אפקה המכללה האקדמית להנדסה בתל-אביב

מחלקה: מדעי המחשב

שם הפרויקט : צילום למימד השלישי

Project name: photography to 3rd dimension

ספר פרויקט

מגישים:

איתי בילסקי – 208994244

סער עמיקם – 314967472

שם המנחה: ד"ר דינה גורן-בר

תאריך הגשה: 18/4/23

2. תודות:

ד"ר – דינה גורן בר על הנחיית פרויקט הגמר.

בוריס יאזמיר על הכוונה ועזרה בפרויקט זה.

Akhil James – refrence open source code.

תוכן עניינים

4. תקציר מנהלים

במסגרת פרויקט זה, פותח ומיושם מודל תלת-ממדי מבוסס על רצפי תמונות בזוויות שונות באמצעות טכניקות מתקדמות של עיבוד תמונה, הפרויקט מציע שיטה מעשית ויעילה ליצירת מודלים תלת-ממדיים מתוך רצפים של תמונות המבוססות על קטע ווידאו של אובייקט.

הפרויקט מבוסס על השילוב בין שיטות מתקדמות בתחום ראיה ממוחשבת, כמו Motion Structure from ותוכנות עיבוד קוד פתוח כמו Meshroom ו-MeshLab, כדי להשיג מודלים תלת-ממדיים היכולת להשיג דיוקים מרשימים של פחות מ-1% שגיאה מרחבית ודיוקים של 0.1-0.5 מילימטר במודלים התלת-ממדיים מעידה על הפוטנציאל של כלים ושיטות אלו ליישומים במגוון תחומים. התהליך כולל שלבים רבים כמו זיהוי נקודות עניין, חיבור תמונות, שיערוך מבנה וטריאנגולציה, ובסוף ייצוא המודל הראשוני.

הפרויקט מספק תיאור מקיף של השיטה המוצעת והיכולות שלה, ומעמיד בפני מפתחים ומשתמשים כלי עשיה וידע על מנת לבצע וליישם פרויקטים דומים בעתיד. כמו כן, הפרויקט עוסק באתגרים ובעיות שנתקלנו בהם במהלך העבודה ומציע דרכים לשיפור והתאמה של השיטה לצרכים ולתנאים השונים של פרויקטים עתידיים.

Executive Summary

This project creates a 3D model based on sequences of images at different angles, developed and implemented using advanced image processing techniques. The project offers a practical and efficient method for creating 3D models from sequences of images based on a video segment of an object.

The project is based on the combination of advanced methods in the field of computer vision, such as Structure from Motion and open source processing software such as Meshroom and MeshLab, to obtain 3D models, the ability to obtain impressive accuracies of less than 1% spatial error and accuracies of 0.1-0.5 millimeters in the 3D models shows the potential of these tools and methods for applications in a variety of fields. The process includes many steps such as identifying points of interest, connecting images, editing structure and triangulation, and finally exporting the initial model.

The project provides a comprehensive description of the proposed method and its capabilities, and provides developers and users with practical tools and knowledge in order to carry out and implement similar projects in the future. Further more the project deals with the challenges and problems we encountered during the work and offers ways to improve and adapt the method to the various needs and conditions of future projects.

5. מבוא

5.1 מוטיבציה

מוטיבציה לפרויקט זה מתחילה מהצורך להבין ולייצג באופן מדויק עצמים וסביבות תלת-ממדיות מתוך תמונות דו-ממדיות. יכולות כאלה חשובות במגוון רחב של תחומים כמו:

- רפואה
- ארכיאולוגיה
- תעשיית המשחקים והבידור
- תכנון עירוני

5.2 הגדרת הבעיה

בעיה זו מתייחסת ליצירת מודל תלת-ממדי של אובייקט מבוסס על רצף תמונות, כאשר המידע המרחבי אינו זמין או ידוע מראש. מטרת הפרויקט היא לפתח פתרון אוטומטי ויעיל שיאפשר לבצע זאת באמצעות כלים וטכניקות מתקדמות מתחום מדעי המחשב. האתגר העיקרי הוא למצוא דרך להבין ולשלב את המידע המרחבי והצבע הקיים בתמונות בצורה מדויקת ורב-ממדית, כך שיתאפשר ייצוג מלא של האובייקט במרחב התלת-ממדי.

5.3 מטרות ויעדים

בפרויקט זה, המטרות והיעדים העיקריים הם לפתח וליישם שיטות ואלגוריתמים מתקדמים ליצירת מודלים תלת-ממדיים איכותיים מתוך רצף תמונות בזוויות שונות. המטרות המפורטות כוללות:

- לחקור וללמוד את הטכניקות והכלים הקיימים בתחום תכנון תלת-ממדי.
- לפתח שיטות לשיפור ולאיחוד התיאוריות השונות וליישום אלגוריתמים מתקדמים ליצירת מודלים תלת-ממדיים מדויקים ואיכותיים.
- לשפר את היעילות של תהליכים של יצירת מודלים תלת-ממדיים על ידי קיצור זמני העיבוד וצמצום דרישות משאבים.

היעד הסופי של הפרויקט הוא להעמיק את הידע המדעי והטכני בתחום היצירה של מודלים תלת-ממדיים, ולספק כלים ושיטות מעשיות ליישום ע"י מתכננים ומפתחים בתעשייה.

