הפקולטה להנדסה ע"ש איבי ואלדר פליישמן
אוניברסיטת תל אביב

בי"ס להנדסת חשמל

פרויקט מס' 22-1-1-2456

ספר פרויקט

שם הפרויקט: פרויקט ייצוב קרן לייזר

מבצעים:

שם: אלמוג אשר, ת"ז: 322417916

שם: בר זהבי, ת"ז: 323014894

מנחה: חן כהן

מנחה נוסף: עופר כפיר

מיקום ביצוע הפרויקט: מעבדת לייזרים בבניין

ברודקום (המעבדה של עופר כפיר)

תקציר עבודה

כדי לייצב קרן לייזר נדרש לדעת את המיקום שלה בנקודת ייחוס, וכן את הזווית בה היא מתקדמת. אלו הן 4 דרגות חופש, שתיים בציר האופקי ושתיים באנכי.

השיטה כיום לאפיין את פרמטרים אלו של קרן לייזר, וכך גם לייצב אותה ע״י מראה ממונעת, היא שימוש בעדשות ומצלמות: מצלמה אחת מצלמת את הקרן כדי לקבוע את מיקומה. המצלמה הנוספת מקליטה את הפוקוס של הקרן לאחר עדשה, אשר נושא מידע על הזווית.

בפרוייקט זה ניתן היה להשתמש בתווך מפזר, דוגמת נייר לבן, חלון שירותים, ספוג או קיר, במקום עדשות. מכיוון שהפיזור יהיה תלוי במיקום הקרן וכן בזוויתה, שילובו עם מצלמה בודדת יאפשר את אפיון שני הפרמטרים האלו של הקרן על מצלמה יחידה, לאחר כיול או אימון המערכת.

הפיזור המתקבל מהקרן נקלט ע״י מצלמה של 12MP עם סנסור מסוג Sony Imx477 שמחוברת לבקר Raspberry-pi.

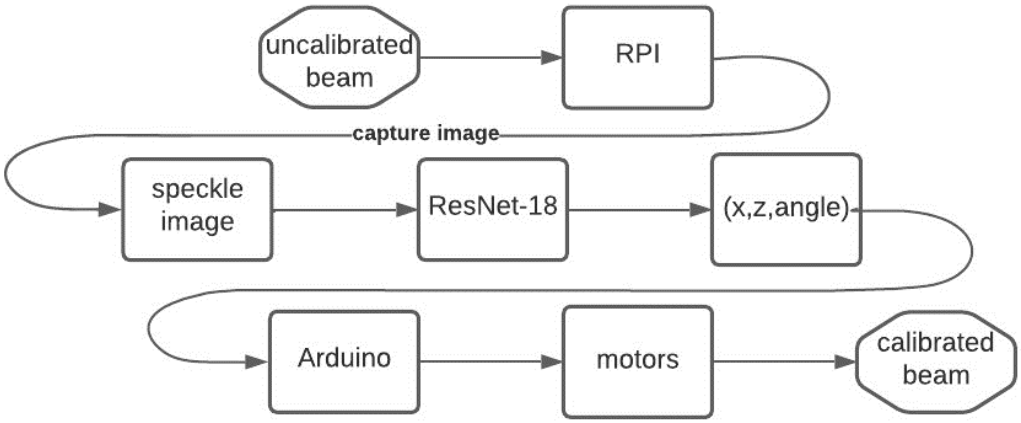
הבסיס התיאורטי לפרויקט הוא שיטה שעדיין לא מיצו את כל הפוטנציאל שלה והיא שימוש בתמונות הפיזורים של לייזר (speckle-images) על מנת ללמוד מאפיינים הן על הקרן והן על התווך המפזר.

השימוש בתמונות פיזורים כיום נעשה בעיקר בתהליכי ניטור[i]- למשל בניטור תהליך התכה של מתכות בהם רוצים לאפיין תכונות גיאומטריות של המתכת והרכב המתכת. או למשל בתהליכים ביו טכנולוגיים בהם יש לנטר את הביו-מסה של האורגניזם.

אנחנו מנסים לנצל את המידע הטמון בפיזורים למטרות אחרות – שיערוך של מיקום הקרן.

תמונות הפיזורים שמתקבלות על המצלמה שלנו משמשות כקלט לרשת למידה עמוקה מסוג Res-Net18, שמוציאה כפלט את החיזוי של הזוויות והמיקום המרחבי שממנו נורתה הקרן. החיזוי משמש אותנו למעגל משוב, שמסתמך על פלט הרשת על מנת להחזיר את קרן הלייזר לראשית הצירים.

הן הזווית והן המיקום המרחבי נשלטים ע״י מנועי צעד שמחוברים לבקר מסוג Arduino, שמאפשרים לנו איסוף נוח של תמונות ומעגל משוב אוטומטי שמחזיר את הקרן לראשית.
בפרויקט הוכחנו שאכן ניתן לכייל את הקרן באמצעות תמונות הפיזורים, לאחר שאימון על מאות קואורדינטות שונות עם אלפי דוגמאות קלט הביא את המודל לשגיאה ריבועית ממוצעת שמתקרבת לאפס הן עבור קבוצת האימון והן עבור קבוצת טסט.



איור 1 - דיאגרמת בלוקים של תהליך הקליברציה

תוכן עניינים

טבלת איורים

הקדמה

פרויקט זה עוסק בכיול של קרן לייזר.
מטרת העל של הפרויקט שלנו היא להצליח לבצע כיול של קרן לייזר אך ורק על סמך תמונות פיזורים של הקרן, שהן תמונות של הקרן שמתקבלות לאחר מעבר שלה דרך תווך מפזר שאינו שקוף.
כיום תמונות פיזורים משמשות בעיקר לחיתוך וריתוך של מתכות ועוד לא מנוצלות לצרכים של שערוך מיקום, והפרויקט שלנו בא לבחון האם ניתן לנצל את המידע שטמון בתוך תמונות הפיזורים של הלייזר גם לצורך שיערוך המיקום של הקרן.
את המשימה הכללית הזו ניתן לפרק למספר תתי משימות שהיה עלינו לבצע:

- ראשית, היה עלינו להקים מערכת אופטית שבבסיסה תכלול לייזר ותווך מפזר, ותאפשר שליטה גם במיקומו המרחבי של הלייזר וגם בזווית הפגיעה.
- המשימה השנייה היא להפוך את המערכת הפיזית ממערכת ידנית למערכת אוטומטית בצורה פשוטה וזולה – מאחר והמטרה בפרויקט היא גם לפשט באופן פרקטי את הכיול הן מבחינת נוחות למשתמש והן מבחינת עלויות הכיול, בנינו בתור התחלה מערכת שניתנת לשליטה ידנית ולכן חלקיה זולים, ולא עשינו שימוש בבמות ממונעות שמחירן יכול להגיע לעשרות אלפי שקלים.
לאוטומציה של המערכת יש תפקיד חשוב בפרויקט שלנו – מערכת אוטומטית יכולה לאסוף דאטה בקלות מבלי שיהיה אפילו צורך לקחת חלק באופן פיזי באיסוף הדאטה, ומערכת אוטומטית גם יכולה לבצע את הכיול של הלייזר באופן עצמאי לאחר אימון חד פעמי של רשת קונבולוציה.
- המשימה השלישית הייתה לתכנן רשת למידה עמוקה שתתאים לבעיה שלנו הן מבחינת הגודל והן מבחינת זמן הריצה העתידי שלה.
- לבסוף, עלינו להשתמש במודל המאומן כדי ליצור מערכת שתוכל לכייל את עצמה באופן אוטומטי.

