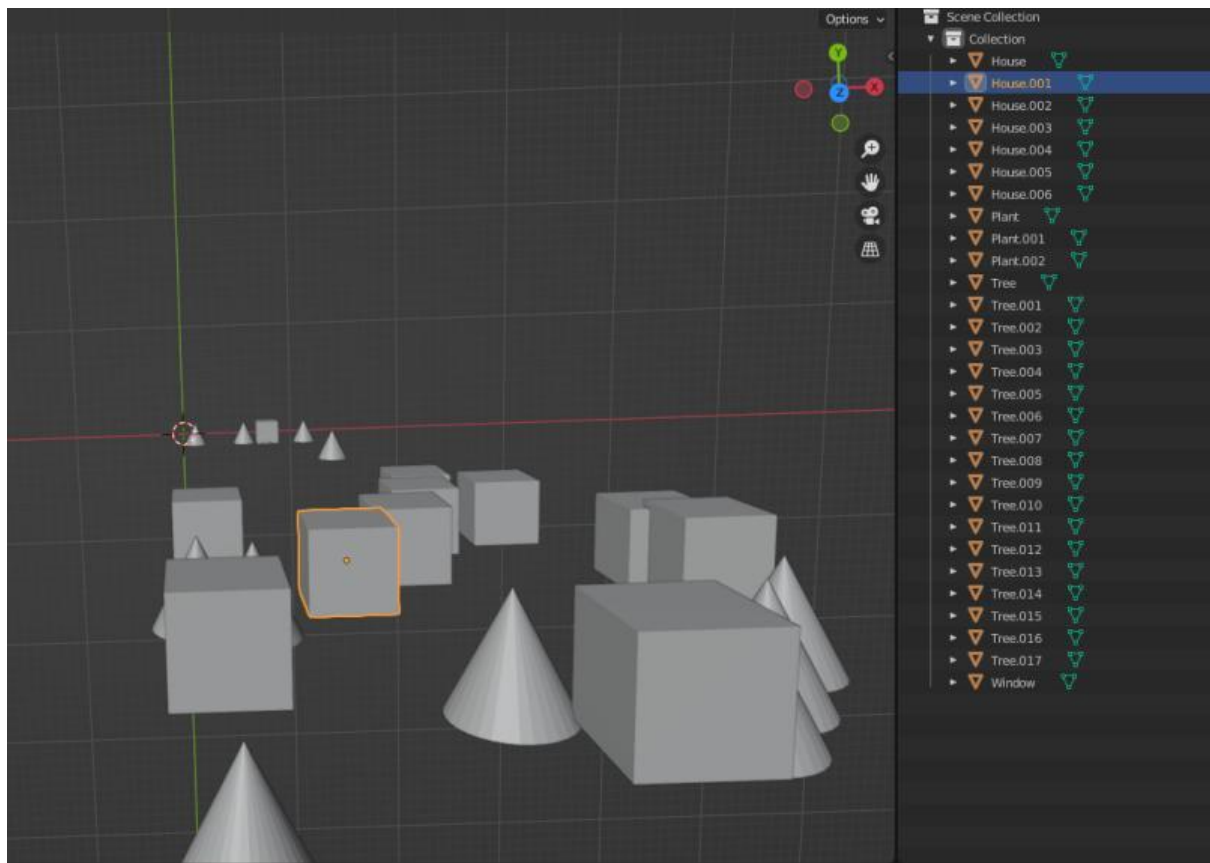
כעת יפתח לנו חלון שבו יהיה רשום קטע הקוד שלנו.



```
1 import bpy
2 import csv
3 from math import radians
4
5 class Pic2Peak(bpy.types.Operator):
6     bl_idname = "object.my_operator"
7     bl_label = "Pic2Peak"
8
9     def execute(self, context):
10
11         def BuildObject (x,y,z=0): # Bulid cube in the x,y place:
12             z = y * 100
13             if(objectName == 'Tree'):
14                 bpy.ops.mesh.primitive_cone_add(enter_editmode=False, align='WORLD',
15                                                  location=(x*64/5, y*48/5, z), scale=(1, 1, 1))
16             else:
17                 bpy.ops.mesh.primitive_cube_add(size=2, enter_editmode=False, align='WORLD',
18                                                  location=(x*64/5, y*48/5, z), scale=(1, 1, 1))
19             bpy.context.active_object.rotation_euler[0] += radians(270)
20             bpy.context.object.name = objectName
21
22         data = []
23         with open("D:\Downloads\DataTable.csv") as f:
24             reader = csv.reader(f)
25             next(reader)
26             for row in reader:
27                 # Extract the location (X,Y) of the object
28                 tmp = row[2].split(",")
29                 xtmp = tmp[0].replace("[", "")
30                 x = float(xtmp)
31                 ytmp = tmp[1].replace("]", "")
32                 y = float(ytmp)
33                 # Extract the name of the Object
34                 objectName = row[0].replace("'", "").replace("b", "", 1)
35                 # Extract the Chance of that object to be in the Enviroment
36                 detectedChance = float(row[1])*100
37
38                 objlist = []
39                 objlist.append(objectName)
40                 objlist.append(detectedChance)
```

