**המחלקה להנדסת תוכנה**

**פרויקט גמר – תשפ"ב**

**מחקרי רשתות נוירונים עמוקות –**

מציאת התאמה טובה ביותר בין ענני נקודות

Point Clouds Registration Based on SuperGlue Algorithm

**מאת**

**שם הסטודנט\ית: נתנאל ראובן ומאור בן יאיר**

**ת.ז סטודנט\ית: 312597784 205463938**

**מנחה אקדמי: דר' חסין יהודה אישור: תאריך:**

מערכות ניהול הפרויקט:

|  |  |  |
| --- | --- | --- |
| # | מערכת | מיקום |
| 1 | מאגר קוד | [GitHab](https://github.com/benyairmaor/Final_Project_DNN_Lidar) |
| 2 | יומן | [Monday](https://view.monday.com/2092185360-9b543b1e6b32c8b2b80614359dd3f553?r=use1) |
| 3 | סרטון הסבר |  |

מידע נוסף:

|  |  |
| --- | --- |
| סוג הפרויקט | **מחקרי ממרצה במכללה** |
| פרויקט מח"ר | **לא** |
| פרויקט ממשיך | **זה פרוייקט חדש** |
| פרויקט זוגי: | **כן** |

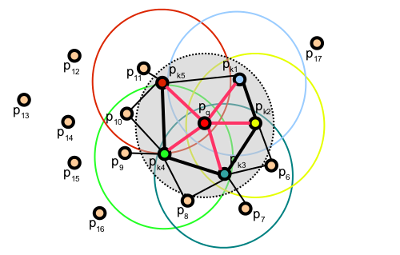
**תוכן עניינים –**

מערכת ניהול הפרויקט..........................................................................................................2   
תוכן עניינים.........................................................................................................................3   
מילון מונחים, סימנים וקיצורים.................................................................................................4

1. מבוא......................................................................................................................5
2. תיאור הבעיה
3. דרישות ואפיון הבעיה.....................................................................................6
4. הבעיה מבחינת הנדסת תוכנה.........................................................................6
5. סקר ספרות.............................................................................................................7
6. תיאור הפתרון..........................................................................................................7
7. קדם מחקר (מה עשינו עד כה)
8. ICP...........................................................................................................9
9. RANSAC + ICP........................................................................................9
10. Sinkhorn + SVD......................................................................................01
11. Sinkhorn + RANSAC + ICP....................................................................11
12. נספחים................................................................................................................12

**מילון מושגים:**

* **Point Cloud (ענן נקודות)** – מערך של נקודות במערכת קואורדינטות מסוימת. במערכת קואורדינטות תלת ממדית, נקודות אלה מסומנות בדרך כלל ב־X ,Y ו־ Z.
* **Correspondence** – זוג נקודות מתאימות כך שעבור ההתאמה של ענני נקודות A ו-B הנקודות a ו-b נמצאות אחת על השניה.
* **Overlap (אחוז חפיפה)** – אחוז החפיפה המבטא את ההתאמה המקסימלית אליה ניתן להגיע בין שני ענני נקודות.
* **Rotation Matrix (מטריצת סיבוב)** – היא מטריצת טרנספורמציה המשמשת לביצוע סיבוב במרחב האוקלידי.
* **Translation Vector (וקטור הזזה)** – המשמשת לביצוע הזזה במרחב האוקלידי.
* **SVD** **Singular value decomposition (פירוק לערכים סינגולריים)** - היא שיטת פירוק באלגברה ליניארית של מטריצה מרוכבת או ממשית. בהתאם לבעיה שלנו ניתן לחשב בהנתן זוגות של correspondence אופטימלים את מטריצת הסיבוב ווקטור הזזה האופטימלים עבור בעית התאמה ספציפית.
* **ICP (Iterative closest point)** – אלגוריתם איטרטיבי המשמש כדי למזער את ההבדל בין שני ענני נקודות. ICP משמש גם ל – 2D או 3D.
* **RANSAC (Random sample consensus)** – שיטה איטרטיבית להערכת פרמטרים של מודל מתמטי מתוך קבוצה של נתונים המכילה נתונים חריגים, כאשר לנתונים החריגים אין יכולת השפעה על ערכי האומדנים.
* **Sinkhorn** – אלגוריתם הפותר את [בעית ההשמה](https://he.wikipedia.org/wiki/%D7%94%D7%90%D7%9C%D7%92%D7%95%D7%A8%D7%99%D7%AA%D7%9D_%D7%94%D7%94%D7%95%D7%A0%D7%92%D7%A8%D7%99) ע"י מציאת מינימום למחיר השגיאה (מרחק בין זוג נקודות), ובכך בבעיה שלנו ניתן למצוא correspondence.
* **Descriptor** – תיאורים של המאפיינים החזותיים של התוכן לדוגמא בתמונות, סרטונים, אלגוריתמים או יישומים נוספים דומים לתיאורים אלו.
* **FPFH (Fast Point Feature Histograms) Descriptor** – תיאור גיאומטרי של נקודה בהתייחס לסביבה המקומית שלה.



