CS@BIU
המחלקה למדעי המחשב
אוניברסיטת בר-אילן

Multi-Robot Adversarial System

Ofir Ben-Shoham

Supervisor: Prof. Noa Agmon

אוניברסיטת בר-אילן
Bar-Ilan University

Goal

Developing an adversarial system for Multi-Robot Formation problem.
Both for simulation (on Gazebo) and for Hamsters (real robots) in the lab.

Hamster Robot

תיאור הפרויקט

תיאור כללי

הפרויקט עוסק בבניית מערכת עבור סימלוץ קבוצת רובוטים בסביבה עוינת, בפרט עבור בעיית ה-Formation, בה הרובוטים פועלים יחד להשגת מטרה משותפת.
משמעות הסביבה העוינת, היא שבכל מיקום בו הרובוטים הולכים, יש להם סיכוי להיפגע (ולכן הסביבה נקראת "עוינת"). המערכת הנ"ל יכולה להיות בעלת שימוש צבאי, למשל, כאשר רוצים לבחון משימת רובוטים בידיעה שיש מוקשים במקומות מסוימים, שעלולים לפגוע ברובוט בהסתברות מסוימת.

במהלך הפרויקט, פיתחתי מערכת סימולציה כזו גם עבור סימולציה ממוחשבת (ב-Gazebo) וגם עבור רובוטים אמיתיים (Hamster-ים) במעבדה. על כן, התוצרים הם: מערכת עבור משימת רובוט יחיד (ממוחשבת, וגם עבור רובוטים אמיתיים במעבדה). בנוסף, מערכת עבור משימה קבוצתית של מספר רובוטים שנעים ביחד (ממוחשבת, וגם עבור Hamster-ים, רובוטים אמיתיים במעבדה).
כמו כן, מעבר לכך שיש בכל מיקום הסתברות שהרובוט ייפגע, הרובוטים נעים במרחב תוך הימנעות ממכשולים/הימנעות מהתנגשות ברובוטים אחרים.

תיאור מפורט

המערכת מורכבת מ-3 מודולים מרכזיים: Stopper, Logger, Adversarial Monitor, כפי שניתן לראות בתרשים הבא:

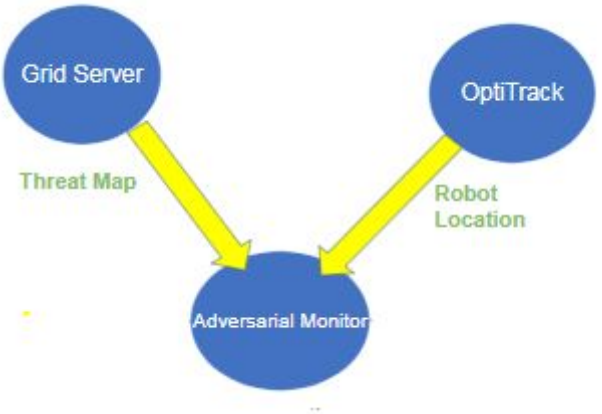
מודול ה-**Logger** - מודול המשמש לצרכי בקרה ותחזוקה. עוזר ל-debugging, אוסף log-ים על אירועים בהם הרובוטים נתפסו באיומים. כמו כן, שומר בקובץ את המסלול שכל רובוט עשה ב-grid עד רגע תפיסתו. בנוסף, שומר לקובץ את מפת האיומים, בה ניתן לראות את ההסתברויות בכל מיקום ומיקום בלוח שהרובוט ייתפס. אפשרות נוספת שה-Logger מאפשר, הוא שליחת דו"ח אוטומטי בסיום ריצת התוכנית, אשר מכיל את דיווחי הרובוטים - באיזו משבצת כל אחד מהם נתפס באיום ב-grid, מתי, ומה ההסתברות בה הוא נתפס. כל הקבצים הנ"ל נוצרים בתוך מודול ה-Logger תחת תיקייה חדשה.

מודול ה- **Adversarial Monitor** - מודול שמכיל את הלוגיקה המרכזית של המערכת. מודול זה מקבל נתוני מיקום רובוטים שהוא מקבל מהממשקות למערכת ה-optitrack (הרחבה בהמשך).

בנוסף, הוא מקבל גם את מפת האיומים מתוך ה-grid server (הרחבה בהמשך). בהינתן שניהם, הוא קובע האם הרובוט נפגע מהאיום ע"פ ההסתברות איום שהוגדרה עבור המשבצת הזו ב-grid. במידה והרובוט נפגע, ה-Adversarial Monitor שולח דיווח על כך למודול ה- Logger (כפי שפורט קודם לכן) וגם ל-Stopper (על מנת לעצור את תנועת הרובוט, כי הוא נתפס).

