מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון 972-73-3783146+ E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

חומרי לימוד בנושא כלים בלתי מאוישים

יישומים פדגוגיים בלמידה עמוקה ורובוטיקה בחינוך טכנולוגי

כתיבה:

אריאל דוברובנסקי

ד"ר שחף רוקר-יואל

ד"ר אמונה אבו-יונס עלי

ייעוץ מדעי ופדגוגי:

פרופ' יהודית דורי

ייעוץ הנדסי:

יעקב הרשקוביץ

גדי הרמן

מור-טק, אוקטובר 2023

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תוכן העניינים

מונחים **1**
1. הקדמה **3**
2. חלק א' של תוכנית ההכשרה – מבוא ורקע בסיסי בנושא יישומים פדגוגיים בלמידה עמוקה ורובוטיקה בחינוך טכנולוגי **5**
2.1 תיאור כללי 5
2.2 ראשי פרקים מרכזיים בחלק א' של תוכנית ההכשרה 5
2.3 מטלה מסכמת – אתר WIKI 7
3. סקירה לכלי התוכנה **8**
4. תיאור מפורט לנושאי הלימוד באתר WIKI **9**
4.1 MorBot - פלטפורמת רובוט חינוכית לתלמידים 9
4.2 מאפיינים 10
4.3 רכיבים עיקריים 10
4.4 מטרות לימודיות 10
4.5 Hardware Setup 12
4.5.1 מושגים מרכזיים 12
4.5.2 בדיקת תכולת הקיט 13
4.5.3 הוראות התקנת החומרה 14
4.6 Jetson Nano Linux Setup 27
4.6.1 מושגים מרכזיים 27
4.6.2 בקר Jetson Nano בגרסה חדשה של Linux 27
4.7 ROS 31
4.7.1 מושגים מרכזיים 31
4.7.2 ROS – Robot Operating System 32
4.7.3 Linux – הסבר קצר 33
4.7.4 הקדמה להתקנת ROS על מחשב שולחני 34
4.7.5 הוראות התקנת ROS Melodic על מחשב שולחני 34
4.7.6 בדיקת ההתקנה 36
4.7.7 יצירת פרויקט חדש - ROS workspace 37
4.8 ROS 101 39
4.8.1 מושגים מרכזיים 39
4.8.2 צומת - Node 40
4.8.3 Topics 41
4.8.4 הודעות - Messages 42
4.9 Creating New Packages in ROS 46
4.9.1 יצירת חבילת ROS חדשה 46
4.9.2 בניית החבילה 46

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.9.3 הוספת צמתים וקבצי הפעלה 47
4.10 ROS PubSubPy 52
4.10.1 מושגים מרכזיים 52
4.10.2 מהי חבילת ה-PUB/SUB? 53
4.10.3 דרישות קדם 53
4.10.4 תרגול בבניית מערכת ההודעות בין מפרסם למנוי למפרסם 53
4.10.5 כתיבת מספר מפרסמים ומאזינים ב-ROS 60
4.11 ROS turtlesim 64
4.11.1 מבוא 64
4.11.2 דרישות קדם 64
4.11.3 שלב 1: להתחיל turtlesim 64
4.11.4 שלב 2: שליטה בצב באמצעות Node קיים בשם turtle_teleop_key 65
4.11.5 שלב 3: חקירת topics ב-turtlesim 65
4.11.6 שלב 4: מעקב אחר מיקום הצב 66
4.11.7 שלב 5: כתיבת קוד ב-Python לשליטה על הצב 66
4.11.8 שינוי צבע הרקע של Turtlesim 68
4.11.9 הזזת הצב Turtle על סמך צבע הרקע 70
4.11.10 הפעלת מספר צבים בחלון Turtlesim אחד 72
4.11.11 ROS Services 73
4.12 ROS Workshop 75
4.12.1 מושגים מרכזיים 75
4.12.2 דרישות קדם 75
4.12.3 תרחישי עבודה 76
4.12.4 חבילת JetBot control 76
4.12.5 כתיבת הקובץ jetbot_controller.py 77
4.12.6 הוספת Teleoperate כדי לשלוט ברובוט דרך מקשי המקלדת 79
4.12.7 הוספת קובץ Launch להפעלת הפרויקט 81
4.12.8 כתיבת תוכנית פעולה לרובוט 82
4.12.9 בניית תרחיש AI driver ל-JetBot 85
4.12.10 יצירת FTL FULL Demo Launch File 92
5. סיכום 96
6. מקורות 97
7. נספחים 100
7.1 נספח א' - מדריך בנושא Linux for Beginners 100
7.1.1 מושגים מרכזיים 100
7.1.2 Linux - רקע היסטורי 100
7.1.3 הכרות עם ההפצה הפופולארית של Linux 101
7.1.4 הסבר על תיקיות ראשיות 102

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.1.5 פקודות עיקריות 104
7.1.6 ניהול הרשאות 107
7.2 נספח ב' - מדריך בנושא Multi robot 108
7.2.1 סנכרון וגילוי Master למערכת מרובת רובוטים 108
7.3 נספח ג' - מדריך בנושא Notebooks Demos 109
7.3.1 שיטה 1: שלוט ב-JetBot על ידי תכנות מדפדפן אינטרנט 109
7.3.2 שיטה 2: שליטה ב-JetBot על ידי תכנות מחברת jupyter ישירות מ-VS 109
7.4 נספח ד' - מדריך בנושא VirtualBox Tutorial for Beginners 111
7.4.1 מבוא ל-VirtualBox 111
7.4.2 הורדת התוכנה הנדרשת 111
7.4.3 יצירת מכונה וירטואלית חדשה 111
7.4.4 הקצאת זיכרון RAM 112
7.4.5 יצירת דיסק קשיח וירטואלי 112
7.4.6 התקנת Ubuntu 18.04 112
7.4.7 הגדרת הגדרות התצוגה 112
7.5 נספח ה' - מדריך בנושא Visual Studio SSH Tutorial 113
7.5.1 מושגים מרכזיים 113
7.5.2 מבוא - SSH 113
7.5.3 איך ה-SSH עובד? 113
7.5.4 אימות משתמשים על ידי SSH 114
7.5.5 יצירה ועבודה עם מפתחות SSH 115
7.5.6 יצירת זוג מפתחות SSH עם מספר גדול יותר של ביטים 117
7.5.7 הסרה או שינוי של ביטוי הסיסמה במפתח פרטי 117
7.5.8 הצגת טביעת האצבע של מפתח SSH 118
7.5.9 הוראות חיבור בסיסיות 119
7.5.10 מדריך SSH ל visual studio code 120

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון 972-73-3783146+ E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

כל הזכויות שמורות למשרד החינוך ©

מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק

הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020

הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

החוברת תצא לאור במימון האגף למדעים במזכירות הפדגוגית ומינהלת מל"מ המרכז הישראלי לחינוך מדעי טכנולוגי.

אין לשכפל, להעתיק, לצלם, להקליט, לתרגם, לאחסן במאגר מידע, לשדר או לקלוט בכל דרך או אמצעי אלקטרוני, אופטי או מכני או אחר כל חלק שהוא מהחומר שבחוברת זו. שימוש מסחרי מכל סוג שהוא בחומר הכלול בחוברת זו אסור בהחלט אלא ברשות מפורשת בכתב מהמו"ל.

הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון 972-73-3783146+ E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

מונחים

מונח	הסבר
ROS	Robot Operating System - אוסף של ספריות תוכנה ואלגוריתמים לבניית מערכות רובוטיות
מערכת הפעלה Linux	מערכת הפעלה למחשב מסוג לינוקס
Ubuntu	גרסת מערכת הפעלה לינוקס הנקראת Ubuntu (קיימות גרסאות רבות אחרות)
Visual Studio CODE	סביבת פיתוח תוכנה עם כלים רבים
SSH	פרוטוקול תקשורת בין מחשבים מבוססי לינוקס
PIP	תוסף העוזר בהתקנת ספריות פייתון
VNC	שירות שולחן עבודה מרוחק המאפשר לראות ולהשתמש במחשב מרוחק עם ממשק גרפי מלא
PID	proportional–integral–derivative controller / בקר פרופורציונלי-אינטגרלי-דיפרנציאלי, הוא מנגנון משוב עבור חוגי בקרה
Playground tensorflow	סביבת הדמיה אינטראקטיבית לעבודה עם תשתיות למידת מכונה
Transfer Learning	שיטת לימוד מחדש של מודלים שנלמדו בעזרת למידת מכונה ולבצע לימוד נוסף בלי לבנות את כל המודל מהתחלה, משתמשים באבני הבניין הקיימות ובעזרתם מלמדים את המודל דברים חדשים
VGG16	רשת נוירונים קונבולוצית בעלת 16 שכבות
Sklearn	ספריית פייתון ללימוד מכונה
MNIST	מאגר מידע המכיל תמונות של אותיות ללימוד מערכות זיהוי כתב יד

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
E-mail: Moretech@ed.technion.ac.il +972-73-3783146 טלפון
https://moretech.technion.ac.il

מונח	הסבר
ANN	Artificial neural network
Cifar10	מאגר מידע המכיל תמונות של אותיות ללימוד מערכות
NODE	רכיב הפעלה בסיסי במערכות מידע בצורת גרף (כמו ros) ומשמש כצומת
Jupyter Notebook	סביבת פיתוח אינטראקטיבית המופעלת ישירות מדפדפן ולרוב יושבת על ענן מרוחק
GPU	מעבד גרפי
SDK	software development kit, ערכת פיתוח המכילה כלים וספריות תוכנה
services	דרך להעביר הודעות ממקום למקום עם בקשה ואישור מסירה
VirtualBox	תוכנה שמאפשרת להפעיל מערכות הפעלה שונות על מחשב מארח אחד

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

1. הקדמה

ארגונים בינלאומיים המתמקדים בהכנת מדענים ומהנדסים לעולם התעסוקה העתידי מגדירים מגוון רחב של מיומנויות המאה ה-21 (ABET, 2019; National Research Council, 2013). מחקרים קודמים בחנו את הקשרים בין שיטות הוראה ולמידה לבין התפתחותן של מיומנויות המאה ה-21 במסגרת בית הספר ובחינוך הגבוה למדע והנדסה (Kember & Leung, 2005; Stehle & Peters-Burton, 2019). מחקרים נוספים מצאו כי שיטות של למידה פעילה המתמקדות בעבודת צוות עם בעיות אותנטיות, מספקות הזדמנויות לביטוי ולהתפתחות של מיומנויות מגוונות המתאימות למאה ה-21 בקרב התלמידים והמורים העוסקים בהוראת מקצועות STEM (Dori & Belcher, 2005; Shwartz-Asher et al., 2020). למידה מבוססת פרויקטים - PBL – Project Based Learning מהווה אחת משיטות הלמידה הפעילה ומודל המארגן למידה סביב פרויקטים והכולל משימות מורכבות על סמך שאלות ובעיות מאתגרות. סגנון למידה זה דורש מהתלמידים לעצב, לפתור בעיות, לקבל החלטות, לחקור פעילויות, לעבוד באופן עצמאי לתקופות ממושכות ולהגיע למוצרים ריאליסטיים (Thomas, 2000). מאפיינים נוספים של למידה מבוססת פרויקטים הם תוכן אותנטי, הערכה אותנטית, מטרות חינוכיות ברורות, למידה שיתופית, רפלקציה ושילוב מיומנויות ברמות חשיבה גבוהות. הגדרות של הוראה ולמידה מבוססת פרויקטים כוללות מאפיינים הקשורים לשימוש בשאלה אותנטית (Barak, 2005; Dori, 2003; Verner & Ahlgren, 2007), פעילות קבוצתית, שימוש בכלים ומיומנויות מגוונות ונושאים רב תחומיים (Thomas, 2000). רובוטים ניידים Mobile Robotics הם תחום חדשני הכולל טווח רחב של רובוטים אשר ניתן לשלבו בלמידה מבוססת פרויקטים למטרת פיתוח מגוון מיומנויות העונות על דרישות המאה ה-21 (Rocker Yoel, & Dori, 2022). בנוסף, רובוטים ניידים מבוססים על תחומים רבים של הנדסה ומדע, כגון, הנדסת מכונות, חשמל, אלקטרוניקה ומדעי המחשב (Siegwart & Nourbachsh, 2004). רובוט נייד הוא מכונה ניידת אוטונומית או מופעלת מרחוק המסוגלת לנוע בסביבה מוגדרת. רובוטים ניידים משתמשים בחיישנים כדי לתפוס את הסביבה שלהם ולקבל החלטות על סמך המידע שנרכש מהחיישנים. האופי האוטונומי של רובוטים ניידים הם מרכיב חשוב בהתפתחות הטכנולוגית לטובת החברה האנושית. רובוטים ניידים משמשים בבתי הספר ככלי הוראה וטכנולוגיה חדשנית, ומשולבים בדרך כלל בתוך המחלקות למדעי המחשב וההנדסה. בחינוך משתמשים ברובוטים ניידים ללימוד קורסים הקשורים ישירות לרובוטיקה, כולל קורסים תיאורטיים ומעבדות (בדרך כלל בשימוש בלימודים

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

גבוהים), וככלי ללמד נושאים אחרים בהנדסה, במדע ואפילו בתחומים שאינם קשורים כמו ביולוגיה ופסיכולוגיה (בדרך כלל בשימוש בבתי ספר) (Malec, 2001).

במטרה לענות על הצורך, שעלה ממשרד החינוך ומהצבא, להכניס לתוכנית הלימודים בתיכון את תחום הכלים הבלתי מאוישים (כתב"מ) האוטונומי ע"י שילובו בלמידה מבוססת פרויקטים, התקיימו מפגשי צוות מומחים וצוות פיתוח בשנת תשפ"ב במסגרת מרכז המורים הארצי למקצועות הטכנולוגיים (לחוברת המתארת את כל התהליך שבוצע לחצו כאן). בעקבות המפגשים פותחה תוכנית הכשרה המיועדת למורי החינוך הטכנולוגי במגמות הנדסת אלקטרוניקה, הנדסת מכונות ותחבורה מתקדמת, בנושא: פיתוחים ויישומים בלמידה עמוקה ורובוטיקה בפרויקטים של כלים בלתי מאוישים. התוכנית כללה שני חלקים: חלק א' - הקניית רקע תיאורטי בסיסי וחלק ב' – הקניית התנסות פעילה בנושאי התכנית במטרה לפתח מערכי שיעור או מערכי מעבדה ליישום עם התלמידים בבתי הספר. בשנת תשפ"ד יפותח חלק ג' – שמטרתו לפתח פרויקטי גמר בתחום הנ"ל במסגרת בתי הספר ויהווה פיילוט לביצוע תחרויות בהתאם.

החוברת הנוכחית סוקרת את חלק א' של תוכנית ההכשרה: השתלמות בנושא "יישומים פדגוגיים בלמידה עמוקה ורובוטיקה בחינוך טכנולוגי – חלק א'" שנערך במהלך יוני – יולי 2023, במסגרת מרכז המורים מור-טק. בשנת תשפ"ד תפותח חוברת נוספת בדגש על חלק ב' וחלק ג' של תוכנית ההכשרה.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

2. חלק א' של תוכנית ההכשרה – מבוא ורקע בסיסי בנושא יישומים פדגוגיים בלמידה עמוקה ורובוטיקה בחינוך טכנולוגי

2.1 תיאור כללי

חלק א' של תוכנית ההכשרה הינו בהיקף של 60 שעות וכלל מפגשים סינכרוניים בזום ומפגשים אסינכרוניים לביצוע מטלות. מטרת חלק א' הייתה להכשיר את המורים המובילים במגמות הנדסת אלקטרוניקה, הנדסת מכונות ותחבורה מתקדמת בנושאים הקשורים לכלים בלתי מאוישים אוטונומיים. במהלך המפגשים המורים רכשו רקע תיאורטי בסיסי סביב טכנולוגיות חדשות: בהיבט של תוכנה וחומרה השימושיים היום בפיתוח כלים בלתי מאוישים אוטונומיים. כל זאת, במטרה להביא לבניית אב טיפוס ראשוני והמתבסס על שילובם של ידע תוכן טכנולוגי וידע פדגוגי חדש אשר נרכש במהלך פיתוח פרויקטים טכנולוגיים המשלבים למידת מכונה ורובוטיקה. מטרת העל של כל תוכנית ההכשרה הינה קידום הוראה מבוססת פרויקטים במסגרת החינוך הטכנולוגי המשלבים דיסציפלינות שונות ולאפשר פיתוח חומרי למידה חדשים בהקשר תחרות פרויקטים בנושא בין בתי ספר בארץ.

2.2 ראשי פרקים מרכזיים בחלק א' של תוכנית ההכשרה

הנושאים המרכזים של חלק א' בתוכנית ההכשרה (טבלה 1), כדלקמן:

- יסודות במערכת הפעלה Linux: התקנת גרסת Ubuntu תחת מערכת הפעלה חלונות, מבנה התיקיות, עבודה עם שורת הפקודה.
- כלים לעבודה עם Linux, כגון: Visual Studio CODE, SSH, PIP,VNC.
- מערכת ROS: התקנה סביבת העבודה, כתיבת Node בסיסי, תרגול תקשורת הודעות בין שני Nodes, כתיבת אלגוריתם PID לבקרת תנועה.
- למידת מכונה בדגש על רשת נוירונים בדגש על ההבדל בין מודל קלאסי ללמידת מכונה, התנסות בזיהוי תמונה, קונבולוציה, tensorflow playground, תרגול fashion_mnist.
- התנסות בפיתוח פרויקט כדוגמת כתיבת קוד Python לביצוע Transfer Learning לצורך זיהוי פריטים ספציפיים תוך שימוש במודל מאומן כדוגמת VGG16.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

טבלה 1. תוכנית מפורטת של נושאי המפגשים בחלק א' של תכנית ההכשרה

מספר מפגש	נושא המפגש
1	מבוא ללמידת מכונה. תכנות בסביבת Google Colab. טעינה והצגה של Dataset. מבוא לרגרסיה לינארית. רגרסיה לינארית עם Sklearn.
2	מבוא לרגרסיה לוגיסטית. רגרסיה לוגיסטית עם Sklearn. מטריצת הבלבול ומדד Accuracy. ויזואליזציה של גבולות החלטה. מוטיבציה לרשתות נוירונים.
3	מבוא לרשת נוירונים. מימוש רשת עמוקה Sklearn עם שתי מחלקות. מימוש רשת עמוקה Sklearn עם ריבוי מחלקות. חלוקה ל- Train, validation, test.
4	Overfitting and underfitting. פונקציות אקטיבציה. הכרות עם Tensorflow playground. כוון היפר פרמטרים Mini batches. Dropout.
5	הכרות עם Tensorflow. כתיבת קוד העושה שימוש ב-MNIST עם ANN. כתיבת קוד העושה שימוש ב-Cifar10 עם ANN.
6	מבוא ל-CNN. אוגמנטציות. כתיבת קוד העושה שימוש ב Flowers dataset עם CNN.
7	מבוא ל-Transfer learning. כתיבת קוד לסיווג כלבים חתולים תוך שימוש ב MobilenetV2
8	יסודות במערכת Linux: התקנת גרסת Ubuntu תחת מערכת הפעלה חלונות, מבנה התיקיות, עבודה עם שורת הפקודה.
9	כלים לעבודה עם Linux , כגון: Visual Studio CODE, SSH, PIP.
10	מערכת ROS - התקנה סביבת העבודה ומושגים בסיסיים על ROS.
11	מערכת ROS - כתיבת חבילה חדשה ותרגול תקשורת הודעות בין שני Node.
12	מערכת ROS - התנסות בסימולטור.

דוגמאות לחומרים שחולקו במהלך תוכנית ההכשרה - חלק א' לחצו כאן.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

2.3 מטלה מסכמת – אתר WIKI

המטלה המסכמת של חלק א' בתוכנית ההכשרה התבססה על האתר של המדריך הלימודי בנושא רובוטיקה ולמידה עמוקה בחינוך הטכנולוגי בכלים אוטונומיים WIKI - לאתר לחצו כאן. האתר בשפה האנגלית, פותח במיוחד כדי לתת מענה וליווי למורים המעוניינים ללמוד ולהעשיר את הידע שלהם בנושא למידת מכונה ורובוטיקה למטרת פיתוח פרויקטים בכלים בלתי מאוישים. האתר כולל פרקים העונים על הנושאים שהוצגו בחלק א' של תוכנית ההכשרה. האתר עדיין בתהליך פיתוח ושדרוג ויותאם בהמשך לחלק ג' של תוכנית ההכשרה. המשתתפים התבקשו לעיין בכל הפרקים ולבחור בפרק או מספר פרקים מתוך האתר. המטלה הוגשה בקבוצות, המורים קיבלו משוב ותיקנו בהתאם והגישו שוב. המטלה הסופית כללה:

- נושאי הפרק או הפרקים
- תתי הפרקים המופיעים
- מטרות לימודיות של הפרק
- מושגים מרכזיים
- סיכום הפרק או הפרקים כך שישמש מורים נוספים בעתיד בהיבט של תוכנה, חומרה, ציוד נדרש, הוראות והנחיות כלליות לעבודה, ידע נדרש וכו'.
- הצעות לשימור בתוכן ובנראות הפרק, כגון: סדר וארגון המידע, ניסוח ברור וכו'.
- הצעות לשיפור בתוכן ובנראות הפרק שיובילו להבנה עמוקה ובהירות מצד המשתמש כולל נימוקים/הסברים. דוגמאות להצעות לשיפור: מידע חסר, סדר וארגון, הוספת תמונות, הוספת סרטונים, הוספת קישורים אחרים וכו'.

בהמשך לחלק א' של תוכנית ההכשרה, התקיים חלק ב' אשר התבסס על למידה והתנסות פעילה (עבודה סדנאית בקבוצות) בלמידת מכונה ורובוטיקה.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

3. סקירה לכלי התוכנה

במהלך חלק א' המשתתפים התנסו בכלי תוכנה מגוונים אותם נציג גם לעומק בפרק 4 המתאר במפורט את הפרקים השונים שנלמדו בהשתלמות.

להלן כלי התוכנה השונים:

- אתר MorBot Wiki הכולל את המידע והחומרים הדרושים בהשתלמות. להלן הקישור: Morbot WIKI.
- הבלוג של אריאל בר-יצחק – הבלוג כולל: חומרי לימוד בנושא למידת מכונה, פתרונות בגרות מדעי המחשב, למידה עצמית לבגרות, מודלים חישוביים ותוכנת JFLAP, סביבת Visual Studio, פתרונות מבחני מפמ"ר לחט"ב. להלן: קישור לבלוג.
- קישור ל-GOOGLE COLAB - סביבת עבודה מבוססת Jupyter Notebook אשר מאוחסנת על השרתים של גוגל, וכוללת: חבילות הכרחיות ללמידת מכונה, גישה ל-GPU, אחסון המידע בענן בתוך ה-Drive, להלן: GOOGLE COLAB.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4. תיאור מפורט לנושאי הלימוד באתר WIKI

חומרי הלימוד מסתמכים על אתר MorBot Wiki הכולל את המידע והחומרים הדרושים בהשתלמות. להלן הקישור: Morbot WIKI. בפרק זה נתאר את נושאי הלימוד השונים תוך כדי הקישוריות לאתר. ההסברים לנושאי הלימודים מסתמכים גם על המטלה המסכמת שהגישו המשתתפים בחלק א' של תוכנית ההכשרה. תתי הפרקים של חלק זה הם בהתאמה לתתי הפרקים באתר.

4.1 MorBot - פלטפורמת רובוט חינוכית לתלמידים

MorBot היא פלטפורמת רובוט חינוכית המיועדת לתלמידים ללמוד על רובוטיקה, רשתות עצביות עמוקות (DNN), למידת מכונה (ML) ותכנות באמצעות מערכת ההפעלה הרובוט (ROS). פלטפורמה אינטראקטיבית זו מציעה לתלמידים חוויה מעשית לחקור את העולם של הרובוטיקה והבינה המלאכותית (AI) בצורה מרתקת וקלה להבנה. תלמידים יכולים ללמוד מיומנויות חשובות כגון פתרון בעיות, יצירתיות וחשיבה ביקורתית. מיומנויות אלה בעלות ערך רב בכל תחומי החיים. הפלטפורמה מבוססת על שימוש בערכה בשם Nano Jetson NVIDIA KIT Robot AI JetBot. ניתן לרכוש ערכה זו מחברת פייטל לחצו כאן.

JetBot הינה ערכת רובוטיקה מבית NVIDIA המאפשרת לבנות אפליקציות AI המשלבות רובוטיקה ולמידת מכונה. JetBot מבוסס על המחשב הקטן בשם NVIDIA Jetson Nano AI המאפשר להריץ רשתות עמוקות. מחשב זה תומך במספר חיישנים ורשתות נויראליות במקביל לזיהוי אובייקטים, הימנעות ממכשולים ומעקב אחרי אובייקטים.

היתרונות של JetBot על פני ערכות אחרות הם:

הוא משתמש ב NVIDIA JetPack שהוא SDK מקיף שמאפשר לנו לנצל את העוצמה של AI המודרני. ה-SDK מבוסס על Linux ומאפשר ללומדים לפתח יישומים בדומה לנעשה בתעשייה.

JetBot מאפשר לנו לשלב מוצרים וטכנולוגיות של צד שלישי.

JetBot מבוסס על Robot Operating System - ROS שהיא מערכת ההפעלה הנחשבת ביותר בעולם הרובוטיקה והכלים הבלתי מאוישים.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.2 מאפיינים

הרכבה קלה: MorBot נועד להיות קל להרכבה על בסיס פלטפורמת JetBot, ללא צורך בהלחמה או חיווט מורכב. תלמידים יכולים לעקוב אחר ההוראות המפורטות ולהרכיב את הרובוט בעצמם, מה שעוזר לפתח את פתרון הבעיות והמוטוריקה העדינה שלהם.

ניתן לתכנות: לאחר ההרכבה, ניתן לתכנת את MorBot באמצעות אינטגרציה עם ROS לשליטה חלקה ואינטראקציה עם הרובוט. ניתן לשלוט על הרובוט באמצעות Python ומודולי AI מובנים מראש לתלמידים להתנסות ולהתאים אישית.

תוכן חינוכי: MorBot מגיע עם סדרה של שיעורים שנועדו ללמד תלמידים על מושגי STEM כגון רובוטיקה, הנדסה וקידוד. השיעורים מוגשים בצורה מהנה והמסייעת בהכרת המושגים.

ניתן להרחבה: MorBot תוכנן להרחבה, עם מגוון אביזרים ותוספות שניתן להשתמש בהן כדי לשפר את יכולות הרובוט. תלמידים יכולים להשתמש באביזרים אלה כדי ליצור פרויקטים וניסויים מותאמים אישית, אשר תורמים לטיפוח יצירתיות וחדשנות.

4.3 רכיבים עיקריים

- JetBot: ה-JetBot, רובוט המורכב בקלות, משתמש ב-Jetson Nano, מחשב קומפקטי אך חזק. אופיו הידידותי למשתמש והיכולת לבצע משימות מורכבות, כמו הפעלת אלגוריתמי AI וניהול תנועות רובוט, הופכים אותו לבחירה אידיאלית.