6. סקירה ספרותית

תיאוריה של פיתוח מרחבי (Structure from Motion - SfM)

מחקרים רבים עוסקים בתיאוריה של פיתוח מרחבי כשיטה לשחזור תלת-ממדי של עצמים וסביבות מתמונות מרובות. טכניקת SfM מבוססות על זיהוי נקודות עניין חשובות בתמונות ושיערוך מבנה תלת-ממדי על בסיס התאמת זוגות תמונות. [1]

צילום גיאומטרי מבוסס עיסוק (Multi-View Stereo - MVS)

ישנם מחקרים המתבוננים בפיתוח של טכניקות MVS המשלימות את הפיתוח המרחבי של SfM בכדי ליצור דיוק גבוה יותר במודלים התלת-ממדיים. MVS מבוסס על שימוש בתמונות מרובות זוויות כדי לחשב מידע על מרחק וצורה של האובייקט או הסביבה בתלת-ממד.[2]

בשנים האחרונות פותחו תוכנות קוד פתוח שמשלבות תיאוריות SfM ו MVS-כגון Meshroom תכנות אלו מציעות פתרונות מקיפים ויעילים ליצירת מודלים תלת-ממדיים מתוך רצף תמונות. הם מפותחים ומתוחזקים על ידי קהילות פועלות ומגוונות של מדענים ומתכנתים, ומספקים כלים רבים לשחזור, עיבוד וניתוח של מודלים תלת-ממדיים.[3]

לאחר יצירת המודל התלת-ממדי, חשוב לבצע שיפורים ואופטימיזציות שונות על מנת להביא למודלים מדויקים ואיכותיים יותר. תוכנות קוד פתוח כגון MeshLab מספקות מגוון רחב של כלים לעיבוד ושיפור מודלים תלת-ממדיים, כולל הסרת רעשים, החלקה, פוליגון-רדוקציה ושיפור צבעים וטקסטורות.

מקורות מידע:

[1] Schönberger, J. L., & Frahm, J. M. (2016).

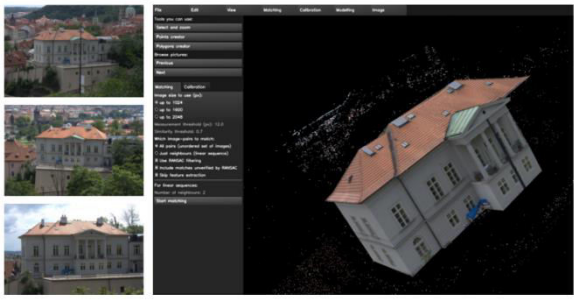
[2] Snavely, N., Seitz, S. M., & Szeliski, R. (2006).

[3] Schönberger, J. L., Zheng, E., Pollefeys, M., & Frahm, J. M. (2016)

7. סקר שוק

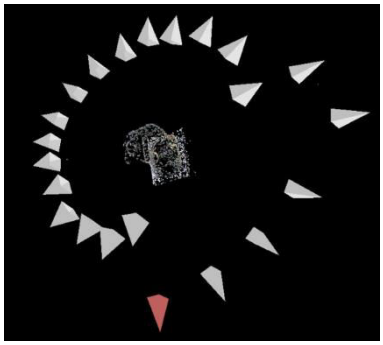
insight3d:

- You give it a series of photos of a real scene (e.g., of a building), it automatically matches



them and then calculates positions in space from which each photo has been taken (plus camera's optical parameters) along with a 3D pointcloud of the scene

- https://insight3d.sourceforge.net/



NVIDIA NeRF:

- use neural networks to represent and render realistic 3D scenes based on an input collection of 2D images.

- https://blogs.nvidia.com/blog/2022/03/25/instant-nerf-research-3d-ai/

Einscan:

- Easiest 3D scanning experience for non-technical users
- Dual scan modes: Auto Scan and Fixed Scan
- Easy to operate and great price-to-performance ratio
- https://www.einscan.com/einscan-se/

Matterandform:

- The Matter and Form Desktop 3D Scanner captures high resolution, amazing looking scans for an amazing market leading price. Its sleek, foldable design means it doesn't take up space on your desk.
- Get up to 0.1 mm accuracy with the precision of these eye-safe red lasers.
- https://matterandform.net/

	SCAN ACCURACY	SCAN DURATION	PRICE
insight3d	Low	Slow	Free
NVIDIA NeRF	High	Fast	Free
Einscan	High	Fast	999$
Matterandfcorm	High	Fast	749$
Our project	Medium - High	Medium(depends on hardware)	Free

8. חלופות

מספר חלופות אפשריות למימוש הפרויקט, בכל אחת מהן נציג יתרונות וחסרונות משלה.

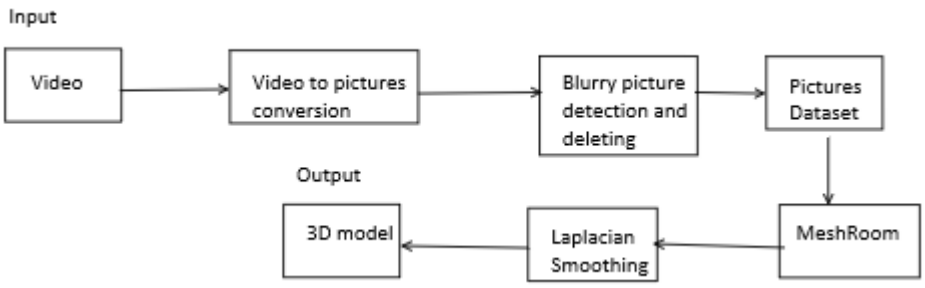
- COLMAP - כלי לבניית מודלים תלת-ממדיים המבוסס גם על SfM ו – MVS (בדומה לMeshroom), יתרונותיו הם קוד פתוח, תמיכה במגוון רחב של פלטפורמות ויכולות תכנות מתקדמות. חסרונותיו הם ממשק משתמש מורכב יותר וזמן עיבוד ארוך במקרים מסוימים.

- Agisoft Metashape - תוכנה מסחרית המאפשר בניית מודלים תלת-ממדיים מתמונות. יתרונותיו הם מהירות עיבוד מהירה, תמיכה טובה וממשק משתמש ידידותי. החסרון הוא מחיר רישיון גבוה.