* **Voxel down sample** – סינון נקודות על-ידי שימוש בייצוג של נקודה בודדת עבור גודל שטח מסוים.

**מבוא**

**Lidar –**

בשנים האחרונות בתחומים שונים בתעשייה נעשה שימוש במכשיר Lidar. Lidar זוהי טכנולוגיה למידול תלת מימדי / מדידת מרחק בעזרת חיישני לייזר. זאת על ידי הארת המטרה בקרן לייזר. הקרן נשברת וחוזרת אל סורק, אשר מחשב ע"י אורך גל הלייזר את מיקום המטרה. פלט הLidar הוא נקודות (x,y,z) במרחב ובעזרתו ניתן למפות את המרחב. דוגמא לשימוש ב-Lidar ניתן לראות בשימוש ברכבים אוטונומים, במפות 3D ועוד.

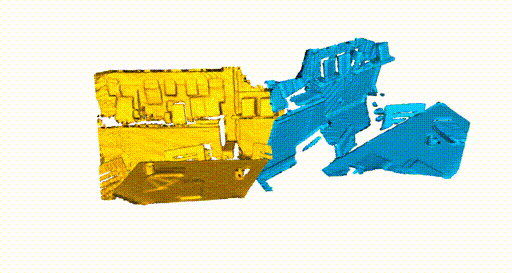
**Deep Neural Network –**

רשתות נוירונים עמוקות מהוות רכיב מרכזי בעולם התוכנה. רשתות נוירונים עמוקות הן מודל מתמטי חישובי, שפותח בהשראת תהליכים מוחיים או קוגניטיביים המתרחשים ברשת עצבית טבעית ומשמש במסגרת למידת מכונה. רשתות נוירונים עמוקות נכנסו לחיינו בשנת 2012 ע"י הרשת AlexNet אשר הצליחה לפתור בעית סיווג של תמונות ל1000 קטגוריות שונות באחוזים גבוהים (יחסית לאותה שנה).

**SuperGlue** – אלגוריתם שפורסם בשנת 2019 ע"י Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich. אלגוריתם אשר מבצע התאמה בין שתי תמונות. האלגוריתם מציע פתרון למציאת נקודות טובות ביותר להתאמה ודחיית נקודות שאינן ניתנות להתאמה. SuperGlue מורכב מרשת נוירונים גרפית (GNN) שלומדת להפיק Descriptor טוב ביותר לכל pixel בכל אחת מהתמונות. לאחר מכאן מבצע התאמה בעזרת אלגוריתם Sinkhorn.

לאור הגדילה בכמות חברות ההייטק המשתמשות הן ברשתות נוירונים עמוקות והן ברכיב הLidar.

בחרנו לבצע פרויקט מחקרי המשלב בין התחומים. **הבעיה שאיתה אנו נתמודד בפרויקט היא בעיית ההתאמה של שני ענני נקודות** (כאשר אחוז החפיפה בניהם >=50). שהיא בעיה יסודית בתחום הנ"ל.



בפרויקט זה אנו נבצע התאמה לאלגוריתם SuperGlue. כך שיאפשר מציאת התאמה טובה ביותר עבור ענני נקודות. בעצם ילמד להפיק Descriptor טוב ביותר עבור כל נקודה מענני הנקודות. לאחר מכאן יבצע התאמה בעזרת Sinkhorn.

**דרישות ואפיון הבעיה**

## **דרישות ואפיון הבעיה -**

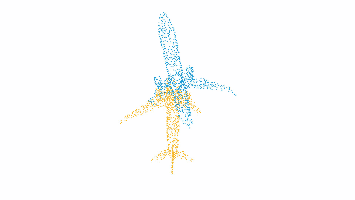
בעיית ההתאמה של שני ענני נקודות –

בהנתן ענני נקודות A ו-B אשר מייצגים את אותו **האובייקט** (או לפחות אחוז חפיפה גדול בניהם) ואחוז חפיפה עבור ההתאמה האופטימלית M.