הסיבה שבגללה בחרנו להשקיע את מאמצינו בלכייל קרן לייזר היא בגלל השימושים ההולכים וגוברים בלייזרים בימינו בישומים שדורשים דיוק גבוה – עבודה עם סיליקון וחיתוך של חומרים כמו מתכות ופלסטיקים, שבהם נדרש דיוק גבוה.
מכשור רפואי שדורש דיוק גבוה, כמו למשל בניתוחים בעין שמצריכים לייזר והדיק של הלייזר בהם הוא קריטי.
מחקרים ביולוגיים, כמו למשל מחקרים של הDNA שמצריכים עבודה עם לייזר וגם הם כמובן מצריכים רזולוציות גבוהות.
כל אלו מצריכים מאיתנו להגיע לרזולוציות גבוהות מאוד בכיול לייזר והופכים את הפוטניצאל האפליקטיבי של שיטת כיול נוחה ומהירה לגדול מאוד.

בפרויקט שלנו רצינו לנצל תכונה מיוחדת של הלייזר – והיא יצירת תבניות פיזורים לאחר מעבר דרך תווך מפזר, ושכל שינוי באחת מדרגות החופש של הקרן יוצר תבנית פיזורים חדשה.
השינוי הזה בתבניות הפיזורים הביא אותנו להאמין שבשילוב עם רשתות למידה עמוקה וזיהוי והתאמת התבניות לקואורדינטות המתאימות, ניתן יהיה לכייל את קרן הלייזר ברזולוציות מאוד גבוהות בהסתמך אך ורק על תמונות הפיזורים שלה.
כיול באופן כזה הוא מאוד פשוט, שכן בשילוב עם אוטומציה של המערכת קל מאוד לתת למערכת לאסוף דאטה לבד ולאמן בעצמה מודל מבלי אפילו להיות צריכים להיות נוכחים פיזית, וכל זאת עם מצלמה אחת בלבד שתצלם את הקרן לאחר פגיעה בתווך המפזר.
זאת לעומת השיטות הקיימות כיום שמחייבות שימוש בשתי מצלמות – אחת למיקום ואחת לזווית.

בכך אנחנו מצמצמים הן את עלות הסטאפ והן את כמות המידע שצריך לעבד (פי 2 פחות מידע כי מספר הפיקסלים קטן בחצי). הרחבה על היתרונות התיאורטיים לעומת השיטות הקיימות תיעשה בתת הסעיף הבא.

רקע תיאורטי

הפרויקט שלנו בראש ובראשונה מתעסק בלייזר – או בשמו המלא,לצורך Light Amplification by Stimulated Emission of Radiation, הוא מכשיר שפולא אור מונוכרומטי וקוהרנטי. מונוכרמטי משמעותו שהלייזר פולט גלים בעלי אורך גל זהה (שאצלנו נמצאים בתחום הצבע האדום), וקוהרנטי משמעותו שאין הפרשי פאזה בין הגלים הנפלטים.

בפרויקט שלנו התמקדנו בתמונות פיזורים של הלייזר[ii] – כלומר תמונות של הקרן כפי שהן מתקבלות לאחר מעבר דרך תווך מפזר כגון נייר, צבע לבן או כל תווך שאינו שקוף וגורם לפיזור של הלייזר.

מתקבלות תמונות של מה שנראה כמו חלקיקים של הלייזר, במה שנוצר בגלל התאבכות של גלים שהיו פעם חלק מחבילת הגלים הפוגעת של קרן הלייזר וחזרו עם פאזות שונות. תיאור פשטני של גל המתפשט במרחב בכיוון x למשל יהיה $Ae^{j(kz-\omega t)}$ והפאזה שלו תהיה $\arg(A)+kz-\omega t$, כש-ω היא התדירות ו-k הוא מספר הגל שפרופורציונלי להופכי של אורך הגל. ניתן למצוא אותו ע״י $k=\frac{2\pi}{\lambda}$.

הייחודיות של תמונות הפיזורים היא בכך שכל שינוי קטן במיקום וזווית הפגיעה מובילים לשינוי בתמונת הפיזורים המתקבלת. זאת מאחר וכל נקודה בתווך המפזר תיתן לנו תבנית התאבכות שונה, בטח אם התווך הוא לא אחיד.

המטרה שלנו בפרויקט היא לנצל את התכונה הזו של תמונות הפיזורים כדי להצליח לכייל קרן לייזר ברזולוציות גבוהות אך ורק ע״י תמונת הפיזורים שלה. הנחת הבסיס היא שמאחר ותבנית הפיזורים תהיה שונה לכל נקודה, איסוף של כמות גדולה של תמונות פיזורים (הכמות כמובן תלויה ברזולוציה אליה רוצים להגיע), תוכל לאפשר שיערוך של מיקום הקרן מתוך התמונה תוך שימוש במודל למידה עמוקה, ומתוך כך גם כיול של הקרן על סמך השיערוך.

כיום השיטות לכיול הקרן אינן מסתמכות על תמונות הפיזורים שלה, אם כי על צילום הקרן עצמה[iii], פעם אחת ישירות כדי למדוד את מיקום X,Y שלה ופעם אחת לאחר פיזור מעדשה וצילום הקרן עם מצלמה נוספת לאחר הפגיעה בעדשה. באמצעות מיקום הפגיעה לאחר העדשה וידיעת התכונות של העדשה המפזרת ניתן לחשב מה הייתה זווית הפגיעה.

שיטה נוספת שכרגע עדיין בגדר מחקר בלבד הינה צילום הקרן במספר נקודות שונות וביצוע אינטרפולציה לשיערוך המיקום של הקרן[iv]

לשיטה שאנחנו יש יתרון משמעותי על פני השיטות האלה – אנחנו צריכים רק מצלמה אחת במקום שתיים. אנחנו מציעים סביבה זולה מאוד לצורך כיול הקרן – מצלמה אחת בלבד, שני בקרים פשוטים ומספר מנועי הצעד כמספר דרגות החופש שבהן רוצים להתחשב.

מעבר לעלות, החשביבות במצלמה אחת באה לידי ביטוי גם בזמן הריצה – יש פי 2 פחות מידע לעבד ולכן ניתן להשיג זמן ריצה משמעותית יותר טוב.

אנחנו גם מתבססים על שיטה פשוטה בהרבה שלמעשה לא מצריכה מאמץ כמעט בכלל – במערכת אוטומטית כמו שלנו כל שצריך לעשות הוא לתת לרשת לצלם תמונות ולאמן מודל בעצמה, ולהגיע יום למחרת כשכבר ישנה רשת קונבולוציה מאומנת שיודעת לשערך את מיקום הלייזר מתוך תמונות הפיזורים.

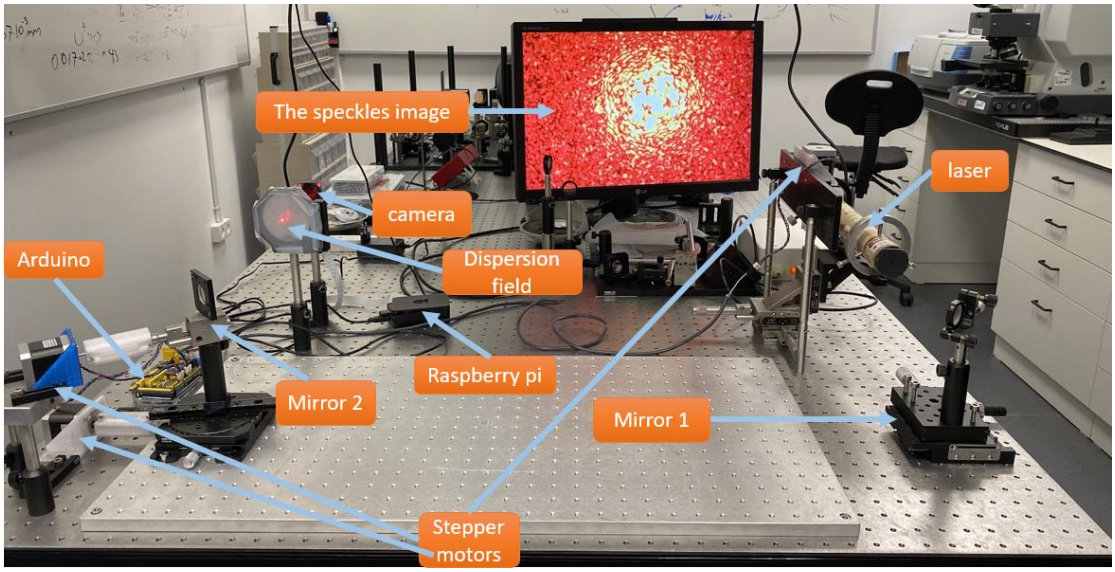
מימוש

המערכת הפיזית:

הרכיב העיקרי במערכת הוא כמובן הלייזר, לייזר אדום מדרגה B3. הלייזר יושב על במה שממנה יוצאת לשונית מסתובבת שמאפשרת לשנות את גובהו של הלייזר.