- 3DF Zephyr: כלי נוסף לבניית מודלים תלת-ממדיים מתמונות באמצעות SfM ו – MVS, יתרונותיו הם ממשק משתמש ידידותי, תמיכה בפורמטים רבים של קבצים ויכולות עיבוד מהירות. חסרונותיו גם כן, מחיר רישיון גבוה ופחות גמיש מאשר תוכנות קוד פתוח.

לסיכום, בחירת הכלי המתאים לבעיה זו תלויה במספר גורמים, כמו קצב עבודה, תקציב ודרישות תוצאה. במקרה של פרויקט זה, החלטנו להשתמש ב Meshroom עקב היתרונות של קוד פתוח, איכות התוצאות הגבוהה וקלות השימוש. עם זאת, חשוב לזכור שכלים אחרים עשויים להתאים לפרויקטים אחרים בהתאם לצרכים ולציפיות של משתמשים שונים. לכן, כדאי לשקול את כל האפשרויות ולבחון את יתרונותיהם וחסרונותיהם לפני בחירת הכלי המתאים לפרויקט מסוים.

9. ארכיטקטורה



תיאור הארכיטקטורה:

המערכת מקבלת כקלט קטע ווידאו של האוביקט, בעת הצילום, האובייקט יהיה במרכז הפריים והמצלמה תנוע מסבבו (כ 360°) ממספר זוויות שונות.

לאחר מכן נמיר את סרטון הווידאו למספר תמונות שישמו כדטא סט עליו האלגוריתם יפעל.

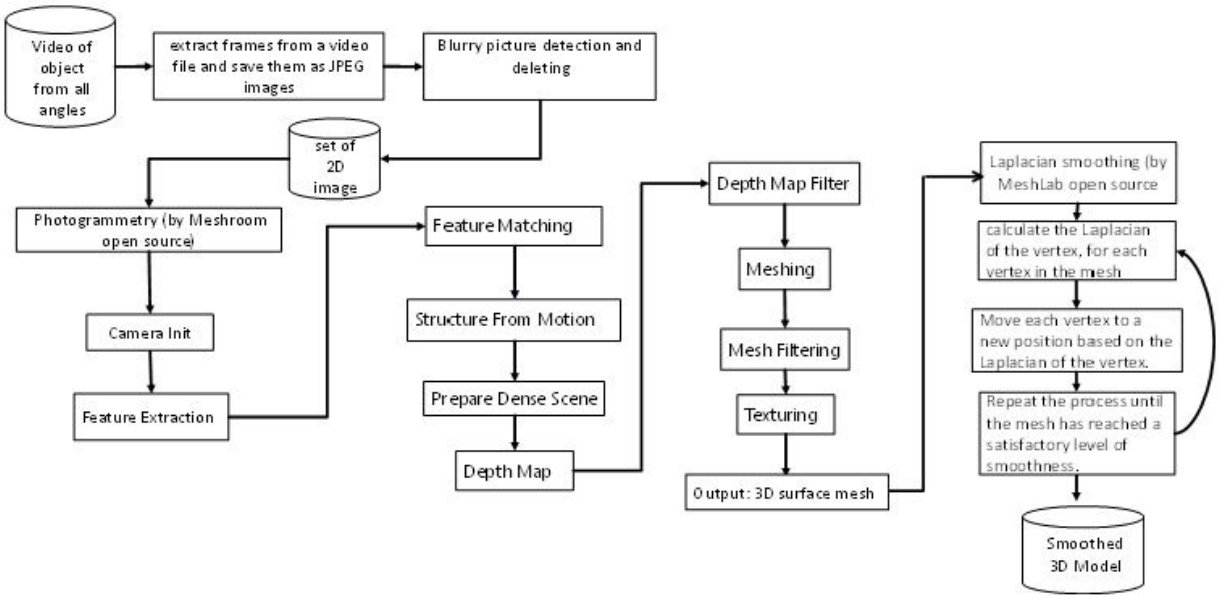
אחר כך עוברים על כל הדטא סט ובעזרת אלגוריתם openCV מזהים ומוחקים תמונות מטושטשות ומרוחות.

בשלב הבא האלגוריתם Meshroom יקרא את הדטא סט ויעבור על כל התמונות ויתחיל תהליך הכולל את השלבים הבאים :

- Camera Init
- Feature Extraction
- Feature Matching
- Structure From Motion
- Prepare Dense Scene
- Depth Map
- Depth Map Filter
- Meshing
- Mesh Filtering
- Texturing

בשלב האחרון על מנת לקבל תוצר סופי גמור אנו לוקחים את המודל המתקבל מMeshroom ומעבירים אותו לאלגוריתם בשם Laplacian smoothing העובר על המודל עצמו ומוריד רעשים לא רצויים במודל(בחלק הבא של הדוח יפורט מה זה רעשים לא רצויים)

10. תכן מפורט



11. תיאור התוצר וגרסת ALPHA

11.1 אלגוריתמים

אלגוריתם ראשון: המרה מסרטון לרצף תמונות

האלגוריתם מקבל PATH לתיקייה שבה נמצא הסרטון, לאחר מכן בודק האלגוריתם שהקובץ עצמו הוא מפורמט: .avi, .MOV, .mp4. ואם כן ידפיס הודעה שמתחיל להשתמש בקובץ בשם (שם הקובץ) אחרת ידפיס הודעה שאין קובץ בתיקייה. לאחר מכן האלגוריתם פותח את הסרטון בעזרת OpenCV.VideoCapture() ומתחיל לעבור על כל פריים בסרטון, בעזרת משתנה בשם step_size אנו יכולים להגדיר לאלגוריתם כל כמה פריימים לשמור בשביל הדטא סט.

בעזרת משתנה זה נוכל לקבוע ולשנות את גודל הדטא סט ובכך נשפיע מצד אחד על איכות מודל התלת מימדי ומצד השני על זמן יצירתו.