File: *D:\Blender files\Pic2PeakForBlender.py (unsaved)

מה שנשאר זה לוודא שהמיקום של הקובץ SCV שאותו הוצאנו מהמודל נמצא במיקום שממנו הקוד מוגדר לפתוח אותו. לאחר מכן ללחוץ על "Run Script" שזה הכפתור עם המשולש (נראה כמו חץ שפונה שמאלה) למעלה בצד ימין. והסביבה התלת-מימדית תיבנה.



איור 25 - הסביבה שנבנתה בבלנדר

והינה לנו הסביבה התלת-מימדית ששוחזרה מהתמונה. בצד שמאל של התמונה קיימת הסביבה, שאפשר לטייל בה ככל שנרצה. בצד שמאל מפורטים האובייקטים שנכנסו לסביבה לאחר סינון שלנו, של איזה אובייקטים יכנסו ואיזה לא. ניתן לראות שלכל אובייקט יש שם, שם זה הוא השמות שהמודל רשת הנוירונים הוציא לנו מהתמונה. בנוסף כדי שהסביבות יהיו יותר מובנות הפרדנו בין יצוג גרפי של אובייקטים ויצגנו עצים כחרוטים. וכל אובייקט אחר מיוצג כקוביה.

מסמך בדיקות

בדיקות משתמש:

מספר בדיקה		1	יוצר הבדיקה		מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)		גבוהה	תאריך יצירת הבדיקה		30/05/2022
תאריך הרצת הבדיקה		30/05/2022	מריץ הבדיקה		מאי יחזקאל
מטרת הבדיקה				נבדוק שהתוכנה מציגה נתונים תלת ממדיים.	
תנאים מקדימים				הפעלת התוכנה, העלאת תמונה והפעלת המכונה.	
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור	עבר / לא עבר	הערות
1	לחץ Upload, ובחר תמונה מתוך סירר הקבצים.	התמונה שנבחרה מוצגת בתוכנה על הקנבס בגודל מלא.	התמונה שנבחרה מוצגת בתוכנה על הקנבס בגודל מלא.	עבר	
2	לחץ Start.	תצוגה תלת ממדית של הסביבה מוצגת על הקנבס בגודל מלא.	נפתח חלון המודיע על סיום התהליך וכפתור לשמירת קובץ.	לא עבר	את הקובץ נעלה אל תוכנה חיצונית על מנת לראות את הנתונים בתלת ממד.