במערכת קיימות גם שתי מראות שמטרתן לאפשר לנו לשלוט גם בזוויות הפגיעה וכן במיקום על ציר X. גם בהן השליטה מתאפשרת באמצעות לשוניות שבולטות מהבמה וניתן לסובב אותן כדי לשנות את המיקום.

לבסוף ישנו תווך מפזר, במקרה שלנו ניילון שאינו לגמרי שקוף, שלאחריו מוצבת מצלמה המחוברת לבקר ראספברי-פאי ויכולה לצלם את תמונות הפיזורים.



איור 2 - המערכת הפיזית במהלך צילום תמונת פיזורים

אוטומציה והתממשקות למנועים:

כפי שניתן לראות באיור של המערכת, כל אחת מהלשוניות ששולטות בקואורדינטות ממומשקת למנועי צעד. כך ניתן לשלוט בקלות במיקום המערכת ולהשיג שתי מטרות חשובות-

הראשונה שבהן היא איסוף מהיר ואוטומטי של תמונות לאימון מודל למידה עמוקה, והשנייה היא להיות מסוגלים לבצע פידבק על סמך השיערוך של המודל ולכייל את הלייזר בצורה אוטומטית ולא ידנית.

ההתממשקות למנועי הצעד נעשתה באופן הבא:

על מנת לחבר את הזרוע של מנועי הצעד ללשוניות של הבמות, תכננו באמצעות תוכנת Fusion 360 מודל תלת מימד של מצמד גמיש.

המצמד שמידלנו בנוי כך שמצד אחד יש חור בקוטר של הזרוע של מנוע הצעד, בעוד שמהצד השני יהיה חור בקוטר של הלשונית. היתרון העיקרי של המצמד שלנו הוא שלאורך הגליל השארנו מרווח כך שנוכל להדק את המצמד על הזרוע של המנוע ועל הלשונית באמצעות בורג ואום M3 וכך להבטיח חיבור חזק של הזרוע ללשונית והשפעה מינימלית של החיכוך.

המדידות של הקטרים נעשו בצורה מדויקת ע"י קליבר אלקטרוני.

בנוסף לכל אחד מהמנועים הדפסנו סטנד שעליו הוא יושב עם חורים במיקומים המתאימים מסביב לזרוע כך שנוכל לקבע את המנוע בתוך הסטנד. הסטנד מאפשר לנו להפחית את החיכוך עם השולחן האופטי, ומאפשר לנו להניע את המנועים בתוך מסילה מוגדרת שבנינו ובכך למנוע מהם תזוזות שיצרו הפרעות וסטיות במערכת.

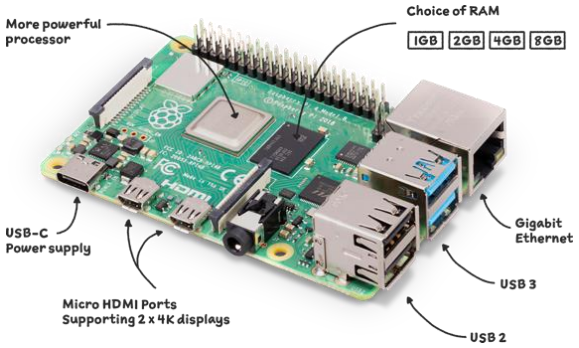
השליטה במנועים נעשתה באמצעות בקר Arduino Mega עם shield מסוג RAMPS 1.4, ועל כך נרחיב בתת הכותרת הבאה (רכיבי המערכת).

רכיבי המערכת הפיזית

ראשית נתחיל מתיאור קצר של הבקרים, יכולותיהם, האופי בו הם מתממשקים ושימושם בפרויקט זה:

Raspberry-Pi

תמונה של הבקר עליו נרחיב כעת[v]:

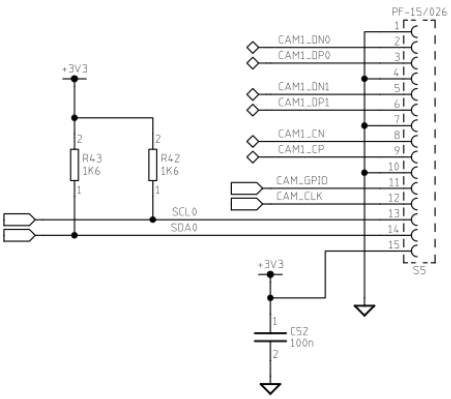


איור 3 - תמונה של בקר הראספברי-פאי 4

בקר זה הינו single board computer שפותח על ידי חברת Broadcom באנגליה, בפרויקט שלנו אנו משתמשים בסדרה 4 של בקר זה בכדי לנצל את השדרוג ביכולות העיבוד שלו מול הגרסה הקודמת והפופולארית שלו RPI 3, הRPI נוצר בכדי להיות כלי המאפשר ללמד על כל תחומי ההנדסה בדגש על תכנות ורובוטיקה.

בחרנו בו מכיוון שהייתה לו יכולת להתממשק עם חיישן Sony Imx477 (בעזרת תושבת הנמכרת עם החיישן) בו אנו משתמשים לצילום הלייזר ועליו נפרט בהמשך, בנוסף ישנה התממשקות עם חיבורי USB ועם חיבור micro-HDMI, והדבר שהכי היה חשוב לנו הייתה האפשרות להריץ סקריפטים של פייתון ועריכה שלהם בעזרת ה-IDE המותקן עליו.

חיבורי ה-CSI camera connector בהם נעשה שימוש הנמצאים על הבקר הם כמו שניתן לראות בתמונה הבאה[vi]:

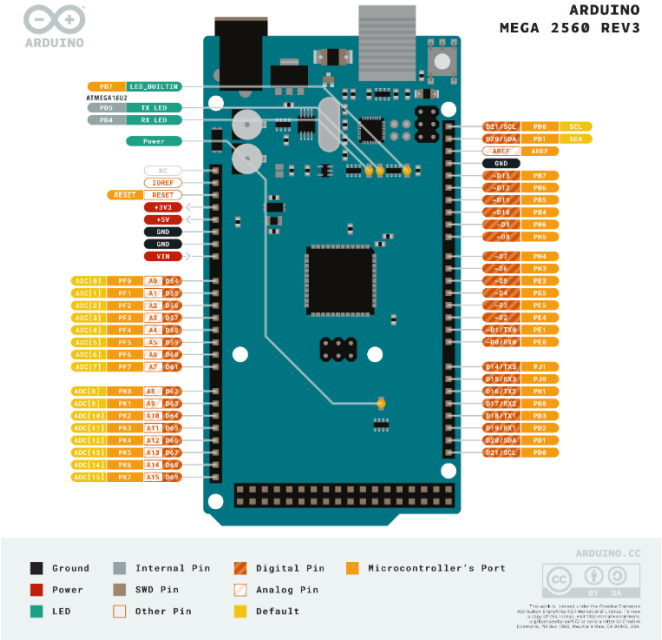


איור 4 - חיבורי המצלמה לפיני ה-GPIO של הבקר

שימוש ב-RPI 4 איפשר דרך התממשקות מה-script שנכתב בשפת python למצלמה מחד ולבקר של המערכת המכנית (ה-Arduino) מאידך, לכן הבחירה בו הייתה זו המתאימה ביותר לצרכינו בפרויקט זה, בהמשך נפרט עוד על המצלמה בחלק זה ועל אופן התקשורת בחלק המימוש התוכנתי, עוד על אופי העבודה של בקר ה-RPI ניתן לפרש מהסכמות המופיעות באתר הרשמי של ה-developers שלו[vii].