נרצה למצוא מטריצה R ווקטור T כך שאחוז החפיפה של (R\*A + T, B) יהיה קרוב ביותר שניתן ל- M ובכך יספק לנו את ההתאמה הקרובה ביותר לאופטמלית.

ב-dataset שאיתו בחרנו לעבוד ETH (הסבר מפורט בהמשך) יש מטריצת סיבוב למיקום מוטעה 'R ווקטור ההזה למיקום מוטעה 'T. כך ש A’ = R’\*A + T’.

לכן נרצה למצוא מטריצה R ווקטור T כך שאחוז החפיפה של (R\*A’ + T, B) קרוב ביותר שניתן ל-M.



## **הבעיה מבחינת הנדסת תוכנה –**

ראשית נבחין כי הבעיה שהצגנו מתחלקת לשתי בעיות האחת קלה והשניה קשה.

הבעיה הקלה: בהנתן וקטור זוגות של correspondence אופטימלים מצא ההתאמה הטובה.  
אז על ידי אלגוריתמם SVD ניתן להגיע למטריצת סיבוב ווקטור הזזה אופטימלים (כך שאחוז החפיפה לאחר התיקון = Overlap).

הבעיה הקשה: מצא את ה correspondenceהטובים ביותר ובעזרתם מצא את ההתאמה הטובה ביותר.

בהנתן שלא ידוע ה-correspondence מציאתן מהווה מרכיב מרכזי בבעיה. ככל שה-correspondence יהיו קרובים יותר לאופטימלים כך ההתאמה תהיה קרובה יותר ל-Overlap. לא ניתן לבדוק את כל ההתאמות בזמן פולינומי ולכן מבצעים קירוב.

כיום קיימים אלגוריתמים למציאת קירוב לפתרון. לדוגמא ICP לבעית ההתאמה או Sinkhorn למציאת correspondence ואז חישוב ההתאמה. אך אינן מהוות פתרון כללי לבעיה, עם פרמטרים X יכול להיות שנקבל ציון גבוה על בעיה a וציון נמוך על בעיה b, אם נשנה את X יכול להיות שנקבל ציון גבוה על בעיה b וציון נמוך על בעיה a. כלומר עבור כל בעיה נצטרך לבחור פרמטרים אחרים.

נרצה להשתמש באלגוריתם SuperGlue כדי לפתור את הבעיה הצגנו. אופן פועלת רשתות הנוירונים עדיין נחקרות, ולכן אנחנו לא יודעים אם פתרון זה יהיה יעיל עבור בעיית ענני הנקודות. מהם הפרמטרים שהרשת צריכה לקבל? האם נצליח? איך הרשת לומדת? מה הרשת לומדת? האם היא טובה על כל סוג של דאטה, גם כשאחוז החפיפה קטן?

**סקירת בספרות**

* (A1) [Deep learning based point cloud registration](https://www.sciencedirect.com/science/article/pii/S2096579620300383)

מחקר המציג פתרון עבור התאמת שני ענני נקודות. מוכיח את העובדה שרשתות נוירונים עמוקות יכולות להפיק פתרון עבור בעית ההתאמה של ענני נקודות.

* (A2) [RANSAC on FPFH Descriptors + ICP](https://iopscience.iop.org/article/10.1088/1757-899X/612/3/032190/pdf)

מאמר המציג פתרון עבור התאמת שני ענני נקודות על ידי שימוש בRANSAC אשר מבצע מה שנקרא Global Registration שזה בעצם התאמה ע"י מעט נקודות בדרך כלל, משיג התאמה טובה אך לא מדויקת. לאחר מכן מבצע ICP כמו שנקרא Local Registration שזה התאמה על ידי שימוש בכל הנקודות בענני הנקודות (עובד אם ההזזה לא רחוקה מידי).

* (A3) [SuperGlue](https://arxiv.org/pdf/1911.11763.pdf)

מחקר המציג פתרון עבור מציאת התאמה בין שתי תמונות (אופן הפעולה מוסבר במבוא). זהו האלגוריתם שאנו נתבסס על מנת לפתור את בעיית ההתאמה עבור ענני הנקודות.

**תיאור הפתרון**

כאמור כפי שמוזכר בסקירת ספרות (A1) ישנו פתרון לבעיה התאמת ענני נקודות המיושמת ברשתות נוירונים עמוקות. מה שגורם לנו לחשוב כי רשתות נוירונים עמוקות מספקות תוצאות טובות לבעית התאמה של ענני נקודות.

מאמר SuperGlue (A3) מציע את הפתרון הטוב ביותר לבעית ההתאמה עבור תמונות נכון לשנת 2019. מאמר זה מציע פתרון ע"י שימוש ברשתות נוירונים. SuperGlue זכה בכל התחרויות של התמאת תמונות בשנת 2019.