מספר בדיקה		3	יוצר הבדיקה		מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)		גבוהה	תאריך יצירת הבדיקה		30/05/2022
תאריך הרצת הבדיקה		30/05/2022	מריץ הבדיקה		מאי יחזקאל
מטרת הבדיקה				נבדוק שהתוכנה מציגה חלון תצוגה בגודל מלא בהפעלה.	
תנאים מקדימים					
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור	עבר / לא עבר	הערות
1	נפתח את תוכנת .PEAK2PIC	התוכנה נפתחת בגודל חלון מלא.	התוכנה נפתחת בגודל חלון מלא.	עבר	

מספר בדיקה	3.1	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהתוכנה מציגה שורת תפריט עם כפתורים.		
תנאים מקדימים			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	נפתח את תוכנת PEAK2PIC.	שורת תפריט ראשי מופיעה בחלק העליון של החלון.	שורת תפריט ראשי מופיעה בחלק העליון של החלון.
הערות	עבר / לא עבר	הערות	עבר

מספר בדיקה	3.2	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהתוכנה מציגה חלון תצוגה בהפעלה.		
תנאים מקדימים			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	נפתח את תוכנת PEAK2PIC.	שטח תצוגה עיקרי מופיע באמצע חלון התוכנה ועליו לוגו התוכנה.	שטח תצוגה עיקרי מופיע באמצע חלון התוכנה ועליו לוגו התוכנה.
הערות	עבר / לא עבר	הערות	עבר

מספר בדיקה	3.3	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהתוכנה מציגה חלון הודעות בהפעלה.		
תנאים מקדימים			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	נפתח את תוכנת PEAK2PIC.	שטח חלון הודעות מופיע בתחתית חלון התוכנה בצבע שחור.	שטח חלון הודעות מופיע בתחתית חלון התוכנה בצבע שחור.
הערות	עבר / לא עבר	הערות	עבר

מספר בדיקה	5.1	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק ששורת התפריט מופיע לאורך כל חלקו עליון של מסך התוכנה.		
תנאים מקדימים			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	נפתח את תוכנת PEAK2PIC.	שורת התפריט מופיעה לכל אורך חלקו העליון של חלון התוכנה.	שורת התפריט מופיעה לכל אורך חלקו העליון של חלון התוכנה.

מספר בדיקה	5.2	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שניתן להעלאות תמונה דרך כפתור "UPLOAD" בשורת התפריט.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	נלחץ על UPLOAD בשורת התפריט.	סייר קבצים נפתח.	סייר קבצים נפתח.
2	בחירת תמונה מתוך סייר קבצים.	התמונה שנבחרה מוצגת בחלון התצוגה הראשי.	התמונה שנבחרה מוצגת בחלון התצוגה הראשי.

מספר בדיקה	5.3	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שכפתור UPLOAD פותח לנו את סייר הקבצים.		
תנאים מקדימים	הפעלת התוכנה ולחיצה על UPLOAD בשורת התפריט.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
שלב לביצוע	שלב לביצוע	עבר / לא עבר	הערות
1	נלחץ על UPLOAD בשורת התפריט.	סייר קבצים נפתח.	סייר קבצים נפתח.

מספר בדיקה	5.4	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שכפתור RESET פותח לנו את סייר הקבצים.		
תנאים מקדימים	הפעלת התוכנה והעלאת תמונה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
שלב לביצוע	שלב לביצוע	עבר / לא עבר	הערות
1	נלחץ על RESET בשורת התפריט.	התמונה נמחקה ובמקומה מופיע חלון התצוגה עם לוגו התוכנה.	התמונה נמחקה ובמקומה מופיע חלון התצוגה עם לוגו התוכנה.