Arduino mega 2560

תמונה של הבקר עליו נפרט בהמשך[viii]:



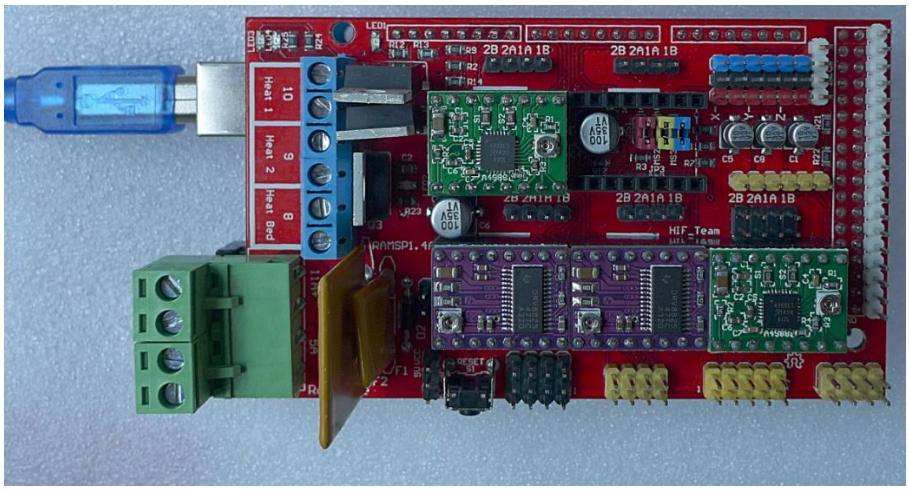
איור 5 - תמונה של בקר הארדואינו-מגה

בחירה בבקר זה הייתה שילוב של צורך בשליטה במנועים אשר להנעתם צריך להשתמש בזרם גבוהה, והעובדה ששליטה זו צריכה להתאפשר באופי תוכנתי (כדי שכל חלקי המערכת יעבדו באופן שלא מצריך עירוב של המשתמש), בפרק של התקשורת של הבקרים יוסבר על אופן התקשורת בה השתמשנו בכדי לעבור מהרצה של תוכנית ב-python לסיבוב הפיזי של המנועים, כל זה מתאפשר רק בזכות השילוב של הסיפריות המוצעות כחלק מה-IDE של בקר זה בשילוב עם ה-ports הפיזיים בהם הוא מצוייד, עוד דבר שהוביל לבחירה זו היה החיבור שלו עם ה-RAMPS 1.4 אשר איפשר ממשק בין פיני ה-I\O של הבקר ל-Drivers של מנועי הצעד.

בקר זה מאפשר שליטה תוכנתית בכל ה-I/O pins שלו, בנוסף גם ישנה אפשרות להעברת מידע סריאלית מפורט ה-USB שלו, שהיא עוד אחת מהסיבות שבחרנו דווקא בו.

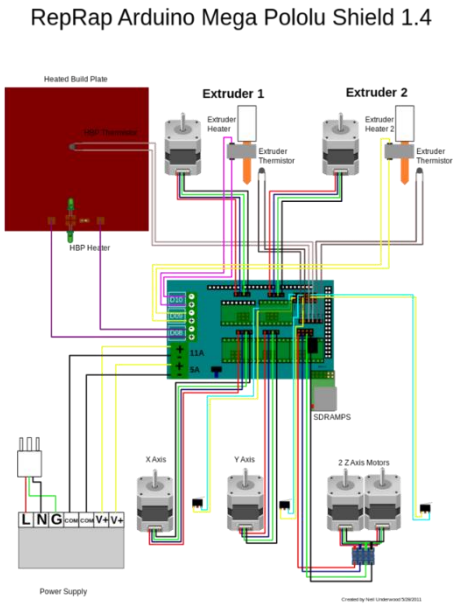
RAMPS 1.4

ה-shield עליו נפרט כעת (עם כמה drivers שונים, נשתמש ב-driver הסגול ה-DRV8825 לאפליקציה שלנו עליו נפרט בהמשך) נראה כך[ix]:



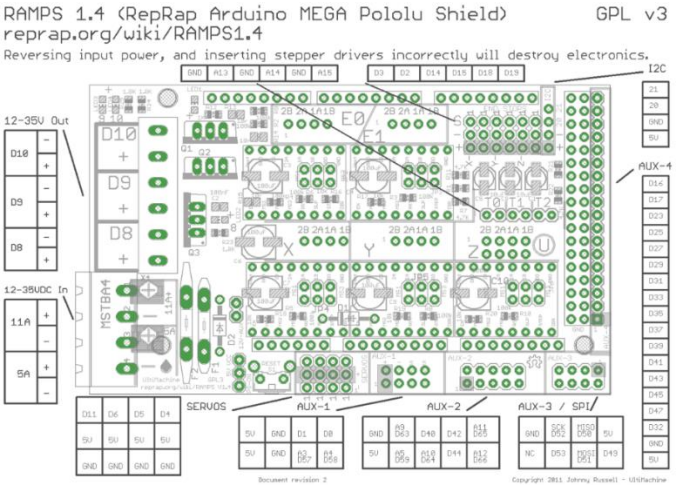
איור 6 - תמונה של ה-RAMPS 1.4

זהו אחד מה-shields המיוצרים לבקר ה-Arduino (RAMPS - RepRap Arduino Mega Pololu Shield) הוא משמש בד"כ לייצור מדפסות תלת מימד בייתיות, אך במקרה שלנו השתמשנו בו בצורה קצת שונה בשל יכולת השליטה שלו ב-5 מנועים שונים (בד"כ מחברים 2 מנועים לציר z) אופן החיבור השכיח שלו באפליקציית הדפסת התלת מימד נראה כך[x]:



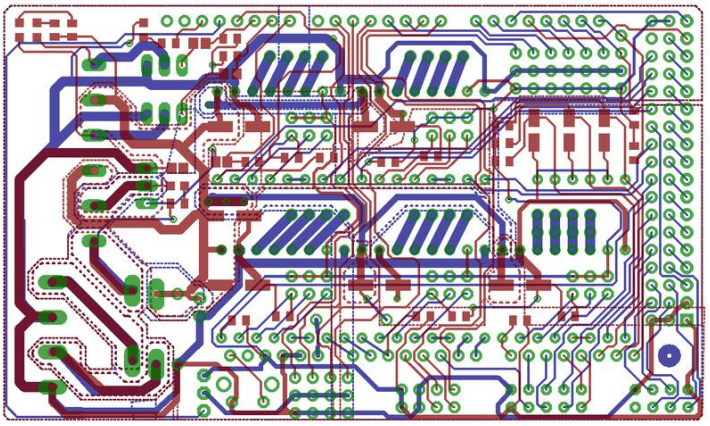
איור 7 - החיבורים הנחוצים בין הארדואינו ל-RAMPS וממנו למנועי הצעד

הסכמה שלו נראית כך:



איור 8 - סכמה של RAMPS 1.4

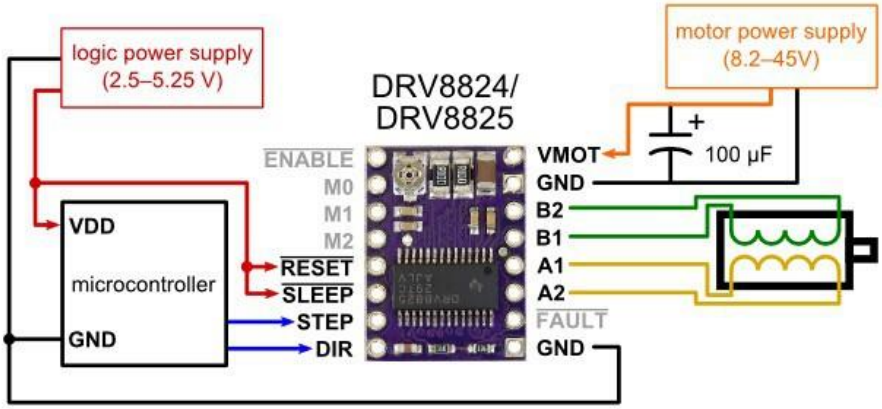
והוא מציג את אופן התממשקות ה-shield עם בקר ה-Arduino mega עם החיבורים הפנימיים שלו באופן הבא:



איור 9 - חיבורי ה-PCB של ה-RAMPS

:DRV8825

תמונה של הדרייבר[xi]:



Minimal wiring diagram for connecting a microcontroller to a DRV8824/DRV8825 stepper motor driver carrier (full-step mode).

איור 10 - הדרייבר שמניע את מנועי הצעד בקונפיגורציית חיבור בסיסית למנוע

רכיב זה נמצא על ה-RAMPS 1.4 שלנו ומהווה את הקשר בין כל ה-stepper motors שלנו לפינים המצוינים בתוכנה, ניתן לראות מן התמונה כי אנו מקבלים 2 סיגנלים בכניסת הדרייבר, הראשונה הינה כניסת ה-step אשר במקרה שלנו מציינת את כמות ה-microsteps שהדרייבר צריך לבצע והשנייה הינה DIR שהיא בעצם סיגנל בינארי המציין אם המנוע שלנו צריך להסתובב בכיוון clockwise (CW) או counter-clockwise (CCW).

ברכיב זה השתמשנו בכדי להגיע לדיוק גבוהה יותר ובכדי לחסוך צורך בבניית H-bridge חיצוני בכדי לשלוט במנועים (נרחיב על H-bridge בהמשך).

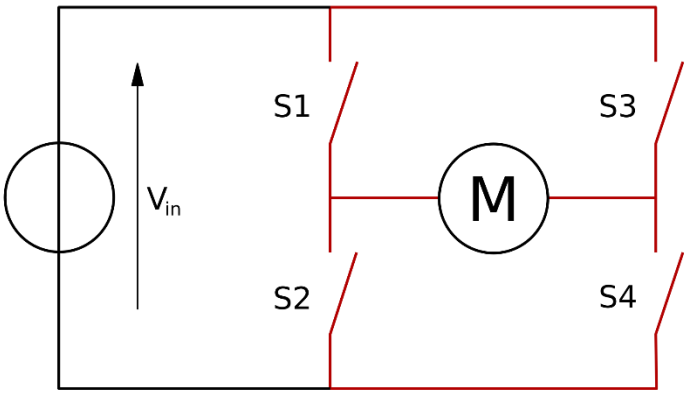
עבור כניסת ה-step הדרייבר הספציפי שלנו ניתן להגיע לרזולוציה של $\frac{1}{32}$ כלומר כל 32 microsteps המנוע שלנו יבצע step יחיד (דבר המאפשר להגיע לרזולוציות גבוהות מאוד בסיבוב הפיזי של הלשונית של הבמות האופטיות), השתמשנו בבקר זה בכדי להגיע לאחוז דיוק גבוהה מאוד בין הסיבובים, כלומר שכמות סיבובים זהה עבור DIR שונה תחזיר את המערכת למיקום הקודם שלה בדיוק המיטבי.

עבור כניסת ה-DIR ניתן לראות בשרטוט כי ה-stepper motor שלנו מחובר בקונפיגורציה של ארבעה חוטים (כי הוא בי-פולארי), הם מחולקים לקבוצות של 2 הנותנות לנו את האפשרות להסתובב בשני הכיוונים (CW וCCW), בסופו של דבר הדרייבר מחלק אותם כך שעבור כל אפשרות של DIR נקבל קבוצה שונה ובכך נקבל את השליטה המלאה על כל הטווחים שנרצה (תרגום של CW וCCW יכול להיות לעלייה/ירידה של אחד מן הצירים בעזרת החיבור ללשוניות).

דרייבר זה משתמש בצ'יפ DRV8825 של Texas-Instruments אשר מצויד ב2 H-bridge drivers ובנוסף יש לו microstepping indexer העוזר להגעה לרזולוציות המדוברות.

H-bridge:

תמונה של המעגל שנלקחה מWikipedia-[xii]:

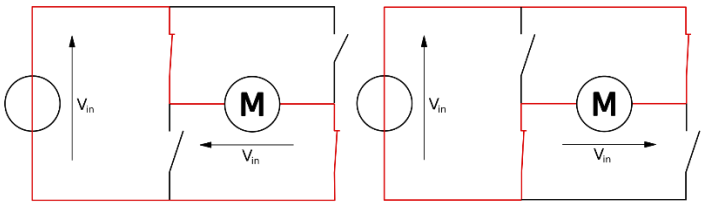


איור 11 - סכמה של מעגל H-Bridge

מעגל ה-H-bridge עובד באופן הבא המיוצג בטבלה הבאה:

mode	S1	S2	S3	S4
Motor moves right	**1**	0	0	**1**
Motor moves left	0	**1**	**1**	0
Motor coasts	0	0	0	0
	1	0	0	0
	0	**1**	0	0
	0	0	**1**	0
	0	0	0	**1**
Short circuit	0	0	**1**	**1**
	1	**1**	0	**1**
Brakes	**1**	0	**1**	0
	0	**1**	0	**1**

כלומר ניתן לראות כי את כל הפונקציונאליות שלו ניתן להשיג משלושת המצבים (Motor moves right Motor moves left ו Brakes) מהם המצבים של סיבוב שמאלה וימינה מצוינים בתמונה הבאה:



איור 12 - קונפיגוריות של H-Bridge עבור סיבובים שמאלה וימינה

מעגל זה משמש בד"כ לאספקת מתח למעגלים בי-פולאריים ועובד על העיקרון בו ניתן לסגור מעגל באיזה אחת מהדרכים שנרצה (ב2 תצורות הזרם האפשריות ב-DC במנוע) ובעזרת סיגנל דיגיטלי לשלוט בו.

מימוש פיזי של רכיב כזה הינו SN754410NE של TI[xiii].

מימוש בתוכנה

תקשורת בין בקר ה-Raspberry pi לבקר ה-Arduino

על מנת לאסוף את הדאטה שלנו באופן אוטונומי, מהיר ויעיל, מימשנו ערוץ תקשורת בין בקר הארדואינו ששולט במנועי צעד לבין בקר ה-RPI ששולט במצלמה שמצלמת את תמונות הפיזורים.

התקשורת בין שני הבקרים הינה תקשורת סיריאלית – כלומר על מנת להעביר הודעות מאחד לשני עליהם להיות מחוברים בכבל.

פיני ה-GPIO (General Purpose Input/Output) של ה-Raspberry pi מחוברים לפינים הסיריאלים שיש לארדואינו, והתקשורת נעשית בפרוטוקול UART שבו הדאטה עוברת מבקר אחד לשני בצורה של ביט-ביט.

עלינו לקבוע baud rate זהה עליו יתקשרו שני הבקרים, כשה-baud rate מייצג את כמות הביטים לשניה שעוברים בערוץ התקשורת. אנחנו בחרנו baud rate של 9600 ביטים לשניה.