מאחר ו-SuperGlue התאימה את הpixels בעזרת Descriptors ולא בעזרת שום דבר שמייחד את התמונה כ"תמונה". גרם לנו לתהות האם האלגוריתם הזה יכול לעבוד על ענני נקודות? ואם כן, כמה טוב יצליח?

מטרתנו בפרויקט תהיה ליצור אלגוריתם אשר יקבל שני ענני נקודות. אלגוריתם זה יפעל כמו SuperGlue וגם הוא יעבוד בשני שלבים:

**בשלב הראשון:** רשת נוירונים גרפית שתאומן בכדי למצוא את הDescriptors טובים ביותר עבור כל נקודה בכל אחד מענני הנקודות.

**בשלב השני:** Sinkhorn שיאמד את המרחקים בין הנקודות על פי הDescriptors. ובכך ימצא את correspondence טובים. השאלה כמה טובים?

כפי שהזכרנו אלגוריתם SuperGlue חדשני ומספק תוצאות מעולות, אין שום סיבה שלא נבחן את יכולתו לעבוד על Data שונה (ענני נקודות).

נשאף להגיע לרמת דיוק גבוהה מהקיים היום. כמו כן להבין מהם הערכים המשמעותים בכל אחד מהשלבים של האלגוריתם, מהם הערכים אשר יובלו את הרשת להתכנס לפתרון מציאת ההתאמה הטובה ביותר ואיך היא בנויה.

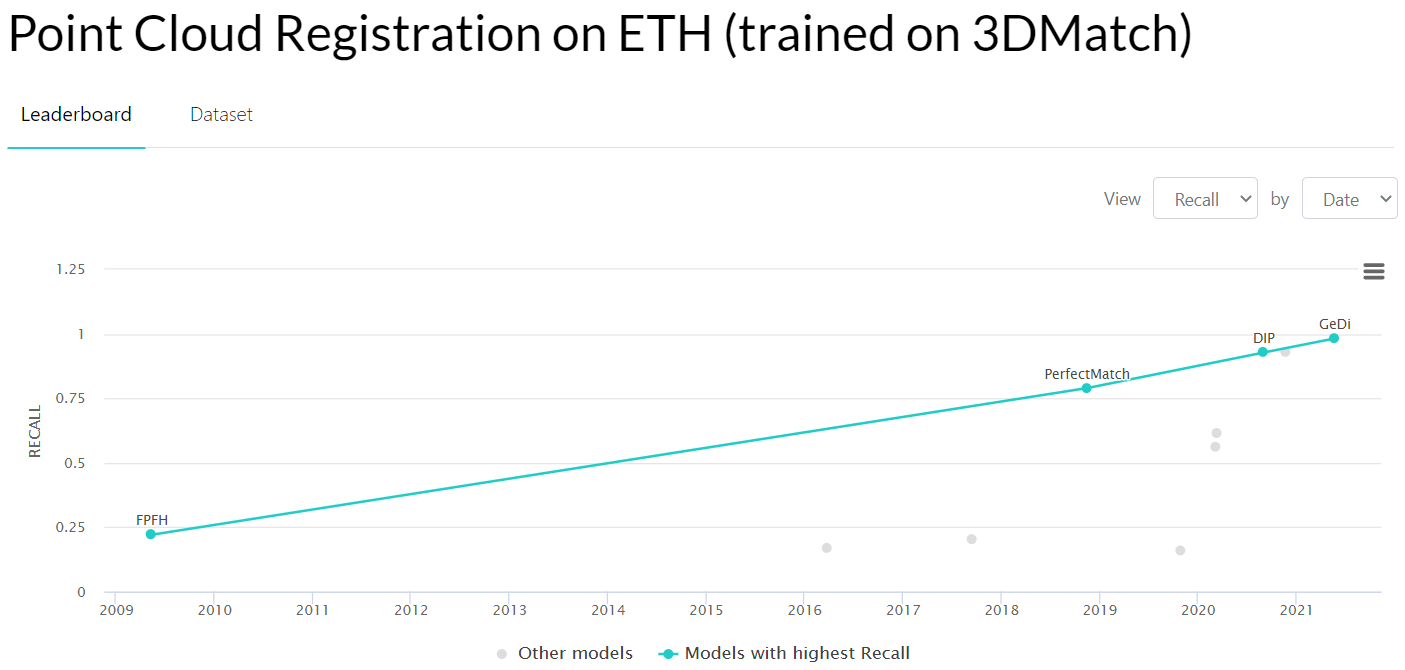
כדי להוכיח שהפתרון אכן אפשרי השתמשנו בארבעה אלגוריתמים המציעים פתרון לבעיית ההתאמה. בחנו את האלגוריתמים המצוינים מטה על עשר בעיות מכל קבוצה בדאטה ETH ובכך הרצנו כל אלגוריתמים על אותן 80 בעיות. לקבצות הבעיות הזו קראנו POC.