- Jetson Nano: Jetson Nano, כ-"מוח" של הרובוט, מעבד מידע ושולט בתנועותיו. מחשב קומפקטי וחזק זה מריץ ביעילות אלגוריתמי בינה מלאכותית ומנהל אנרגיה היטב, ומבטיח זמני פעולה ארוכים בטעינת סוללה אחת.

- הבנת Linux ו-Python היא חיונית ברובוטיקה ו-ROS מכיוון ש-Linux מספקת מערכת הפעלה חזקה ורב-תכליתית אידיאלית למשימות המורכבות והמקבילות ברובוטיקה, בעוד ש-Python, בשל הפשטות והתמיכה הרבה בספרייה, נמצא בשימוש נרחב עבור סקריפטים והטמעת אלגוריתמים ב-ROS.

4.4 מטרות לימודיות

להלן מגוון מטרות לימודיות של כל הפרקים שיפורטו בהמשך:

<u>מטרות קוגניטיביות</u>

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- התלמידים יזהו את רכיבי ערכת הרכב הרובוטי (הבנה).
- התלמידים יפעלו בהתאם למערכת הנחיות ויבחנו את מידת הבנתו ומיומנויותיו הטכניות - הנדסיות (יישום).

מטרות אופרטיביות

- התלמידים יבנו רכב רובוטי (JetBot) (יישום).
- התלמידים יתקינו מערכת הפעלה (תוכנה) על הרובוט ויבחנו את תקינותה (אנליזה).
- הכרת המושגים המרכזיים בעולם ה-ROS (הבנה).
- הכרת דרישות הקדם להתקנת ROS (הבנה).
- בדיקה שדרישות סביבת העבודה מוכנות, והתקנת החסר (יישום).
- התקנת מערכת ROS על פי ההוראות במסמך ההתקנה (יישום).
- בדיקת תקינות ההתקנה על פי ההוראות במסמך ההתקנה (יישום).
- הכרות ראשונית עם Linux ומערכות ההפעלה.
- רקע היסטורי של Linux.
- הכרות בסיסית עם גרסאות הפצה של Linux.
- הבדלים בין Linux ל-Windows.
- התנסות בסביבת Linux בעיקר עם Ubuntu.
- הבנה ויישום של הרשאות לקבצים ותיקיות במערכת Linux.
- ממשק עבודה לפרסום הודעות ורישומן.
- יצירת מפרסם (Publisher).
- יצירת מנוי למפרסם (Subscriber) (יכול להיות יותר ממנוי אחד).
- העברת הודעות בין מפרסם למנוי למפרסם (הכולל פעולת האזנה של המערכת).
- הבנת העקרונות של מערכת ההפעלה ROS.
- השגת הבנה איך מתבצעת התקשורת במערכת מבוססת ROS.
- השגת יכולת לתכנן וליישם את התקשורת.
- הכרת סביבת העבודה ROS והרובוט JetBot.
- מימוש והפעלת חבילת ROS בשם jetbot_control_mode והסעת הרובוט.
- היכרות ועבודה עם SSH ו-Visual Studio Code.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מט"ח המרכז הישראלי לחינוך מדעי טכנולוגי ע"ש עמוס דה-שליט
מור-טק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

<u>מטרות חברתיות</u>

- התלמידים יעבדו בצוות תוך שימוש במיומנויות "רכות" להערכה וניהול תהליכי הבניה (הערכה).
- עזרה לצוותים מתקשים או צוותים עם תקלות (מיומנויות רכות והעמקת הידע במערכת - אנליזה והערכה).

הפתיחה לאתר כוללת הנחיות מפורטות להרכבת הרובוט, מסמכים, קישורים, מגוון הדגמות ואתגרים שנועדו לעזור למשתמשים ללמוד על מושגי STEM ורובוטיקה. השלבים הבאים מפרטים את הפעולות הדרושות להתחלת העבודה עם MorBot:

1. בניית הרובוט MorBot לפי ההוראות בסעיף 4.1 Hardware Setup.
2. הגדרת Jetson Nano ו-Linux - לפי ההוראות בסעיף 4.2 Jetson Nano Linux Setup.
3. התקנת ROS – לפי ההוראות בסעיף 4.3 ROS.
4. התקנת והפעלת חבילת MorBot – הדף בבנייה.

4.5 Hardware Setup

התקנת חומרת הרובוט (Hardware Setup) - בפרק זה מפורטות הפעילויות הנדרשות להרכבת והתקנת החומרה עד לקבלת רובוט מכני. בסיום פרק זה הרובוט עומד פיזית אולם עדיין אינו בר הפעלה. לפני תחילת העבודה יש לקרוא את הוראות הבטיחות ולוודא שליטה והכרת אופן העבודה עם כלי העבודה שהוגדרו לביצוע.

4.5.1 מושגים מרכזיים

- הרכבת והידוק חלקים (הסבר על אופן ביצוע הידוק חלקים, באילו כלים ובאיזה עוצמה).
- בקר ודוחף זרם מנוע (מהו הבקר ועקרון פעולתו - חומרה ותוכנה).
- גולה כלל כיווניות (Ball Caster Wheel) – (כדור המשמש כנקודת תמיכה צפה המאפשרת היגוי קל ומישורי של הרכב).
- צלעות קירור (התקן במערכות אלקטרוניות שתפקידו לקרר את הרכיב האלקטרוני באמצעות יצירת משטחי מגע גדולים עם האוויר).
- גלגל הינע (גלגל שתפקידו לדחוף ולהניע את הרובוט לכיוון הרצוי).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.5.2 בדיקת תכולת הקיט

כלי העבודה: מברג פיליפס ומפתח ייעודי פתוח.

הערה: בכלי העבודה הנ"ל יש לעבוד עם ידיים יבשות.

טבלה 1. חלקים ורכיבי החומרה

שם החלק / רכיב	כמות
בסיס רובוט	1
אנטנת WiFi	1
קיט מנוע DC (מנוע כולל 2 בורגי חיבור ואומים)	2
ברגים לתושבת בסיס דוחף מנועים (M3x4mm)	6
מחברי מרווח להפרדה בין הבסיס וכרטיס דוחף המנוע	6
מחזיק מצלמה	1
ברגים לתותבי בקר מנועים (M2.5x5mm)	4
מחברי מרווח להפרדה בין כרטיס הבקרה וכרטיס דוחף המנוע (M2.5x25mm)	4
בורג (ניילון) לחיבור מצלמה לבסיס (M2x8mm)	4
אום (ניילון) לחיבור מצלמה (M2)	4
בורג ראש מחוספס (M3x6mm)	2
כרטיס דוחף מנועים (Motor Driver) ובית הסוללות	1
גלגל רב כיווני (Omnidirectional wheel)	2
גלגלי צד (גלגל ראשי)	2
בורגי חיבור גלגלי צד (KA2x16)	4
כרטיס בקר עם צלעות קירור (JetSon)	1
מאוורר	1
סוללות 1.5V AA	3

משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מור-טק | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

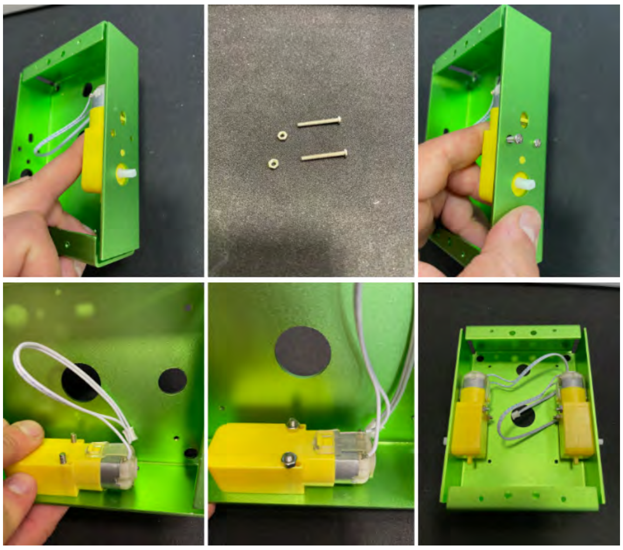
4.5.3 הוראות התקנת החומרה

4.5.3.1 התקנת מנועים

- הכינו את קיט המנועים ובסיס הרובוט להרכבה (תמונה 1).

תמונה 1. קיט מנועים

- מקמו את המנועים במקומם על גבי בסיס הרובוט וחברו למסגרת הבסיס באמצעות בורגי החיבור והאומים (כמתואר באיורים הבאים). שימו לב לא להפעיל כוח הידוק חזק מידי בעת סגירת אומי הברגים (תמונה 2).



תמונה 2. הרכבת המנועים למסגרת

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מור-טק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.5.3.2 התקנת תושבות לכרטיס דוחק מנועים

- הכינו את מערך תושבות דוחף המנועים (Motors Driver) (תמונה 3).

תמונה 3. תושבות מערך דוחף מנועים

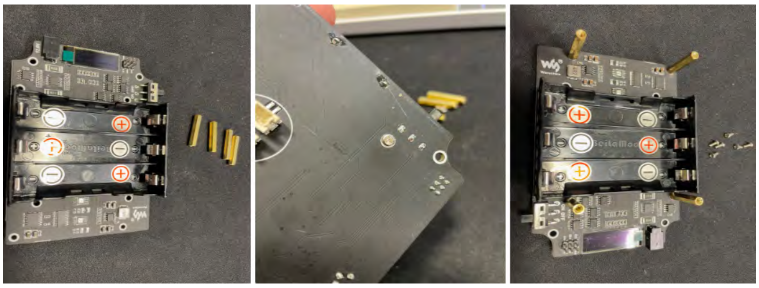
- הרכיבו את התושבות לקדחים המתאימים בבסיס הרובוט והדקו למקומם (תמונה 4).

תמונה 4. תושבות מורכבות על בסיס הרובוט

4.5.3.3 התקנת מחברי מרווח (Spacer screws) בין כרטיס ה-Jetson לכרטיס דוחף המנועים

- חברו את מחברי המרווח המיועדים להפרדה בין כרטיס הבקרה וכרטיס דוחף המנוע (תמונה 5).
- הרכיבו את הברגים מצידו התחתון של כרטיס הבקר ואליהם חברו מצידו השני של הכרטיס את מחברי המרווח (תמונה 5).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



תמונה 5. מחברי מרווח בין כרטיס ה-Jetson לכרטיס דוחף המנועים

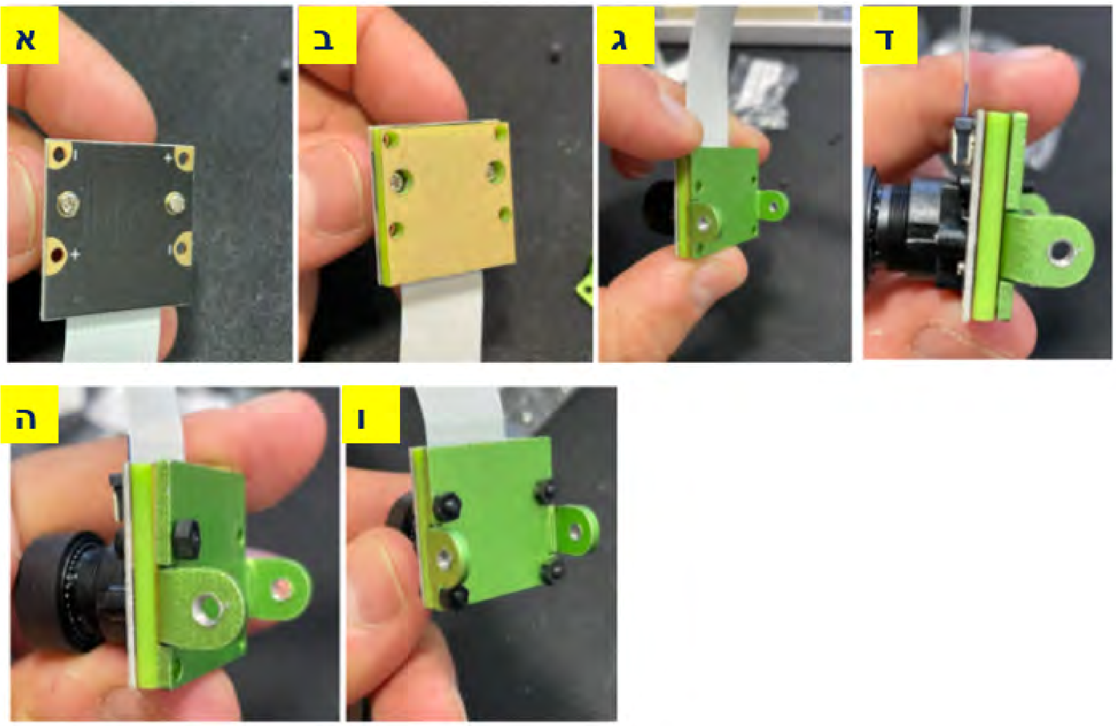
4.5.3.4 הרכבת מודול המצלמה

- הכינו את לוח המצלמה, מתקן המצלמה ובורגי החיבור להרכבה (תמונה 6).

תמונה 6. לוח מצלמה, מתקן המצלמה ובורגי החיבור

- הרכיבו את הרכיבים על גבי מתקן המצלמה. שימו לב שהצד השטוח בכבל המצלמה פונה כלפי מעלה (לצד השני של המחבר). חברו את החלקים באמצעות בורגי הניילון והדקו את האומים למקומם כפי שמפורט בתמונה 7.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



תמונה 7. הרכבת מתקן המצלמה והמצלמה

4.5.3.5 הרכבת מודול מצלמה לגוף הרובוט

- הרכיבו את מחזיק המצלמה למודול המצלמה (שהורכב קודם לכן) באמצעות שני בורגי ההידוק (תמונה 8).

תמונה 8. הרכבת מודול המצלמה

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- חברו את מחזיק המצלמה לבסיס הרובוט באמצעות זוג בורגי M3x6mm (תמונה 9).

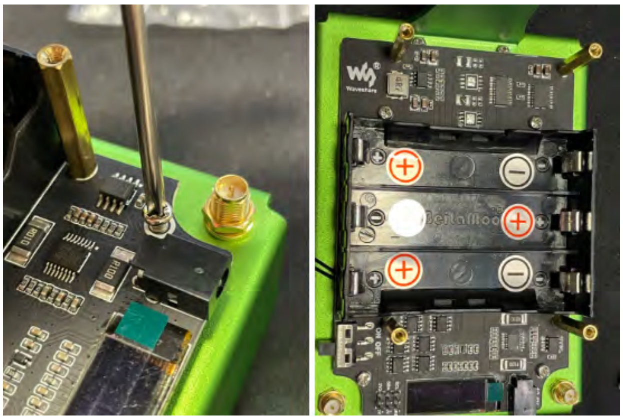
תמונה 9. הרכבת מחזיק המצלמה לבסיס הרובוט

4.5.3.6 הרכבת כרטיס דוחף מנועים (Motor Driver) ובית הסוללות

- הרכיבו את כרטיס דוחף המנועים (הכולל את בית הסוללות) על גבי בסיס המנוע (תמונה 10).
- חברו את הכרטיס למקומו באמצעות 4 בורגי חיבור M3x6mm. וודאו סגירת הברגים והצמדת הכרטיס לתושבות (שימו לב לא להדק חזק מדי את הברגים).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- ודאו שכבל יציאות אנטנות ה-WiFi נמצא בצד ימין (חזית המצלמה)



תמונה 10. הרכבת כרטיס דוחף מנועים ובית הסוללות

4.5.3.7 התקנת סוללות

- הכניסו את שלושת הסוללות למקומן בבית הסוללות. שימו לב לכיוון הכנסת הסוללות (תמונה 11).

תמונה 11. הכנסת הסוללות

4.5.3.8 התקנת גלגלים כלל כיווניים (קדמי ואחורי)

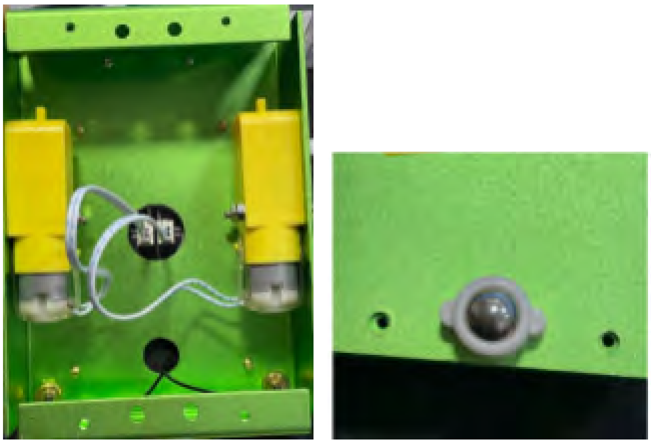
- הרכיבו את הגלגלים הכלל כיווניים על גבי מכסה בית הרובוט.
- חברו את הגלגלים למקומם באמצעות בורגי החיבור הייעודיים (תמונה 12).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תמונה 12. הרכבת גלגלים כלל כיווניים

4.5.3.9 סגירת בית הרובוט

- חברו את מחברי המנועים (קונקטורים) לכרטיס דוחף המנועים, והרכיבו את מכסה בית הרובוט למקומו (שימו לב לכוון ההרכבה - תמונה 13).

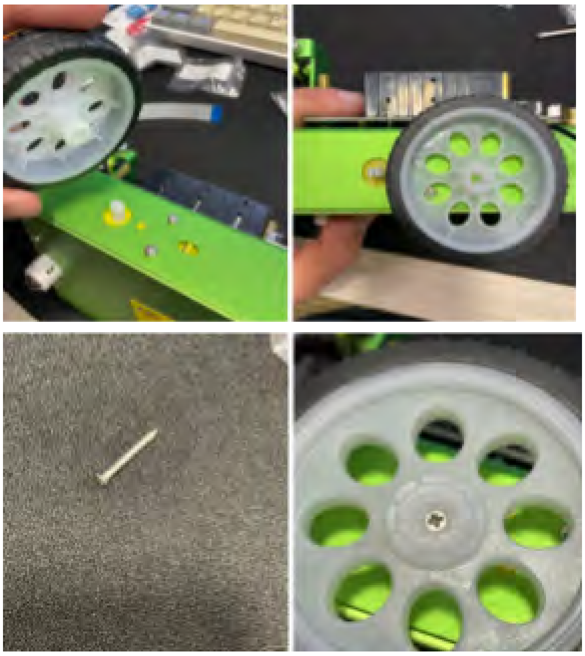


תמונה 13. הרכבת מכסה בית הרובוט

4.5.3.10 התקנת גלגלים צדדיים (ימני ושמאלי)

- הרכיבו את הגלגלים הצדדיים למקומם על גבי צירי המנועים. הדקו הגלגלים למקומם באמצעות בורגי ההידוק KA2x16 (תמונה 14).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



תמונה 14. הרכבת גלגלים צדדיים (הינע)

4.5.3.11 חיבור כרטיס ה-Jetson לגוף הרובוט

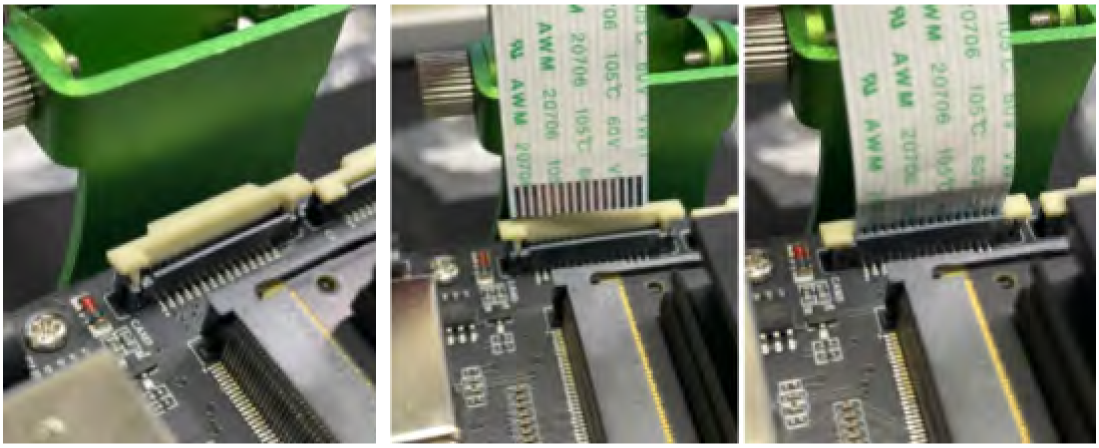
- הרכיבו את כרטיס הבקרה (Jetson) על גבי כרטיס דוחף המנוע (שהורכב קודם לכן). שימו לב שמחבר אנטנת ה-WIFI נמצא בצד ימין והמצלמה מקדימה.
- חברו את שני הכרטיסים באמצעות הידוק בורגי החיבור M2.5x5 למחברי מרווח (תמונה 15).

תמונה 15. הרכבת כרטיס ה-Jetson לגוף הרובוט

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.5.3.12 חיבור כבלי המצלמה לכרטיס הבקר (Jetson)

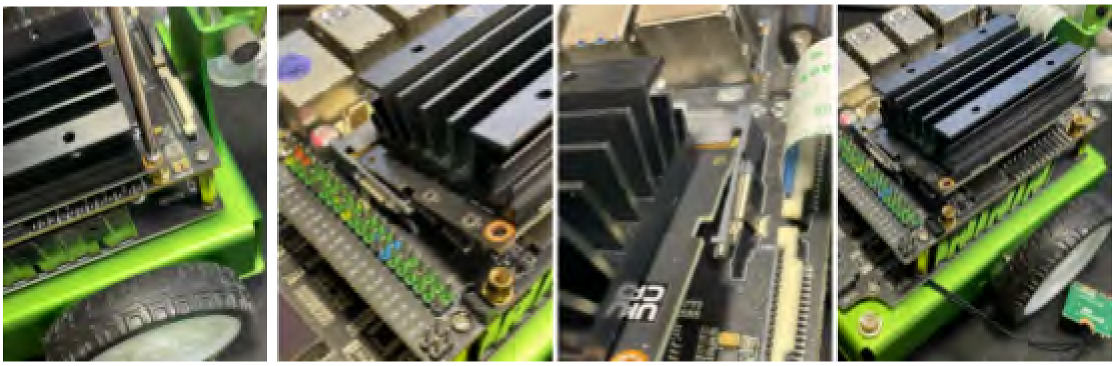
מחבר המצלמה על גבי כרטיס הבקר נמצא מאחורי מחזיק המצלמה (CAM). הסיטו את תופסן הכבל, השחילו את הכבל בעדינות לתוך המחבר והדקו אותו למקומו באמצעות החזרת התופסן למקומו (תמונה 16).



תמונה 16. חיבור כבלי המצלמה לכרטיס הבקר

4.5.3.13 פירוק צלעות קירור של Jetson Nano בקיט הפיתוח (לטובת התקנת מודול WiFi)

- שחררו את בורגי הידוק צלעות הקירור לכרטיס הבקרה.
- סובבו את צלעות הקירור כלפי מעלה והוציאו את היחידה מחוץ לכרטיס הבקרה, עד קבלת גישה פנויה להכנסת כרטיס ה – WiFi (תמונה 17).



תמונה 17. פירוק צלעות הקירור

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תמונה 18. מיקום הרכבת כרטיס ה-WiFi

4.5.3.14 התקנת כרטיס ומחבר אנטנת WiFi

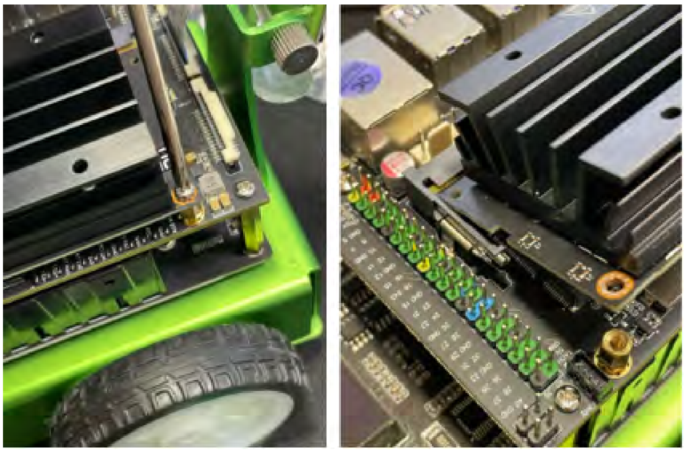
- הכניסו את כרטיס (מודול) ה–WiFi למקומו על כרטיס הבקרה. הרכיבו את כבל אנטנת ה–WiFi למקומו (מאחורי מודול ה–WiFi) והדקו אותו באמצעות בורג החיבור (תמונה 19).

תמונה 19. הרכבת כרטיס ומחבר אנטנת WiFi

4.5.3.15 התקנה חוזרת של צלעות הקירור במודול Jetson Nano מקיט הפיתוח

- הכניסו את צלעות הקירור למקומן על גבי כרטיס הבקרה, סובבו אותו כלפי מטה והדקו אותו למקומו באמצעות ברגי חיבור. שימו לב בהידוק הברגים למקומם (לא להדק בכוח רב) (תמונה 20).

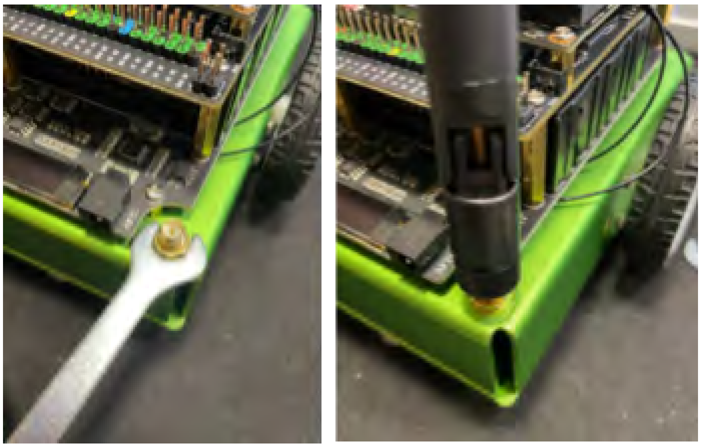
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



תמונה 20. הרכבת צלעות הקירור

4.5.3.16 חיבור אנטנת WiFi

- הרכיבו את מחבר אנטנת ה–WiFi למקומו על גבי בסיס הרובוט (צד ימין מאחור).
- בסיום הבריגו את האנטנה למקומה (תמונה 21).