מימוש התקשורת שעשינו מתבסס על עיקרון מאוד בסיסי בתקשורת שהוא ack messages (ack כקיצור ל-acknowledgement).

ראשית הראספברי-פאי מחכה להודעת ack ראשונה וחד פעמית מהארדואינו שמציינת שהוא סיים את שלב ה-setup ומוכן לשמוע ולשלוח הודעות.

לאחר מכן מבצעים עבור כל קואורדינטה שעבורה רוצים לאסוף תמונת פיזורים את התהליך הבא:

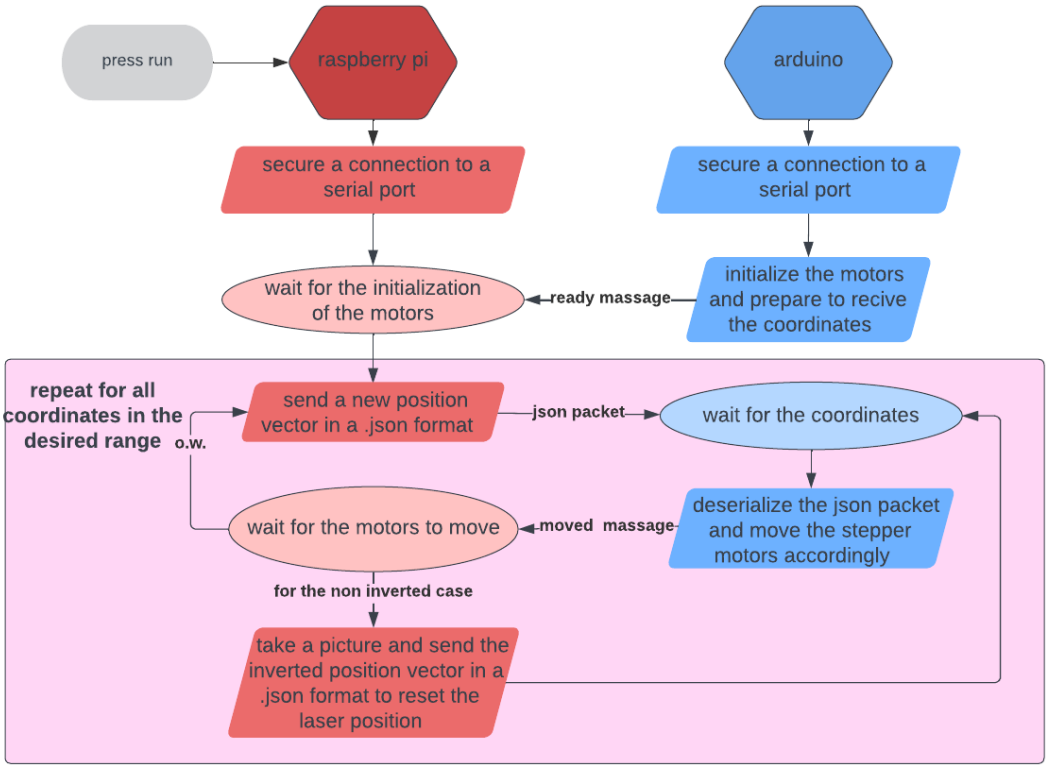
ראשית, הראספברי-פאי מקודד לקובץ json. את הקואורדינטה שבה רוצים לצלם. הוא מקודד לכל אחד מהמנועים כמה סיבובים עליו לבצע ובאיזה כיוון.

הארדואינו קולט את את ההודעה ומבצע דיסריאליזציה של קובץ ה-json ובכך מפענח כמה סיבובים כל מנוע צריך לבצע ובאיזה כיוון.

לאחר הזזת המנועים הארדואינו ישלח ack message אל הראספברי-פאי שמשמעותה שהוא סיים להזיז את המנועים למקום המבוקש והראספברי יכול לצלם תמונה. הראספברי מצלם תמונה, שומר אותה בכונן חיצוני שמיקומו הלוקאלי מוגדר ע"י המשתמש בקוד, ולאחר מכן שולח הודעה לארדואינו שהוא יכול להחזיר את המנועים למיקומם המקורי, והארדואינו שוב ישלח ack message שהוא סיים להחזיר את המנועים.

תהליך זה חוזר עבור כל נקודה שנרצה לצלם ויחזור על סט הנקודות שנבחר לכמות מסוימת של איטרציות המוגדרת בקוד (הדאטה שלנו נאספה כך שיש 30 תמונות שצולמו לכל קואורדינטה).

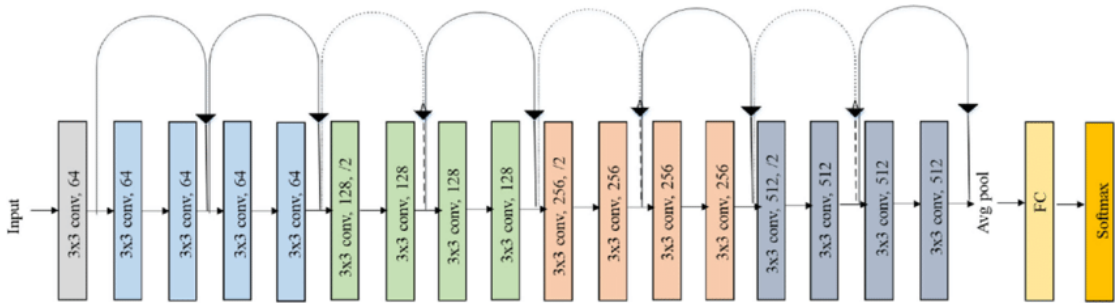
הסכמה הבאה מציגה בצורה טובה את תהליך התקשורת בין הבקרים וצילום הדאטה:



איור 13 התקשורת בין הראספברי-פאי לארדואינו בעת איסוף הדאטה

רשת למידה עמוקה:

שיערוך המיקום שלנו למעשה מתבסס לחלוטין על מודל למידה עמוקה מסוג supervised learning.
כלומר, אנחנו מכניסים למודל שלנו דוגמאות של תמונות פיזורים יחד עם הקואורדינטות המתאימות לכל תמונה, ובכך מלמדים את המודל להבחין בין הקואורדינטות השונות לפי השוני בין התמונות.
זוהי גם הסיבה שלשמה היינו צריכים כמות גדולה של דאטה – עם כל היתרונות המהותיים של רשתות למידה עמוקה והתוצאות יוצאות הדופן שניתן להוציא מהן, החיסרון המשמעותי העיקרי שלהן הוא שהן מצריכות כמות גדולה של דוגמאות קלט כדי להצליח "ללמוד".
בפרויקט שלנו בחרנו להשתמש בספריית pyTorch, שהיא ספריה מובנית ומוכרת בפייתון שמאפשרת שימוש נוח ברשתות קונבולוציה ולמידה עמוקה.
בחרנו להשתמש בארכיטקטורת Res-Net18, שבנויה כלהלן:



איור 14 - דיאגרמת בלוקים של השכבות ב-ResNet-18

ניתן לראות שהרשת מורכבת משרשרת ארוכה של שכבות קונבולוציה, שבסופן Avg pool שהיא למעשה להעביר מסנן מיצוע על הפלט של שכבות הקונבולוציה. לאחר מכן מגיע שיטוח של פלט ה-pooling ומעבר בשכבת fully-connected ולבסוף softmax.

הסיבה שבחרנו להשתמש דווקא בארכיטקטורה זו היא ראשית בגלל שהיינו צריכים ארכיטקטורה שבה תהיה כמות גדולה יחסית של פרמטרים שניתן "ללמד". תמונות הקלט שלנו הן יחסית גדולות - גודלן 1280x690x3 ובנוסף יש לנו אלפי תמונות קלט לעבוד איתן וכן מדובר בבעיית רגרסיה שמצריכה מאיתנו לשערך כמות גדולה מוד של קואורדינטות אפשריות.