אלגוריתמים להוכחת הטענה:

1. ICP
2. RANSAC + ICP
3. Sinkhorn + SVD
4. Sinkhorn + RANSAC + ICP

למעשה כל שנרצה לעשות זה למצוא correspondenceטובים יותר מאלגוריתמים אלו על מנת לבצע התאמה טובה יותר על כל זוג ענני הנקודות. בנוסף לכך נרצה לספק התאמה עבור כל זוג ענני נקודות ללא קשר לאוביקט שמייצג או להזזה של אחד מענני הנקודות. שזהו כמו שהזכרנו מקודם נקודת חולשה של אלגוריתמים אלו.

כאמור, נשים לב כאלגוריתמים שהצגנו מקודם אינם מהווים את הציונים הטובים ביותר עבוד בעיית ההתאמה על הדאטה של ETH וישנם אלגוריתמים שפותרים את הבעיה הזו בציונים טובים בהרבה. נרצה להשוות את יכולות האלגוריתם שנבנה ליכולתיהם.



ציוני האלגוריתמים הטובים ביותר עבור בעיית ההתאמה ב-ETH dataset.

נרצה להגיע לציונים של אלגוריתמים כמו GeDi ו-DIP עבור בעית ההתאמה של שני ענני נקודות על dataset ETH.

**קדם מחקר (מה עשינו עד כה)**

## כמו שצינו בתיאור הפתרון בחנו את האלגוריתמים להוכחת הטענה על אותן 80 בעיות התאמה. בכל הרצה כיווננו את הפרמטרים על בכדי להגיע לציונים הטובים ביותר שניתן על קבוצת הPOC. את ההרצות בצענו על המחשבים האישים שלנו ולכן היו מעט הגבולות. (בהמשך נריץ את אלגוריתמים אלו על מחשב יעודי, ובכך נוכל אף להגיע לתוצאות טובות יותר).

## **Data ([לינק לדאטה](https://projects.asl.ethz.ch/datasets/doku.php?id=laserregistration:laserregistration)) –**

עבור התאמת ענני הנקודות השתמשנו ב-[ETH Dataset](https://projects.asl.ethz.ch/datasets/doku.php?id=laserregistration:laserregistration). מכיל שמונה סוגי אובייקטים שונים המיוצגים על ידי ענני נקודות להתאמה. ובעל כ-800 בעיות התאמה שונות (נספחים A).

כמו כן מכיל קובץ המכיל:

1. שמות של זוגות הענני נקודות עליהם ניתן לבצע התאמה (מקור ויעד).
2. T - מטריצת הטרנספורמציה ליצירת בעיית ההתאמה.
3. Overlap – ערך המייצג אחוז התאמה מקסימלי בין ענני נקודות.

על-ידי שימוש בנתונים אלה נוכל למדוד דיוק של התאמה.

## **ICP –**

## על-ידי שימוש בספריית [Open3D](http://www.open3d.org/docs/release/tutorial/pipelines/global_registration.html?highlight=ransac) השתמשנו באלגוריתם ICP על כל זוג של ענני נקודות. כאמור ICP טוב לבעיות בהן האתחול (המיקום הראשוני של ענני הנקודות) לא משמעותי. במידה וענני הנקודות רחוקים אחד מהשני (הן בסיבוב או בהזזה), סביר להניח שהאלגוריתם לא יתכנס או שישנה את המיקום במעט אבל עדיין יהיה רחוק מתוצאת ההתאמה הרצויה. וזה אכן מה שקרה אצלנו.

המיקום ההתחלתי של המקור ב-ETH גם מסובב וגם מוזז מה שמקשה על ICP להתכנס מאחר ומתייחס לכל הנקודות בענני הנקודות (בין 90,000-370,000 נקודות). תוצאות על הPOC עמדו על ~ 5%

## **RANSAC + ICP (A2) ([לינק לקוד](https://github.com/benyairmaor/Final_Project_DNN_Lidar/blob/master/RANSAC_ICP_Reference.py)) –**

## על-ידי שימוש בספריית [Open3D](http://www.open3d.org/docs/release/tutorial/pipelines/global_registration.html?highlight=ransac) השתמשנו באלגוריתמים RANSAC ו-ICP על כל זוג של ענני נקודות. בשלבים הבאים:

## בצענו Voxel down sample עבור כל ענן נקודות. בכדי לצמצם את מספר הנקודות בשטח מסוים (לדוגמה - במקום 50 נקודות ב10 סמ"ר נסתכל על נקודה אחת בכל 10 סמ"ר).