לאחר מכן כאשר האלגוריתם הגיע לסוף הסרטון הוא יסגור אותו וישחרר אותו מהזיכרון ויספור כמה תמונות יצר וידפיס לנו.

אלגוריתם שני: מציאת תמונות מטושטשות ומחיקתן

כאן לצורך שיפור המודל התלת מימדי, מכיון שאנו גוזרים תמונות מהסרטון יכולות להיווצר תמונות מטושטשות לכן האלגוריתם עובר על הדטא סט כדי למצוא את אותן תמונות וימחק

אותן, בכך אנו משפרים את הדטא סט כדי לקבל מודל יותר מדויק. לבסוף עוברים שוב ומדפיסים כמה תמונות נשארו בדטא סט.

אלגוריתם שלישי: Meshroom

בשלב הראשוני של האלגוריתם אנו מייצרים תיקיית עבודה זמנית ובתוכה שלוש תיקיות: input, output, Meshroom. תקיית ה - Meshroom תכיל בתוכה את כל קבצי ההתקנה.

לאחר מכן מעתיקים את כל התמונות מתיקיית הדטא סט אל תוכן תיקיית העבודה input.

בשלב הבאה מפעילים את Meshroom ומגדירים לקחת את התמונות מתיקיית העבודה input ולייצא את המודל התלת מימד לתיקיית עבודה /output

לבסוף הקבצים של המודל מועתקים אל תוך תיקיית output שלנו וכל הקבצים בתיקיות העבודה נמחקים (חוץ מקבצים בתיקיית Meshroom).

להלן פירוט השלבי עבודה של Meshroom:

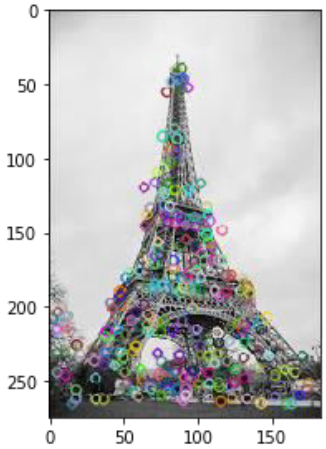
1. **CameraInit** – השלב הראשוני אחראי לאתחל את פרמטרי המצלמה, כגון פרמטרים פנימיים וחיצוניים, עבור כל תמונת קלט מהדטא סט.

   פרמטרים פנימיים כוללים אורך מוקד גודל חיישן תמונה והנקודה העיקרית, המגדירים כיצד המצלמה מייצגת את התמונה באופן פנימי. פרמטרים חיצוניים מתארים את המיקום והכיוון של המצלמה בחלל תלת מימד.

   השלב מעריך את פרמטרי המצלמה על סמך מטא נתונים של התמונה, כגון מידע EXIF, ומסד נתונים פנימי של חיישני מצלמה. מסד נתונים זה כולל מידע על דגמי מצלמות שונים וגדלי החיישנים שלהם. אם המטא נתונים חסרים או אינם שלמים, השלב עדיין יכול להעריך חלק מהפרמטרים באמצעות ערכי ברירת מחדל או מידע שסופק על ידי המשתמש

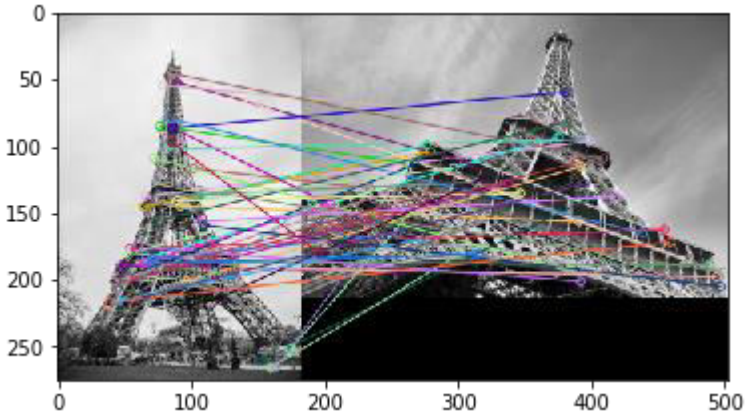
   מכיוון שאנו משתמשים בטלפון נייד כמצלמה נשתמש בערכי ברירת מחדל.

2. **FeatureExtraction** - בשלב זה, נקודות מפתח ייחודיות ומתארי התכונות התואמים להן מזוהות ומחולצות מכל תמונות הדטא סט. נקודות מפתח אלו הן נקודות ייחודיות בתמונה, כגון פינות או קצוות, שניתן להתאים באופן מהימן בין תמונות שונות במערך הנתונים.
   בשלב זה קיים שימוש בשיטת:
   SIFT (Scale-Invariant Feature Transform) כדי לבצע מיצוי תכונות כברירת מחדל. אלגוריתם SIFT חסין לשינויים בקנה המידה, הסיבוב והתאורה של התמונה, מה שהופך אותו לבחירה פופולרית לחילוץ תכונות במשימות ראייה ממוחשבת.



3. **FeatureMatching** - בשלב זה, התכונות (נקודות המפתח) שחולצו מכל תמונת הקלט במהלך השלב הקודם מותאמות על פני מספר תמונות כדי ליצור התאמה ביניהן. תכונות אלו עוזרות לזהות נקודות משותפות בתמונות שונות, המשמשות מאוחר יותר להערכת מבנה התלת-ממד של הסצנה.

Meshroom משתמש ב-Cascade Hashing Matcher, שיטה מהירה ומדויקת להתאמת תכונות תמונה. שיטה זו תואמת את מתארי SIFT שהזכרנו בשלב הקודם על ידי הערכת הדמיון שלהם ומציאת ההתאמות הטובות ביותר עבור כל נקודת מפתח. על ידי התאמת תכונות על פני תמונות, האלגוריתם יכול לקבוע אילו חלקים מהתמונות תואמים לאותן נקודות תלת מימד בסצנה.