ועם זאת מדובר ברשת לא מאוד גדולה יחסית לארכיטקטורות נוספות כמו ResNet50 ולכן גם תגובת ה-Real time שלה היא לא מאוד ארוכה, שזה חשוב לנו בהתחשב בעובדה שנרצה לבצע משוב בזמן אמת ע"מ לכייל את הקרן בצורה אוטומטית. זמן התגובה של הרשת כשמריצים איתה חיזוי על Raspberry pi הוא כ-0.1 שניות, כשברוב הרשתות הגדולות יותר זמן התגובה גדל פי כמה, שזה משמעותי כשמדברים בסדרי גודל של שניות.

הקלט לרשת שלנו היה התמונות המקוריות בלי שינויים כלשהם – בדיקה של תוצאת החיזוי עם טרנספורמציות שונות לתמונות הקלט העלתה ששימוש בתמונות בגודלן המקורי ובלי מניפולציות על ערכי הפיקסלים נותן את התוצאה הטובה ביותר.

כדי לעדכן את הפרמטרים ברשת, השתמשנו ב-Adam optimizer:

זהו למעשה עדכון אדפטיבי של הפרמטרים במערכת שהוא בדרך כלל יעיל יותר מאלגוריתמים ישנים כמו SGD למשל.

הרקע המתמטי לאלגוריתם[xiv]:

$$m_{t+1} \leftarrow \beta_1 m_t + (1-\beta_1)\nabla_\theta \mathcal{L}(\theta)$$ Momentum

$$v_{t+1} \leftarrow \beta_2 v_t + (1-\beta_2)\nabla_\theta \mathcal{L}(\theta)^2$$ RMS Prop

$$\hat{m}_{t+1} \leftarrow \frac{m_{t+1}}{1-\beta_1^t}$$
$$\hat{v}_{t+1} \leftarrow \frac{v_{t+1}}{1-\beta_2^t}$$ Bias Correction

$$\theta_j \leftarrow \theta_j - \frac{\epsilon}{\sqrt{v_{t+1}} + 1e^{-5}} m_{t+1}$$ RMS Prop + Momentum

Eq. 6 Bias Correction for Adam Optimizer

כאשר θ מייצג את הפרמטרים שאותם אנו רוצים לעדכן, m_t הוא מומנטום אדפטיבי שמתעדכן בכל איטרציה, v_t הוא גורם RMS שגם הוא מתעדכן, ו-β_1, β_2 הם hyperparameters שאנחנו מאתחלים אותם.

האדפטיביות שלו לעומת אלגוריתמים אחרים מאפשרת לו בדרך כלל להשיג תוצאות טובות יותר במיוחד במקרים כמו שלנו שבהם מאגר הדאטה הוא לא ענק, והוא הוכיח את עצמו גם עבור הדאטה שלנו כשנתן התכנסות מהירה יותר מאלגוריתמים אחרים.
כקריטריון השתמשנו בשגיאה ריבועית ממוצעת – MSE loss . הבחירה נעשתה מתוך התחשבות בסוג הבעיה איתה אנו מתמודדים – אנחנו למעשה מנסים לשערך מיקום, שהוא משתנה רציף, גם אם כמות הקואורדינטות שאספנו בדאטה שלנו היא על פניו מוגבלת ועברה דיסקרטיזציה. יש לה יתרון מוכח בבעיות של שיערוך מיקום, ומעבר לכך יש לה תכונה שרצינו ליישם במודל שלנו- ה"קנס" שלה הוא גדול יותר ככל שהשגיאה המרחבית (כלומר הסטייה של הקואורדינטות המשוערכות ביחס לקואורדינטות האמיתיות של אותה התמונה) גדולה יותר. ככה למעשה כשביצענו דיסקרטיזציה למיקום יכולנו "לקנוס" יותר על מקרים שבהם החיזוי של המודל חורג לקואורדינטה אחרת, ופחות על מקרים שבהם עיגול נכון של החיזוי יתן לנו את הקואורדינטה הנכונה.

את הדאטה שאספנו חילקנו באופן הבא – 80% מהתמונות שאספנו שימשו כדוגמאות לאימון, וה-20% הנותרות שימשו כדוגמאות מבחן – כלומר בכל פעם שביצענו איטרציה של אימון ועדכון פרמטרי המודל, בדקנו את הביצועים על קבוצת המבחן. בכך מנענו מהמודל להתאים את עצמו רק לדוגמאות האימון ולהיכשל במבחן על תמונות אחרות. החלוקה לאימון וטסט נעשתה באופן רנדומלי.
בנוסף בחרנו להשתמש ב-batch size של 16 – כלומר כל עדכון של המשקולות נעשה רק לאחר עיבוד של 16 תמונות בכל פעם. בכך יכולנו לנצל יתרון של המשאבים שהיו בידינו – יכולנו לאמן במקביל על 4 GPU שונים ולקיחת batch size של 16 אפשרה לנו למקבל את תהליך האימון בין ארבעתם, מעבר לכך ש-batch size גדול יותר הוא באופן כללי כלי להאצת תהליך האימון.

קצב האימון ההתחלתי שבחרנו הינו learning rate = 0.01 שהוא אמנם נראה כגדול יחסית, אבל בשל ה-batch size היחסית גדול רצינו קצב למידה גדול לאיטרציות הראשונות. בכל 8 איטרציות קצב הלמידה קטן ב-70% .

תוצאות

התוצאות שקיבלנו לאחר אימון המודל היו מעודדות למדי – ייצאנו גרף של השגיאה הממוצעת על דוגמאות האימון בכל איטרציה וכן של השגיאה על דוגמאות הטסט בכל איטרציה, וקיבלנו את הגרף הבא:

איור 15 - גרף השגיאה הריבועית הממוצעת בעת אימון המודל עבור קבוצת האימון וקבוצת הטסט

קל מאוד להבחין בכך שהשגיאה יורדת מהר מאוד לקרוב לאפס, במה שמצביע באופן ישיר על כך שהמודל מצליח לחזות בקירוב גבוה מאוד את הקואורדינטות עבור התמונות עליהן הוא מתאמן, וחשוב בהרבה – הוא מצליח לחזות נכון גם עבור דוגמאות הטסט.

השגיאה הריבועית הממוצעת ירדה עד כדי פי 100 מהשגיאה ההתחלתית.

לאחר אימון המודל, שמרנו את פרמטרי המודל המאומן (לאחר 30 איטרציות) בקובץ pth שניתן לטעון בקלות ולהשתמש בספריית pyTorch על מנת לבצע איתו חיזוי לתמונות חדשות.

התוצאות שלנו גם עונות על הדרישות שלנו בתחילת תוכנית העבודה – השאיפה הראשונית הייתה שתוך 5 איטרציות של חיזוי מיקום והזזה לפי החיזוי נצליח להביא את הקרן לראשית הצירים.

בפועל החיזוי שלנו מדויק מה שציפינו להשיג וברוב הפעמים מצליח תוך פעם אחת בלבד לחזות את הקואורדינטות הנכונות.

תוצאה נוספת מתייחסת לאוטומציה שביצענו למערכת – הצלחנו להגדיל את קצב איסוף הדאטה מכ-10 תמונות לשעה כשאספנו אותן ידנית, לכ-400 תמונות בשעה עם המערכת האוטומטית.

מסקנות

המסקנה הראשונה והחשובה שבהן – תמונות פיזורים של קרן לייזר אכן מכילות מידע שימושי על המיקום המרחבי של הקרן ועל זווית הפגיעה, וניתן ללמוד מתוך תמונת פיזורים את כל 4 דרגות החופש של הקרן.
מסקנה זו ממשיכה את המגמה המחקרית שרואה בתמונות פיזורים כלי חשוב ויעיל לפיתוח יישומים הקשורים בקרני לייזר, למחקר המשטחים דרכם הקרן מתפזרת בדגש למבנה שלהם (אפילו ברמה מיקרונית).