## חשבנו את FPFH Descriptors עבור אחד מענני הנקודות – מבטא את הקשר של הסביבה הלוקאלית של כל נקודה עבור כל נקודה.

## הרצנו RANSAC על ה-Descriptors FPFH לקבלת מטריצת טרנספורמציה. RANSAC מקרב את המקור ליעד בהתייחסות למעט נקודות ולכן לא מניבה תוצאת התאמה טובה אך התוצאה מספיק טובה כדי ש-ICP יוכל להתכנס ולקרב את ההתאמה להתאמה טובה מאוד.

## הרצנו ICP כאשר השתמשנו במצטריצת התוצאה של שלב 3 כמיקום התחלתי עבור ICP. ומאחר וההתאמה יחסית קלה ההתאמה השתפרה.

## C:\Users\Maor Ben-Yair\Desktop\Maor Ben-Yair\לימודים\שנה ד\Final_Project_Lidar_Super_Glow\Result ICP 4 7 50 150.pngתוצאות על POC:

כמו שניתן לראות הציון הנמוך ביותר התקבל על wood summer והיה 78%.

הציון הכללי על כל קבוצת ה-POC היה 88%.

**Sinkhorn + RANSAC + ICP ([לינק לקוד](https://github.com/benyairmaor/Final_Project_DNN_Lidar/blob/master/sinkhorn_ransac_icp.py)) –**

## על-ידי שימוש בספריות [Open3D](http://www.open3d.org/docs/release/tutorial/pipelines/global_registration.html?highlight=ransac) ו-[POT](https://pythonot.github.io/index.html), השתמשנו באלגוריתם Sinkhorn RANSAC ו-ICP.

## בשלבים הבאים:

## עבור כל זוג ענני נקודות אתחלנו וקטורים a(source) ו-b(target) המייצגים את משקל כל אחת מנקודות הענן (מחושב על פי 1 חלקי מספר הנקודות בענן).

## בצענו Voxel down sample עבור כל זוג מענני נקודות. על מנת לצמצם את מספר הנקודות בשטח מסוים.

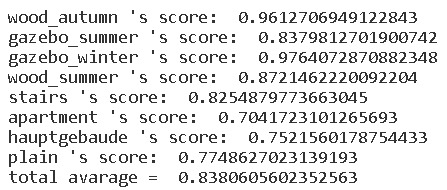
## חשבנו את ה- FPFH Descriptors עבור כל אחד מענני הנקודות בכדי לחשב עבור כל זוג ענני נקודות את מטריצת המחיר M (מחשבת מרחק בין נקודה במקור ליעד).

## על-ידי המשקלי הנקודות a ו-b ועל-ידי מטריצת המחיר M אנו מריצים רשת נוירונים בשם Sinkhorn בכדי למצוא השמה טובה ביותר וממנה לבחור את הCorrespondence set שממנה נרצה להריץ התאמת נקודות.

## עבור מטריצת המשקלים M הוספנו עמודה ושורה של אפסים. ולוקטורים a ו-b הוספנו ערך 0. ובכך יצרנו Dust Bin (עבור נקודות שאין התאמה טובה אל תתאים).

## לאחר מכן הרצנו RANSAC על הCorrespondence set לקבלת מטריצת טרנספורמציה שמקרבת את המקור ליעד בהתייחסות למעט נקודות.

## הרצנו ICP – כאשר השתמשנו בתוצאת RANSAC כאתחול לICP .



## תוצאות על POC:

כמו שניתן לראות הציון הנמוך ביותר התקבל על apartment והיה 70%.

הציון הכללי על כל קבוצת ה-POC היה 84%.

\*\* נבחין כי Sinkhorn נבחן עם voxel גבוה יותר (עקב מגבלות מקום בשל יצירת מטריצות גדולות), ובכך התייחס למספר קטן יותר של נקודות. בהמשך נבחן על מחשב יעודי וכך נדע אם Sinkhorn טוב יותר מ-RANSAC + ICP.