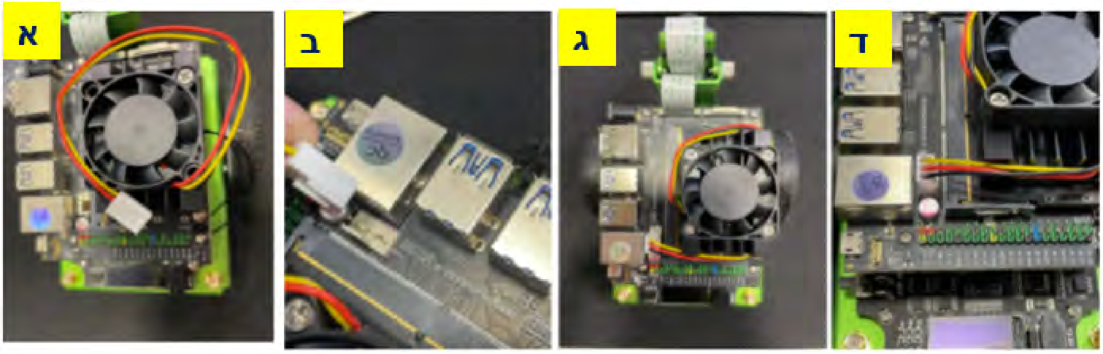
תמונה 21. חיבור אנטנת ה WiFi

4.5.3.17 התקנת המאוורר (על גבי צלעות הקירור)

- הניחו את המאוורר על צלעות הקירור, וודאו חורי הברגים מתאימים מעל להברגות בבסיס של בית צלעות הקירור.
- חברו את המאוורר למקומו באמצעות הידוק בורגי החיבור (שימו לב לא להדק בכוח רב).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- חברו את מחבר כבל החשמל לשקע המתאים (מסומן כ-*FAN*) על גבי צלעות הקירור, סדרו את הכבל מסביב למאוורר.



תמונה 22. הרכבת המאוורר על גבי צלעות הקירור

4.5.3.18 התקנת צמת תקשורת ואספקת מתח

- מקמו את שקע המתח (צבע לבן) מעל המחבר שלו (5V_3V3) שעל גבי כרטיס הבקרה (מודול ה-Jetson Nano). שימו לב שהחוט הצהוב מתחבר לפין מספר 1 בשקע.
- חברו את צמת התקשורת (שקע צבע שחור) למקומו על גבי כרטיס דוחף המנוע. **<u>שימו לב</u>** שהחוט הצהוב מתחבר מול פין *3V3* במחבר השחור (תמונה 23).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תמונה 23. הרכבת צמת תקשורת ואספקת מתח

4.5.3.19 סיום

- בסיום ההרכבה יתקבל הרובוט כמתואר בתמונה 24:

תמונה 24. הרובוט הסופי אחרי הרכבה

כעת עברו ל-Jetson Nano Linux Setup (סעיף 4.6).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.6 Jetson Nano Linux Setup

הגדרות כרטיס Jetson Nano עבור Jetson Nano Linux Setup – בפרק זה נדרש להמשיך ולבצע את הגדרות המערכת השונות והפעלת כרטיס הבקר במטרה לשלוט על הרובוט. בסיום הפרק ניתן יהיה להפעיל את הרובוט להתממשק אליו עם מערכות בקרה ושליטה נוספות.

4.6.1 מושגים מרכזיים

- מודול WiFi - משמש ליצירת תקשורת אלחוטית בין הרובוט ומחשב לצורך העברת פקודות מובנות לרובוט.
- אתחול תוכנה - תהליך מחיקת התוכנה הקיימת ואתחולה בגרסת תוכנה חדשה.
- Visual Studio Code - סביבת פיתוח (IDE) המשמשת לתכנות הרובוט והעברת פקודות לרובוט

4.6.2 בקר Jetson Nano בגרסה חדשה של Linux

ניתן להשתמש בכרטיס JetBot SD שנבנה מראש הכלול בערכה, הוא מוכן לשימוש. אך במקרה של תקלות, יש להשתמש בשיטה להלן:

4.6.2.1 צריבת תוכנה לכרטיס הזיכרון

- הורידו למחשב שולחני את תוכנת הכרטיס מהקובץ המצורף (מבוסס על JetPack 4.5):

jetbot-043_nano-4gb-jp45.zip

- הכניסו את כרטיס הזיכרון למחשב השולחני.
- באמצעות תוכנת הצריבה Etcher, בחרו את קובץ התוכנה שהורד קודם לכן
- jetbot-043_nano-4gb-jp45.zip וצרבו אותו על גבי כרטיס הזיכרון.
- הוציאו את כרטיס הזיכרון מהמחשב השולחני.

4.6.2.2 אתחול בקר Jetson Nano

- הכניסו את כרטיס הזיכרון למקומו בחריץ הייעודי על מחשב Jetson Nano נמצא מתחת למודול הבקר עצמו.
- חברו מקלדת עכבר ומסך לבקר Jetson Nano.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- הפעילו את בקר Jetson Nano באמצעות חיבור כבל החשמל לרובוט והעברת מתג ההפעלה למצב ON.

4.6.2.3 חיבור תקשורת WiFi לרובוט

- בצעו Login למערכת ההפעלה תוך שימוש בשם משתמש: jetbot וסיסמה: jetbot.
- התחברו לרשת ה-WiFi (השתמשו בתקשורת המחשב המחובר לרובוט). בשלב זה בקר ה-Jetson Nano צריך להתחבר אוטומטית ל-WiFi בהפעלתו ולהציג את כתובת ה–IP על גבי מסך הרובוט (piOLED display).
- כבו את הרובוט באמצעות המחשב החיצוני המחובר אליו.
- נתקו את רכיבי המחשב (מסך, מקלדת, עכבר וספק כוח) המחוברים מכרטיס הבקרה (Jetson Nano).
- הפעילו את הרובוט באמצעות מערך הסוללות שלו.
- המתינו לעליית התוכנה.
- בדקו את כתובת ה – IP של הרובוט ע"ג מסך הבקר. לאחר הופעת הכתובת ע"ג המסך הרובוט מוכן להפעלה ושליטה מרחוק.

4.6.2.4 התחברות מרחוק בעזרת מחשב שולחני לרובוט תוך שימוש בתוכנת Visual studio Code

- לאחר חיבור הרובוט ל– WiFi ניתן להפעיל את הרובוט מרחוק תוך מחשב שולחני המריץ מערכת הפעלה שונה מזו שבבקר (כדוגמת Windows) באמצעות התוכנה (Visual Studio Code) כל זאת ללא צורך במקלדת, עכבר ומסך המחוברים ישירות לבקר ה-Jetson Nano.
- Visual studio Code מאפשר לתקשר עם מחשבים מרוחקים באמצעות פרוטוקול תקשורת הנקרא SSH - Secure Shell , בעזרתו נתחבר עם מחשב הjetson nano שעל הרובוט ונעבוד עליו בנוחות בסביבת עבודה מרוחקת על מחשב שולחני בנוסף נוכל להשתמש בכלים רבים שקיימים בתוך ה Visual studio Code שעוזרים בפיתוח.

4.6.2.5 הגדרת תצורת ממשקים ברובוט

שימו לב! לצורך הגדרת ממשקי הרובוט משתמשים ב- - Linux Command Line Interface CLI. וודאו שיש ברשותכם ידע בסיסי ב-Linux (ניתן להיעזר במדריך Linux Tutorial wiki במידת הצורך).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020 . הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

פקודות עיקריות ב Linux

i. sudo – פקודה מבוססת מערכת Unix המאפשרת למשתמש להפעיל פקודה ברמת "משמש על" ומשמשת לבחינת רישיונות משתמש לאבטחת הגישה לבקר.
ii. apt-get – פקודה שמאפשרת להתקין, לעדכן ולהסיר תוכנה במערכת. זוהי דרך מהירה להתקין תוכנה ממאגרי התוכנה הסטנדרטיים של אובונטו.
iii. apt-get update – פקודה אשר בודקת אם קיימים עדכוני תוכנה זמינים ומתקינה אותם בהתאם. פקודה זו שימושית להורדה מהירה של עדכוני תוכנה.
iv. apt-get install – פקודה שמתקינה חבילה חדשה ממאגר התוכנה.
v. chmod - פקודה מבוססת מערכת Unix המאפשרת למשתמש לשנות הרשאות שימוש בקבצים ובספריות.
vi. cd – שינוי ספריה.
vii. ls – פקודה להצגת רשימת הקבצים והתיקיות.
viii. setup.py – קובץ המגדיר ל Python כיצד להתקין את מערכת הבקרה שלנו בספריית ה- Python.

שימוש ב- Git

Git הנה סביבת מעקב גרסאות תוכנה שעוזרת לעקוב אחר שינויים בקבצים, במיוחד כאשר עובדים על קוד. Git מאפשר למספר אנשים לעבוד בו זמנית על אותה קטעי קוד מבלי לדרוס שינויים שאחרים הכניסו.

repo - Git repository הנו אחסון דיגיטלי שבו נשמרים קובצי הפרויקט, כולל ההיסטוריה של השינויים שבוצעו בהם. ניתן לשמור ואחסון קטעי קוד במספר אתרים כמו: GitHub, GitLab, Bitbucket, כך שמספר אנשים יכולים לעבוד עמה בו זמנית.

פקודות Git נפוצות:

- git clone - פקודה המשמשת ליצור העתק של המאגר קוד במחשב מקומי.
- git pull - פקודה המשמשת לעדכון גרסת קוד באתר מרוחק. תחילה מבוצע עדכון המאגר מרחוק ואח"כ מיזוגו עם המאגר המקומי.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- git push - פקודה המשמשת להעלאת מאגר מקומי למאגר מרחוק. משמשת לשיגור פקודות מהמערכת (מאגר) מקומית למערכת מרחוק.

להורדת התוכנית לרובוט לחצו כאן.

להכרת הרובוט Basic Motion והקוד, נסו את ה-NVIDIA JetBot המוכן מראש במדריך notebooks-demos או התחילו לפתח את הפרויקט ע"י התקנת ROS (סעיף 4.7).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.7 ROS

תתי הנושאים שיוצגו בפרק זה: מה זה ROS?, מה זה Linux?, מה הקשר בין Linux לבין ROS?

לפני שמתקינים – דרישות קדם, התקנת המערכת (ROS) ובדיקת ההתקנה.

4.7.1 מושגים מרכזיים

- ROS Framework - סביבת פיתוח (middleware) המבוססת על קוד פתוח שתוכננה במיוחד לבנייה ושליטה במערכות רובוטיות. הסביבה מספקת אוסף של ספריות, כלים ומוסכמות המסייעות בפיתוח יישומי רובוט. הסביבה מאפשרת פיתוח תוכנה מודולרי.
- ROS Nodes - קובץ הרצה במערכת ההפעלה הרובוטית (ROS) המבצעת משימה ספציפית בתוך מערכת רובוטית.
- ROS publisher-subscriber system - מנגנון תקשורת מרכזי במערכת הפעלה רובוטיות שבה ROS Nodes יכולים לפרסם הודעות בנושאים ספציפיים, וצמתים אחרים יכולים להירשם להודעות אלה כדי לקבל ולבצע עליהם עיבוד בתוכנה.
- ROS Melodic Morenia - שם גרסת ה-ROS המתאימה למערכת ההפעלה - Ubuntu 18.04.
- Linux Ubuntu 18.04 היא אחת ממערכות ההפעלה המבוססת על Linux. גרסה זו מתאימה לעבודה עם בקר Jetson.
- VirtualBox - תוכנת וירטואליזציה חינמית המבוססת על קוד פתוח שמאפשרת להריץ מספר מערכות הפעלה בו זמנית על מחשב אחד תוך כדי יצירת מכונות וירטואליות. תוכנה זו מאפשרת להריץ גרסה מלאה של Ubuntu 18.04 על גבי מחשב המריץ מערכת הפעלה Windows.
- Jetson - בקר מבוסס על שבבים של חברת Nvidia שתוכננו במיוחד עבור יישומים ברובוטיקה ובקרה המשלבים בינה מלאכותית.
- Terminal- ממשק שורת פקודה, ממשק מבוסס טקסט שבו משתמשים יכולים ליצור אינטראקציה עם מערכת מחשב באמצעות פקודות מוקלדות.
- Bash Shell Scripting - תהליך יצירה והרצה של קטעי קוד (סקריפטים) שנכתבו בשפת Bash. קטעי קוד אלו מאפשרים למשתמשים לבצע אוטומטיזציה של משימות

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

כולל התניות (תנאים) ולולאות כל זאת במקום להקליט מערך גדול של הוראות לתוך חלון ה-Terminal. דוגמה לכך ניתן לראות בקישור הבא:

https://www.freecodecamp.org/news/bash-scripting-tutorial-linux-shell-script-and-command-line-for-beginners/

4.7.2 ROS – Robot Operating System

ROS אינה מערכת הפעלה קלאסית כדוגמת Linux ו-Windows. מערכת ההפעלה ROS היא תשתית - Framework לפיתוח ובניית רובוטים המבוססת על קוד פתוח. ROS מספקת אוסף של ספריות תוכנה, כלים וכללים המסיעים למפתחים ליצור ולנהל יישומי רובוטים. ROS תוכננה להיות גמישה, מודולרית ומבוזרת וכך מאפשרת לרכיבים שונים של מערכת רובוטית לתקשר זה עם זה בקלות. הביזור מאפשר ליצור תקשורת בין רכיבים שונים המותקנים על מערכות רובוטיות כדוגמת חיישנים ומנועים הממוקמים במקומות שונים ואפילו לייצר אינטראקציה בין רובוטים שונים כאילו היו מותקנים על מערכת אחת.

ROS מספקת תשתית תקשורת המאפשרת חילופי נתונים והעברת מסרים בין מודולי תוכנה הנקראים ROS Nodes בתוך מערכת רובוטית (ה-ROS nodes יכולים לרוץ על אותו המחשב או על מחשבים שונים). ROS מנהלת מערכת פרסום והרשמה (publisher-subscriber) שבה Nodes יכולים לפרסם הודעות בנושאים ספציפיים, ו-Nodes אחרים יכולים להירשם לנושאים אלה כדי לקבל את ההודעות שהם מעוניינים בהן.

ROS כוללת גם מגוון רחב של ספריות וכלים המפשטים משימות רובוטיות נפוצות, כגון שילוב חיישנים, אלגוריתמי בקרה, הדמיה וסימולציה. ROS תומכת בשפות תכנות שונות, כולל Python ו-C++ מה שמאפשר למפתחים לבחור את השפה איתה נוח להם לעבוד.

יתרון נוסף של ROS הוא הקהילה הגדולה והפעילה שלה המהווה מאגר גדול של חבילות ומשאבים שנתרמו על ידי חוקרים, מפתחים וחברות ברחבי העולם. זה מאפשר למפתחים למנף את הקוד הקיים, לשתף את עבודתם עם אחרים ולשתף פעולה בבניית מערכות רובוטיות מתקדמות.

לסיכום ROS מספקת סביבת פיתוח המאפשרת שימוש חוזר בקוד (Code reusability), פיתוח מודולרי ושיתוף פעולה, דבר המקל על פיתוח יישומים רובוטיים מורכבים. ROS צברה

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

פופולריות משמעותית בקהילת הרובוטיקה ונמצאת בשימוש נרחב במחקר, באקדמיה ובתעשייה לפיתוח מגוון רחב של רובוטים ומערכות אוטונומיות.

ל-ROS ישנן שתי גרסאות ROS ו-ROS2, לכל אחת מהגרסאות הנ"ל יש הפצות (distributions) שונות, בדרך כלל יש קשר בין ההפצה ומערכת ההפעלה עליה הפצה מסוימת נתמכת. במקרה שלנו אנחנו נעבוד עם ROS Melodic Morenia הרצה על Linux Ubuntu 18.04, אומנם מערכת ישנה אבל זאת המערכת הנתמכת על ידי בקר ה-Jetson ברובוט שמשמש אותנו, ומאחר והמערכת מבוזרת נוכל להתקין ROS Nodes על מחשב ה-PC ולתקשר עם הרובוט מרחוק לצורך להפעלה, פיתוח ובדיקות. לכן גם ה-ROS שירוץ על ה-PC יהיה Melodic בסביבה של Ubuntu 18.04.

4.7.3 Linux – הסבר קצר

Linux הינה מערכת הפעלה בקוד פתוח המבוססת על Unix, מערכות אלו ידועות בשם הפצות Distributions. המערכת פותחה בתחילה על ידי Linus Torvalds בתחילת שנות ה-90 ומאז הפכה לאחת ממערכות ההפעלה הנפוצות ביותר בעולם.

Linux ידועה ביציבות, אבטחה וגמישות. הגמישות מאפשר למשתמשים ולמפתחים להתאים אותה לצרכים הספציפיים שלהם. ליבת Linux מספקת את פונקציונליות הליבה של מערכת הפעלה מודרנית, כולל מנהלי התקנים, ניהול זיכרון, תזמון תהליכים ותמיכה במערכת הקבצים.

קוד המקור של ליבת Linux זמין באופן חופשי ומאפשר למפתחים ברחבי העולם לתרום לפיתוחו ולשיפורו. גישה שיתופית זו הביאה לקהילה תוססת שיצרה אלפי חבילות תוכנה, כלים ויישומים הפועלים על Linux.

Linux נמצאת בשימוש נרחב בתחומים שונים, החל ממחשבים אישיים דרך שרתים ועד למערכות משובצות מחשב ומחשבי-על. הפצות רבות של Linux פופולריות, כגון Ubuntu, Debian, CentOS. ההפצות השונות נוצרו כדי לארוז את ליבת Linux יחד עם תוכנות וכלי עזר נוספים כדי לספק חוויית מערכת הפעלה מלאה.

Linux תומכת במגוון רחב של ארכיטקטורות חומרה, כולל x86, ARM ו-MIPS, מה שהופך אותה לגמישה המתאימה למגוון רחב של מכשירים.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מערכת ההפעלה Linux מאפשרת ממשק משתמש גרפי (GUI) כדוגמת זה הקיים במערכת ההפעלה Windows כמו כן ניתן להפעיל את מערכת ההפעלה Linux ללא ממשק גרפי, במצב זה תפעול התוכנה יתבצע על ידי שורת הפקודה (CLI) בדומה לחלון ה-Command prompt של מערכת ההפעלה Windows.

לסיכום, Linux מציעה מערכת הפעלה רבת עוצמה המתאימה למשתמשים יחידים, לעסקים ולארגונים ברחבי העולם. Linux היא פלטפורמה אמינה וחסכונית עבור מגוון יישומים כגון מערכות משובצות מחשב כדוגמת הבקר שלנו, מחשוב אישי ושרתים.

4.7.4 הקדמה להתקנת ROS על מחשב שולחני

כפי שכבר למדנו יש להתקין את ROS על גבי מערכת הפעלה Ubuntu 18.04. בגלל שלרובינו יש מחשב שולחני או נייד הכולל מערכת הפעלה Windows ניתן להוסיף על גבי ה-Windows מערכת הפעלה Ubuntu 18.04 במספר אופנים:

- על מחשב PC עצמאי.
- על מחשב וירטואלי בתוך Windows תוך שימוש בתוכנה VirtualBox.

4.7.5 הוראות התקנת ROS Melodic על מחשב שולחני

4.7.5.1 הוספת רשימת המקורות של ROS

ההתקנה של ROS תתבצע בעזרת פקודת ההתקנה הרגילה של המערכת, חברי הקהילה שפתחו את ROS שומרים את כל הקבצים במאגר גדול ומסודר אותו הם מנהלים, מאגר זה נקרא ROS Repository, והוא מאורגן על פי גרסאות ההפצה השונות של ROS.

כדי שההתקנה תתבצע בעזרת כלי ההתקנה apt, הוסיפו למאגר התוכנות של apt את המקור של מאגר גרסת ה-ROS המתאימה ל-Ubuntu 18.04. בצעו זאת בעזרת הפקודה הבאה, את ההוראה הקלידו בחלון ה-Terminal של המחשב השולחני:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/Ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

4.7.5.2 התקנת המפתח הציבורי של ROS

תוכנת ROS חתומה קריפטוגרפית על ידי PGP - Pretty Good Privacy כדי להבטיח מצד אחד שלא הייתה תקלה בהורדה הקובץ ומצד שני מבחינת אבטחת מידע להבטיח שאף גורם

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

שלישי לא שינה את הקוד הנ"ל. לכן הוסיפו למערכת את המפתח הציבורי של ROS. להלן ההוראה שיש לכתוב בחלון ה-Terminal של המחשב השולחני:

```
sudo apt install curl # if you haven't already installed curl

curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

4.7.5.3 עדכון אינדקס חבילות ההתקנה

לאחר הוספת ה-ROS למאגר התוכנות של apt ולאחר עדכון המפתח של ROS, עדכנו את אינדקס החבילות של המערכת (כל החבילות כולל ה-dependencies שלהם) עם המידע החדש. להלן ההוראה שיש לכתוב בחלון ה-Terminal של המחשב השולחני:

```
sudo apt-get update
```

4.7.5.4 התקנת ROS

לאחר ביצוע הוראות ההגדרה, מערכת ההפעלה מוכנה להתקנה, ניתן להתקין התקנה מלאה, התקנה חלקית או לבחור להתקין חבילות תוכנה על פי דרישה. בצעו התקנה מלאה. להלן ההוראה שיש לכתוב בחלון ה-Terminal של המחשב השולחני:

```
sudo apt-get install ros-melodic-desktop-full
```

4.7.5.5 הוספת משתני סביבה של ROS לקובץ .bashrc

שם הקובץ .bashrc. הוא קיצור של המשפט bash run commands. קובץ זה הוא קטע קוד בשפת bash שרץ אוטומטית בכל פעם שחלון bash shell מופעל, תפקידו להתאים את ההגדרות של ה-shell לכל משתמש. לצורך פעולה תקינה של ROS יש לאתחל מספר משתני סביבה, כדי שמשתנים אלו יאותחלו בכל הפעלת חלון bash shell חדש שנפתח הוסיפו את הדרישות של ROS לקובץ .bashrc. להלן הקוד:

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.7.5.6 עדכון משתני הסביבה שב-bashrc.

הקובץ bashrc. מופעל אוטומטית בכל פעם שנוצר העתק חדש של bash. במצב זה ה-bashrc. מופעל אוטומטית. פעולת ה-source מאפשרת לנו להפעיל את bashrc. בכל עת שרוצים וזאת במטרה לעדכן את משתני הסביבה. להלן ההוראה:

```
source ~/.bashrc
```

* לפקודה source יש קיצור שהוא נקודה, לדוגמה:

```
. ~/.bashrc
```

(שימו לב, יש רווח בין הנקודה השמאלית ל- ~)

4.7.5.7 התקנת ROS Dependencies

לצורך כתיבה והרצה של מודולים חדשים נדרשים כלים נוספים הנקראים ROS Dependencies, חלקם כלים כלליים כגון סביבות פיתוח ל-Python, קומפיילר ל-C++ וחלקם ייעודיים להרצה ובדיקה של ROS Nods. התקינו את כולם בשורת התקנה אחת:

```
sudo apt install python-rosdep python-rosinstall python-rosinstall-generator
python-wstool build-essential
```

4.7.5.8 תיחול של ROS Dependencies

אתחלו את חבילות התוכנה על ידי ההוראות הבאות:

```
sudo rosdep init
rosdep update
source /opt/ros/melodic/setup.bash
```

4.7.6 בדיקת ההתקנה

בצעו מספר בדיקות בכדי לבדוק האם ההתקנה הצליחה. סגרו את הטרמינל ופתחו אחד חדש. הריצו את הפקודה הבאה:

```
$ printenv | grep ROS
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

פקודה זו תחפש את כל משתני הסביבה הקשורים ל-ROS, בפעולה זו ניתן לבדוק שהמשתנים קיימים. הפעלה תקינה תספק את הפלט הבא:

```
ROS_ETC_DIR=/opt/ros/melodic/etc/ros
ROS_ROOT=/opt/ros/melodic/share/ros
ROS_MASTER_URI=http://localhost:11311
ROS_VERSION=1
ROS_PYTHON_VERSION=2
ROS_PACKAGE_PATH=/opt/ros/melodic/share
ROSLISP_PACKAGE_DIRECTORIES=
ROS_DISTRO=melodic
```

4.7.7 יצירת פרויקט חדש - ROS workspace

סביבת עבודה של ROS מספקת מספר כלים ליצירת פרויקט ROS חדש, פיתוח ובנייה של חבילות ROS מותאמות למשתמש, דבר המאפשר עבודה יעילה לצורך ניהול פרויקט התוכנה. סביבת עבודה אחת יכולה להכיל מספר חבילות (ROS packages) שהמשתמש יוצר, ובכל חבילה יכולים להיות מספר קבצי הרצה הכתובים בשפת תכנות שונות כדוגמת Python ו-++C.

יצירת תיקייה חדשה עבור פרויקט ROS

1. במטרה ליצור תיקייה חדשה עבור פרויקט ROS כתבו את ההוראות הבאות:

```
mkdir -p ~/MorBot/morbot_ros/morbot_ws/src
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
```

פעולה זו תיצור תיקייה חדשה בשם morbot_ws בתיקייה morbot_ros.

ההוראה catkin_make תבנה את סביבת העבודה ותיצור מספר תתי תיקיות כדוגמת devel ו-build בתוך סביבת העבודה.

