# תרגיל בית 2 – ראייה ממוחשבת

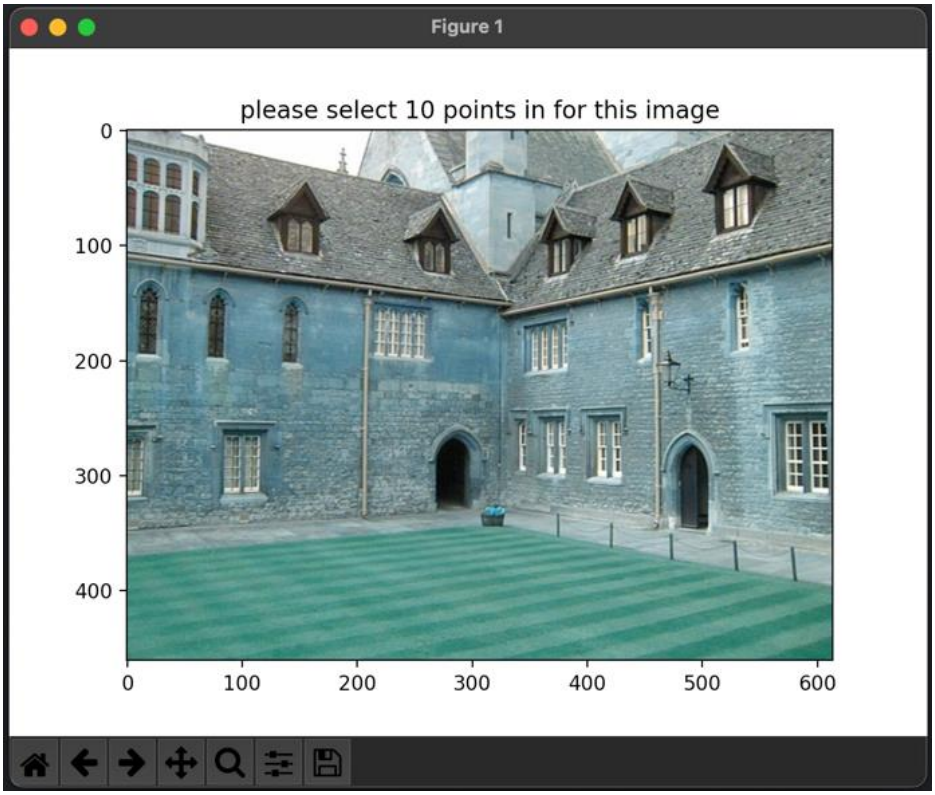

**כרים גבארין - 211406343**

**מאלק אגבאריה - 318585627**


## שאלה 1:

בהתחלה מימשנו פונקציה בשם getImagePts שבעזרתה אנחנו מייצרים ושומרים את הקואורדינטות של הנקודות של שני הסטים הנדרשים: סט החישוב וסט הביקורת).

להלן צילום מסך של התוצאה שמקבלים לאחר הקריאה לפונקציה הנ"ל, התוכנית עוצרת ומחכה למשתמש שיבחר את הנקודות בתמונה על ידי לחיצה על העכבר.