**Sinkhorn + SVD ([לינק לקוד](https://github.com/benyairmaor/Final_Project_DNN_Lidar/blob/master/sinkhorn_svd.py)) –**

## על-ידי שימוש בספריות [Open3D](http://www.open3d.org/docs/release/tutorial/pipelines/global_registration.html?highlight=ransac) ו-[POT](https://pythonot.github.io/index.html), השתמשנו באלגוריתימים Sinkhorn ו-SVD על מנת למצוא נקודות חשובות ביותר, כרדוקציה לבעיית ההשמה (בעית ההשמה הינה מציאת התאמה חד-חד ערכית לכל נקודה, כך שהמחיר לבחירת נקודות תואמות הוא מינימלי).

## בשלבים הבאים:

## עבור כל זוג ענני נקודות אתחלנו וקטורים a(source) ו-b(target) המייצגים את משקל כל אחת מנקודות הענן (מחושב על פי 1 חלקי מספר הנקודות בענן).

## בצענו Voxel down sample עבור כל זוג מענני נקודות. על מנת לצמצם את מספר הנקודות בשטח מסוים.

## חשבנו את ה- FPFH Descriptors עבור כל אחד מענני הנקודות בכדי לחשב עבור כל זוג ענני נקודות את מטריצת המחיר M (מחשבת מרחק בין נקודה במקור ליעד).

## עבור מטריצת המשקלים M הוספנו עמודה ושורה של אפסים. ולוקטורים a ו-b הוספנו ערך 0. ובכך יצרנו Dust Bin (עבור נקודות שאין התאמה טובה אל תתאים).

## על-ידי המשקלי הנקודות a ו-b ועל-ידי מטריצת המחיר M הרצנו את Sinkhorn בכדי למצוא השמה טובה ביותר וממנה לבחור את הCorrespondence set שממנה נרצה להריץ התאמת נקודות.

## השתמשנו באלגוריתם SVD על מנת לקבל את מטריצת הסיבוב ווקטור ההזזה.

תוצאות: בתהליך..

**נספחים**

## **תכנון הפרויקט**

|  |  |
| --- | --- |
| 17.10.21 | סיום למידה על רשתות נוירונים עמוקות |
| 12.11.21 | בחירת נושא –Lidar cloud view matching in והתחלת מחקר |
| 07.12.21 | סיום למידה על אלגוריתמים של Lidar |
| 09.01.22 | בניית References שישמשו כציון בסיס אותו נרצה לעבור |
| 28.02.22 | בניית שלד של SuperLidarGlue |
| 31.03.22 | מציאת פרמטרים טובים ל- SuperLidarGlue |
| 31.04.22 | השוואת התוצאות לתוצאות האלגוריתמים המובילים בתחום |
|  |  |
|  |  |