2. הוסיפו סביבת העבודה החדשה כחבילה של ROS.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

הוספת החבילה החדשה לתוך ROS תאפשר להפעיל את הקודים שנכתבו יחד עם חבילות קוד אחרות שכבר קיימות בסביבת העבודה. לשם כך, הוסיפו את השורה הבאה לקובץ ה-bashrc. :

```
echo "source ~/MorBot/morbot_ros/morbot_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

3. בדקו את סביבת העבודה.

כדי לוודא שסביבת העבודה נוצרה כהלכה, הפעילו את הפקודה הבאה:

```
echo $ROS_PACKAGE_PATH
```

ניתן לראות את הפלט התקין הבא:

```
/home/your_username/MorBot/morbot_ros/morbot_ws/src:/opt/ros/melodic/share
```

לאחר ביצוע התקנה של ROS Melodic על גבי מערכת הפעלה Ubuntu 18.04 ניתן כעת להתחיל לפתח ולהריץ יישומי ROS. מידע נוסף ניתן לקבל בקישור הבא:

https://github.com/arieldo/MorBot/wiki/Ros-101

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון +972-73-3783146 E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

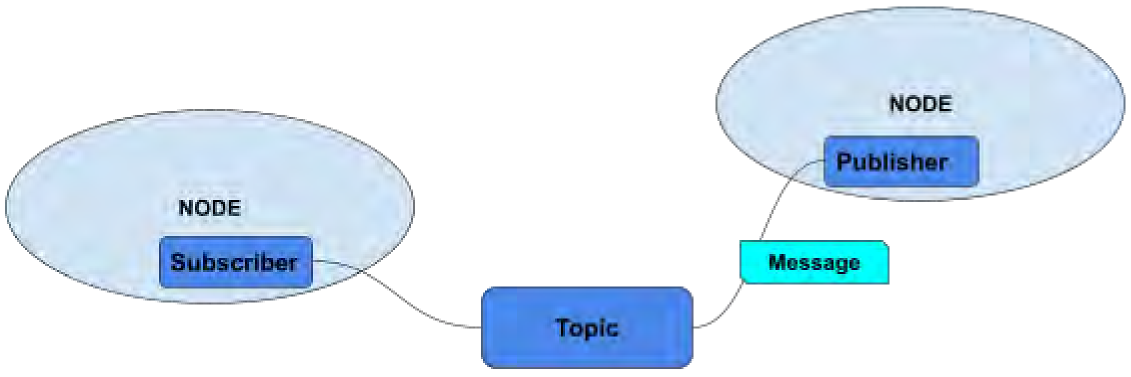
ROS 101 4.8

בפרק זה יוצגו המרכיבים העיקריים של מערכת ההפעלה ROS והאופן בו המרכיבים השונים במערכת רובוטית מתקשרים זה עם זה.

4.8.1 מושגים מרכזיים

- Node - יחידה חישובית ייעודית למטרה ספציפית, למשל דרייבר המפעיל חיישן ומייצר action או service.
- Topic - זרם הודעות מוגדר, היכול להתקבל ממספר מקורות, ומספר צמתים יכולים להשתמש בו. כפי שלמדנו מוקדם יותר במדריך זה כאשר דיברנו על nodes ועל הנגנון publish & subscribe.
- Message - מבנה הנתונים בהודעות תקשורת.

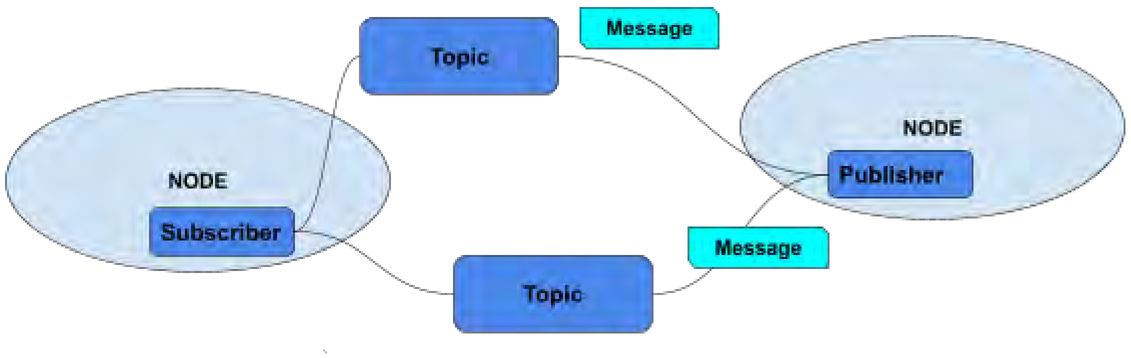
איור 1 מציג דוגמה של מערכת הפעלה בסיסית, בעלת שני צמתים ו-topic אחד.



איור 1. דוגמה למערכת הכוללת שני צמתים ו-topic אחד

איור 2 מציג דוגמה למערכת הכוללת שני צמתים ושני topics.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



איור 2. מערכת עם שני צמתים ושני topics

בנוסף קיים ב-ROS מרכיב חשוב נוסף הקרוי services ומיועד לשימוש כאשר נדרשת תקשורת מסוג request / reply interaction. יישום של בקשה / תשובה נעשה באמצעות שירותים (services), המוגדרים על ידי זוג מבני הודעות: אחד לבקשה ואחד לתשובה.

זה הוא רכיב שמוכל לרוב במערכות מתקדמות יותר ולא יהיה בשימוש בדוגמאות בחוברת זו.

4.8.2 צומת (Node)

צומת היא 'יחידת חישוב', כלומר התקן בעל מטרה ספציפית אחת או יחידה המיועדת למשימה (task) אחת. בדרך כלל כל רובוט בנוי ממספר צמתים, אשר מתקשרים ביניהם דרך פרסום הודעה ל-topic, או קבלת הודעה מ-topic, כמו כן קיימת אפשרות לעשות שימוש ב-service קיים של מערכת ההפעלה. גישה זו מאפשרת לבצע יישום מודולרי של רובוט.

צמתים מבצעים משימות ספציפיות ארכיטקטורת התוכנה של רובוט, מה שהופך את המערכת הכוללת נוחה לתחזוקה, להרחבה.

צמתים מתקשרים באמצעות topics, המאפשר לחבר מספר צמתים אחד לשני.

כמו כן קיים ב-ROS רכיב העברת מידע נוסף הנקרא services אשר מאפשר לבקש מידע מצומת אחד לשני וגם כן לקבל אישור קבלה של אותו מידע בניגוד ל-topic אשר רק שולח את המידע לכל הנרשם אליו ללא דרך לדעת אם המידע התבקש או נקלט .

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

לכל צומת כולל שם ייחודי, נהוג לכלול בשם הצומת גם את ה-'משפחה' שאליה משתייכת הצומת, לדוגמה צומת שמטפלת באחד החיישנים תוגדר תחת השם /sensors והשם הספציפי של החיישן.

הרצת צומת (Node) נעשית על ידי קובץ Launch, קובץ שהסיומת שלו תהיה launch. , קובץ זה מסוגל להפעיל בו זמנית מספר צמתים יחד. טבלה 2 מתארת מספר פקודות שימושיות לעבודה עם צומת.

טבלה 2. מספר פקודות שימושיות לעבודה עם צומת (Node)

פקודה	משמעות	דוגמה
rosnode list	קבלת רשימת הצמתים במערכת שלך	rosnode list
rosnode info <<node_name	קבלת אינפורמציה על צומת מסוים	rosnode info */listener_12*
rosnode ping <<node_name	לבדיקת ה connectivity בין הצומת נוכחית לצומת מסוימת	rosnode ping */listener_12*

את כל הפקודות הנ"ל יש לבצע בטרמינל.

אפשר לממש צומת באופן כזה שרק תפרסם (publish) מידע ל-topic מסוים, או רק תקבל מידע, כלומר כל צומת יכולה לבצע רישום (subscribe) לקבלת מידע מ-topic מסוים. כמובן ניתן גם לשלב ביניהם.

4.8.3 Topics

topic הוא מרכיב חיוני בפרויקט מבוסס ROS ומהווה צינור העברת המידע מצומת לצומת, הוא מעביר את ההודעה שבנויה מסוג ההודעה, שם הצינור, המפרסם והנרשם.

צומת שולחת הודעה על-ידי פרסומה topic ספציפי. שם ה-topic הוא שם המשמש לזיהוי תוכן ההודעה. צומת המעוניינת בסוג מסוים של נתונים צריכה להירשם ל-topic המתאים.

ייתכנו מספר מפרסמים ומנויים בו-זמניים עבור topic אחד, וצומת יחיד עשוי לפרסם ו/או להירשם כמנוי למספר topics.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

הכלל הוא שהמנויים אינם מודעים זה לקיומו של זה, כי הרעיון הוא לנתק את הפקת המידע מצריכתו.

אפשר לחשוב על topic כעל bus הודעות מוגדר היטב. לכל bus יש שם, וכל אחד יכול להתחבר ל - bus כדי לפרסם או לרשום הודעות, כל עוד הן מהסוג הנכון.

מספר מאפיינים עיקריים:

לכל topic יש שם ייחודי, המאורגן בצורה היררכית, למשל /camera/rgb/image_raw

לכל topic יש סוגי תקשורת מוגדרים מראש, והוא יתקשר רק עם סוגי ההודעות הללו, למשל nav_msgs/Odometry או sensor_msgs/Image.

התקשורת בין ה-topics היא א-סינכרונית, וניתן לחבר מספר רב של topics למספר רב של topics אחרים. במצב זה יש לשים לב גם לשיקולים של רוחב הפס הנדרש לתקשורת במערכת כזו. טבלה 3 מתארת פקודות עיקריות לעבודה עם Topics.

טבלה 3. פקודות עיקריות לעבודה עם topics

פקודה	משמעות	דוגמה
rostopic list	קבלת רשימת ה topics	rostopic list
rostopic echo <topic>	בצע רישום ל-topic, כדי לקבל את ההודעות המגיעות ממנו	rostopic echo *chatter*
rostopic info <topic>	קבלת מידע על topic ספציפי	rostopic info *chatter*
rostopic pub <topic> <msg_type> <args>	פרסום הודעה בודדת אל topic ספציפי	rostopic pub *chatter std_msgs/String "hello there"*

4.8.4 הודעות (Messages)

הודעות הן מבני הנתונים המשמשים לתקשורת בין צמתים ב-ROS. לכל topic משויך סוג הודעה ספציפי. ROS מספק סוגי הודעות סטנדרטיים רבים בחבילת std_msgs, כמו כן חבילות אחרות עשויות להגדיר הודעות מותאמות מסוגים שונים.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.8.4.1 סוגי הודעות

להלן רשימה של סוגי ההודעות הבסיסיות:

std_msgs/String

std_msgs/Int32

std_msgs/Float32

std_msgs/Bool

כפי שניתן לראות ניתן לבנות הודעות העושות שימוש בטיפוסי נתונים סטנדרטיים כדוגמה מחרוזת, מספר שלם, מספר ממשי ומשתנים מטיפוס בוליאני. בנוסף על כך ניתן ליצור הודעות מותאמות אישית כלומר הודעה המתאימה במבנה הנתונים שלה למשימה רובוטית ספציפית, לדוגמה הודעות גיאומטריות geometry_msgs המאפשרות להציג מיקום גופים, על פני מרחב תלת מימדי. להלן דוגמה למספר הודעות מותאמות אישית:

- **geometry_msgs/Point:** A message type for representing a 3D point.
- **geometry_msgs/Pose:** A message type for representing a 3D pose.
- **geometry_msgs/Transform:** A message type for representing a 3D transformation.
- **geometry_msgs/Quaternion:** A quaternion for representing orientations.
- **geometry_msgs/Twist:** A combination of linear and angular velocities.

4.8.4.2 הגדרת הודעה

סוג הודעה מוגדר בקובץ msg. המכיל את מבנה הנתונים של ההודעה. הקובץ מורכב מרצף של הצהרות, אחת בכל שורה, כאשר לכל שדה יש סוג ושם. להלן דוגמה למבנה הודעות msg:

```
flaot 64  x
flaot 64  y
flaot 64  z
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.8.4.3 יצירת הודעה מותאמת אישית

כאשר סוגי ההודעות הרגילות, כפי שתיארנו בתחילה, אינם מתאימים לצרכים שלנו ניתן ליצור הודעות מותאמות אישית עבור חבילות ה-ROS. כדי לעשות זאת יש לבצע את השלבים הבאים:

- צרו תיקייה חדשה בשם msg בתוך בחבילה.
- בתיקייה msg, צרו קובץ msg. עם מבנה ההודעה הרצוי.
- יש לשנות את קבצי CMakeLists.txt ו-package.xml של החבילה כדי לכלול את התלות הנדרשות והוראות בנייה.

4.8.4.4 הודעות בתוך הודעה (Nested Messages)

הודעות יכולות לכלול גם הודעות אחרות כשדות בתוך ההודעה. הדבר מאפשר ליצור מבני נתונים מורכבים על-ידי שילוב סוגי הודעות קיימות.

לדוגמה, סוג ההודעה geometry_msgs/Pose כוללת את ההודעות geometry_msgs/Point ו-geometry_msgs/Quaternion כשדות:

```
geometry_msgs/Point position
geometry_msgs/Quaternion orientation
```

4.8.4.5 הודעות בתוך קוד (Messages in Code)

ניתן לייבא הודעות לתוך קוד הכתוב בשפת Python או C++, ולהשתמש בסוגי הודעות כמו כל מודול או מחלקה אחרים. לדוגמה, ב-Python, ניתן לייבא הודעת String מחבילת std_msgs ולאחר מכן ליצור אותה באופן הבא:

```
from std_msgs.msg import String
msg = String ()
msg.date = “Hello World”
```

לסיכום, הבנת הודעות ROS חיונית לפיתוח מערכות רובוטיקה מבוססות ROS מכיוון שהן מספקות את האמצעים לתקשורת בין רכיבים שונים של רובוט. ניתן ליצור ולהשתמש בהודעות כדי להחליף מידע בין חיישנים, מפעילים ואלגוריתמים. כמו כן לאפשר לרובוט

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מור-טק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

לתפקד ביעילות. ישנם מספר פקודות המיועדות לשורת הפקודה (Command) להלן מספר דוגמאות (טבלה 4):

טבלה 4. מספר דוגמאות להוראות שימושיות לעבודה עם הודעות

פקודה	משמעות	דוגמה
rosmsg list	קבלת המידע על כל סוגי ההודעות	rosmsg list
rosmsg show <<MessageType	קבלת אינפורמציה על סוג הודעות מסוים	rosmsg show *turtle_actionlib/Velocity*

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.9 Creating New Packages in ROS

בפרק זה נציג כיצד ניתן ליצור חבילות חדשות ב-ROS. מטרת המדריך היא לספק הסבר ברור של השלבים הנדרשים ליצירה וניהול של חבילות ROS.

ב-ROS, חבילה היא היחידה העיקרית לארגון תוכנה. היא מכילה קוד מקור, קובצי תצורה ומשאבים אחרים הדרושים לבנייה והפעלה של יישום ספציפי. החבילות מאפשרות למפתחים להפיץ ולעשות שימוש חוזר בקוד שלהם בקלות.

4.9.1 יצירת חבילת ROS חדשה

כדי ליצור חבילת ROS חדשה, בצעו את השלבים הבאים:

1. פתחו חלון Terminal חדש.
2. נווטו והגיעו אל ספריית src של סביבת העבודה שלכם, לדוגמה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src
```

3. השתמשו בפקודה catkin_create_pkg כדי ליצור חבילה חדשה.

```
catkin_create_pkg your_package_name dependency_1 dependency_2 ...
```

החליפו את המילים your_package_name בשם הרצוי עבור החבילה שלכם, ואת dependency_1, dependency_2, ... עם התלות הנדרשת לכם לדוגמה, כדי ליצור חבילה בשם morbot_control עם התלות std_msgs ,roscpp, rospy ו-sensor_msgs, הפעילו את הפקודה הבאה:

```
catkin_create_pkg morbot_control rospy roscpp std_msgs sensor_msgs
```

החבילה החדשה תיווצר בתוך התיקייה src של סביבת העבודה שלכם. החבילה תכיל מספר קבצים וספריות, בין היתר את הקבצים הבאים:

- CMakeLists.txt: קובץ ה-Build CMake עבור החבילה.
- package.xml: מניפסט החבילה, המכיל מטא נתונים על החבילה, כגון השם, הגרסה, התיאור והתלות שלה.

4.9.2 בניית החבילה

כדי לבנות את חבילת ה-ROS החדשה שלנו, יש לבצע את השלבים הבאים:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

1. נווטו אל תיקיית השורש של סביבת העבודה ROS שלנו תוך שימוש בהוראה הבאה:

```
cd ~/MorBot/morbot_ros/morbot_ws
```

2. הפעילו את הפקודה catkin_make לצורך בניית התבנית לחבילה:

```
catkin_make
```

3. לאחר שהחבילה נוצרה יש להריץ את הקובץ setup.bash שבתוך סביבת העבודה כדי להפוך את החבילה החדשה לזמינה בסביבת ה-ROS שלך:

```
source devel/setup.bash
```

4.9.3 הוספת צמתים וקבצי הפעלה

לאחר יצירת ובניית חבילת ה-ROS, יש להוסיף צמתים וקבצי הפעלה כדי ליישם את הפונקציונליות הרצויה. צמתים הם קבצי הפעלה המבצעים משימות ספציפיות, קבצי Launch עוזרים להפעיל מספר צמתים ולהגדיר פרמטרים בו זמנית , מוסבר בהמשך .

4.9.3.1 הוספת צומת - Node

כדי להוסיף צומת חדשה הכתובה בשפת Python לחבילה, יש לבצע את השלבים הבאים:

1. צרו תיקייה חדשה בשם scripts בתוך תיקיית העבודה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/your_package_name

 mkdir scripts
```

2. צרו קובץ קוד חדש הכתוב בשפת Python בתוך התיקייה scripts. לדוגמה, כדי ליצור קובץ Python בשם my_node.py כתבו את ההוראות הבאות:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
cd scripts

touch my_node.py

sudo chmod 777 my_node.py
```

3. פתחו קובץ המקור וכתבו את הקוד הרצוי, לדוגמה:

```
#!/usr/bin/env python

import rospy

print("My Cool ROS python Node!!")
```

במידה ותרצו לשלב קטעי קוד הכתובים ב- C++ כחלק מהצומת בצעו את השלבים הבאים:

1. צרו תיקייה חדשה בשם src בתוך התיקייה המכילה את קבצי העבודה, לדוגמה:

~/morbot_ws/src/your_package_name

ניתן ליצור את התיקייה על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/your_package_name

mkdir src
```

לדוגמה, החבילה בשם morbot_control תיכתב כך:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/morbot_control

mkdir src
```

2. צרו קובץ קוד חדש בתוך ספריית src. לדוגמה, כדי ליצור קובץ C++ בשם my_node.cpp הקלידו את ההוראות הבאות:

```
cd src

touch my_node.cpp
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

3. פתחו את קובץ המקור וכתבו את קוד הצומת, בהתאם להנחיות התכנות של ROS ושימוש בספריות ROS המתאימות.
4. שנו את קובץ CMakeLists.txt בספריית החבילה של הפריקט כדי לכלול את ה-Node החדש במהלך תהליך הבנייה. לדוגמה, אם יצרתם קובץ בשם my_node.cpp הקלידו את ההוראות הבאות לקובץ CMakeLists.txt:

```
add_executable(my_node nodes/my_node.cpp)

target_link_libraries(my_node ${catkin_LIBRARIES})
```

5. בנו שוב את החבילה באמצעות הפקודה catkin_make:

```
cd ~/MorBot/morbot_ros/morbot_ws

catkin_make
```

6. לאחר שהחבילה נוצרה, הריצו את הקובץ setup.bash שבתוך סביבת העבודה כדי להפוך את החבילה החדשה לזמינה בסביבת ה-ROS:

```
source devel/setup.bash
```

4.9.3.2 יצירת קובץ הפעלה Launch

קבצי Launch של ROS הם קובצי XML המשמשים להגדרה והפעלה של מספר צמתים בו זמנית כולל אפשרות להעביר פרמטרים והגדרות בהתאם לדרישות ההפעלה של רובוט ספציפי. כמו כן קבצי Launch מספקים דרך נוחה להפעיל מערכת מורכבת בפקודה אחת. קבצים אלה מקבלים את הסיומת launch. כדי להוסיף קובץ Launch חדש לחבילה, בצעו את השלבים הבאים:

1. צרו תיקייה חדשה בשם launch בתוך תיקיית העבודה שלכם:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/your_package_name

mkdir launch
```

2. צרו קובץ קוד חדש בשם launch_file.launch בתוך התיקייה launch. לדוגמה, כדי ליצור קובץ launch בשם my_launch_file.launch כתבו את ההוראות הבאות:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
cd launch

touch my_launch_file.launch
```

3. פתחו את הקובץ באמצעות עורך טקסט וכתבו את קוד ה-XML כדי להגדיר את הצמתים והפרמטרים שתרצו להפעיל. לדוגמה, כדי להפעיל את my_node שיצרתם קודם לכן כתבו את הקוד הבא:

```
<launch>

<node name="my_node" pkg="your_package_name" type="my_node.py"
output="screen" />

</launch>
```

* יש להחליף את המילה your_package_name בשם החבילה שלכם. התכונת name מציינת את שם ה-Node, בעוד שהתכונה pkg מצביעה על החבילה המכילה את ה-Node. תכונת ה-type מציינת את שם ההפעלה, שאמור להתאים לשם שניתן בקובץ CMakeLists.txt. לדוגמה, אם שם החבילה הוא morbot_control לכן:

```
<launch>

 <node name="my_node" pkg="morbot_control" type="my_node.py"  output="screen"
/>

</launch>
```

4. שמרו את הקובץ וסגרו את העורך.
5. בנו שוב את החבילה שלכם באמצעות הפקודה catkin_make:

```
cd ~/MorBot/morbot_ros/morbot_ws

catkin_make
```

6. לאחר שהחבילה נוצרה הריצו את הקובץ setup.bash שבתוך סביבת העבודה כדי להפוך את החבילה החדשה לזמינה בסביבת ה-ROS:

```
source devel/setup.bash
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7. הפעילו את קובץ launch באמצעות הפקודה roslaunch:

```
roslaunch your_package_name my_launch_file.launch
```

לדוגמה, החבילה בשם morbot_control שיצרתם קודם לכן:

```
roslaunch morbot_control my_launch_file.launch
```

כעת, הוספתם בהצלחה צמתים וקבצי הפעלה לחבילת ה-ROS. ניתן להמשיך ולפתח את החבילה על ידי הוספת צמתים נוספים, קבצי launch או משאבים אחרים לפי הצורך. יש לזכור לבנות את החבילה על ידי הקובץ setup.bash לאחר ביצוע השינויים.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.10 ROS PubSubPy

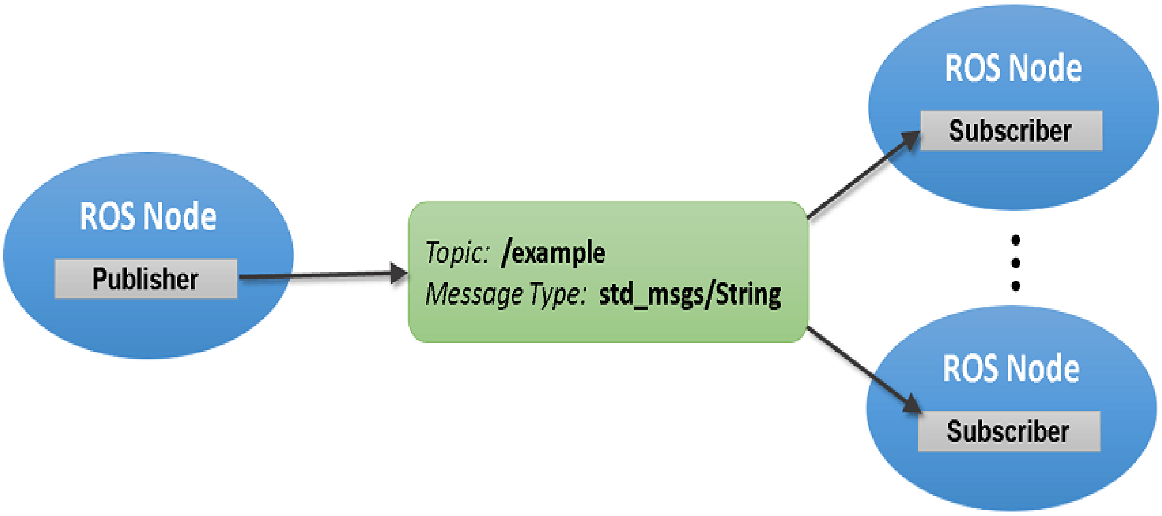
נושאי הפרק הם:

- יצירת חבילה חדשה.
- יצירת מפרסם חדש publisher.
- יצירת מנוי למפרסם חדש subscriber.
- הפעלת מפרסם ומנוי למפרסם.
- הצגת publisher and subscriber nodes.
- הצגת publisher and subscriber topics.
- Multiple Publishers and Subscribers in ROS.

4.10.1 מושגים מרכזיים

- Publisher – קוד תוכנה המייצר Node שפרסם את הודעות.
- Subscriber- קוד תוכנה המייצר Node הנרשם לקבלת הודעה.
- Callback - קריאה לפונקציה ע"י Node הנרשם שמקבלת את ההודעה המפורסמת ומבצעת עליה פעולה מסויימת , לדוגמא קריאה והדפסה על המסך של המסר שהתקבל
- Node - צומת מידע.

איור 3 מתאר מבנה שליחת הודעות בין מפרסם למנוי למפרסם:



איור 3. תרשים המתאר את מהלך העברת המידע בין המפרסם למנוי למפרסם

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.10.2 מהי חבילת ה-PUB/SUB?

Pub/Sub הוא שירות הודעות אסינכרוני המאפשר לשירותים שונים לתקשר עם חבילות מידע שעוברות ברשת. ROS משמש להזרמת מידע וקבלתו על ידי יצירת צינורות המעברים מידע משירותים שונים כמו: ROBOTICS, CLIENT SERVICES, IOT.

לדוגמה ניתן לבנות חבילת Pub/Sub שמצד אחד יהיה Node המאפשר לקרוא נתונים מחיישן טמפרטורה ולפרסם אותם. מצד שני לכתוב Node הנרשם כמנוי במטרה לקבל את נתוני הטמפרטורה ולהפעיל מאוורר לצורך ביצוע קירור.

המטרה בעולם הרובוטיקה היא ליצור מערכת הודעות בין המפרסם והמנוי בסביבת ROS ובעזרת מערכת הודעות אילו ניתן לתקשר בין המערכות השונות של הרובוט במטרה להניע אותו ולבצע משימות.

4.10.3 דרישות קדם

- ידע במערכת הפעלה Linux.
- שימוש במכונה וירטואלית Ubuntu.
- כתיבת nodes.
- ידע בכתיבת תוכנית בשפת Python.

4.10.4 תרגול בבניית מערכת ההודעות בין מפרסם למנוי למפרסם

1. לצורך יצירת חבילה בסביבת עבודה של ROS, היכנסו לסביבת העבודה של ROS לתיקיה src

   cd ~/MorBot/morbot_ros/morbot_ws/src

2. צרו חבילה חדשה בשם simple_pub_sub בעזרת הפקודה catkin_create_pk

```
catkin_create_pkg   simple_pub_sub    rospy std_msgs
```

3. בתוך התיקיה של החבילה צרו תיקייה בשם scripts

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub

mkdir scripts
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4. צרו Node חדש המיועד לפרסם publisher הודעות חדשות. הדגימו קוד מפרסם המדפיס הודעה כול שנייה מטיפוס מחרוזת במבנה הבא:

```
"Hello, ROS! %s" % rospy.get_time()%s
```

הפעולה rospy.get_time מחזירה את השעה הנוכחית של המחשב. להלן השלבים בתהליך יצירת Node מפרסם:

1. צרו קובץ בשם talker.py בתיקיה scrips

```
cd scripts

touch talker.py
```

2. לאחר יצירת הקובץ הפכו את הקוד שכתוב בו לקובץ שניתן להרצה. לשם כך היכנסו לתיקייה scripts וכתבו את ההוראה הבאה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts

sudo chmod +x talker.py
```

3. פתחו את הקובץ talker.py במצב עריכה והעתיקו אליו הקוד הבא:

```
#!/usr/bin/env python
# Import the required modules
import rospy
from std_msgs.msg import String

# Define the main function for the publisher
def talker():
    # Initialize the ROS node with a unique name
    rospy.init_node('talker', anonymous=True)

    # Create a publisher object that will publish messages to the 'chatter' topic
    pub = rospy.Publisher('chatter', String, queue_size=10)
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    # Set the publishing rate to 1 message per second
    rate = rospy.Rate(1)

    # Keep publishing messages until the node is shut down
    while not rospy.is_shutdown():
        # Create the message with the current ROS time
        message = "Hello, ROS! %s" % rospy.get_time()

        # Log the message to the console
        rospy.loginfo(message)

        # Publish the message to the 'chatter' topic
        pub.publish(message)

        # Sleep for a while to maintain the publishing rate
        rate.sleep()
# Check if the script is being run as the main program
if __name__ == '__main__':
    try:
        # Call the main function to start the publisher node
        talker()
    except rospy.ROSInterruptException:
        # If an exception occurs, exit gracefully without displaying an error message
        pass
```

הסבר הקוד:

שורה זו מייבאת את ספריית המפרסם rospy ואת ספרית String מהספרייה std_msgs.msg

```
import rospy
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
from std_msgs.msg import String
```

פונקציה ראשית של המפרסם, הודעה – המשתנה message נשלחת בקצב של 10 פעמים בשנייה.

```
def talker():
    rospy.init_node('talker', anonymous=True)
    pub = rospy.Publisher('chatter', String, queue_size=10)
    rate = rospy.Rate(1)
```

הלולאה הראשית יוצרת את ההודעה, כותבת את ההודעה לקובץ הלוג, מפרסמת, המאפשרת לשלוח את ההודעה הבאה בקצב-זמן הנדרש כך שהיא גורמת למערכת ההפעלה להשהות את ההרצה כדי לאפשר לשלוח את ההודעה הבאה – כלומר סוג של סינכרון בין ההודעות בעזרת sleep:

```
while not rospy.is_shutdown():
    message = "Hello, ROS! %s" % rospy.get_time()
    rospy.loginfo(message)
    pub.publish(message)
    rate.sleep()
```

זימון הפונקציה הראשית עם אפשרות לטיפול בחריגה, כדי שהתוכנית לא תקרוס – טיפול בחריגות:

```
if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

יצירת Node המקבל את הודעות הפרסם:

1. צרו Node חדש new subscriber node על ידי יצירת קובץ בשם listener.py בספריה scripts:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch listener.py
```

2. הפכו את הקוד כך שניתן יהיה להריץ אותו:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
chmod +x listener.py
```

3. פתחו את הקובץ listener.py בעורך טקסט, והעתיקו אילו את הקוד הבא:

```
#!/usr/bin/env python
import rospy
from std_msgs.msg import String


def callback(msg):
    rospy.loginfo("I heard: %s", msg.data)


def listener():
   rospy.init_node('listener', anonymous=True)
   rospy.Subscriber('chatter', String, callback)
   rospy.spin()


if __name__ == '__main__':
    listener()
```

קוד זה מגדיר Node שמאזין ל-topic בשם chatter עבור הודעות מחרוזת. כאשר מתקבלת הודעה, נקראת פונקציית ה-callback, אשר רושמת את נתוני ההודעה המתקבלת באמצעות הפעולה rospy.loginfo

הסבר הקוד

השורה להלן מציינת למערכת ההפעלה שזהו קוד בשפת Python:

```
#!/usr/bin/env python
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

שורה להלן מייבאת את הספריית rospy ואת סוג הודעת מהחבילה std_msgs.

```
import rospy
from std_msgs.msg import String
```

הפונקציה callback מזומנת כאשר מקבלת הודעה. במצב זה תוכן ההודעה נרשם כפלט של התוכנית.

```
def callback(msg):
    rospy.loginfo("I heard: %s", msg.data)  # Log the received message data
```

הפונקציה listener רושמת את ה- Node כמנוי ל-chatter ומזמנת לבד את הפונקציה callback כאשר מתקבל String המכיל מידע.

```
def listener():
    rospy.init_node('listener', anonymous=True)
    rospy.Subscriber('chatter', String, callback)
    rospy.spin()
```

הקוד הבא מזמן בתחילת התוכנית את הפונקציה listener

```
if __name__ == '__main__':
    listener()
```

4. לצורך בניית החבילה בסביבת morbot_ws כתבו את הפעולות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source ~/MorBot/morbot_ros/morbot_ws/devel/setup.bash
```

5. לצורך הפעלת כל החבילה יחד בצעו את הפעולות הבאות: פתחו שלוש חלונות של Command. בחלון ראשון הפעילו את rosscore. בחלון השני הפעילו את ה-Node המפרסם את ההודעות. ובחלון השלישי הפעילו את ה-Node המנוי להודעות שמפרסם ה-Node הראשון.

להלן שלוש ההוראות שיש לכתוב, כל הוראה צריכה לרוץ בחלון Command מפרד.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מור-טק | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

חלון ראשון

```
rosscore
```

חלון שני

```
rosrun simple_pub_sub talker.py
```

חלון שלישי

```
rosrun simple_pub_sub listener.py
```

6. לצורך בדיקה הציגו מספר נתונים על המסך. לשם כך פתחו חלון נוסף, רביעי במספר והריצו בו את ההוראה הבאה:

```
rosnode list
```

הוראה זו תציג על גבי המסך את רשימת כל ה-Nodes הקיימים.

```
rosnode info /talker
```

הוראה זו תציג על גבי המסך מידע על ה-talker

```
rosnode info /listener
```

הוראה זו תציג על גבי המסך מידע על ה-listener

7. הצגת ה- topic על גבי המסך:
- פתחו חלון חדש.
- הציגו רשימת הנושאים הפעילים על ידי ההוראה: rostopic list.
- הציגו מידע על ה-topic על ידי ההוראה /chatter topic: rostopic info /chatter.
- הציגו את תוכן ההודעה שמפורסמת על ידי ה-Node תוך שימוש בהוראה הבאה: chatter topic: rostopic echo /chatter/
- צפו בהודעות המתפרסמות על ידי ה-topic כפי שמתקבלות ב-listener.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.10.5 כתיבת מספר מפרסמים ומאזינים ב-ROS

בפעילות זו תרחיבו את שני קטעי הקוד שכתבתם "Talker" ו-"Listener" במטרה להדגים כיצד שני צמתים "Nodes" מעבירים ביניהם מספר Topics.

בעבודה עם ROS, ניתן למצוא תרחישים שבהם מספר מפרסמים ומנויים פועלים במקביל. מצב זה נוצר מכיוון שצמתים (Nodes) שונים נדרשים להתמודד עם היבטים שונים של מערכת רובוטית, כגון קליטת נתונים ממספר חיישנים, עיבוד נתונים ובקרת מפעילים שונים. לכן, יש צורך במספר מפרסמים ומנויים כדי להקל על התקשורת בין הצמתים.

4.10.5.1 יצירת multiple_publisher node

1. צרו קובץ קוד חדש בשם multitopic_talker.py בתוך התיקייה scripts.

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch multitopic_talker.py
```

הפכו את הקובץ שיצרתם בצורה כזו שניתן להרצה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
sudo chmod +x multitopic_talker.py
```

2. פתחו את הקובץ multitopic_talker.py בעורך טקסט, והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python
# Import the required modules
import rospy
from std_msgs.msg import String, Int32
# Define the main function for the publisher
def talker():
    # Initialize the ROS node with a unique name
    rospy.init_node('multitopic_talker', anonymous=True)
   # Create a publisher object that will publish messages to the 'chatter' topic
    pub = rospy.Publisher('chatter', String, queue_size=10)
   # Create a second publisher that will publish messages to the 'numbers' topic
    pub2 = rospy.Publisher('numbers', Int32, queue_size=10)
```


משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מור-טק | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
 # Set the publishing rate to 1 message per second
    rate = rospy.Rate(1)
    # Initialize an integer to publish
    number = 0
    # Keep publishing messages until the node is shut down
    while not rospy.is_shutdown():
        # Create the message with the current ROS time
        message = "Hello, ROS!"
       # Log the message to the console
        rospy.loginfo(message)
        # Log the integer to the console
        rospy.loginfo(number)
       # Publish the message to the 'chatter' topic
        pub.publish(message)
       # Publish the integer to the 'numbers' topic
        pub2.publish(number)
        # Increment the number
        number += 1
        # Sleep for a while to maintain the publishing rate
        rate.sleep()
# Check if the script is being run as the main program
if __name__ == '__main__':
    try:
        # Call the main function to start the publisher node
        talker()
   except rospy.ROSInterruptException:
        # If an exception occurs, exit gracefully without displaying an error message
        Pass
```

להסברים אודות הקוד לחצו כאן.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.10.5.2 יצירת multiple_ subscriber node

1. צרו קובץ קוד חדש בשם multitopic_listener.py בתוך התיקייה scripts.

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch multitopic_listener.py
```

הפכו את הקובץ לניתן להרצה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
chmod +x multitopic_listener.py
```

2. פתחו את הקובץ multitopic_listener.py בעורך טקסט, והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python
# Import the rospy and String, Int32 message type from the std_msgs
import rospy from std_msgs.msg import String, Int32
# Define the callback function that will be executed when a message is received on the 'chatter' topic
def chatter_callback(msg):
    rospy.loginfo("I heard on chatter: %s", msg.data)  # Log the received message data
# Define another callback for the 'numbers' topic
def numbers_callback(msg):
    rospy.loginfo("I heard on numbers: %d", msg.data)
# Define the main function for the subscriber node
def listener():
    rospy.init_node('multitopic_listener', anonymous=True)  # Initialize a ROS node named 'listener', with the anonymous flag set to True
    rospy.Subscriber('chatter', String, chatter_callback)  # Create a subscriber that listens to the 'chatter' topic and calls the 'chatter_callback' function when a message is received
    rospy.Subscriber('numbers', Int32, numbers_callback)  # Create another subscriber that listens to the 'numbers' topic and calls the 'numbers_callback' function when a message is received
    rospy.spin()  # Keep the node running and processing callbacks until it is shut down
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
# Check if the script is being run as the main program and not imported as a module
if __name__ == '__main__':
    listener()  # Call the listener() function to start the subscriber node
```

הסברים אודות הקוד לחצו כאן.

4.10.5.3 בניית החבילה - package

1. פתחו חלון Command חדש.
2. עברו לתיקיית הפרויקט ובנו את החבילה על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source ~/MorBot/morbot_ros/morbot_ws/devel/setup.bash
```

4.10.5.4 הפעלת multitopic publisher and subscriber nodes

פתחו שלוש חלונות Command חדשים, בכל אחד הפעילו הוראה אחרת:

1. בחלון הראשון הריצו master node: roscore

```
roscore
```

2. בחלון השני הריצו publisher node:

```
rosrun simple_pub_sub multitopic_talker.py
```

3. בחלון השלישי הריצו subscriber node

```
rosrun simple_pub_sub multitopic_listener.py
```

ע"י ביצוע הקוד הנ"ל יצרתם בהצלחה מפרסם ומנוי באמצעות Python ב-ROS.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.11 ROS turtlesim

<u>מטרות לימודיות</u>

- הכרת הפלטפורמה לסימולציה המבוססת על turtle.
- הכרת הספריות שמשתמשים בהן.
- חקירת הנושאים של ROS ב-turtlesim.
- מעקב אחרי תנועות הצב המדמה את הרובוט.
- שליטה בתכנות תנועות הצב המדמה את הרובוט.

4.11.1 מבוא

כמו בכל פרויקט הנדסי אמיתי, צריך וכדאי לבדוק את נכונות התכן בסימולטור, לפני הפעלתו על המערכת האמיתית. הדבר נכון לפני שליחת חללית לירח ועל אחת כמה וכמה לפני הפעלת הרובוטים במעבדה.

במדריך זה תלמדו כיצד להשתמש ב-turtlesim - סימולטור תנועה פשוט. Turtlesim מספק דרך קלה ללמוד את היסודות של ROS, כגון publishing, subscribing ו-services, על ידי שליטה בצב בסביבת סימולציה דו-ממדית.

כדי לקבל את חבילת turtlesim, אין צורך להתקין אותה בנפרד מכיוון שהיא כלולה כחלק מהתקנת ה-ROS הרגילה.

4.11.2 דרישות קדם

- היכרות עם מושגי ROS, כגון nodes, topics ו-messages.
- הבנה בסיסית יצירת חבילות חדשות בפרק ROS.
- כתיבת Publisher פשוט ו-Subscriber ב-Python. לצורך כך יש לחזור לפרק ROS PubSubPy.
- הבנה בסיסית ב-Python.

4.11.3 שלב 1: להתחיל turtlesim

כדי להפעיל את turtlesim, יש פתחו חלון Command והפעילו את הפקודה הבאה:

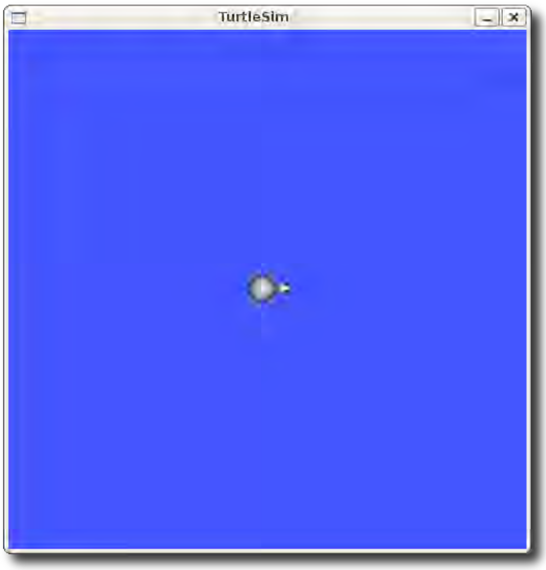
```
roscore
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

פקודה זו מפעילה את ה-ROS Master, המנהל את התקשורת בין צמתים. כעת, פתחו חלון מסוף חדש והפעילו את הפקודה הבאה:

```
rosrun turtlesim turtlesim_node
```

מתקבל חלון חדש עם רקע כחול וצב במרכז. זהו סימולטור turtlesim (תמונה 25):



תמונה 25. חלון סביבת הצבים

4.11.4 שלב 2: שליטה בצב באמצעות Node קיים בשם turtle_teleop_key

פתחו חלון טרמינל נוסף והקלידו את הפקודה הבאה:

```
rosrun turtlesim turtle_teleop_key
```

כעת, לחצו בחלון הטרמינל שבו הפעלתם את turtle_teleop_key והשתמשו במקשי החצים כדי לשלוט בצב. ניתן לראות את הצב נע בתוך הסימולטור.

4.11.5 שלב 3: חקירת topics ב-turtlesim

בזמן שה-turtlesim פועל, פתחו חלון טרמינל נוסף והפעילו את הפקודה הבאה כדי לרשום את רשימת כל ה-topics הזמינים:

```
rostopic list
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מהפלט ניתן ללמוד שיש מספר topics זמינים ל-turtlesim, בין היתר:

/turtle1/cmd_vel

/turtle1/color_sensor

/turtle1/pose

4.11.6 שלב 4: מעקב אחר מיקום הצב

כדי לראות את מיקומו של הצב בזמן אמת, רשמו כמנוי ל-topics לנושא /turtle1/pose באמצעות ההוראה הבאה:

```
rostopic echo /turtle1/pose
```

כעת ניתן לראות שבזמן הזזת הצב באמצעות מקשי החצים, תקבלו topic המציג את מיקומו של הצב בתוך הסימולציה. בין הפרטים המתקבלים ניתן לראות שיש לנו מידע על הכיוון המהירויות הזוויתיות של הצב בתוך חלון הטרמינל.

4.11.7 שלב 5: כתיבת קוד ב-Python לשליטה על הצב

נווטו אל התיקייה scripts בחבילה simple_pub_sub וצרו קובץ קוד חדש בשם move_turtle.py על ידי ההוראה הבאה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch move_turtle.py
chmod +x move_turtle.py
```

פתחו את הקובץ move_turtle.py בעורך טקסט, והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python
# Import necessary modules
import rospy
from geometry_msgs.msg import Twist
# Define the move_turtle function
def move_turtle():
    # Initialize the ROS node with a unique name
```


משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
E-mail: Moretech@ed.technion.ac.il +972-73-3783146 טלפון
https://moretech.technion.ac.il

```
    rospy.init_node('move_turtle', anonymous=True)
    # Create a publisher object to publish messages to the /turtle1/cmd_vel topic
    pub = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
    # Create a rate object to control the loop rate (1 message per second)
    rate = rospy.Rate(1)
    # Continue publishing messages until the ROS node is shut down
    while not rospy.is_shutdown():
        # Create a Twist message object to store linear and angular velocities
        vel = Twist()
        # Set linear and angular velocities for the turtle
        vel.linear.x = 2.0
        vel.angular.z = 1.0
        # Log the velocities as an info-level log message
        rospy.loginfo("Moving the turtle: linear_x = %s, angular_z = %s", vel.linear.x, vel.angular.z)
        # Publish the velocities to the /turtle1/cmd_vel topic
        pub.publish(vel)
        # Sleep for the required time to maintain the desired message rate (1 message per second)
        rate.sleep()
# Check if the script is being run as the main program and not being imported as a module
if __name__ == '__main__':
    # Try to call the move_turtle function and start the publisher node
    try:
        move_turtle()
        # Catch any ROSInterruptException that may occur (e.g., when the script is terminated using Ctrl+C)
    except rospy.ROSInterruptException:
        # Gracefully exit the script without displaying an error message
        pass
```


הפקולטה לחינוך למדע וטכנולוגיה
הטכניון מכון טכנולוגי לישראל
מור-טק
המינהל לטכנולוגיה ומערכות תקשוב
משרד החינוך המזכירות הפדגוגית אגף א' למדעים
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

שמרו את הקובץ וסגרו את עורך הטקסט.

קוד זה יוצר Node חדש בשם move_turtle המפרסם הודעת Twist לתוך topic /turtle1/cmd_vel. הודעת Twist מכילה שני שדות: כיוון מהירויות זוויתיות. כל שדה בנוי מהודעת Vector3 המכילה שלושה ערכים: x, y ו-z. בדוגמה זו, אתם משנים את הערכים x ו-z של המהירות הזוויתית ל-2.0 ו-1.0 בהתאמה. המשמעות היא שהצב ינוע קדימה במהירות ליניארית של 2.0 מ' לשנייה ומסתובב במהירות זוויתית של 1.0 רד/שנייה.

הפעילו את הקוד, וודאו שה-turtlesim פועל. לאחר מכן הפעילו את הפקודה הבאה בחלון Command חדש:

```
rosrun simple_pub_sub move_turtle.py
```

ניתן לראות את הצב נע בסיבובים בתוך הסימולטור.

למדתם את היסודות של turtlesim והשתמשנו בהם בכדי לחקור את מושגי ROS כגון topics, publishers ו-subscribers.

4.11.8 שינוי צבע הרקע של Turtlesim

נווטו אל תיקיית scripts שבחבילת simple_pub_sub וצרו קובץ קוד חדש בשם change_color.py להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch change_color.py
chmod +x change_color.py
```

פתחו את הקובץ change_color.py בעורך טקסט והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python
import rospy
from std_srvs.srv import Empty
def change_background_color(r, g, b):
    rospy.init_node('change_background_color', anonymous=True)
    # Set the background color parameters
    rospy.set_param('/turtlesim/background_r', r)
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    rospy.set_param('/turtlesim/background_g', g)
    rospy.set_param('/turtlesim/background_b', b)

    # Wait for the clear service and call it
    rospy.wait_for_service('clear')
    try:
        clear_background = rospy.ServiceProxy('clear', Empty)
        clear_background()
    except rospy.ServiceException as e:
        print("Service call failed: %s"%e)
if __name__ == "__main__":
    # Change the background color to red
    change_background_color(255, 0, 0)
```

יש לשמור את הקובץ ולסגור את עורך הטקסט.

קוד זה יוצר Node בשם change_background_color שמשנה את צבע הרקע של ה-turtlesim לאדום. צבע הרקע נקבע באמצעות הפרמטרים:

/turtlesim/background_r

/turtlesim/background_g

/turtlesim/background_b

לאחר מכן, ה-service מנקה את סביבת העבודה שבסימולטור ומציג את צבע רקע חדש.

פתחו חלון Command חדש והפעילו בו את הקוד שכתבתם. וודאו שה-turtlesim פועל, ולאחר מכן הפעילו את הפקודה הבאה:

```
rosrun simple_pub_sub change_color.py
```

הפונקציה change_background_color לוקחת שלושה משתנים, שכל אחד מהם מייצג את עוצמת הצבע האדום, הירוק והכחול בהתאמה. הערכים יכולים לנוע בין 0 ל-255, כאשר:

- 0 אומר שהצבע נעדר לחלוטין (ללא עוצמה).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- 255 אומר שהצבע קיים במלואו (העוצמה הגבוהה ביותר).

לאחר הפעלת הקוד ניתן לראות את צבע הרקע של סביבת הצבים משתנה בהדרגה לאדום.

4.11.9 הזזת הצב Turtle על סמך צבע הרקע

השתמשו בחיישן צבע כדי לזהות את צבע הרקע והזיזו את ה-Turtle בהתאם לערכי ה-RGB שצוינו (אדום במקרה זה). צבע הנתיב שהצב מצייר מצוין על ידי path_color. חיישן הצבע מתעלם מצבע זה. נווטו אל התיקייה scripts של חבילת simple_pub_sub וצרו קובץ קוד חדש בשם color_moving.py להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch color_moving.py
chmod +x color_moving.py
```

פתחו את הקובץ color_moving.py בעורך טקסט, והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python

import rospy
from geometry_msgs.msg import Twist
from turtlesim.msg import Color

# Global variable to hold the color detected by the turtle
color_detected = None

# Color of the path drawn by the turtle
path_color = Color(r=179, g=184, b=255)

# Callback for color sensor
def color_callback(color):
    global color_detected

    # Ignore if the color matches the path color
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    if color.r == path_color.r and color.g == path_color.g and color.b == path_color.b:
        return
    color_detected = color
# Function to move the turtle in a circle
def move_in_circle():
    # global color_detected
    # Create a publisher for the turtle velocity
    publisher = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
    # Initialize the node
    rospy.init_node('move_in_circle', anonymous=True)
    # Create a subscriber for the color sensor
    rospy.Subscriber('/turtle1/color_sensor', Color, color_callback)
    print(color_detected)
    # Set the rate
    rate = rospy.Rate(1)  # 1Hz
    # Loop until shutdown
    while not rospy.is_shutdown():
        if color_detected is None:
            continue
        # Only move in circle if the color beneath the turtle is red
        if color_detected.r == 255 and color_detected.g == 0 and color_detected.b == 0:
            rospy.loginfo("MOVING")
            # Create a Twist message and set linear.x and angular.z
            twist = Twist()
            twist.linear.x = 2.0  # Speed value
            twist.angular.z = 2.0  # Turning rate
            # Publish the Twist message
            publisher.publish(twist)
        rate.sleep()
if __name__ == '__main__':
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
try:
    move_in_circle()
except rospy.ROSInterruptException:
    pass
```

שמרו את הקובץ וסגרו את עורך הטקסט.

קוד זה יוצר Node בשם move_in_circle שמזיז את הצב במעגל אם הצבע מתחת לצב אדום. חיישן הצבע המקושר ל-topic בשם turtle1/color_sensor/. פונקציית color_callback מזומנת בכל פעם שמתקבלת הודעה דרך topic זה. פונקציית color_callback משנה את הצבע שזוהה על ידי הצב במשתנה גלובלי color_detected. הפונקציה move_in_circle בודקת את הערך של color_detected ומזיזה את הצב במעגל אם הצבע שזוהה הוא אדום.

פתחו חלונות Command חדש והפעילו את הקוד שכתבתם. יש לוודא שה-turtlesim פועל, ולאחר מכן הפעילו את הפקודה הבאה:

```
rosrun simple_pub_sub color_moving.py
```

ניתן לראות את הצב נע במעגל אם הצבע שמתחתיו אדום.

עכשיו שנו את צבע הרקע של חלון Turtlesim, והשתמשו בו כדי להדגים תנועה של רובוט בהתאם לקלט מחיישן.

4.11.10 הפעלת מספר צבים בחלון Turtlesim אחד

נווטו אל תיקיית scripts של חבילת simple_pub_sub וליצור בה קובץ קוד חדש בשם spawn_turtle.py להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/simple_pub_sub/scripts
touch spawn_turtle.py
chmod +x spawn_turtle.py
```

פתחו את הקובץ spawn_turtle.py בעורך טקסט, והעתיקו לתוכו את הקוד הבא:

```
#!/usr/bin/env python
import rospy
from turtlesim.srv import Spawn
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
rospy.init_node('spawn_turtle')
rospy.wait_for_service('spawn')
spawn_service = rospy.ServiceProxy('spawn', Spawn)
# Spawn a new turtle at x=5, y=5, theta=0
spawn_service(5, 5, 0, "turtle2")
```

שמרו את הקובץ וסגרו את עורך הטקסט.

4.11.11 ROS Services

Services הם צורת תקשורת המאפשרת לצומת אחד לבקש משימה מצומת אחר ולקבל מענה. שירות זה מבוסס על בקשה ותגובה לבקשה. בדרך זו אנו מתמודדים עם משימות העשויות להימשך פרק זמן לא יודע.

נקודות מפתח:

1. השירותים הם סינכרוניים: ה-Node המבקש (לקוח) ממתין ל-Node המגיב (שרת) כדי להשלים את המשימה לפני שהוא ממשיך.
2. ה-Services ב-ROS מוגדרים על ידי קבצי srv. המציינים את מבנה הבקשה והודעות התגובה.
3. השתמשו ב-rosservice list כדי לרשום שירותים, rosservice info כדי לקבל פרטים, וב-rosservice call כדי להתקשר לשירות.

הסבר הקוד:

```
rospy.init_node('spawn_turtle')
```

מאתחל Node חדש בשם spawn_turtle.

```
rospy.wait_for_service('spawn')
```

ממתין עד שה-service ששמו spawn יהיה זמין. זהו שירות הניתן על ידי Turtlesim המאפשר לייצר צבים חדשים.

```
spawn_service = rospy.ServiceProxy('spawn', Spawn)
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

יוצר שירות עבור ה-service ששמו spawn. זה מאפשר להתקשר לשירות באמצעות תחביר קריאת פונקציות רגיל.