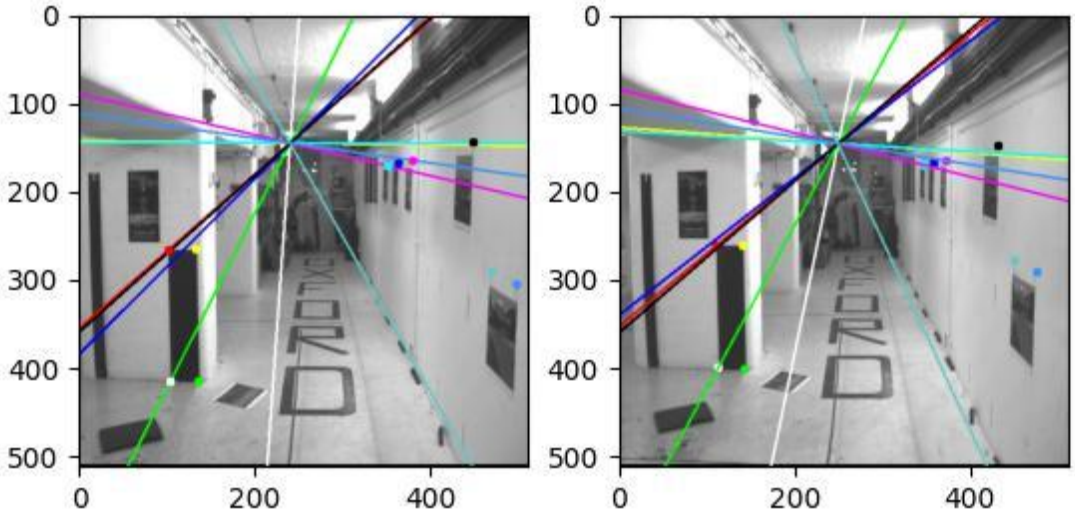



אחר כך, קראנו את הקואורדינטות של הנקודות מהדיסק, חישבנו את מטריצת ה- fundamental בעזרת סט החישוב.
בעזרת מטריצת ה- fundamental חישבנו את הישרים האפיפולריים, ואז שרטטנו אותם בעזרת OpenCV, ואז הצגנו אותם בעזרת matplotlib.

בתום העבודה על השאלה, חישבנו את ערכי ה SED עבור שני הסטים ואז הצגנו אותו בציורים, ושמרנו את התוצאות כקבצים מסוג JPG.

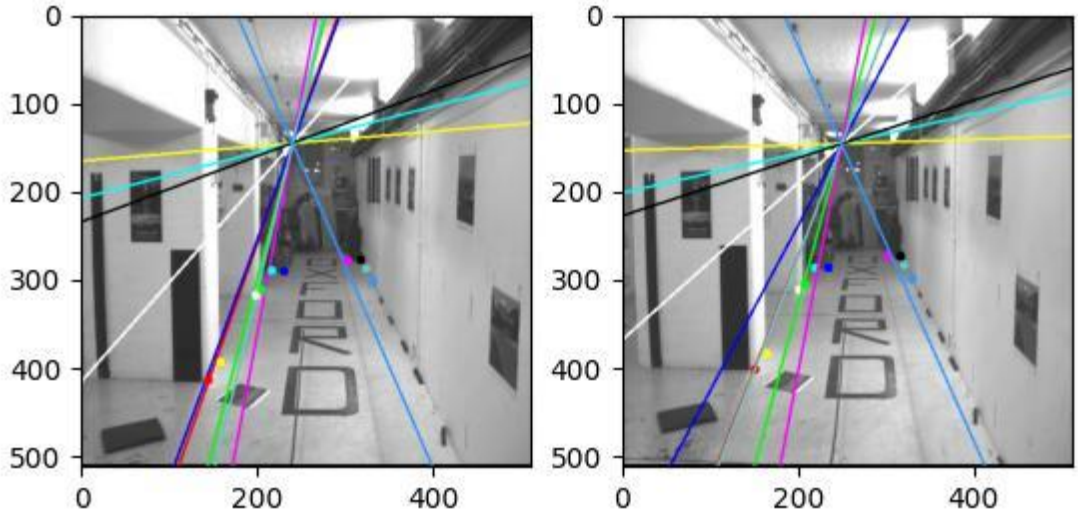
להלן צילום מסך של הישרים האפיפולריים עבור סט החישובים של המיקום הראשון:

SED = 0.07044432954144213



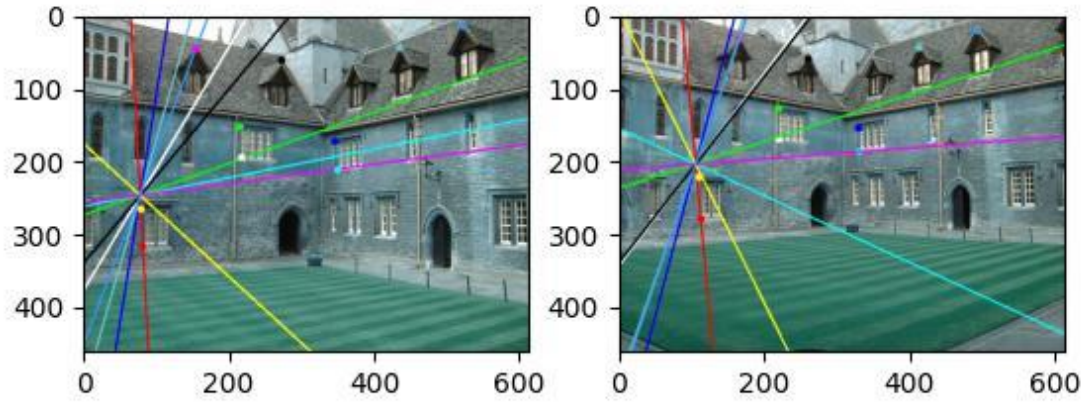
ולהלן הישרים האפיפולריים עבור סט הביקורת של המיקום הראשון:

SED = 5.894055801941706



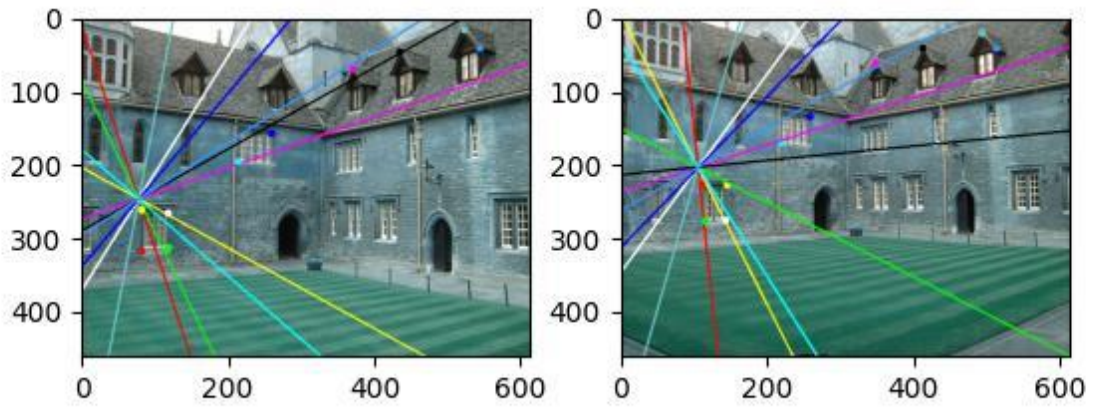
עבור המיקום הראשון, קיבלנו את התוצאות הבאות בסט החישוב:

SED = 0.5478305591852989



וקיבלנו את התוצאות הבאות באותו מיקום עבור סט הביקורת:

SED = 5.122436921908383



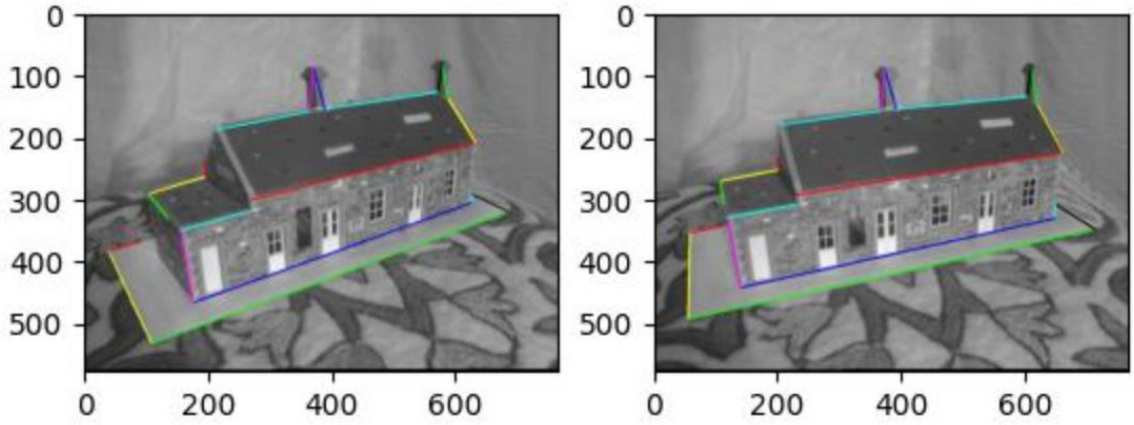

## הסבר לתוצאות:

כאשר אנחנו מחשבים את המטריצה ה- fundamental מסט החישוב, נקבל מעבר מדויק מתמונת המקור לתמונת היעד עבור סט החישוב ולהפך. ומכאן ניתן לראות שכאשר אנחנו מחשבים את ערך ה- SED על אותו סט, אנחנו מקבלים ערך קטן יחסית בגלל שיש סיכוי יותר טוב לחיתוך ואפילו התלכדות בין זוגות הקווים של שתי התמונות.

שימוש במטריצה שנוצרה עבור סט החישוב כדי לחשב מעבר של נקודות בסט אחר יכול לתת לנו דיוק בסדר אבל לא יהיה יותר טוב מהמעבר של הסט המקורי ולכן קיבלנו SED יותר גדול כשהשתמשנו במטריצה הנ"ל עבור סט הביקורת.