מספר בדיקה	5.5	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שכפתור EXIT יוצא מהתוכנה.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	נלחץ על EXIT בשורת התפריט.	חלון קופץ ששואל האם אתה בטוח.	חלון קופץ ששואל האם אתה בטוח.
2	לחץ "כן אני בטוח"	התוכנה תיסגר.	התוכנה נסגרה.
הערות	עבר / לא עבר	הערות	הערות

מספר בדיקה	5.6	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שכפתור HELP פותח לנו מדריך עזרה למשתמש.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	נלחץ על HELP בשורת התפריט.	יפתח חלון ובו מדריך למשתמש.	נפתח חלון ובו מדריך למשתמש.
הערות	עבר / לא עבר	הערות	הערות

מספר בדיקה	9	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהחלון מנצל את מלוא שטח התצוגה.		
תנאים מקדימים			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	נפתח את תוכנת PEAK2PIC.	חלון התוכנה נפתח על כל שטח חלון המסך.	חלון התוכנה נפתח על כל שטח חלון המסך.

מספר בדיקה	10	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהודעות מערכת מוצגות ברכיב שורת ההודעות.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור RESET בשורת התפריט.	הודעה בחלון השגיאות המודיעה על שגיאה באיפוס. (מכוון ולא הועלתה תמונה)	הודעה בחלון השגיאות המודיעה על שגיאה באיפוס. (מכוון ולא הועלתה תמונה)

מספר בדיקה	11	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהודעות מערכת המוצגות ברכיב שורת ההודעות, מספקות הנחיות לגבי השגיאה.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור RESET בשורת התפריט.	הודעה בחלון השגיאות המנחה מה לעשות לגבי סוג השגיאה וכיצד להתגבר עליה.	הודעה בחלון השגיאות המנחה מה לעשות לגבי סוג השגיאה וכיצד להתגבר עליה.

מספר בדיקה	12	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת תומכת בשפה האנגלית.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	נפתח את תוכנת PEAK2PIC.	התוכנה תפתח בשפה האנגלית.	התוכנה תפתח בשפה האנגלית.

בדיקות מערכת:

מספר בדיקה	1	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת יודעת לזהות אובייקטים בתמונה.		
תנאים מקדימים	הפעלת התוכנה, העלאת תמונה והפעלה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	לחץ על כפתור SAVE במסך התצוגה.	נשמר קובץ ובו שמות האובייקטים הנמצאו.	נשמר קובץ ובו שמות כל האובייקטים שנמצאו.
הערות	עבר / לא עבר	הערות	עבר

מספר בדיקה	2	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת יודעת לזהות את וקטור מיקום האובייקטים בתמונה.		
תנאים מקדימים	הפעלת התוכנה, העלאת תמונה והפעלה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
1	לחץ על כפתור SAVE במסך התצוגה.	נשמר קובץ ובו וקטורי המיקום של האובייקטים הנמצאו.	נשמר קובץ ובו וקטורי המיקום של כל האובייקטים שנמצאו.
הערות	עבר / לא עבר	הערות	עבר

מספר בדיקה	3	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת יודעת לשמור את שמות האובייקטים יחד עם וקטור המיקום שלהם.		
תנאים מקדימים	הפעלת התוכנה, העלאת תמונה והפעלה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור SAVE במסך התצוגה.	נשמר קובץ ובו שמות האובייקטים יחד עם וקטורי המיקום של האובייקטים הנמצאו.	נשמר קובץ ובו שמות האובייקטים יחד עם וקטורי המיקום של האובייקטים הנמצאו.

מספר בדיקה	4	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת יודעת להתמודד עם אובייקטים שאינה מוכרים לה.		
תנאים מקדימים	הפעלת התוכנה, העלאת תמונה והפעלה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור SAVE במסך התצוגה.	נשמר קובץ ובו אובייקטים שנמצאו, אובייקטים שלא מוכרים אינם נמצאים בקובץ.	נשמר קובץ ובו אובייקטים שנמצאו, אובייקטים שלא מוכרים אינם נמצאים בקובץ.