1. **Data -**

1. Apartment – תמונות של הדירה שנדגמה –

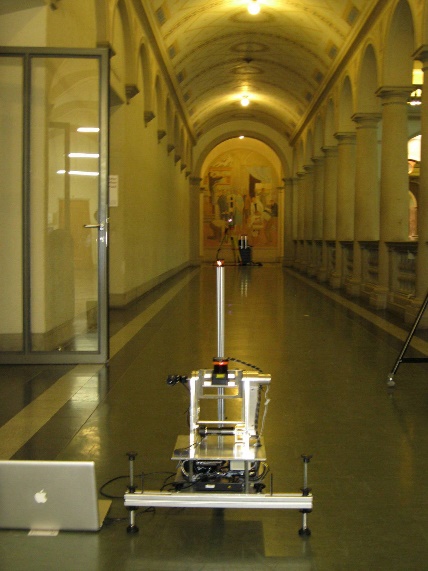


1. Gazebo Summer / Gazebo Winter – תמונה של העץ שנגדם –

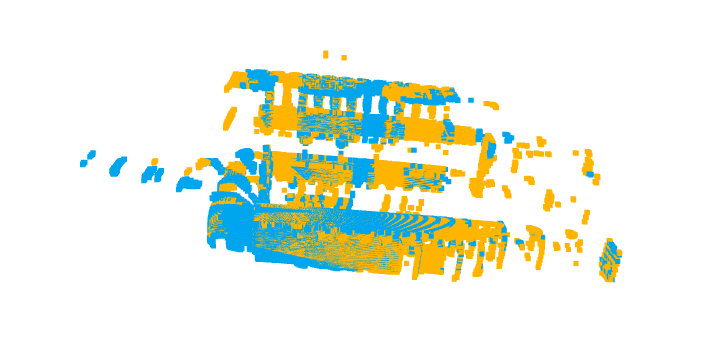


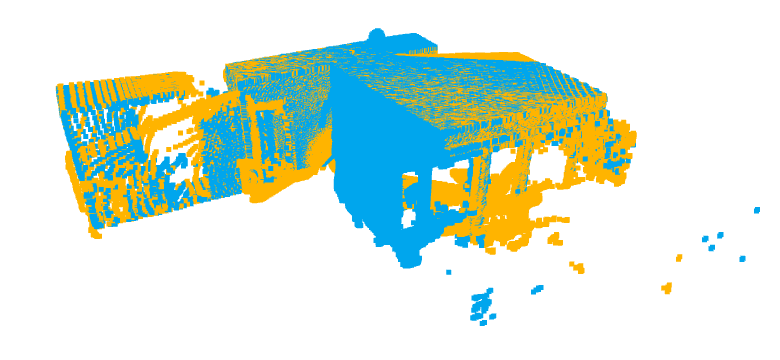
בקיץ בחורף

1. Hauptgebaude – תמונה של המבנה הנדגם



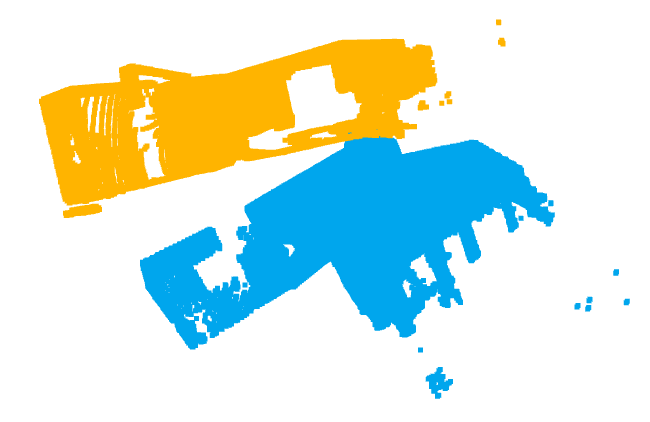
1. **דוגמאות לבעיית ההתאמה ודוגמא של אלגוריתמים כגון ICP / Sinkhorn / RANSAC –**
2. דוגמאות לתוצאות רצויות

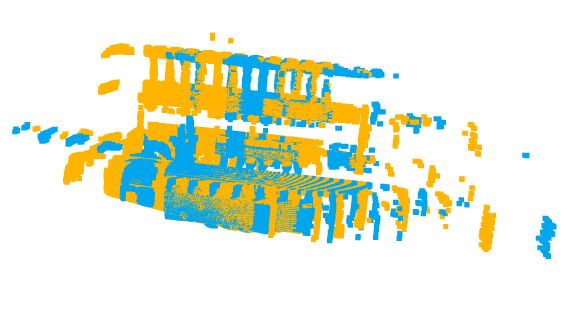


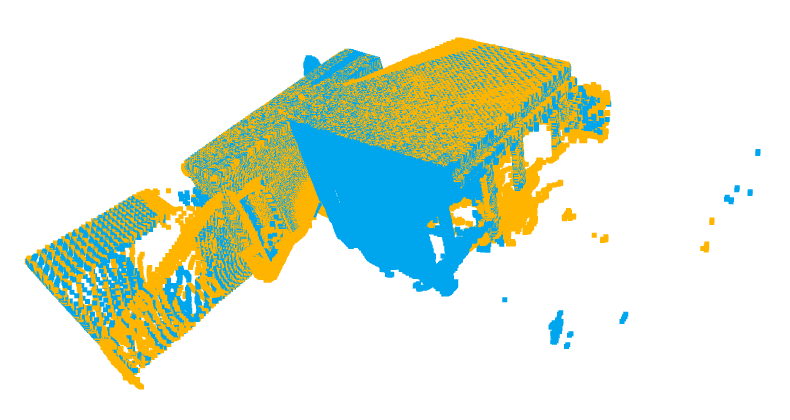


1. דוגמאות לבעיות

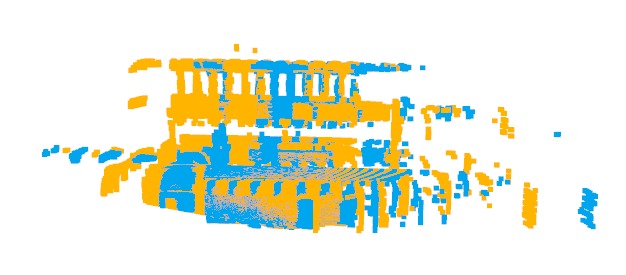




1. דוגמאות לתוצאות RANSAC + ICP



1. דוגמאות לתוצאות Sinkhorn + RANSAC + ICP



1. דוגמאות לתוצאות Sinkhorn + SVD