```
spawn_service(5, 5, 0, "turtle2")
```

קורא לשירות המייצר צב חדש בקואורדינטות (5,5) עם כיוון 0 והשם "turtle2".

כעת ניתן ליצור אינטראקציה עם כל צב בנפרד באמצעות ה-service הייחודיים שלו (לדוגמה, turtle2/cmd_vel עבור פקודות מהירות).

פתחו חלון Command חדש והפעילו את הקוד שכתבם, וודאו שה-turtlesim פועל, ולאחר מכן הפעילו את הפקודה הבאה:

```
rosrun simple_pub_sub spawn_turtle.py
```

פלט התוכנית יראה שני צבים על גבי המסך. כדי לראות את מיקומו של הצב החדש בזמן אמת, הירשמו ל-topic בשם /turtle2/pose באמצעות הפקודה rostopic echo:

```
rostopic echo /turtle2/pose
```

ניתן ליצור אינטראקציה עם כל צב בנפרד באמצעות ה-topics והשמות הייחודיים שניתן לכל צב. לדוגמה:

turtle1/cmd_vel

turtle2/cmd_vel

turtle3/cmd_vel

בשלב הבא ניתן לבנות שוב את ההדגמה הראשונה של MorBot לפי פרק 4.12 ROS-Workshop.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.12 ROS Workshop

בפרק הקודם למדתם להפעיל סימולטור רובוטי בשם turtlesim, נעזרתם בו בכדי לבדוק את הסביבה ולמדתם כיצד לעבוד עם ה-ROS. בפרק זה תבנו חבילת תוכנה אשר תעזור לכם להסיע את הרובוט על גבי מערכת ההפעלה ROS, ותבנו מספר תרחישים אפשריים להפעלת התוכנה. לצורך כך תעשו שימוש בסביבת הפיתוח של Visual Studio Code. את סביבת הפיתוח תתקינו על מחשב שולחני ותתחברו אל הרובוט מרחוק על ידי SSH.

Visual Studio SSH היא תכונה בסביבת הפיתוח של Visual Studio Code המאפשרת חיבור מאובטח לשרתים רחוקים כולל רובוטים. באמצעותה, מפתחים יכולים לתכנת, לשלוט ולנתח את הרובוטים מתוך סביבת הפיתוח Visual Studio Code. בדרך זו ניתן יהיה לאפשר עריכה נוחה של קוד, שליטה בתנועה, גישה לחיישנים ותיעוד בזמן אמת.

בפעילות זו תבנו ROS package חדשה בשם jetbot_control_node בסביבת עבודה morbot_ws שתממש את פעולות ROS המופקדות על תנועה של הרובוט על מנת שנוכל להסיע את JetBot שלכם.

תוך כדי בניית החבילה תלמדו כיצד מפתחים פרויקטים ב-ROS וב-JetBot. כדי להניע את הרובוט דרך ROS תממשו חבילה הכוללת הרכיבים הבאים: ros cmd_vel topics geometry_msgs/Twist ו-JetBot Motion liberis. בפרק זה, תעתיקו מ-MorBot git repository או תיצרו קובץ קוד jetbot_controller.py בתיקיית scripts של החבילה ותתקינו את geometry_msgs ואת rospy.

4.12.1 מושגים מרכזיים

- JetBot – הרובוט שבו נשתמש בסדנה זו הוא קל לבניה ותומך באופן מלא ב-ROS.
- Rospy – ספריה המקשרת בין חבילות ROS לבין Python.
- Topics – ערוצים שבו Nodes מחליפים הודעות ב-ROS.

4.12.2 דרישות קדם

- התקנה של סביבת העבודה ROS, Linux ו-JetBot Nano.
- הבנה בעקרונות ROS כגון node, topic והודעות.
- הבנה בעקרונות Publisher ו-Subscriber.
- יצירת חבילות ב-ROS.
- ידע בשפת התכנות Python.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מל"מ המרכז הישראלי לחינוך מדעי טכנולוגי ע"ש עמוס דה-שליט
מור-טק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.12.3 תרחישי עבודה

סביבת עבודה אחת של ros יכולה לכלול פרויקטים רבים המשתמשים באותה חבילה אך עם מטרות שונות, כדי לארגן טוב יותר פרויקטים שונים, צרו את תיקיית התרחישים בתוך ספריית סביבת העבודה. על כן יש ליצור תיקייה חדשה בשם scenarios בתוך סביבת העבודה שלכם. להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src
mkdir scenarios
```

המטרה העיקרית של פעילות זו היא לספק ניסיון מעשי ולהעמיק את ההבנה במושגי ROS, ובסופו של דבר לאפשר יישום הידע ביישומי רובוטיקה בעולם האמיתי. לצורך כך בנו חבילה חדשה. להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios
catkin_create_pkg workshop rospy geometry_msgs
```

4.12.4 חבילת JetBot control

חבילת jetbot_control_node היא חבילת ROS המיישמת את Node האחראי על השליטה ברובוט JetBot. הקוד של ה-Node צפוי להירשם ל-topic ששמו cmd_vel/ ומפרסם את מהירויות המנוע המתאימות לרובוט JetBot.

cmd_vel הוא שם נפוץ בנושא ROS לשליחת פקודות מהירות לרובוטים. ה-topic ממומש בדרך כלל עם הודעות מסוג geometry_msgs/Twist, כיוון ומהירות.

צרו חבילה חדשה בשם jetbot_control_node בתוך תיקיית סביבת העבודה שלכם על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src
catkin_create_pkg jetbot_control_node rospy geometry_msgs
```

צרו תיקייה חדשה בשם scripts בתוך ספריית החבילה:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
cd ~/MorBot/morbot_ros/morbot_ws/src/jetbot_control_node
 mkdir scripts
```

צרו קובץ קוד חדש בשם jetbot_controller.py בתוך התיקייה scripts.

```
cd scripts
touch jetbot_controller.py
cd ~/MorBot/morbot_ros/morbot_ws/src/jetbot_control_node/scripts
chmod +x jetbot_controller.py
```

4.12.5 כתיבת הקובץ jetbot_controller.py

קוד Python זה שולט ברובוט JetBot באמצעות סביבת ROS. הקוד נרשם ל-topic בשם cmd_vel, המספק פקודות מהירות וכיוון, ולאחר מכן מגדיר את מהירויות המנוע של הרובוט בהתאם.

התרחיש משתמש בספריית JetBot Python כדי לשלוט במנועים של הרובוט. הספרייה מספקת מחלקה ייעודית לך שניתן להשתמש בה כדי לשלוט במנועי הרובוט. למחלקה זו יש פונקציה שם set_motors שניתן להשתמש בה כדי להגדיר את מהירויות המנוע, ופונקציה נוספת בשם set_motors המקבלת שני פרמטרים: הראשון מהירות המנוע השמאלי והפרמטר השני מהירות המנוע הימני. מהירויות המנוע מוגדרות כערכים בין -1.0 ל-1.0, כאשר -1.0 היא מהירות מלאה אחורה, 0.0 עצירה ו-1.0 היא מהירות מלאה קדימה. להן קוד התוכנית:

```
#!/usr/bin/env python3
import rospy
from geometry_msgs.msg import Twist
from jetbot import Robot
def cmd_vel_callback(msg, robot):
    # Extract linear and angular velocity from the message
    linear_x = msg.linear.x
    angular_z = msg.angular.z
    # Calculate motor speeds based on linear and angular velocities
    left_motor_speed = linear_x - angular_z
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    right_motor_speed = linear_x + angular_z
    # Set motor speeds
    robot.set_motors(left_motor_speed, right_motor_speed)
def main():
    print("Start jetbot controller")
    # Initialize the ROS node
    rospy.init_node('jetbot_controller', anonymous=True)
    # Create a Robot object
    robot = Robot()
    # Subscribe to the '/cmd_vel' topic and define the callback function
    rospy.Subscriber('/cmd_vel', Twist, cmd_vel_callback, callback_args=robot)
    rospy.spin()
if __name__ == '__main__':
    try:
        main()
    except rospy.ROSInterruptException:
        # Handle the ROSInterruptException gracefully
        pass
```

להסבר אודות הקוד לחצו כאן.

לאחר שכתבתם את הקוד בנו את החבילה על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```

לצורך הרצת הקוד פתחו מספר חלונות Command ורשמו את ההוראות הבאות:
בחלון הראשון:

```
roscore
```

בחלון השני:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/jetbot_control_node/scripts
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
rosrun jetbot_control_node jetbot_controller.py
```

בחלון השלישי:

```
rostopic pub /cmd_vel geometry_msgs/Twist "linear:
  x: 0.5
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0"
```

ההוראה בחלון השלישי שולחת הודעת Twist ל-topic בשם cmd_vel כדי שיעביר הוראות דרך הקוד למנועים של הרובוט.

כדי לעצור את הרובוט שלחו את הקוד הבא:

```
rostopic pub /cmd_vel geometry_msgs/Twist "linear:
  x: 0.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0"
```

4.12.6 הוספת Teleoperate כדי לשלוט ברובוט דרך מקשי המקלדת

השימוש בחבילת teleop_twist_keyboard מאפשרת לשלוט בתנועות הרובוט על ידי הוצאת פקודות מהמקלדת. קוד זה בדומה למה שכתבתם במפרסם הודעות geometry_msgs/Twist ל-topic בשם cmd_vel.

כדי לעשות זאת בצעו את השלבים הבאים:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

התקינו את חבילת teleop_twist_keyboard על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src
git clone https://github.com/ros-teleop/teleop_twist_keyboard.git
```

בנו את החבילה שהורדתם:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```

הריצו את ה-Node שהורדתם teleop_twist_keyboard

```
rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

וודאו ש-roscore פועל בחלון אחד והקוד jetbot_controller.py שמפעיל את הרובוט עובד בחלון נוסף בטרמינל נפרד. עכשיו כאשר שהכל הופעל לחצו על הלחצנים במקלדת כדי לגרום לרובוט לזוז:

- `i` - forward
- `k` - backward
- `j` - left
- `l` - right
- `u` - forward-left
- `o` - forward-right
- `m` - backward-left
- `.` - backward-right
- `,` - stop

כדי לעצור את הצומת, לחצו Ctrl+C בחלון שבו ה-Node פועל.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.12.7 הוספת קובץ Launch להפעלת הפרויקט

קבצי launch משמשים להגדרה והפעלה של מספר nodes, פרמטרים והגדרות buxpu, הנדרשות עבור יישום רובוט ספציפי. הם מספקים דרך נוחה להפעלת מערכת מורכבת בפקודה אחת. קובץ Launch מכיל את הסיומת "launch".

כדי להוסיף קובץ launch חדש לחבילה בצעו את השלבים הבאים:

הוסיפו תיקייה חדשה בשם launch בתוך תיקיית החבילה על ידי ההוראות הבאות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/jetbot_control_node
mkdir launch
cd launch
touch jetbot_controller.launch
```

פתחו את קובץ ה-launch הכתוב כקוד XML והגדירו בו את ה- Nodes והפרמטרים שאתם רוצים להפעיל. להלן דוגמא לקובץ כזה:

```
<launch>
            <node name="jetbot_controller"
            pkg="jetbot_control_node"
            type="jetbot_controller.py"
            output="screen" />
</launch>
```

שימרו את קובץ ה-launch ובנו שוב את החבילה באמצעות הפקודה catkin_make:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```

הפעילו את קובץ ה-launch באמצעות הפקודה roslaunch:

```
roslaunch jetbot_control_node jetbot_controller.launch
```

כדי לעצור את קובץ ההפעלה, לחצו Ctrl+C בטרמינל שבו הוא פועל.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

לצורך בדיקת החבילה פרסמו הודעת Twist ל-topic ששמו cmd_vel להלן דוגמה:

```
rostopic pub /cmd_vel geometry_msgs/Twist "linear:
  x: 0.5
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0"
```

ההוראה בחלון השלישי שולחת הודעת Twist ל-topic בשם cmd_vel כדי שיעביר הוראות דרך הקוד למנועים של הרובוט.

כדי לעצור את הרובוט שלחו את אותה הודעה אך הפעם עם פרמטר x:0.0 באופן הבא:

```
rostopic pub /cmd_vel geometry_msgs/Twist "linear:
  x: 0.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0"
```

4.12.8 כתיבת תוכנית פעולה לרובוט

הקוד שתכתבו באתחול ה-Node תוך שימוש בפעולה init_node. לאחר מכן הוא יוצר עצם Publisher שמפרסם ל-topic בשם cmd_vel. הפרמטר queue_size=10 מגדיר את גודל תור ההודעות היוצאות.

לאחר מכן הרוד נכנס ללולאה שנמשכת עד כיבוי ה-Node. בתוך הלולאה, תיצרו הודעת Twist ותגדירו את המהירויות הליניאריות והזוויתיות. המהירות הליניארית לאורך ציר ה-x מוגדרת ל-0.5 (מטר/שנייה), והמהירות הזוויתית סביב ציר ה-Z מוגדרת ל-0.1 (רדיאנים/שנייה).

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

לצורך כך ממשו את השלבים הבאים:

צרו קובץ קוד חדש בתיקייה חדשה בשם scripts שבתוך הפרויקט שלכם, להלן ההוראות:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop
mkdir scripts
cd scripts
touch jetbot_motion_control.py
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop/scripts
 chmod +x jetbot_motion_control.py
```

פתחו את הקובץ שיצרתם וכתבו בו את הקוד הבא:

```
#!/usr/bin/env python
import rospy
from geometry_msgs.msg import Twist


def driver():
    # Initialize the Publisher Node with name 'jetbot_motion_control'
    rospy.init_node('jetbot_motion_control', anonymous=True)

    # Create a Publisher object, publishing to the '/cmd_vel' topic
    pub = rospy.Publisher('/cmd_vel', Twist, queue_size=10)

    # Set the publishing rate (Hz)
    rate = rospy.Rate(10)

    while not rospy.is_shutdown():
        # Initialize a Twist message
        twist_msg = Twist()

        # Set the linear and angular velocities
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
        twist_msg.linear.x = 0.5

        twist_msg.angular.z = 0.1


        # Publish the message

        pub.publish(twist_msg)


        # Sleep for the remaining time of the rate

        rate.sleep()


if __name__ == '__main__':

    try:

        driver()

    except rospy.ROSInterruptException:

        pass
```

כפי שלמדתם מוקדם יותר צרו קובץ launch חדש:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop

mkdir launch

cd launch

touch workshop_jetbot_controller.launch
```

פתחו את הקובץ workshop_jetbot_controller.launch וכתבו את ההוראות הבאות:

```
  <launch>

          <node name="jetbot_controller"

          pkg="jetbot_control_node"

          type="jetbot_controller.py"

          output="screen" />

          <node name="jetbot_motion_control"

          pkg="workshop"

          type="jetbot_motion_control.py"
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
            output="screen" />
</launch>
```

שימרו את קובץ ה-launch ובנו שוב את החבילה באמצעות הפקודה catkin_make:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```

הפעילו את קובץ ה-launch באמצעות הפקודה roslaunch:

```
roslaunch workshop workshop_jetbot_controller.launch
```

זהירות!! הרובוט צפוי להתחיל לזוז.

כדי לעצור את קובץ ההפעלה, לחצו Ctrl+C בטרמינל שבו הוא פועל.

4.12.9 בניית תרחיש AI driver ל-JetBot

בחלק זה תיצרו תרחיש המבוסס על למידת מכונה (AI) עבור JetBot באמצעות ROS detectNet ומספר סקריפטים נוספים. הרעיון הבסיסי הוא להשתמש במצלמה המחוברת ישירות ל-Jetbot ולבצע object detection על התמונות באמצעות DetectNet ולאחר מכן להזיז את Jetbot בהתבסס על המיקומים שקיבלתם detected objects.

AI_driver: הינו קוד בשפת Python שנרשם לתוצאות הזיהוי ומחשב את פקודות התנועה הנדרשות עבור ה-Jetbot לאחר מכן הוא מפרסם את הפקודות הללו בנושא cmd_vel. להלן הסבר קצר על הקוד:

```
detections_callback(data)
```

פונקציה זו מזומנת לצורך עיבוד התוצאות בעקבות זיהוי האובייקטים. הפונקציה מעבדת כל זיהוי ומחשבת את פקודות התנועה הנדרשות כדי לגרום ל-Jetbot לעקוב אחר האובייקט שזוהה. הפעולה מפרסמת את הפקודות הללו תחת topic ששמו cmd_vel.

```
size_callback(data)
```


משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מרכז המורים הארצי למקצועות הטכנולוגיים | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

פונקציה זו משמשת לעדכון המצלמה דרך topic התמונה של camera_publisher.
הפונקציה מעדכנת את המשתנים הגלובליים של image_width ו-image_height בכל פעם שמגיעה הודעה חדשה.

```
listener()
```

זוהי הפונקציה העיקרית, היא מאתחלת את ה-Node, מגדירה את ה-publishers וה-subscribers ומתחילה את לולאת האירועים של ROS.

דרישות קדם

- DetectNet ROS Node
- ROS Jetbot Camera

לצורך מימוש התרחיש צרו קובץ Python חדש בשם ai_driver.py:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop/scripts
touch ai_driver.py
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop/scripts
chmod +x ai_driver.py
```

פתחו את הקובץ ai_driver.py וכתבו בו את הקוד הבא:

```
#!/usr/bin/env python


import rospy
from vision_msgs.msg import Detection2DArray
from std_msgs.msg import Int32MultiArray
from geometry_msgs.msg import Twist


def detections_callback(data):
    global cmd_vel_pub, target_id, image_width, image_height
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    # Assume no detection at first
    detection_found = False

    for detection in data.detections:

        # Ensure we only follow the specified target_id
        if detection.results[0].id != target_id:
            continue

        # rospy.loginfo('       Cente_x: %s', detection.bbox.center.x)
        # rospy.loginfo('       size_x: %s', detection.bbox.size_x)
        # rospy.loginfo('       size_y: %s', detection.bbox.size_y)

        twist = Twist()

        # Calculate the error as the distance of the object's center from the image's center
        error = detection.bbox.center.x - image_width / 2

        # Control gain
        Kp = 0.0005  # Lowered for slower movements

        # Calculate the angular speed with a proportional control
        twist.angular.z = -Kp * error

        # Calculate the distance error as the difference of the size of the bounding box from a desired size
        # Use the mean of the width and height of the bounding box
        bbox_size = (detection.bbox.size_x + detection.bbox.size_y) / 2
        desired_size = image_width * 0.4  # let's say we want the object to take up SOME% of the image width
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
        size_error = bbox_size - desired_size

        # Proportional control for forward speed
        Kp_distance = 0.0009  # Lowered for slower movements
        twist.linear.x = -Kp_distance * size_error

        # print(twist.linear.x , twist.angular.z)

        cmd_vel_pub.publish(twist)

        # Detection was found
        detection_found = True

    # If no detection was found, stop the robot
    if not detection_found:
        twist = Twist()
        cmd_vel_pub.publish(twist)

def size_callback(data):
    global image_width, image_height

    image_width, image_height = data.data

    # rospy.loginfo('Image size: %s', data.data)

def listener():
    global cmd_vel_pub, image_width, image_height, target_id

    rospy.init_node('detection_listener', anonymous=True)
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
    image_width = 0
    image_height = 0

    # Fetch the target_id parameter, defaulting to 0 if it's not set
    target_id = rospy.get_param("~target_id", None) # from list ssd_coco_labels.txt

    cmd_vel_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)

    rospy.Subscriber('/detections/info', Detection2DArray, detections_callback)
    rospy.Subscriber('/image_size', Int32MultiArray, size_callback)
    rospy.spin()

if __name__ == '__main__':
    listener()

##################### more controls ##################
#  if detection.bbox.center.x < image_width / 2:
#            # Move left
#            twist.angular.z = 0.1
#        elif detection.bbox.center.x > image_width / 2:
#            # Move right
#            twist.angular.z = -0.1

#######################################################
```

בנו את החבילה:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תרחיש AI driver כולל תיאום של מספר חבילות, ולכן מומלץ מאוד להשתמש בקובץ הפעלה מקיף. קבצי הפעלה ב-ROS מאפשרים להפעיל מספר צמתים בבת אחת ולהגדיר פרמטרים, מה שהופך אותו לכלי בעל ערך עבור מערכות מורכבות.

לצורך כך צרו קובץ Launch המתאים להרצת המשימה באופן הבא:

שלב ראשון צרו הקובץ:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop/launch
touch FTL_demo.launch
```

שלב שני פתחו את הקובץ וכתבו את הקוד הבא:

```
<launch>
    <!-- Argument for whether to include the jetbot_controller -->
    <arg name="with_controller" default="false"/>

    <!-- Start the camera_publisher node -->
    <node name="camera_publisher"
            pkg="camera_publish"
            type="camera_publisher.py"
            output="screen">
   <param name="camera_type" value="jetbot" /> <!-- "webcam" or "jetbot" -->
    </node>

    <!-- Start the web_video_server node -->
    <node name="web_video_server"
            pkg="web_video_server"
            type="web_video_server"
            output="screen" />

    <!-- Start the detectnet node -->
    <node name="detectnet"
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
            pkg="detectnet"
            type="detectnet_ros_node.py"
            output="screen" />

    <!-- Start the AI_driver node -->
    <node name="AI_driver"
            pkg="workshop"
            type="ai_driver.py"
            output="screen">
        <!-- Set the desired ID (1 for a person) -->
        <param name="target_id" value="1" />
    </node>

    <!-- Start the jetbot_controller node if with_controller argument is true -->
    <group if="$(arg with_controller)">
        <node name="jetbot_controller"
            pkg="jetbot_control_node"
            type="jetbot_controller.py"
            output="screen" />
    </group>
</launch>
```

בנו את החבילה:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

4.12.10 יצירת FTL FULL Demo Launch File

ההדגמה של "Follow The Leader" מאפשרת ל-JetBot לעקוב אחר אובייקט מסוים (במקרה זה, אדם). כדי להקל על הדגמה זו, צרו קובץ הפעלה המשלב את כל צמתי ה-ROS הדרושים. התחילו קודם ביצירת קובץ הפעלה חדש בספריית תרחישי הסדנה:

```
cd ~/MorBot/morbot_ros/morbot_ws/src/scenarios/workshop/launch
touch FTL_demo.launch
```

פתחו לעריכה את הקובץ וכתבו את ההוראות הבאות:

```
<launch>
 <!-- Argument for whether to include the jetbot_controller -->
 <arg name="with_controller" default="false"/>

 <!-- Start the camera_publisher node -->
 <node name="camera_publisher"
        pkg="camera_publish"
        type="camera_publisher.py"
        output="screen">
 <param name="camera_type" value="jetbot" /> <!-- "webcam" or "jetbot" -->
 </node>

 <!-- Start the web_video_server node -->
 <node name="web_video_server"
        pkg="web_video_server"
        type="web_video_server"
        output="screen" />

 <!-- Start the detectnet node -->
 <node name="detectnet"
        pkg="detectnet"
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

```
            type="detectnet_ros_node.py"
            output="screen" />

    <!-- Start the AI_driver node -->
    <node name="AI_driver"
            pkg="workshop"
            type="ai_driver.py"
            output="screen">
        <!-- Set the desired ID (1 for a person) -->
        <param name="target_id" value="1" />
    </node>

    <!-- Start the jetbot_controller node if with_controller argument is true -->
    <group if="$(arg with_controller)">
        <node name="jetbot_controller"
            pkg="jetbot_control_node"
            type="jetbot_controller.py"
            output="screen" />
    </group>
</launch>
```

בנו את החבילה:

```
cd ~/MorBot/morbot_ros/morbot_ws
catkin_make
source devel/setup.bash
```

כדי לזהות אדם יש להגדיר את הפרמטר 1 target_id לצומת AI_driver בתוך קובץ ההפעלה. פרמטר זה מתאים ל-ID עבור person.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מומלץ מאוד להקדיש זמן לכוונון עדין של ערכי בקר ה-PID. שינוי של ערכים אלו משפיע על ההיענות והיציבות של תנועות הרובוט. על ידי כך ניתן להשיג את האיזון האופטימלי עבור התרחיש הספציפי שלנו.

כברירת מחדל, קובץ ההפעלה אינו מפעיל את Node הבקר JetBot זה אומר שהרובוט לא יזוז פיזית. תכונה זו שימושית במיוחד עבור כוונון עדין ובדיקת המערכת לפני הפעלת הרובוט פיזית.

לכוונון עדין ולריצת בדיקה:

```
roslaunch workshop FTL_demo.launch with_controller:=false
```

כדי לראות את זיהוי העצמים פתח דפדפן אינטרנט ונווטו אל:

http://<jetbot_ip_address>:8080/stream?topic=/detections/overlay

כדי לבדוק את התמונות שפורסמו:

```
rostopic echo /detections/info
```

כדי לבדוק את ה- cmd_vel:

```
rostopic echo /cmd_vel
```

לאחר בדיקה יסודית של המערכת, ניתן לאפשר לצומת הבקר של JetBot להפעיל תנועה פיזית. כדי לעשות זאת, הפעילו את המערכת כשהפרמטר with_controller מוגדר כ-true.

להפעלת קובץ ההפעלה באמצעות הפקודה roslaunch:

```
roslaunch workshop FTL_demo.launch with_controller:=true
```

פתחו דפדפן אינטרנט ונווטו אל:

http://<jetbot_ip_address>:8080/stream?topic=/detections/overlay

כדי להציג את המצלמה.

כדי לעצור את קובץ ההפעלה, לחצו Ctrl+C בטרמינל שבו פועל הקובץ.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון 972-73-3783146+ E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

לרשימת Class ID List for ssd coco labels לחצו כאן.

לתרחישים נוספים לחצו כאן.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון +972-73-3783146 E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

5. סיכום

במטרה לענות על הצורך שעלה ממשרד החינוך והצבא לגבי פיתוח תוכנית לימודים בנושא כלים בלתי מאוישים אוטונומיים פותחה תוכנית הכשרה המיועדת למורי החינוך הטכנולוגי במגמות הנדסת אלקטרוניקה, הנדסת מכונות ותחבורה מתקדמת. נושא תוכנית ההכשרה: פיתוחים ויישומים בלמידה עמוקה ורובוטיקה בפרויקטים של כלים בלתי מאוישים. התכנית מורכבת משלושה שלבים. שני שלבים שפותחו בשנת תשפ"ג : חלק א' - הקנית רקע תיאורטי בסיסי וחלק ב' – הקנית התנסות פעילה בנושאי התכנית במטרה לפתח מערכי שיעור או מערכי מעבדה ליישום עם התלמידים בבתי הספר. בשנת תשפ"ד יפותח חלק ג' – שמטרתו לפתח פרויקטי גמר בתחום הנ"ל במסגרת בתי הספר ויהווה פיילוט לביצוע תחרויות בהתאם.

החוברת הנוכחית סוקרת את חלק א' של תוכנית ההכשרה אשר נערך במהלך חודשים יוני – יולי 2023, במסגרת מרכז המורים מור-טק. בשנת תשפ"ד תפותח חוברת נוספת בדגש על חלק ב' וחלק ג' של תוכנית ההכשרה.

בחוברת זו הצגנו את המטרה, המוטיבציה והתוכן של השלב הראשון בתוכנית ההכשרה ואשר התייחס למבוא ורקע בסיסי בנושא יסודות Linux , כלים לעבודה עם Linux, מערכת הפעלה בסביבת ROS (התקנת סביבת העבודה, כתיבת Node בסיסי, תרגול תקשורת הודעות, למידת מכונה בדגש על רשת נוירונים, התנסות בזיהוי תמונה וכו').

שלב זה בתוכנית ההכשרה בנושא למידה עמוקה ורובוטיקה, מהווה את הבסיס לחלק השני של התוכנית שנערכה בחודשים יולי – אוגוסט 2023. התכנים של החלק השני של יוצגו בחוברת ההמשך.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

6. מקורות

Linux 2023 אוחזר מתוך ויקיפדיה:

https://he.wikipedia.org/wiki/%D7%9C%D7%99%D7%A0%D7%95%D7%A7%D7%A1#:~:text=%D7%9C%D7%99%D7%A0%D7%95%D7%A7%D7%A1%20(%D7%91%D7%90%D7%A0%D7%92%D7%9C%D7%99%D7%AA%3A%20Linux),%D7%9C%D7%99%D7%A0%D7%95%D7%A7%D7%A1%20(GNU%2FLinux).&text=%D7%94%D7%A4%D7%99%D7%AA%D

מדריכי Linux 2023 אוחזר מתוך רשת טק:

https://reshetech.co.il/linux-tutorials/index

מערכת הקבצים של Linux 2023, אוחזר מתוך רשת טק:

https://reshetech.co.il/linux-tutorials/the-file-system

ניהול הרשאות משתמשים וקבוצות. (2023). אוחזר מתוך רשת טק:

https://reshetech.co.il/linux-tutorials/users-groups-permissions

ABET (Accreditation Board for Engineering and Technology) (2019). "General criterion 3. Student outcomes". Criteria for Accrediting Engineering Programs, 2018–2019. Retrieved from:

https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-engineering-programs-2018-2019/#GC3

Barak, M. (2005). From order to disorder: The role of computer-based electronics projects on fostering of higher-order cognitive skills. *Computers & Education*, *45*(2), 231–243.