מספר בדיקה	7	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמערכת יודעת לייצר קובץ FBX.		
תנאים מקדימים	הפעלת התוכנה, העלאת תמונה והפעלה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור SAVE במסך התצוגה.	נשמר קובץ FBX.	נשמר קובץ CSV.
		לא עבר	

מספר בדיקה	8	יוצר הבדיקה	מאי יחזקאל
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)	גבוהה	תאריך יצירת הבדיקה	30/05/2022
תאריך הרצת הבדיקה	30/05/2022	מריץ הבדיקה	מאי יחזקאל
מטרת הבדיקה	נבדוק שהמשתמש יכול להכניס תמונה כקלט לתוכנה.		
תנאים מקדימים	הפעלת התוכנה.		
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור
עבר / לא עבר	הערות		
1	לחץ על כפתור UPLOAD בשורת התפריט.	סייר קבצים נפתח וניתן לבחור תמונה.	סייר קבצים נפתח וניתן לבחור תמונה.
		עבר	

מספר בדיקה		8.1	יוצר הבדיקה	מאי יחזקאל	
חשיבות הבדיקה (נמוכה/ בינונית גבוהה)		גבוהה	תאריך יצירת הבדיקה	30/05/2022	
תאריך הרצת הבדיקה		30/05/2022	מריץ הבדיקה	מאי יחזקאל	
מטרת הבדיקה		נבדוק שהמשתמש יכול להוריד את הקובץ בסיום.			
תנאים מקדימים		הפעלת התוכנה, העלאת תמונה, התחלת המכונה.			
שלב לביצוע	תיאור השלב	תוצאה צפויה	תוצאה במקור	עבר / לא עבר	הערות
1	לחץ על כפתור SAVE בחלון התצוגה.	סייר קבצים נפתח וניתן לבחור מיקום להורדת הקובץ.	סייר קבצים נפתח וניתן לבחור מיקום להורדת הקובץ.	עבר	

1. Alhashim, Ibraheem, and Peter Wonka. "High quality monocular depth estimation via transfer learning." arXiv preprint arXiv: 1812.11941 (2018).
2. Guo, Yanming, Yu Liu, Theodoros Georgiou, and Michael S. Lew. "A review of semantic segmentation using deep neural networks." International journal of multimedia information retrieval 7, no. 2 (2018): 87-93.
3. Hedman, Peter, Tobias Ritschel, George Drettakis, and Gabriel Brostow. "Scalable inside-out image-based rendering." ACM Transactions on Graphics (TOG) 35, no. 6 (2016): 1-11.
4. Wang, Chen, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. "Densefusion: 6d object pose estimation by iterative dense fusion." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 3343-3352. 2019.
5. LaViola Jr, Joseph J. "Bringing VR and spatial 3D interaction to the masses through video games." IEEE Computer Graphics and Applications 28, no. 5 (2008): 10-15.
6. Laina, Iro, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. "Deeper depth prediction with fully convolutional residual networks." In 2016 Fourth international conference on 3D vision (3DV), pp. 239-248. IEEE, 2016.
7. O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv: 1511.08458 (2015).
8. Sharma, Pulkit. "Image classification vs. object detection vs. image segmentation". Analytics Vidhya. (2019) <https://medium.com/analytics-vidhya/image-classification-vs-object-detection-vs-image-segmentation-f36db85fe81>
9. The Metaverse and How We'll Build It Together -- Connect 2021, YouTube, 20.11.2021 https://www.youtube.com/watch?v=Uvufun6xer8&t=1040s&ab_channel=Meta
10. What are the best game engines for python? game engine. Slant. 20/06/2022 <https://www.slant.co/topics/10908/~game-engines-for-python>

11. Yuan, Honglin, and Remco C. Veltkamp. "PreSim: A 3D photo-realistic environment simulator for visual AI." *IEEE Robotics and Automation Letters* **6**, no. 2 (2021): 2501-2508.
12. Zou, Zhengxia, Zhenwei Shi, Yuhong Guo, and Jieping Ye. "Object detection in 20 years: A survey." *arXiv preprint arXiv:1905.05055* (2019).