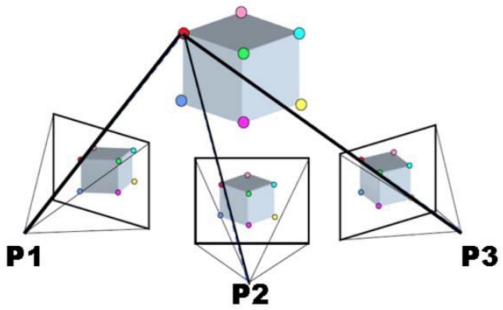


הפלט של שלב זה הוא קבוצה של התאמות זוגיות בין תמונות.

4. **StructureFromMotion (SfM)** - שלב זה אחראי להערכת מבנה התלת מימד של הסצנה וזווית המצלמה (מיקומים וכיוונים).

שלב זה הוא מכריע בתהליך השחזור התלת-ממדי, שכן הוא משחזר את הגיאומטריה התלת-ממדית של הסצנה ואת מיקומי המצלמה היחסיים. שלב ה-SfM משתמש בשילוב של טכניקות טריאנגולציה והתאמת צרור כדי לייעל את תנוחות המצלמה ואת ענן הנקודות התלת-ממדי הדליל.

טריאנגולציה היא תהליך של הערכת מיקום תלת-ממד של נקודה על ידי מדידת ההקרנה שלה בשתי תמונות או יותר. התאמת צרור היא תהליך אופטימיזציה גלובלי שמעדן את תנוחות המצלמה ואת ענן הנקודות התלת-ממדיות הדליל על ידי מזעור שגיאת ההקרנה מחדש בין תכונות התמונה שנצפו ונקודות התלת-ממד המוקרנות מחדש.



הפלט של שלב זה הוא ענן נקודות תלת מימדי דליל המייצג את מבנה הסצנה ואת תנוחות המצלמה.

5. **PrepareDenseScene** -שלב זה מכין את הנתונים הנדרשים ליצירת שחזור תלת מימדי צפוף של הסצנה, המתבצע בשלב הבא (DepthMap) .
בשלב זה תמונות הקלט, פרמטרי המצלמה וענן הנקודות התלת-ממד הדליל המתקבל משיטת SfM מומרים לפורמט מתאים ומאורגנים לתהליך הערכת העומק הבא. שלב זה יוצר קובץ ,(Multi-View Stereo) MVS. שהוא ייצוג קומפקטי של הסצנה המכיל את המידע הדרוש לשלבים הבאים.
קובץ ה-.mvs כולל את תמונות הקלט, הפרמטרים הפנימיים והחיצוניים של המצלמה התואמים, וענן הנקודות התלת-ממד הדליל.

6. **DepthMap** שלב זה מחשב מפות עומק צפופות עבור כל תמונת קלט על ידי הערכת העומק (המרחק) של כל פיקסל מהמצלמה ועד לסצינת התלת-ממד האמיתית. הוא עושה זאת על ידי ניתוח והתאמת תמונות מרובות במערך הנתונים, תוך מינוף אפקט הפרלקסה הנגרם על ידי מיקומי מצלמה שונים.
בשלב זה נעשה שימוש באלגוריתם MVS כדי להתאים את הפיקסלים על פני התמונות ולהעריך את מידע העומק של התמונה. האלגוריתם מסתמך על תמונות הקלט, פרמטרים המצלמה וענן הנקודות התלת-ממד הדליל שנוצר בשלבים הקודמים.
הפלט הוא קבוצה של מפות עומק צפופות, אחת לכל תמונת קלט. מפות עומק אלו מספקות מידע מפורט על הגיאומטריה והמבנה של הסצנה.

7. **DepthMapFilter** - בשלב זה, מפות העומק הצפופות שנוצרות בשלב הקודם מעודנות ומסוננות כדי לשפר את הדיוק והאמינות של מידע העומק של התמונה.
שלב זה מעבד את מפות העומק על ידי החלת סדרה של מסננים ובדיקות עקביות כדי להסיר חריגות, רעש וחפצים. המטרה היא לשמור רק על ערכי העומק המדויקים והאמינים ביותר עבור כל תמונת הקלט. שלב זה משווה את מפות העומק של תמונות שכנות ומאמת את ערכי העומק בהתבסס על עקביות ואילוצים גיאומטריים.
הפלט של שלב זה הוא קבוצה של מפות עומק מסוננות המספקות ייצוג מדויק ועקבי יותר של הגיאומטריה של הסצנה.

8. **Meshing** - בשלב זה, מפות העומק המסוננות מהשלב הקודם משמשות ליצירת רשת תלת מימדית צפופה המייצגת את הגיאומטריה של הסצנה. שלב ה Meshing- ממזג את מידע העומק ממספר תמונות למשטח תלת-ממדי אחד וקוהרנטי.
בשלב הזה נשתמש בטכניקה הנקראת שחזור משטח מבוסס Delaunay כדי ליצור את הרשת התלת-ממדית. אלגוריתם זה משחזר משטח תלת מימד על ידי חיבור נקודות התלת מימד הצפופות במפות העומק המסוננות תוך שמירה על העקביות הגיאומטרית של הסצנה. התוצאה היא רשת תלת-ממדית שמקרבת אל הגיאומטריה האמיתית של האובייקטים בסצנה.
הפלט של שלב זה הוא רשת תלת-ממדית צפופה בצורה של קובץ obj, שהוא פורמט קובץ תלת-ממדי נפוץ.