Dori, Y. J. (2003). A framework for project-based assessment in science education. In M. Segers, F. Dochy, & E. Cascallar (Eds.), Optimizing New Modes of Assessment: *In search of Qualities and Standards*, Dordrecht, The Netherlands: Kluwer, 89-118.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

Dori, Y. J., & Belcher, J. (2005). How Does Technology-Enabled Active Learning Affect Undergraduate Students' Understanding of Electromagnetism Concepts? *The Journal of the Learning Sciences*, *14*(2), 243–279. https://doi.org/10.1207/s15327809jls1402_3

Kember, D., & Leung, D. Y. P. (2005). The influence of active learning experiences on the development of graduate capabilities. *Studies in Higher Education*, *30*(2), 155–170.

https://doi.org/10.1080/03075070500043127

Malec, J. (2001). Some Thoughts on Robotics for Education, 2001 AAAI Spring Symposium on Robotics and Education, Stanford University, USA, March 2001

National Research Council (2013). *Education for life and work: Developing transferable* knowledge *and skills in the 21st century*. National Academies Press

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

Rocker Yoel, S., & Dori, Y. J. (2022). FIRST high school students and FIRST graduates: STEM exposure and career choices. *IEEE Transactions on Education*, 65(2), 167-176. doi: 10.1109/TE.2021.3104268.

Shwartz-Asher, D., Reiss, S. R., Ali, A. A.-Y., & Dori, Y. J. (2020). Engineering teachers' assessment knowledge in Active and Project-Based Learning. In *Mintzes J., Walter E. (eds) Active Learning in College Science. Springer, Cham* (pp. 851–869). Springer International Publishing. https://doi.org/10.1007/978-3-030-33600-4_53

Siegwart, R. & Nourbakhsh, I. (2004). *Introduction to Autonomous Mobile Robots*, MIT Press, ISBN: 0-262-19502-X, Cambridge, MA, USA.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

Stehle, S. M., & Peters-Burton, E. E. (2019). Developing student 21st Century skills in selected exemplary inclusive STEM high schools. *International Journal of STEM Education*, *6*(1), 39. https://doi.org/10.1186/s40594-019-0192-1

Thomas, J. W. (2000). *A Review of Research on Project-based Learning*.

Verner, I., M. & D. Ahlgren (2007). Robot Projects and Competitions as Education Design Experiments, Intelligent Automation and Soft Computing, Special Issue *Global Look at Robotics Education*, *13*(1), 57-68.

https://github.com/arieldo/MorBot/wiki/PubSubPy

https://cloud.google.com/pubsub/docs/overview

http://wiki.ros.org/roscpp/Overview/Publishers%20and%20Subscribers

https://github.com/arieldo/MorBot/wiki/Ros-101

https://opencloudware.com/robot-operating-system-ros-the-key-to-the-future-of-robotics-programming/

http://wiki.ros.org/ROS/Introduction

https://tikshuv.site/ss/

https://www.digitalocean.com/community/tutorials/ssh-essentials-working-with-ssh-servers-clients-and-keys#generating-and-working-with-ssh-keys

https://code.visualstudio.com/docs/remote/ssh-tutorial

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7. נספחים

7.1 נספח א' - מדריך בנושא Linux for Beginners

בנספח זה נדון בנושא: מערכת הפעלה Linux, נלמד פקודות בסיסיות, עבודה עם קבצים, תיקיות והרשאות.

7.1.1 מושגים מרכזיים

- מערכת הפעלה – מערכת הפעלה (OS) היא תוכנה המנהלת משאבי חומרת מחשב ותוכנה ומספקת פלטפורמה להפעלת יישומי תוכנה אחרים. מערכת הפעלה משמשת כמתווך בין חומרת המחשב למשתמש, ומקלה על תקשורת ותיאום בין רכיבים שונים.
- KERNEL (ליבה) – היא מרכיב מרכזי של מערכת הפעלה. היא משמשת כגשר בין החומרה של המחשב לתוכנה הפועלת מעליו.
- Linux distribution – (הפצת Linux), המכונה גם distro, היא חבילת מערכת הפעלה מלאה המורכבת מליבת Linux יחד עם אוסף של יישומי תוכנה, ספריות, כלי עזר וכלי מערכת. ישנן מספר הפצות "distros" פופולריות, להלן רשימה חלקית:

7.1.2 Linux - רקע היסטורי

Linux היא ליבת מערכת הפעלה בקוד פתוח שפותחה במקור על ידי Linus Torvalds בשנת 1991 (Linux, 2023). השם "Linux" מתייחס הן לליבה עצמה והן למשפחה הרחבה יותר של מערכות הפעלה המבוססות עליה. Linux ידועה באופי הפתוח והחופשי שלה, מה שאומר שקוד המקור של מערכת ההפעלה זמין לציבור לעריכה, שיפור ותיקון באגים. זה מאפשר למפתחים ולקהילת Linux הרחבה יותר לשפר את המערכת ולהתאים אותה לצרכי משתמשים שונים. Linux הפכה לפלטפורמה מועדפת עבור שרתים ומחשבים אישיים, והיא גם מתרחבת לעולם הנייד והמכשירים המשובצים. עם זאת, Linux היא הליבה של מערכת

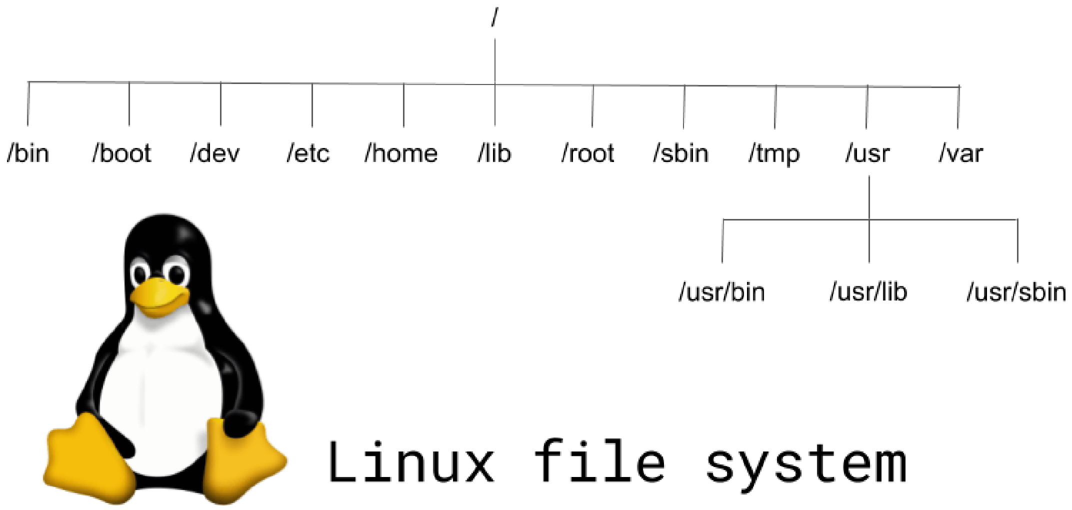
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

הפעלה ומשמש בשילוב עם מגוון תוכנות וכלים אחרים המספקים פונקציות שונות. כל הפצה מספקת חוויות משתמש שונות ומתאימה לצרכי משתמש שונים.

7.1.3 הכרות עם ההפצה הפופולארית של Linux

Ubuntu - היא הפצת Linux פופולרית המבוססת על ארכיטקטורת Debian. Ubuntu ידוע בממשק משתמש ידידותי וקל לשימוש. Debian מתאפיין בנגישות לסביבת שולחן עבודה. Ubuntu פותחה על ידי Canonical Ltd וצברה קהילת משתמשים גדולה ופעילה.

סוגי משתמשים – יש מספר סוגי משתמשים במערכת Ubuntu. ההבדל בין המשתמשים הוא ברמת בהרשאות הגישה והפעולה שלהם. שני משתמשים עיקריים הם: המנהל Super User ומשתמשים רגילים. מערכת הקבצים של Linux מתוארת באיור 4.



איור 4. מבט כללי על מערכת הקבצים (מערכת הקבצים של Linux, 2023)

כל תיקייה נקראת directory, ובתוך התיקיות ניתן למצוא תיקיות משנה מה שיוצר היררכיה (עץ) של תיקיות. שמם של התיקיות נגזר ממיקומם בהיררכיה.

כל מערכת הקבצים קיימת בתוך תיקייה אחת מיוחדת ששמה root - תיקיית השורש.

בתוך תיקיית השורש קיימות כמה תיקיות top level. דוגמת: boot, home ו-Linux. usr משתמש בתיקיות אילה בתהליך האתחול. כדי להגיע לכל אחת מהתיקיות אנחנו יכולים להשתמש בקו נטוי ואח"כ שם התיקייה.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.1.4 הסבר על תיקיות ראשיות

- שורש המערכת - תיקיית בסיס עץ התיקיות. מורשה לכתיבה או מחיקה עליו רק על ידי המשתמש root שסיסמתו היא סיסמת המערכת.
- Root - תיקיית הבית של המשתמש root. ניתנת לצפייה, כתיבה ומחיקה רק על ידי המשתמש root שסיסמתו היא סיסמת המערכת.
- bin - מכילה קבצי פקודות מסוף (Command) בינאריים הניתנים להרצה. המדובר בקבצים המיועדים לשימוש כל המשתמשים כגון ls, cp, grep, ps ו-ping, הקיימים במערכת כברירת מחדל אף ללא פעולת התקנה מצד המשתמש. במידה ויש משתמש בודד למערכת (מלבד root, שאינו נחשב) עשויים להתווסף קובצי פקודה נוספים.
- sbin - מכילה קבצי פקודות מסוף בינאריים הניתנים להרצה. המדובר בקבצים המיועדים לשימוש על פי רוב על ידי מנהל המערכת לשם תחזוקת המערכת. דוגמאות לקבצים אלו הן הפקודות iptables, reboot, fdisk, ifconfig ו-swapon. אין הכוונה כי התיקייה היא הקובעת את רמת ההרשאה אלא כך מסודרות הפקודות מלכתחילה.
- lib - מכילה ספריות בשימוש התוכניות הנמצאות ב-bin/ וב-sbin/.
- usr - מכילה קבצים בינאריים, ספריות, תיעוד ומקורות קוד לתוכניות

תיקיות משנה חשובות בתיקיה זו:

usr/bin/ - מכילה קבצים בינאריים של תוכניות משתמש.

usr/sbin/ - מכילה קבצים בינאריים של תוכניות משתמש לשימוש על ידי מנהל המערכת.

usr/lib/ - מכילה ספריות בשימוש התוכניות הנמצאות ב-usr/bin/ וב-usr/sbin/.

usr/local/ - מכילה תוכניות שהותקנו ממקורות שונים, שאינם כלולים בהפצה הרשמית. בנוסף עשויה לאחסן מידע ותוכניות אישיות במצב שקיימת רשת משותפת בה גם התיקייה usr/ משותפת.

usr/share/ - מכילה קבצי הגדרות כולל הגדרות גרפיות של תוכניות המשתמש.

usr/src/ - מכילה את קוד המקור לתוכניות המערכת כולל הקוד של הליבה.

usr/include/ = מכילה קבצי כותרת (include) הנדרשים בעת הידור תוכנית.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- etc - מכילה קבצי הגדרות הדרושים לתוכניות המערכת. בנוסף, תיקייה זו מכילה קבצי מעטפת (קבצי תצורה) של תסריטים הנוגעים לצורת הפעלת וכיבוי תוכניות במערכת, דוגמת הקובץ fstab, המכיל מידע אודות מערכות קבצים כגון CD-ROM, החסנים ניידים, ונקודות העגינה שלהם. במערכת הפעלה Debian, תופיע בפנים תיקייה בשם apt המכילה בין היתר קובץ בשם sources.list המגדיר את כל הנתיבים אליהם apt-get פונה.
- dev - קיצור ל-device files. מכילה קבצים הנוגעים להתקנים ורכיבים המחוברים למערכת, אם ברמה הפיזית כהתקני USB ואם ברמת התכנה כמסופי TTY. הסיבה לשילוב שני סוגי הרכיבים בתיקייה אחת היא, כי במערכות Linux גם התקנים פיזיים (כלומר חומרה על כל סוגיה) מטופלים כקבצים ולכן גם כונן פיזי יטופל כקובץ.
- var - מכילה קבצים המשתנים תוך כדי פעולת המערכת. אין מדובר בהכרח בקבצי משתנים במובן התכנותי, אלא בקבצים שונים המשתנים בעת פעילות המערכת. דוגמאות לקבצים בתיקייה זו הם קובצי יומן של המערכת (log), קבצים משתנים השייכים לתוכניות (lib), קובץ תור הדפסה (spool), קבצי נעילת קבצים (lock), קבצי דוא"ל (mail) וקבצים זמניים הנותרים לאחר אתחול (tmp).
- boot - מכילה קבצים הקשורים למנהל האתחול. כולל את קבצי GRUB, ו־vmlinux
- run - תיקייה המכילה קבצים זמניים שנטענים ל-RAM, ומכילה משתני זמן ריצה של מערכת ההפעלה.
- srv - מכילה מידע השייך לשירותים מסוימים על פי השירות.
- sys - קיצור של system. תיקייה המכילה מידע על התקנים, מנהלי התקנים וליבת Linux. כמו run, התיקייה משתמשת ב-RAM ולכן נמחקת בכל כיבוי ונוצרת מחדש בכל עלייה.
- tmp - תיקייה זו מכילה קבצים זמניים הנוצרים על ידי המערכת והמשתמש. כל אלו נמחקים בזמן אתחול, בניגוד ל־/var/tmp, בה הם נשארים לאחר אתחול.
- home - תיקייה זו מכילה את קובצי המשתמש הפרטיים הנוצרים על ידו. כוללת את כל התיקיות הנמצאות בשימוש רגיל כגון "שולחן עבודה", "מסמכים", "הורדות" וכדומה. קבצי משתמש מסוים מכונסים בתיקיית בת ב-/home הנושאת את שמו. במידה ויש מספר משתמשים כל אחד יקבל תיקייה פרטית בעלות הרשאות ייחודיות אליו.
- mnt - תיקייה המיועדת לעגינה זמנית של מערכות קבצים אחרות.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

מרכז המורים הארצי למקצועות הטכנולוגיים
הפקולטה לחינוך למדע וטכנולוגיה, הטכניון, מכון טכנולוגי לישראל, חיפה 320003
טלפון 972-73-3783146+ E-mail: Moretech@ed.technion.ac.il
https://moretech.technion.ac.il

- media - מיועדת לעגינת מדיה נתיקה, כגון נגני מדיה, כונני CD-ROM, החסנים ניידים, כוננים קשיחים חיצוניים, ומערכות קבצים נוספות. המדיה לרוב תעגון בתיקייה באופן אוטומטי על ידי מערכת ההפעלה.
- lost+found - בתיקיה זו מאוחסן מידע שקיים על המערכת בעת קריסתה, או במצב של אי ניתוק התקנים לפני כיבוי מערכת.

בכל פעם שתפתחו את הטרמינל תגלו שאתם נמצאים בתיקיית הבית שלכם - התיקייה שבה נמצאים הקבצים האישיים שלכם.

התיקייה האישית שלכם היא מאוד חשובה ועל כן היא זכתה בקיצור דרך. זה לא משנה היכן אתם נמצאים במערכת הקבצים, אם תקלידו : cd תגיעו לתיקיית הבית שלכם.

7.1.5 פקודות עיקריות

- **sudo** - תוסף מקדים לפקודה שמציין למערכת להריץ את הפקודה הבאה עם הרשאות מנהל מערכת. כשמריצים פקודה כזו המערכת מבקשת להכניס את סיסמת המנהל. לדוגמא:

```
sudo apt install <package name>
```

פקודה זו מבקשת להתקין חבילה חדשה למערכת ההפעלה

- **pwd** - מדפיסה למסך את המיקום הנוכחי שלנו בתוך עץ הספריות של מערכת ההפעלה למידע נוסף על מבנה עץ הספריות ניתן לקרוא באתר רשת-טק (Linux, 2023), לדוגמא:

```
pwd
```

להלן דוגמא לתוצאה המתקבלת:

/home/joe/dir1

- **ls** - פקודה זו מדפיסה למסך את רשימת הקבצים והתיקיות הנמצאים בתוך התיקייה הנוכחית.

למשל:

```
ls
```

ונקבל:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

file1 file2 file3 file4

ניתן להפעיל פקודה זו בצירוף פרמטרים שונים המשפיעים על סוג המידע שיודפס למסך. למשל:

ls -l מציג את הרשימה בצירוף שמות בעלי הקבצים, תאריך יצירה אחרון, הרשאות ועוד.

רשימה מתקבלת, למשל:

total 8
drwxr-xr-x 9 joe joe 4096 Nov 28 08:09 backups
drwxr-xr-x 4 joe joe 4096 Nov 28 12:45 Desktop
drwxr-xr-x 2 joe joe 4096 Nov 16 21:15 Documents
drwxr-xr-x 2 joe joe 4096 Dec 4 10:08 Downloads
-rw-r--r-- 1 joe joe 23459 Nov 13 07:06 hello.py
drwxr-xr-x 2 joe joe 4096 Nov 23 18:19 Music
drwxr-xr-x 7 joe joe 4096 Dec 4 09:29 Pictures
drwxr-xr-x 2 joe joe 4096 Dec 3 18:24 Videos

כדי לקבל מידע על כל אפשרויות ההפעלה של הפקודה ניתן להריץ פקודת: man ls אשר תציג את המדריך המלא לשימוש בפקודה. בהמשך מופיע הסבר לפקודה man.

- change directory - **cd** פקודה זו מאפשרת לנווט בתוך עץ התיקיות של מערכת ההפעלה. כשמריצים אותה יש לציין לאיזה תיקייה ברצונכם לעבור. יש תחביר מדויק לציון תיקיית היעד וישנם גם קיצורים כגון: ..cd יגרום לעבור אל התיקייה שנמצאת רמה אחת מעל, /cd יגרום לעבור לתיקיית השורש של מערכת ההפעלה.

הפקודה cd /myDir תעביר אותנו לתוך תיקייה myDir שנמצאת בתוך תיקיית הבית שלכם.

הפקודה ~cd תעביר אותכם לתיקיית הבית שלכם.

הפקודה ..cd תעביר אותכם לתיקייה אחת מעל זו הנוכחית, אם הפעלתם פקודה זו כשאתם בתוך תיקיית myDir אתם עוברים לתיקיית הבית שלכם.

הפקודה ../.. cd תעביר אותכם שתי רמות למעלה בעץ התיקיות.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

- **manual man** - פקודה זו תציג את המדריך המלא לשימוש בפקודה רצויה. הפעלת הפקודה נעשית על ידי הוספת שם הפקודה שעליה רוצים לקבל מידע, למשל: man cd תדפיס את המידע על אפשרויות השימוש בפקודה cd.
- **mkdir** - פקודה ליצירת תיקיה חדשה בתוך התיקיה הנוכחית או בתיקייה קיימת אחרת. הפקודה יכולה לקבל מספר פרמטרים, שהעיקרי ביניהם הוא שם התיקיה החדש, למשל:

```
mkdir ./temp/myNewFolder
```

אם תריצו פקודת ls אחרי פקודה זו, תראו כי התיקייה החדשה נוספה לרשימת הקבצים והתיקיות.

- **touch** – פקודה זו מאפשרת ליצור קובץ חדש בתיקייה הנוכחית או בתיקייה קיימת אחרת. אם הקובץ קיים כבר אז הפקודה מעדכנת את זמן העדכון האחרון של הקובץ. למשל:

```
touch ../../testFile.py
```

פקודה זו תיצור קובץ חדש בתיקייה שנמצאת שתי רמות מעל. אם תעברו לתיקייה בעזרת ../..cd ואז תבצעו ls תראו את הקובץ החדש שנוצר בגודל אפס – ריק.

- **cp** - פקודה זו מאפשרת להעתיק קובץ קיים או תיקייה קיימת למיקום רצוי בעץ התיקיות. ניתן לתת מספר פרמטרים לפקודה כדי להשפיע על פעולת ההעתקה (למשל, העתקה רקורסיבית של כל הקבצים בתוך תיקייה). שני הפרמטרים העיקריים הם שם ומיקום הקובץ להעתקה ומיקום היעד שאליו מעתיקים.

למשל:

```
cp myFile.txt ../
```

פקודה זו תעתיק את הקובץ myFile.txt שנמצא בתיקייה הנוכחית, אל התיקייה מעל. אם תעברו לתיקייה מעל בעזרת ..cd והקלידו ls תראו את הקובץ שהעתקתם.

- **mv** – פקודה זו מאפשרת להעביר קובץ או תיקייה קיימת ממיקום נוכחי למיקום חדש. כמו כן היא מאפשרת לשנות שם קובץ או תיקייה. הפרמטרים העיקריים הם שם ומיקום

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

נוכחי של הקובץ/תיקייה ושם ומיקום היעד. כמו בכל הפקודות יש פרמטרים נוספים שניתן לתת לפקודה בכדי להשפיע על אופן ביצועה. למשל:

```
mv myTest.file myTestFile.txt
```

אם תבצעו ls מיד לאחר הפקודה הזו, תראו את שם הקובץ החדש, ואילו השם הישן יעלם.

- **rm** - פקודה זו מאפשרת למחוק קובץ או תיקייה קיימת במיקום נוכחי. הפרמטרים העיקריים הם שם ומיקום נוכחי של הקובץ/תיקייה שמיועד למחיקה. כמו בכל הפקודות יש פרמטרים נוספים שניתן לתת לפקודה בכדי להשפיע על אופן ביצועה. למשל:

```
rm -r myFolder
```

פקודה זו תמחק את התיקייה ואת כל עץ התיקיות מתחת.

7.1.6 ניהול הרשאות

רמות הרשאה (ניהול הרשאות משתמשים וקבוצות, 2023) – לכל קובץ/תיקייה ב-Linux יש הרשאות לשלוש ישויות: בעל הקובץ, קבוצה וכל המשתמשים. הרשאה מורכבת משלוש פעילויות אפשריות: קריאה (r), כתיבה (w), הרצה (x). מתן הרשאה מתייחס למי נותנים ומה סוג ההרשאה.

כמו שהוזכר לעיל, הרצת הפקודה ls תציג את רשימת הקבצים בתיקייה הנוכחית כולל ההרשאות לכל קובץ. המידע על ההרשאות מורכב מ-10 תווים שיכולים להיות : r, w, x, d, -

בכל מקום בו אין הרשאה יופיע סימן –

התו הראשון משמאל מציין האם הקובץ הוא תיקייה – d

יתר 9 התווים מחולקים לשלוש קבוצות של הרשאות לפי שלוש הישויות שהוזכרו – השלשה הראשונה משמאל (מיד אחרי סימן התיקייה) מגדירה הרשאות של בעלי הקובץ, אחר כך שלשה נוספת מגדירה הרשאות של קבוצות, והשלשה האחרונה הרשאות לכלל יתר המשתמשים.

למשל:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

תיקייה עם הרשאות בעלים בלבד

drwx------

קובץ עם הרשאות כתיבה וקריאה לבעלים ורק קריאה לקבוצה

-rw-r-----

קובץ עם הרשאות מלאות לכולם

rxwrwxrwx

ניתן לציין את רמת ההרשאה גם בספרות. האות r ערכה 4, האות w ערכה 2, האות x ערכה 1. כדי להגדיר הרשאה מסוימת ניתן לסכום את הערכים המספרים ולציין את ההרשאה על ידי מספר תלת ספרתי מתאים. לדוגמא:

קובץ עם הרשאת כתיבה וקריאה לבעלים, הרשאת קריאה בלבד לקבוצות ואחרים. הרשאת הבעלים תהיה בערך 6, ואילו ההרשאה האחרת תהיה בערך 4 .

כדי ליישם הרשאה ספציפית השתמשו בפקודה chmod אשר מקבלת כפרמטרים את רמת ההרשאה ואת שם הקובץ, למשל: chmod 644 myFile.txt

אם תבצעו ls -l מיד אחרי הפקודה הזו, תראו כי הקובץ מופיע עם הרשאות חדשות למשל:

-rw-r--r-- 1 gabbys gabbys 151 Jul 19 07:06 myFile.txt

7.2 נספח ב' - מדריך בנושא Multi robot

סנכרון וגילוי Master הם היבטי מפתח במערכות ROS מרובות רובוטים, המאפשרים למספר רב של ROS Masters לתקשר ולהחליף מידע.

7.2.1 סנכרון וגילוי Master למערכת מרובת רובוטים

סנכרון Master: תהליך זה מבטיח עקביות נתונים על פני ROS מאסטרים שונים. חבילה פופולרית עבור זה ב-ROS Melodic היא multimaster_fkie.

גילוי Master: תהליך זה מאפשר ל-ROS מאסטר לגלות מאסטרים אחרים במערכת. זה מושג באמצעות פרוטוקול גילוי וגם בסיוע multimaster_fkie.

כדי להשתמש ב-multimaster_fkie:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

1. התקינו את החבילה:

```
sudo apt-get install ros-melodic-multimaster-fkie
```

2. השיקו מסוף חדש עבור roscore:

```
roscore
```

3. הפעילו מסוף חדש עבור הצומת master_discovery_fkie:

```
rosrun master_discovery_fkie master_discovery
```

4. הפעילו מסוף חדש עבור הצומת master_sync_fkie:

```
rosrun master_sync_fkie master_sync
```

עם multimaster_fkie, מערכות ROS יכולות להתנהג ביעילות כמערכת אחת גם כשהן מופצות על פני מספר מכונות או רובוטים.

7.3 נספח ג' - מדריך בנושא Notebooks Demos

7.3.1 שיטה 1: שלוט ב-JetBot על ידי תכנות מדפדפן אינטרנט

1. התחברו לרובוט שלכם על ידי ניווט אל:

http://<jetbot_ip_address>:8888

2. היכנסו עם סיסמה ברירת המחדל jetbot
3. נווטו אל:

~/Notebooks/basic_motion/

4. פתחו ופעלו לפי המחברת basic_motion.ipynb
5. וודאו של-JetBot יש מספיק מקום לנוע.

7.3.2 שיטה 2: שליטה ב-JetBot על ידי תכנות מחברת jupyter ישירות מ-VS

1. התחברו לרובוט שלכם מרחוק מ- VS Code SSH
2. נווטו אל:

~/MorBot/jetbot/notebooks/basic_motion

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

3. פתחו ופעלו לפי המחברת basic_motion.ipynb

4. וודאו של-JetBot יש מספיק מקום לנוע.

5. עקבו אחר המחברת.

בשלב הבא – פתחו את הפרויקט שלכם עם ROS, והתקינו לפי פרק ROS.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מורטק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.4 נספח ד' - מדריך בנושא VirtualBox Tutorial for Beginners

זהו מדריך קצר של VirtualBox למתחילים, אשר עוזר ללמוד כמה פקודות ומושגים בסיסיים כדי להתחיל עם VirtualBox.