בנוסף, ניתן לראות כי השערוך שלנו הצליח להתבצע על כל אחת מההזזות המרחביות בנפרד, במה שמוכיח את היכולת של המודל להפריד ולהבחין בין הקואורדינטות השונות מבלי שקואורדינטה אחת תשפיע על החיזוי של אחרת.

מסקנה נוספת הינה שאוטומציה של המערכת הייתה קריטית המיוחד ליכולת שלנו לאסוף מספיק דאטה כדי שהמערכת תצליח ללמוד לשערך את המיקום בצורה מיטבית.
לפני כן קצב איסוף הדאטה היה איטי מאוד וגם לאחר ימים של איסוף דאטה ידני לא הצלחנו להתקרב להתכנסות של המודל.

מסקנה שלישית היא שמודל מסוג supervised learning הוכיח את עצמו גם בבעיית קליברציה, ושדיסקרטיזציה של המיקום אכן מובילה אותנו לתוצאות מיטביות. החיסרון היחסי בשיטה כזו היא שדיסקרטיזציה מאלצת אותנו לקבוע רזולוציה ספציפית שבה אנו מסוגלים לשערך את המיקום.
עם זאת, הצעה שכדאי לבחון בהמשך ועשויה אף לשפר את הרזולוציה אליה ניתן להגיע ואת ביצועי השיערוך תהיה שילוב של supervised learning עם מודל של reinforcement learning. הסיבה לכך נעוצה בעובדה שכשמתקרבים לנקודת האפס של הקרן, ניתן לראות בבירור את מה שפעם היה מרכז האלומה. לכן קל מאוד לעבוד בשיטת DQN ולהגדיר פונקציית Q – בסה"כ נרצה שהמודל ינסה למרכז את העיגול הבהיר שנוצר מהתפזרות של מרכז האלומה.
הבאת העיגול הבהיר למרכז שקולה למעשה לכיול הקרן, מכיוון שזה אומר שמרכז הקרן פוגע בחיישן.
הסיבה שבגללה כדאי לשלב גם supervised learning ולא רק DQN היא שלא בכל תמונה ניתן לראות בבירור סימן למרכז האלומה, וכשמתרחקים מהמרכז ניתן לראות רק תבניות של חלקיקי הפיזור. ולכן כדי לנוע בכיוון הנכון עבור נקודות רחוקות אלה כדאי לשלב גם supervised learning כדי להאיץ את תהליך ההתכנסות ולמנוע איטרציות מיותרות.
רק כאשר ניתן להבחין בבהירות יחסית שמצביעה על כך שישנו סימן למרכז האלומה בתמונת הפיזור הנוכחית, נרצה להשתמש בלמידה מחיזוקים כדי להגיע לרזולוציות גבוהות במהירות.

מסקנה נוספת נובעת מעצם מהירות ההתכנסות של המודל שלנו – מאחר ועם מאגר דאטה שבאופן יחסי אינו מאוד גדול ביחס לבעיה ונאסף בך הכל ביום אחד הצלחנו באיטרציות בודדות של אימון להגיע לשגיאה כמעט אפסית, הרי שאת הרזולוציה שבחרנו ניתן לשפר משמעותית גם מבחינה תיאורטית וגם מהבחינה הטכנית.
אנחנו בחרנו ברזולוציה של סיבוב אחד המנוע, שהיא אמנם גם כך מתבטאת בשינוי מאוד קטן במיקום הלייזר בפועל, אבל המנועים שלנו מסוגלים להגיע לרזולוציות גבוהות בהרבה – מאחר ומנוע הצעד יכול גם לבצע מיקרו-צעד, אנחנו יכולים להגיע לרזולוציות של עד כדי $\frac{1}{12800}$ סיבוב.

הצעה חשובה להמשך תהיה לבדוק לאיזו רזולוציה אנו מסוגלים להגיע ועדיין לשמור על טווח שגיאה סביר יחסית ולהצליח לאמן מודל מתכנס.

זה ככל הנראה יצריך משמעותית יותר דאטה משהספקנו לייצר ולכן מפאת קוצר זמן התפשרנו על רזולוציה יותר ריאלית שהיא עדיין יחסית טובה.

אם אכן יצליחו לגלות במחקר מעמיק יותר שניתן להגיע לרזולוציות מאוד גבוהות השימוש של הפרויקט שלנו בתעשיה ובמחקר אכן יוכל להחליף את השיטות הקיימות לקליברציה של קרני לייזר.

באפליקציות שדורשות דיוק רב כגון ייצור צ׳יפים וחיתוכם, או באמל״ח ומערכות צבאיות, השיטה שלנו שמאפשרת תגובה מהירה יותר מהשיטות הקיימות בשילוב עם רזולוציה גבוהה יכולה להיות משמעותית ופורצת דרך.

תיעוד

התיעוד לפרוייקט נעשה בעיקרו ב-GitHub . הקישור לריפו :

https://github.com/AsherAlmog/Laser_calibration_proj.git

בגיט ניתן למצוא את הסקריפט בארדואינו שבאמצעותו הנענו את מנועי הצעד, את הסקריפט בפייתון שרץ על ה-RPI ששימש לתקשורת בין הראסברי לבין הארדואינו ולצילום התמונות עם הראסברי פאי. כמו כן יש שם את הקוד שבאמצעותו אימנו את הרשת וכן את פרמטרי המודל המאומן.

כבונוס ניתן למצוא שם גם את מודלי התלת מימד ששימשו להתממשקות הסטאפ למנועים, במידה ויהיה מי שירצה לשחזר את מה שעשינו בעתיד ולהעמיק את המחקר בנושא.

פירוט יינתן בקובץ ה-README שנמצא גם הוא בגיט.

ביבליוגרפיה

[i] Laser speckle photometry – Optical sensor systems for condition and process monitoring: Lili Chen, Ulana Cikalova, Beatrice Bendjus, Andreas Gommlich, Shohag Roy Sudip, Carolin Schott, Juliane Steingroewer, Dresden, Matthias Belting, Aachen, and Stefan Kleszczynski, Duisburg, Germany

[ii] https://www.rp-photonics.com/laser_speckle.html - laser speckles, speckle images

[iii] https://www.sitechwest.com/content/uploads/2016/01/hv101-calibrationcheck.pdf - laser calibration method

[iv] https://opg.optica.org/oe/fulltext.cfm?uri=oe-28-19-27588&id=437846 – laser calibration using interpolation

[v] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ - Raspberry-pi 4 product page

[vi] https://www.raspberrypi.com/documentation/accessories/camera.html#installing-a-raspberry-pi-camera – CSI camera connector

[vii] https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-reduced-schematics.pdf?_gl=1*t6cfry*_ga*MTc1ODQ0MzgwNC4xNjY2NzY4NTM3*_ga_22FD70LWDS*MTY4NTk1Njk5MS4xLjEuMTY4NTk1NzE3OS4wLjAuMA. – Raspberry pi full schematic

[viii] https://store.arduino.cc/products/arduino-mega-2560-rev3 –Ardunio mega

[ix] https://reprap.org/wiki/RAMPS_1.4 – RAMPS 1.4

[x] https://reprap.org/wiki/File:Rampswire14.svg - Appropriate connections between Arduino, RAMPS 1.4 and stepper motors

[xi] https://reprap.org/wiki/MKS_DRV8825\ - DRV8825

[xii] https://en.wikipedia.org/wiki/H-bridge - H-Bridge

[xiii] https://html.alldatasheet.com/html-pdf/28616/TI/SN754410NE/169/6/SN754410NE.html - SN754410NE datasheet

[xiv] https://towardsdatascience.com/optimization-algorithms-in-deep-learning-191bfc2737a4 - ADAM optimizer theoretical background