9. **MeshFiltering-** שלב זה מחדד את רשת התלת-ממד הצפופה שנוצרת בשלב הקודם על ידי הסרת רעש, חפצים ורכיבים קטנים מנותקים. זה גם מפשט את הרשת על ידי הפחתת המורכבות שלה, מה שהופך אותה לניתנת יותר לעיבוד נוסף. שלב זה מיישם מספר טכניקות עיבוד רשת לניקוי ואופטימיזציה של רשת התלת מימד. הפלט יהיה רשת תלת מימד נקייה ומיוצאת בפורמט קובץ obj.

10. **Texturing -** בשלב זה המידע על הצבע והטקסטורה מתמונות הקלט "מולבש" על רשת התלת מימד הנקייה והאופטימלית המתקבלת משלב הקודם, זה יוצר מודל תלת מימד ריאליסטי עם מרקם הדומה מאוד למראה של הסצנה בפועל.
שלב זה משתמש בטכניקה הנקראת Texture Atlas Generation, המשלבת ומייעלת את מידע הטקסטורה ממספר תמונות. הוא מקרין את מידע הצבע מתמונות הקלט על גבי רשת התלת מימד, תוך התחשבות בפרמטרים של המצלמה ומידע הנראות שחושב בשלבים קודמים. המטרה היא ליצור מרקם חלק הממקסם את השימוש במידע מרקם זמין וממזער תפרים או חפצים גלויים.
הפלט של שלב ה- Texturing הוא מודל תלת-ממדי בעל טקסטורה בפורמט קובץ .obj, מלווה בסט תמונות טקסטורה בפורמט קבצי .png. קבצים אלה יחד מייצגים את מודל התלת-ממד הסופי.

11. **Exporting** שלב האחרון הוא שלב היצאו של הקבצי המודל התלת מימדי לתיקיה רצויה וסיום האלגוריתם.

אלגוריתם רביעי: Laplacian smoothing

אלגוריתם המשמש לשיפור האיכות של רשת תלת מימדית על ידי הפחתת רעש וחריגות. זהו תהליך איטרטיבי שמעביר את קודקודי הרשת למיקום הממוצע של שכניהם.

להלן הסבר מפורט על אלגוריתם:

עבור כל קודקוד ברשת נמצא את הקודקודים הסמוכים לו: האלגוריתם מתחיל בזיהוי השכנים של כל קודקוד. קודקודים שכנים הם אלו שחולקים קצה עם הקודקוד המדובר.

חישוב המיקום הממוצע של הקודקודים השכנים: עבור כל קודקוד, נחשב את המיקום הממוצע של שכניו. זה נעשה על ידי סיכום המיקומים של כל הקודקודים הסמוכים וחלוקת התוצאה במספר השכנים.

מיקום הקודקוד למיקום הממוצע המחושב: הקודקוד מועבר למיקום הממוצע החדש שחושב. שלב זה מחליק את הרשת על ידי הזזת קודקודים לכיוון המיקום הממוצע של שכניהם.

איטרציה: תהליך חישוב המיקומים הממוצעים וקודקודים נעים חוזר על עצמו במשך מספר מוגדר של איטרציות (משתנה אלפא אותו אנו קובעים מראש). עם כל איטרציה, הרשת נעשית חלקה יותר, כאשר קודקודים מתקרבים למיקום הממוצע של שכניהם.

עדכון הרשת: לאחר מספר האיטרציות הרצוי, הרשת המוחלקת מתעדכנת במיקומי הקודקוד החדשים. לרשת המעודכנת הזו תהיה רעש מופחת ומשטח רגיל יותר.

הערה: אלגוריתם זה יכול להפחית בשגיאות רעש וחריגות במודל, אולם קביעת משתנה אלפא גדול מידי עלול לגרום לאובדן של תכונות ופרטים חדים.

11.2 קוד

```
import os
import cv2
from google.colab import drive   # Mount Google Drive
drive.mount('/content/drive')
currentDir = "/content/Project-3D" #work directory
```

Folder paths

```
videoInput_path = '/content/drive/MyDrive/videoInput/'#This folde
r holds the video input
frames_path = '/content/drive/MyDrive/input'#This is the input fi
le, it holds the photos for the 3D model
inputFolder = "/content/drive/MyDrive/input" #Same input folder t
hat holds the photos
outputFolder = "/content/drive/MyDrive/output" #This folder holds
 the first output of the 3D model
finalOutput = "/content/drive/MyDrive/finalOutput"#This folder ho
lds the finished 3D model after smoothing
```

To delete Data, so to create new ones

```
# Input path
count = 0
# Iterate directory
for path in os.listdir(frames_path):
    # check if current path is a file
    if os.path.isfile(os.path.join(frames_path, path)):
        count += 1
print('Input\nBefore delete\nfile count:', count)


# use os.listdir() to get a list of all the files in the director
y
files = os.listdir(frames_path)

# use a loop to delete each file in the directory
for file in files:
    file_path = os.path.join(frames_path, file)
    os.remove(file_path)

# folder path
count = 0
# Iterate directory
for path in os.listdir(frames_path):
    # check if current path is a file
```

```
    if os.path.isfile(os.path.join(frames_path, path)):
        count += 1
print('\nAfter delete\nfile count:', count)

#Output path
# folder path
count = 0
# Iterate directory
for path in os.listdir(outputFolder):
    # check if current path is a file
    if os.path.isfile(os.path.join(outputFolder, path)):
        count += 1
print('\nOutput\nBefore delete\nfile count:', count)


# use os.listdir() to get a list of all the files in the director
y
files = os.listdir(outputFolder)

# use a loop to delete each file in the directory
for file in files:
    file_path = os.path.join(outputFolder, file)
    os.remove(file_path)

# folder path
count = 0
# Iterate directory
for path in os.listdir(outputFolder):
    # check if current path is a file
    if os.path.isfile(os.path.join(outputFolder, path)):
        count += 1
print('\nAfter delete\nfile count:', count)

#FinalOutput path
# folder path
count = 0
# Iterate directory
for path in os.listdir(finalOutput):
    # check if current path is a file
    if os.path.isfile(os.path.join(finalOutput, path)):
        count += 1
print('\nFinalOutput\nBefore delete\nfile count:', count)


# use os.listdir() to get a list of all the files in the director
y
files = os.listdir(finalOutput)

# use a loop to delete each file in the directory
```

```
for file in files:
    file_path = os.path.join(finalOutput, file)
    os.remove(file_path)

# folder path
count = 0
# Iterate directory
for path in os.listdir(finalOutput):
    # check if current path is a file
    if os.path.isfile(os.path.join(finalOutput, path)):
        count += 1
print('\nAfter delete\nfile count:', count)
```