במדריך זה נתקין את Ubuntu 18.04 על VirtualBox.

7.4.1 מבוא ל-VirtualBox

VirtualBox היא תוכנת קוד פתוח רבת עוצמה ליצירה ולניהול של מכונות וירטואליות. פותח על ידי Oracle, הוא מאפשר להפעיל מספר מערכות הפעלה בו-זמנית במחשב. עם VirtualBox, ניתן להפעיל מערכת הפעלה אחרת בחלון על שולחן העבודה, ללא צורך באתחול כפול. הוא תומך במערכות הפעלה מארחות רבות, כולל Windows, Linux, Mac OS ואחרות, ובמגוון רחב של מערכות הפעלה אורחות. הוא משמש לעתים קרובות לפיתוח תוכנה, בדיקות וחינוך, ומציע סביבה בטוחה להתנסות מבלי להשפיע על המערכת המארחת.

7.4.2 הורדת התוכנה הנדרשת

1. הורידו והתקינו את הגרסה האחרונה של VirtualBox על ידי בחירת חבילות הפלטפורמה שלך (Windows , Mac) מהאתר הרשמי:

https://www.virtualbox.org/wiki/Downloads

2. הורידו את קובץ ה-ISO של Ubuntu 18.04 מהאתר הרשמי של אובונטו:

https://releases.Ubuntu.com/18.04/

7.4.3 יצירת מכונה וירטואלית חדשה

1. פתחו את VirtualBox ולחצו על הלחצן 'חדש' כדי ליצור מכונה וירטואלית חדשה.
2. תנו שם ל-VM (למשל, "Ubuntu 18.04"), בחרו "Linux" כסוג, ובחרו "Ubuntu 64-bit" כגרסה, ולאחר מכן לחצו על "הבא".
3. בחרו 'תמונת ISO' כדי להיות קובץ ה-Ubuntu 18.04" ISO" שהורדתם מוקדם יותר ולחצו על "הבא".
4. ודאו שתיבת הסימון "skip unattended guest installation" מסומנת.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מור-טק | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.4.4 הקצאת זיכרון RAM

החליטו כמה זיכרון RAM ומעבדים להקצות ל-VM. הדרישה המינימלית עבור Ubuntu 18.04 היא 2GB ולפחות 2 מעבדים, אבל מומלץ יותר. לחצו על "הבא" לאחר שתסיימו.

7.4.5 יצירת דיסק קשיח וירטואלי

1. בחרו "Create a virtual hard disk now "
2. הגדירו את גודל הדיסק הקשיח. הדרישה המינימלית עבור Ubuntu 18.04 היא 25GB. לחצו על "צור" בסיום.

7.4.6 התקנת Ubuntu 18.04

1. הפעילו את ה-VM שלכם על ידי לחיצה על כפתור "התחל".
2. בחרו " Install Ubuntu" והקישו "Enter".
3. בחרו את השפה המועדפת ולחצו על "המשך".
4. בחרו את פריסת המקלדת ולחצו על "המשך".
5. בחרו "Normal Installation" וסמנו "Download updates while installing Ubuntu". ניתן לבחור להתקין תוכנת צד שלישי עבור גרפיקה, Wi-Fi וכו'.
6. בחרו "Erase disk and install Ubuntu" and "Install Now" ולחצו על "Install Now".
7. לחצו על "Continue" כדי לאשר שברצונכם למחוק את הדיסק ולהתקין את Ubuntu.
8. בחרו את אזור הזמן שלכם ולחצו על "המשך".
9. צרו שם משתמש וסיסמה חדשים ולאחר מכן לחצו על "המשך".
10. המתינו לסיום ההתקנה ולאחר מכן לחצו על "Restart Now" כדי לאתחל את ה-VM שלכם.
11. היכנסו ל-Ubuntu 18.04 VM החדש שלכם.

7.4.7 הגדרת הגדרות התצוגה

1. בחרו את ה-VM שלכם ולחצו על כפתור "Settings".
2. תחת הכרטיסייה "Screen" < "Display", הקצו זיכרון וידאו ל-128 מגה-בייט לפחות.

לאחר אתחול מחדש, כעת ניתן להשתמש ב-Ubuntu 18.04 VM. מכאן, ניתן להתקין תוכנה, להתאים אישית את הסביבה ולהשתמש בדומה למכונה רגילה של Ubuntu.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

בשלב הבא - כדי לעבוד בקלות על הקוד שלכם ולנהל את הסביבה שלכם, התקינו Visual Studio Code.

Linux – תזכורת ל- Linux Ubuntu יש לעבור על ה-linux Tutorial wiki.

תזכורת ל-ROS יש לחזור אל פרק ROS 4.3.

7.5 נספח ה' - מדריך בנושא Visual Studio SSH Tutorial

פרק זה מהווה הדרכה בשלבי ההגדרה והשימוש ב-SSH בתוך Visual Studio, והוא כולל הנושאים הבאים:

- מה זה SSH?
- דרישות קדם
- הוספת תמיכת SSH ב-Visual Studio
- התחברות לשרת מרוחק
- ביצוע פקודות מרחוק

7.5.1 מושגים מרכזיים

- SSH - Secure Shell הוא פרוטוקול בשימוש נרחב לגישה מרחוק מאובטחת וביצוע פקודות במחשבים מרוחקים.
- Visual Studio, סביבת פיתוח משולבת פופולרית (IDE), מספקת תמיכה מובנית עבור SSH, המאפשרת למפתחים לעבוד עם שרתים מרוחקים להפעיל פקודות ישירות מתוך ה-IDE.

7.5.2 מבוא - SSH

הדרך הנפוצה ביותר להתחבר לשרת Linux מרוחק היא באמצעות SSH. SSH מייצג Secure Shell ומספק דרך בטוחה ומאובטחת לביצוע פקודות, ביצוע שינויים והגדרת שירותים מרחוק. בעת התחברות דרך SSH, ניתן להיכנס באמצעות חשבון משתמש אשר קיים בשרת המרוחק.

7.5.3 איך ה-SSH עובד?

בעת התחברות דרך SSH, ניתן לתפוס לסשן מעטפת, שהוא ממשק מבוסס טקסט שבו ניתן ליצור אינטראקציה עם השרת שלכם. במשך הפעלת SSH, כל הפקודות שניתן להקליד

משרד החינוך המזכירות הפדגוגית אגף א' למדעים | המינהל לטכנולוגיה ומערכות תקשוב | מור-טק | הטכניון מכון טכנולוגי לישראל | הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

במסוף המקומי נשלחות דרך מנהרת SSH מוצפנת ומבוצעות בשרת שלכם. חיבור ה-SSH מיושם באמצעות מודל שרת-לקוח. משמעות הדבר היא שכדי שייווצר חיבור SSH, המחשב המרוחק חייב להפעיל תוכנה הנקראת SSH daemon. תוכנה זו מאזינה לחיבורים ביציאת רשת ספציפית, מאמתת בקשות חיבור ומביאה לסביבה המתאימה אם המשתמש מספק את האישורים הנכונים.

המחשב של המשתמש חייב להיות בעל לקוח SSH. זוהי תוכנה שיודעת לתקשר באמצעות פרוטוקול SSH וניתן לתת מידע על המארח המרוחק שאליו יש להתחבר, שם המשתמש לשימוש והאישורים שיש להעביר לאימות. הלקוח יכול גם לציין פרטים מסוימים לגבי סוג החיבור שהוא ירצה ליצור.

7.5.4 אימות משתמשים על ידי SSH

לקוחות בדרך כלל מבצעים אימות באמצעות סיסמאות (פחות מאובטחות ולא מומלץ) או מפתחות SSH, שהם מאובטחים מאוד.

כניסות לסיסמה מוצפנות וקלות להבנה עבור משתמשים חדשים. עם זאת, בוטים אוטומטיים ומשתמשים זדוניים ינסו פעמים רבות לבצע אימות לחשבונות המאפשרים התחברות מבוססות סיסמה, מה שעלול להוביל לבעיות אבטחה. מסיבה זו, אנו ממליצים תמיד להגדיר אימות מבוסס מפתח SSH עבור רוב התצורות.

מפתחות SSH הם קבוצה תואמת של מפתחות קריפטוגרפיים שניתן להשתמש בהם לצורך אימות. כל סט מכיל מפתח ציבורי ופרטי. ניתן לשתף את המפתח הציבורי בחופשיות ללא חשש, בעוד המפתח הפרטי חייב להישמר בדריכות ולעולם לא להיחשף לאיש.

כדי לאמת באמצעות מפתחות SSH, למשתמש חייב להיות זוג מפתחות SSH במחשב המקומי שלו. בשרת המרוחק, יש להעתיק את המפתח הציבורי לקובץ בתוך ספריית הבית של המשתמש בכתובת

~/.ssh/authorized_keys.

קובץ זה מכיל רשימה של מפתחות ציבוריים, אחד לכל שורה, המורשים להיכנס לחשבון זה.

כאשר לקוח מתחבר למארח, שרוצה להשתמש באימות מפתח SSH, הוא יודיע לשרת על כוונה זו ויאמר לשרת באיזה מפתח ציבורי להשתמש. לאחר מכן השרת בודק את קובץ ה-autorized_keys שלו עבור המפתח הציבורי, יוצר מחרוזת אקראית ומצפין אותו באמצעות המפתח הציבורי. ניתן לפענח הודעה מוצפנת זו רק באמצעות המפתח הפרטי המשויך.

משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
מור-טק
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

השרת ישלח את ההודעה המוצפנת הזו ללקוח כדי לבדוק אם באמת יש לו את המפתח הפרטי המשויך.
לאחר קבלת הודעה זו, הלקוח יפענח אותה באמצעות המפתח הפרטי וישלב את המחרוזת האקראית שמתגלה עם מזהה הפעלה שנערך בעבר.

7.5.5 יצירה ועבודה עם מפתחות SSH

חלק זה יסקור כיצד ליצור מפתחות SSH במחשב לקוח ולהפיץ את המפתח הציבורי לשרתים שבהם יש להשתמש בהם. זהו סעיף טוב להתחיל איתו אם לא יצרתם מפתחות בעבר בשל האבטחה המוגברת שהוא מאפשר לחיבורים עתידיים.
יצירת צמד מפתחות SSH ציבורי ופרטי חדש במחשב המקומי שלכם הוא הצעד הראשון לקראת אימות עם שרת מרוחק ללא סיסמה. אלא אם כן יש סיבה טובה שלא לעשות זאת, יש תמיד לבצע אימות באמצעות מפתחות SSH.
ניתן להשתמש במספר אלגוריתמים קריפטוגרפיים ליצירת מפתחות SSH, כולל RSA, DSA ו-ECDSA. מפתחות RSA מועדפים בדרך כלל והם סוג מפתח ברירת המחדל.
כדי ליצור זוג מפתחות RSA במחשב המקומי שלכם, הקלדו:

```
$ ssh-keygen
```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/demo/.ssh/id_rsa):
```

הנחיה זו מאפשרת לבחור את המיקום לאחסון מפתח ה-RSA הפרטי שלכם. יש להקיש ENTER כדי להשאיר את זה כברירת המחדל, שתשמור אותם בספרייה הנסתרת .ssh בספריית הבית של המשתמש שלכם. השארת מיקום ברירת המחדל תאפשר ללקוח ה-SSH שלכם למצוא את המפתחות באופן אוטומטי.

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```


משרד החינוך המזכירות הפדגוגית אגף א' למדעים
המינהל לטכנולוגיה ומערכות תקשוב
הטכניון מכון טכנולוגי לישראל
הפקולטה לחינוך למדע וטכנולוגיה
כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

ההנחיה הבאה מאפשרת להזין ביטוי סיסמה באורך שרירותי כדי לאבטח את המפתח הפרטי. כברירת מחדל, הזינו כל ביטוי סיסמה אשר מוגדר כאן בכל פעם שיעשה שימוש במפתח הפרטי, כאמצעי אבטחה נוסף. לא להסס ללחוץ על ENTER כדי להשאיר את זה ריק במקרה ולא רוצים ביטוי סיסמה. עם זאת, יש לזכור שזה יאפשר לכל מי שישיג שליטה על המפתח הפרטי שלכם להתחבר לשרתים שלכם. בעת בחירת הזנת ביטוי סיסמה, שום דבר לא יוצג בזמן ההקלדה. זהו אמצעי זהירות אבטחה.

```
Output
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
8c:e9:7c:fa:bf:c4:e5:9c:c9:b8:60:1f:fe:1c:d3:8a root@here
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|                 |
|       +         |
|      o S   .    |
|     o   . * +   |
|      o + = O .  |
|       + = = +   |
|      ....Eo+    |
+-----------------+
```

הליך זה יצר זוג מפתחות RSA SSH, הממוקם בספרייה הנסתרת ssh. בתוך ספריית הבית של המשתמש שלכם. הקבצים האלה הם:

~/.ssh/id_rsa

המפתח הפרטי. אין לשתף את הקובץ הזה!

~/.ssh/id_rsa.pub

המפתח הציבורי המשויך. ניתן לשתף את זה בחופשיות ללא השלכות.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.5.6 יצירת זוג מפתחות SSH עם מספר גדול יותר של ביטים

מפתחות SSH הם 2048 סיביות כברירת מחדל. זה נחשב בדרך כלל לטוב מספיק עבור אבטחה, אבל ניתן לציין מספר גדול יותר של סיביות עבור מפתח מוקשה יותר.

לשם כך, כללו את הארגומנט b- עם מספר הביטים שרוצים. רוב השרתים תומכים במפתחות באורך של לפחות 4096 סיביות. ייתכן שלא יתקבלו מפתחות ארוכים יותר למטרות הגנת DDOS

```
$ ssh-keygen -b 4096
```

בעת יצירת מפתח אחר בעבר, זה יוביל לשאלה האם יש להחליף את המפתח הקודם שלך:

```
Overwrite (y/n)?
```

בעת בחירת "כן", המפתח הקודם יוחלף ולא תתאפשר עוד גישה לשרתים באמצעות המפתח הזה. עקב כך, יש להקפיד על החלפת המפתחות בזהירות.

7.5.7 הסרה או שינוי של ביטוי הסיסמה במפתח פרטי

בעת יצירת ביטוי סיסמה עבור המפתח הפרטי ואם תרצו לשנות או להסיר אותו, ישנה אפשרות לעשות זאת בקלות.

הערה: כדי לשנות או להסיר את ביטוי הסיסמה, יש לדעת את משפט הסיסמה המקורי. אם איבדתם את ביטוי הסיסמה למפתח, אין מנוס ויש ליצור זוג מפתחות חדש.

כדי לשנות או להסיר את ביטוי הסיסמה, הקלידו:

```
$ ssh-keygen -p
```

```
Enter file in which the key is (/root/.ssh/id_rsa):
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

ניתן להקליד את המיקום של המפתח במקרה ותרצו לשנות או לחוץ על ENTER כדי לקבל את ערך ברירת המחדל:

```
Enter old passphrase:
```

הזינו את משפט הסיסמה הישן שברצונכם לשנות. לאחר מכן הזינו ביטוי סיסמה חדש:

```
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
```

כאן, יש להזין את משפט הסיסמה החדש שלכם או להקיש ENTER כדי להסיר את ביטוי הסיסמה.

7.5.8 הצגת טביעת האצבע של מפתח SSH

כל זוג מפתחות SSH חולק "טביעת אצבע" קריפטוגרפית אחת שניתן להשתמש בה כדי לזהות את המפתחות באופן ייחודי. זה יכול להיות שימושי במגוון מצבים.

כדי לגלות את טביעת האצבע של מפתח SSH, הקלד:

```
$ ssh-keygen -l
```

```
Enter file in which the key is (/root/.ssh/id_rsa):
```

ניתן ללחוץ על ENTER אם זה המיקום הנכון של המפתח, אחרת יש להזין את המיקום המתוקן. תתקבל מחרוזת המכילה את אורך הסיביות של המפתח, טביעת האצבע והחשבון והמארח עבורו נוצר, ואת האלגוריתם שבו תעשה שימוש:

```
Output
4096 8e:c4:82:47:87:c2:26:4b:68:ff:96:1a:39:62:9e:4e  demo@test (RSA)
```


כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

7.5.9 הוראות חיבור בסיסיות

הסעיף הבא יכסה כמה מהיסודות על איך להתחבר לשרת עם SSH.

<u>התחברות לשרת מרוחק</u>

כדי להתחבר לשרת מרוחק ולפתוח שם הפעלת מעטפת, ניתן להשתמש בפקודה ssh.

הצורה הפשוטה ביותר מניחה ששם המשתמש שלך במחשב המקומי שלך זהה לזה שבשרת המרוחק. אם זה נכון, ניתן להתחבר באמצעות:

```
$ ssh remote_host
```

אם שם המשתמש שלכם שונה בשרת המרוחק, יש להעביר את שם המשתמש המרוחק כך:

```
$ ssh username@remote_host
```

בפעם הראשונה של התחברות למארח חדש, ניתן לראות את ההודעה הבאה:

```
The authenticity of host '111.111.11.111 (111.111.11.111)' can't be established.
ECDSA key fingerprint is fd:fd:d4:f9:77:fe:73:84:e1:55:00:ad:d6:6d:22:fe.
Are you sure you want to continue connecting (yes/no)? yes
```

הקלידו "Yes" כדי לקבל את האותנטיות של המארח המרוחק.

בעת שימוש באימות סיסמה, הזינו את הסיסמה עבור החשבון המרוחק. בעת שימוש במפתחות SSH, יש להזין את משפט הסיסמה של המפתח הפרטי שלך אם מוגדר כזה, אחרת תבוצע כניסה אוטומטית.

<u>הפעלת פקודה בודדת בשרת מרוחק</u>

כדי להפעיל פקודה בודדת בשרת מרוחק במקום להוליד הפעלת מעטפת, הוסיפו את הפקודה אחרי פרטי החיבור, כך:

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

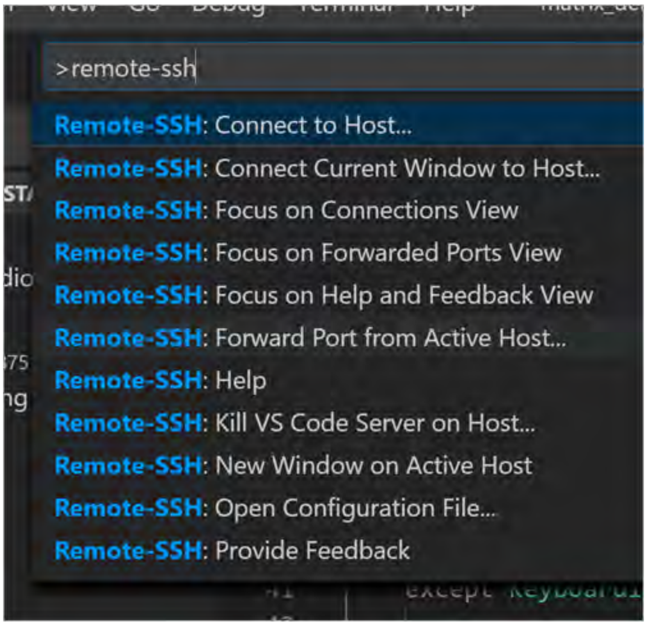
```
$ ssh username@remote_host command_to_run
```

זה יתחבר למארח המרוחק, יאמת עם האישורים ויבצע את הפקודה שציינתם. החיבור ייסגר מיד לאחר מכן.

7.5.10 מדריך SSH ל visual studio code

שלב 1: התקנת תוסף "Remote - SSH"

1. פתחו את Visual Studio Code.
2. עברו לחלון "Extensions" (תפריט צד ימין או תוך לחיצה על Ctrl+Shift+X).
3. חפשו את התוסף "Remote - SSH" ולחצו על "Install" כדי להתקין אותו.
4. אחרי התקנת התוסף, לחצו על "Reload" כדי לטעון מחדש את Visual Studio Code.



שלב 2: התקנת OpenSSH בשרת

(הערה: באופן כללי, אין צורך להתקין open SSH מכיוון שאם ה-Ubuntu לא מגיע כבר עם ה-SSH אפשר להסתמך על ה-SSH הרגיל של Linux: sudo apt-get install ssh)

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

1. בשרת שבו ברצונך להתחבר באמצעות SSH, יש לוודא שהותקנה החבילה OpenSSH. לדוגמה, במערכות מבוססות Ubuntu, ניתן להשתמש בפקודה הבאה על מנת להתקין את OpenSSH:

```
sudo apt-get install openssh-server
```

2. השתמשו בפקודה הבאה על מנת לבדוק את סטטוס השירות:

```
sudo service ssh status
```

3. אם השירות לא פועל, השתמשו בפקודה הבאה כדי להפעילו:

```
sudo service ssh start
```

שלב 3: חיבור לשרת באמצעות SSH

1. פתחו את Visual Studio Code.
2. לחצו על כפתור התפריט בחלק התחתון השמאלי של החלון.
3. בחרו את "Remote-SSH: Connect to Host" ואז "Configure SSH Hosts".
4. בחרו "New Host" על מנת להגדיר מארח SSH חדש.
5. בחרו בתיבת הטקסט שמופיעה, הזינו את פרטי החיבור של השרת:
[user]@[host]:[port]

כאשר:

- [user] הוא שם המשתמש שלך בשרת.
- [host] הוא כתובת ה-IP או שם הדומיין של השרת.
- [port] הוא מספר הפורט בו השירות SSH מקשיב (בדרך כלל 22).

6. לחצו על "Enter" כדי לשמור את השינויים.

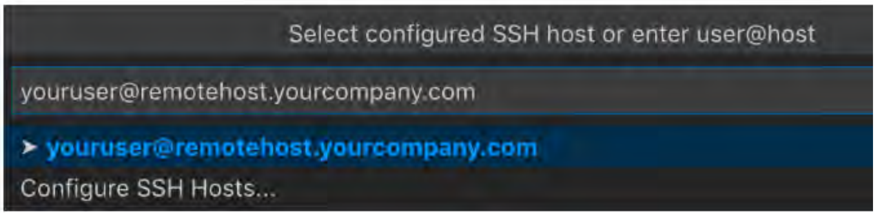
```
ssh user@hostname
# Or for Windows when using a domain / AAD account
ssh user@domain@hostname
```

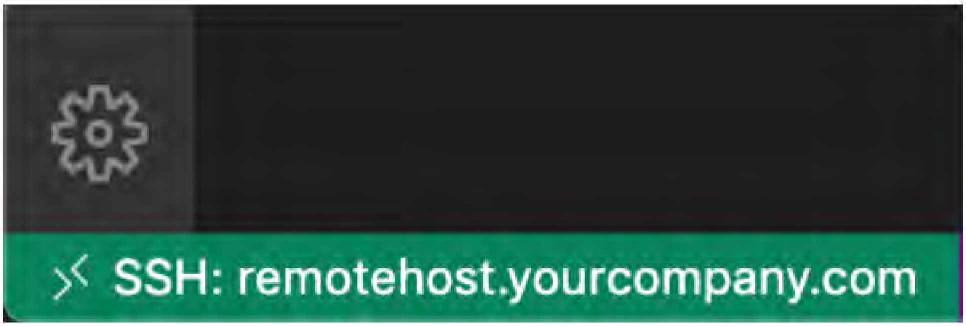
שלב 4: חיבור לשרת עם SSH

8. פתחו את Visual Studio Code.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך

9. לחצו על כפתור התפריט בחלק התחתון השמאלי של החלון.
10. בחרו "Remote-SSH: Connect to Host" ובחרו את המארח שברצונכם להתחבר אליו.
11. התייחסו לחלון חדש שנפתח בתוך Visual Studio Code, והמתינו בסבלנות עד שהחיבור לשרת יסתיים.
12. לאחר ההתחברות המוצלחת, יוצג התוכן של השרת בחלון הפרויקט של Visual Studio Code.



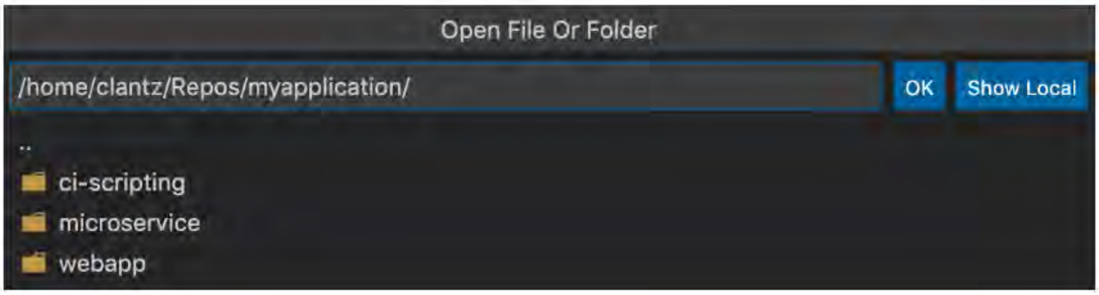


שלב 5: שמירת שינויים ויציאה מהשרת

בעת עבודה עם הקבצים בשרת דרך Visual Studio Code, בצעו שמירות בצורה רגילה על ידי לחיצה על Ctrl+S. בעת יציאה מהשרת, בחרו "Remote-SSH: Disconnect from Host" מתוך תפריט Visual Studio Code.

זהו! ההתחברות לשרת באמצעות SSH בעזרת Visual Studio Code בוצעה. ניתן לערוך ולפתוח קבצים על השרת, לבצע שמירות ועריכות, ולבצע פעולות אחרות ישירות מתוך הסביבה המשתמשת שלכם.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך



דוגמא לחיבור שרת לקוח:

לקוח:

- משתמש: john
- כתובת IP של השרת: 192.168.0.100
- פורט SSH: 22

שרת:

- משתמש Ubuntu:

כדי להקים חיבור SSH מהלקוח לשרת, ניתן להשתמש בפקודה הבאה בטרמינל:

```
ssh Ubuntu@192.168.0.100
```

לאחר ביצוע הפקודה, הזינו את הסיסמה של המשתמש "Ubuntu" בשרת. לאחר הזנת הסיסמה הנכונה, החיבור ל-SSH יתמקם וההתחברות לשרת תבוצע.

מכאן, ניתן לבצע פקודות בשרת, להעביר קבצים, לערוך קבצים ולבצע מגוון משימות ישירות מממשק השורת הפקודה.

כדי לסיים את החיבור ל-SSH ולחזור לטרמינל של הלקוח, הקלידו exit או השתמשו בשילוב המקשים Ctrl+D.

שימו לב: דוגמה זו מניחה תצורת SSH טיפוסית עם שם משתמש, כתובת IP של השרת ופורט SSH ברירת המחדל. הגדרות התצורה שלכם עשויות להשתנות בהתאם להגדרות השרת.

כל הזכויות שמורות למשרד החינוך, מרכז המורים הארצי למקצועות הטכנולוגיים, מור-טק
הפרויקט מבוצע על ידי מוסד הטכניון עפ"י מכרז 22/11.2020. הפרויקט מבוצע עבור המזכירות הפדגוגית, משרד החינוך