## שאלה 2 :

קודם כל קראנו את הדאטה מהדיסק בעזרת שתי הפונקציות read_matrix ו- get_matches, ואז שרטטנו את ההתאמות והצגנו אותם בעזרת matplotlib. להן צילום מסך של התוצאות:

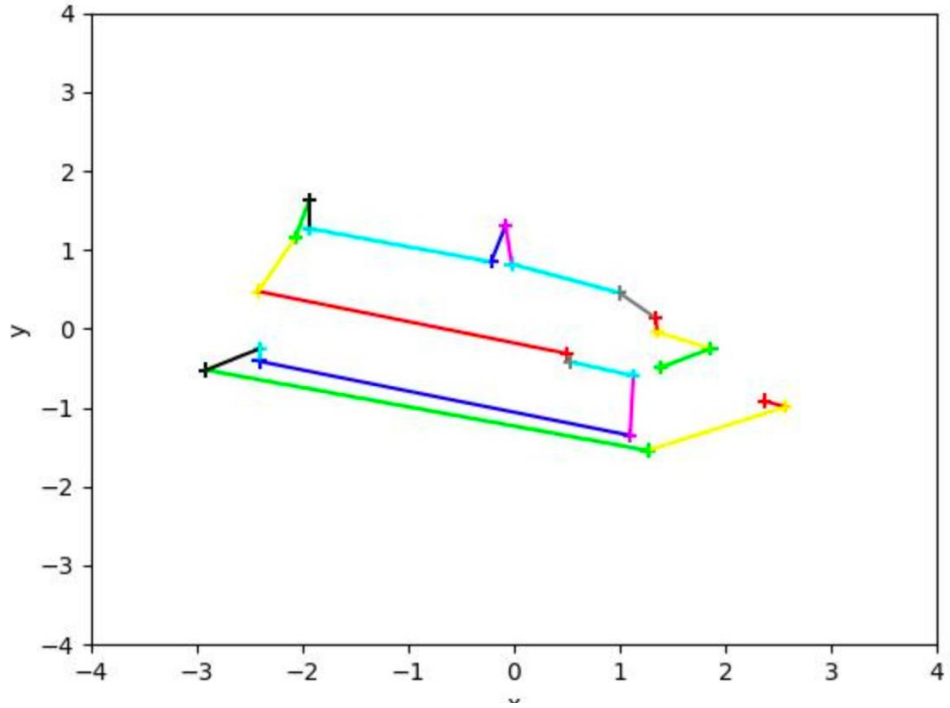



את התמונה הזאת שמרנו בקובץ JPG בשם our_connected_matches.

בהמשך העבודה, הוספנו את הפונקציה DLT שמצורפת לקובץ התרגיל לקוד שלנו. השתמשנו בפונקציה DLT כדי לקבל נקודות תלת ממדיות מהנקודות שיש לנו ביד יחד עם שתי מטריצות המצלמה שנתונות לנו בקבצי הטקסט.

אחר כך, סרקנו את הנקודות התלת ממדיות שקיבלנו וחישבנו את הממוצע שלהם שבעזרתו הצלחנו למרכז את הנקודות התלת ממדיות. ואז ציירנו את ההתאמות על הנקודות הנ"ל ושמרנו את התוצאות בקובץ: our_matches_xy_projected.jpg.

להלן צילום מסך של תוכן הקובץ our_mathces_xy_projected.jpg:




כדי ליצור את ה- gif, אנחנו צריכים מקום לשמור את התמונות שמשמשות ביצירת ה- gif. לכן, יצרנו תיקייה בשם ImagesForGIF-KareemAndMalik, שבה שמרנו את 74 התמונות שייווצרו את קובץ ה- gif.

את התמונה הראשונה בתיקייה קיבלנו מביצוע רוטציה אקראית על הנקודות התלת ממדיות בעזרת הפונקציה get_random_rotation_matrix.

ואז בעזרת שתי הלולאות שמתוארות בקובץ התרגיל, יצרנו את 73 התמונות האחרות ושמרנו אותם כקבצי JPG באותה תיקייה שיצרנו.

בסוף, יצרנו את קובץ ה- gif בעזרת הפונקציה imageio.mimsave, ושמרנו אותו בתיקיית העבודה תחת השם our_reconstruction.gif.

**<u>הערה:</u>** לא צריך ליצור תיקייה כלשהי באופן ידני, הקוד מטפל ביצירת התיקייה בתוך- Path של קובץ הקוד.