To convert video to photos

```
# Set the path to the video file
#video_path = '/content/drive/MyDrive/videoInput/IMG_2698.MOV'
#import os

videoInput_path = '/content/drive/MyDrive/videoInput/'

# use os.listdir() to get a list of all the files in the director
y
files = os.listdir(videoInput_path)

# use a list comprehension to filter for only video files (e.g.,
mp4, mov, avi)
video_files = [f for f in files if f.endswith(('.mp4', '.MOV', '.
avi'))]

# select the first video file in the list to use in your program
if video_files:
    file_name = video_files[0]
    video_path = os.path.join(videoInput_path, file_name)
    print(f"Using video file: {video_files}")
else:
    print("No video files found in the directory.")


# Set the path to the directory where the frames will be saved
#frames_path = '/content/drive/MyDrive/input'

# Create the directory for the frames
if not os.path.exists(frames_path):
    os.makedirs(frames_path)

# Open the video file using OpenCV
cap = cv2.VideoCapture(video_path)
```

```
# Initialize frame count and step size
count = 0
step_size = 5

# Loop through each frame and save as an image file
while cap.isOpened():
    # Read the current frame
    ret, frame = cap.read()

    # If the frame is not valid, break out of the loop
    if not ret:
        break

    # Save the current frame as a JPEG image file every step_size
 frames
    if count % step_size == 0:
        frame_path = os.path.join(frames_path, f'frame_{count}.jp
g')
        cv2.imwrite(frame_path, frame)

    # Increment frame count
    count += 1

# Release the video capture object
cap.release()

count = 0
# Iterate directory
for path in os.listdir(frames_path):
    # check if current path is a file
    if os.path.isfile(os.path.join(frames_path, path)):
        count += 1
print('File count:', count)
```

Checks for blurry and smudge photos, and deletes them.

```
path = '/content/drive/MyDrive/input'
threshold = 100 # adjust threshold as needed

for filename in os.listdir(path):
    img = cv2.imread(os.path.join(path, filename))
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    laplacian = cv2.Laplacian(gray, cv2.CV_64F).var()
    if laplacian < threshold:
        os.remove(os.path.join(path, filename))
```

```
# folder path
count = 0
# Iterate directory
for path in os.listdir(frames_path):
    # check if current path is a file
    if os.path.isfile(os.path.join(frames_path, path)):
        count += 1
print('photo count check:', count)
```

To create 3d model

```
import pathlib  #to check the path and verify if the required SW
is allready installed


print("\nLaunching 3D-
Reconstruction application on Google colab\n")

# Folder paths
print("Locating input images in google drive...\n")
#inputFolder = "/content/drive/MyDrive/input" # Folder in Google
drive with input images
#outputFolder = "/content/drive/MyDrive/output" # Folder in Googl
e drive where output will be finally store/moved
inputFiles = currentDir + "/input" # folder in work directory wit
h the input images
outputFiles = currentDir + "/output" # folder in work directory w
ith output files
MeshroomFolder = currentDir + "/Meshroom" # folder in work direct
ory with all Meshroom files

# Creating all directories
!mkdir -v $currentDir # will be skipped if allready existing
#inputFolder should allready be created in google drive and shoul
d contain images used for 3D reconstruction
!mkdir -v $outputFolder  # will be skipped if allready existing
!mkdir -
v $inputFiles #temporary folder in work directory to store input
images. Should be deleted once 3D is made.
!mkdir -
v $outputFiles #temporary folder in work directory to store 3D ou
tput files. Should be deleted once 3D output is moved to google d
rive folder.
!mkdir -v $MeshroomFolder # will be skipped if allready existing

# Go to work directory
%cd $currentDir
```

```
# Make copy of input image files from the google drive folder to
the input folder of work directory
print("\nCopying input images from google drive to work directory
...\n")
%cp -arv $inputFolder/* $inputFiles

# Check if Meshroom is allready installed. If not, download and i
nstall Meshroom
Meshroom_files = pathlib.Path("/content/Project-
3D/Meshroom/Meshroom-2019.2.0")

if not Meshroom_files.exists ():
  print("\nMeshroom not found.\nInstalling Meshroom...")
  !wget -
N https://github.com/alicevision/Meshroom/releases/download/v2019
.2.0/Meshroom-2019.2.0-linux.tar.gz
  !tar xzf Meshroom-2019.2.0-linux.tar.gz -C $MeshroomFolder
  %mv -v /content/Project-3D/Meshroom/Meshroom-
2019.2.0/* $MeshroomFolder
else :
   print ("\nMeshroom is allredy installed. Skipping a new instal
lation...")

# Execute Meshroom
startMeshroom = MeshroomFolder+"/Meshroom_photogrammetry"
!$startMeshroom --input $inputFiles --output $outputFiles

# Copy Output from work directory to the output folder in google
drive
print("\nMoving output files to google drive...\n")
%cp -arv $outputFiles/* $outputFolder

# Remove the input and output folder in work directory
print("\nRemoving all temporary files...\n")
%rm -Rv $inputFiles
%rm -Rv $outputFiles

print('##############################################################
##################################################################
##################################################################
#############')
```

```
!pip install pymeshlab
import pymeshlab as ml

input_obj_file = "/content/drive/MyDrive/output/texturedMesh.obj"
```

```
ms = ml.MeshSet()
ms.load_new_mesh(input_obj_file)
ms.apply_filter('laplacian_smooth', stepsmoothnum = 3)
#apply some other filters....

#.. and then save the script
ms.save_filter_script('/content/drive/MyDrive/MLX folder/my_scrip
t.mlx')
```

```
ms.load_new_mesh(input_obj_file)
ms.load_filter_script('/content/drive/MyDrive/MLX folder/my_scrip
t.mlx')
ms.apply_filter_script()
ms.save_current_mesh('/content/drive/MyDrive/finalOutput/textured
Mesh.obj')
```

קישור ל Google Colab :

https://colab.research.google.com/drive/1Dtbf6VlwCF6P320m6j6cpSpkdWIjqfgr?usp=sharing

11.3 הדגמה

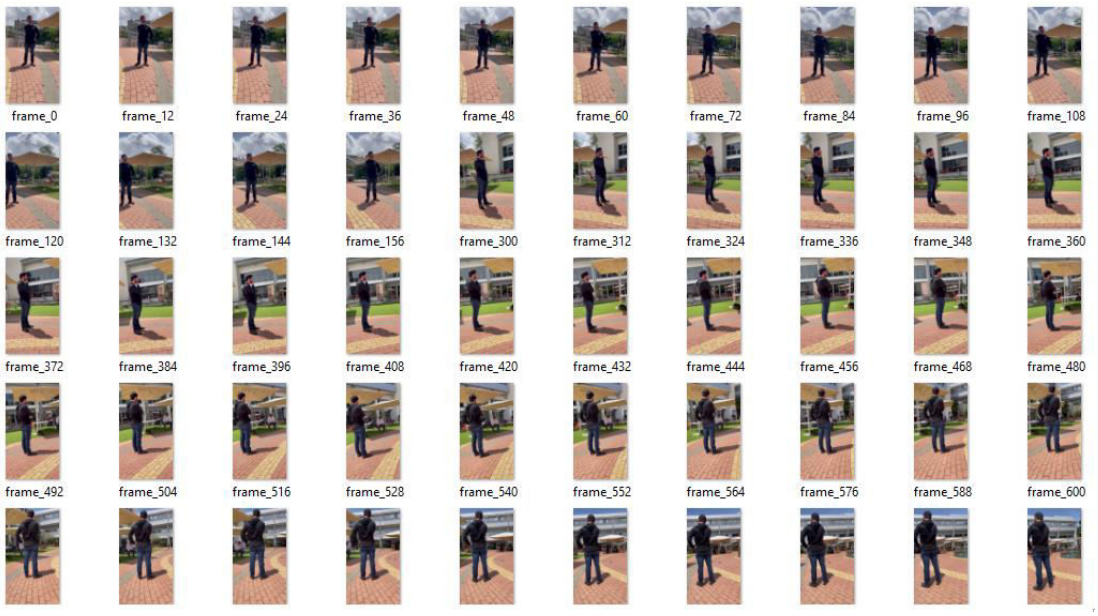
מצורף סרטון הדגמה

12. הערכה

12.1 DATASET

בפרויקט זה הדטא סט אינו קבוע, לכל מודל תלת מימדי רצוי יש דטא סט ייחודי הנוצר מסרטון ומומר לרצף תמונות.

לדוגמה:



12.2/3 מדדים וצורת בדיקה

בגלל ייחודיות הפרויקט שלנו אין ברשותנו דרך למדוד את רמת הדיוק של המודל התלת מימדי ביחס לאובייקט המציאותי.

13. תוצאות

מצורף סרטון הדגמה, בנוסף להלן דוגמת מודלים שלנו (יוצרי הפרויקט):

14. סיכום ומסקנות

בפרויקט זה השגנו את כל המטרות שהצבנו לעצמנו ועוד, התוצר הסופי הוא אכן מודל תלת מימדי שנוצר מרצף תמונות של אובייקט ממספר זוויות צילום, במהלך הפרויקט שיפרנו מספר דברים:

- הוספת אופציה לצילום סרטון כאשר ממנו נגזור את רצף התמונות המבוקש שישמש כדטא סט.
- זיהוי ומחיקת תמונות מטושטשות מרצף התמונות, בכך שיפור הדטא סט והקלה על אלגוריתם בניית המודל התלת מימדי.

איכות המודל התלת מימדי שיצרנו נחשב ברמה בינונית עד גבוה (תלוי בכמות התמונות איכותן, וזמן ריצת האלגוריתם) וזאת בשל הטכנולוגיה המתפתחת והפיתוחים החדשים שנעשים עם AI.

המלצות להמשך לשיפור הפרויקט:

- שימוש במצלמה מקצועית.
- שימוש רחב במגוון פונקציות הקיימות בספרית open source MeshLab.

15. מקורות

[1]

https://aristeksystems.com/blog/features-of-3d-modeling/

[2]

https://insight3d.sourceforge.net/

[3]

https://igl.ethz.ch/projects/Laplacian-mesh-processing/Laplacian-mesh-optimization/lmo.pdf

[4]

https://hal.archives-ouvertes.fr/hal-02190840/document

[5]

https://www.cse.iitd.ac.in/~mcs112609/poission.pdf

[6]

https://github.com/alicevision/Meshroom

[7]

https://www.meshlab.net/

[8] Schönberger, J. L., & Frahm, J. M.(2016) .

[9] Snavely, N., Seitz, S. M., & Szeliski, R.(2006) .

[10] Schönberger, J. L., Zheng, E., Pollefeys, M., & Frahm, J. M(2016) .