מודול ה- **Stopper**- מודול שאחראי לתנועת הרובוט במרחב. בנוסף, אחראי להתחמקות ממכשולים (לא איומים מהמפת איומים, אלא ממש מכשולים פיזיים בעולם, לדוגמה כיסא). כאשר מקבל event מה-Adversarial Monitor שהרובוט נתפס באיום, מפסיק להזיז אותו, כי הוא נתפס ואינו יכול להיות יותר בפעולה.

הקשר בין ה-grid server והמידע מה-Optitrack ו- ל-Adversarial Monitor:



ה- **Optitrack** מספק לנו את מיקומי הרובוטים במעבדה. המיקום נשלח אל ה-Adversarial monitor, שמתרגם את המיקום לאינדקסים (row, colulmn) ב-grid שלו (מפת האיומים), וכך יודע מה ההסתברות של הרובוט להיפגע במשבצת הזו ב-grid. הסבר מלא איך לתפעל את ה-optitrack מבחינה טכנית, מופיע בהמשך המסמך.

ביצוע אינטגרציה עם מערכת ה-Optitrack:

בעצם ה-optitrack זו מערכת שאחראית להגיד לנו מה מיקומי הרובוטים בשטח תחום. היא עושה זאת ע"י מצלמות שנמצאות על התקרה במעבדה. אלו למעשה לא ממש מצלמות רגילות, הן שולחות קרני לייזר לזיהוי הרובוטים. לכן, על כל רובוט, שמתי מחזירי אור כך שהמצלמות יוכלו לזהות אותו בשטח המעבדה ולדווח על מיקומו.

התמונה הבאה ממחישה את המתואר לעיל:

חשוב לציין, שעבור כל רובוט מחזירי האור נמצאים במקומות אחרים על גופו - זאת, כדי שמערכת ה-optitrack שמאפיינת את הרובוטים ע"פ זוויות ההחזרה של האור, לא תתבלבל בין שני רובוטים שעבורם יש זוויות החזרה דומות. על כן, לא קיימים שני רובוטים שעבורם כל מחזירי האור נמצאים על גופם במקומות זהים, מתוך רצון שהמערכת תבדיל בין הרובוטים השונים ותספק ל-Adversarial monitor את מיקומי הרובוט המתאימים עבור כל רובוט.

הסבר על ה-grid server

ה-grid server הוא סרבר המכיל את ה-grid. למעשה זו רשימה דו ממדית, כך שבכל תא בה יש מספר בין 0-1 המסמל את ההסתברות של רובוט שנמצא במשבצת הזו, להיפגע מאיום.
ההסתברויות האלו נוצרו באופן אקראי ע"י הגרלת פיקסלים לתמונה בגודל ה-grid, לאחר מכן, מנרמלים את ערך כל פיקסל בקבוע כלשהו, על מנת לקבל הסתברויות בין 0-1.
לאחר שה-Adversarial monitor מתרגם את מיקום הרובוט שהתקבל מה-Optitrack למשבצת ב-grid, הוא יכול לפנות ל-grid server כדי לקבל את ההסתברות של הרובוט להיפגע במשבצת זו.
אולם, משיקולי יעילות, במקום לבצע כל פעם פניה מחדש לסרבר, ניתן לשמור פשוט את המערך הדו ממדי שהתקבל מה-grid server באופן סטטי בתוך ה- Adversarial monitor, וכך לבצע פניה אחת בלבד ל-grid server, שמחזירה את הרשימה הדו ממדית של ההסתברויות, או בשמה השני - מפת האיומים.

פירוט אתגרי הפרויקט + כיוונים עתידיים

התמודדות עם אתגרי הפרויקט

- למדתי חומר רב במהלך הפרויקט שלא נחשפתי אליו בעבר. עשיתי זאת באופן עצמאי לחלוטין ושילבתי תרגול והתנסות אמיתית במעבדה על מנת לחדד את ההבנה.

- התממשקות למערכת ה-optitrack. מדובר במערכת די חדשה, שאין עליה הרבה מידע באינטרנט כך שאין מדריכים לתפעול והתממשקות אליה. על כן, נדרש זמן רב להבנתה ופיתוח קוד רלוונטי המתממשק עימה.

- אתגר נוסף בפרויקט, היה הצורך להגיע למעבדה באופן תדיר מאוד על מנת לעבוד עם הרובוטים. זאת מאחר ומערכת ה-optitrack מרושתת אך ורק במעבדה. על מנת להתמודד עם אתגר זה, פיתחתי את הקוד עצמו בבית, בדקתי עד כמה שאפשר ע"י הרצת טסטים. לאחר מכן, הגעתי למעבדה ובדקתי את הקוד על הרובוטים האמיתיים.

כיוונים עתידיים

- מחקר שיעסוק במציאת המסלול האופטימלי עבור רובוט יחיד שנדרש להתחמק מאיומים. המערכת שפיתחתי תעזור משמעותית בבדיקה האם המסקנות מהמחקר עובדות במבחן המציאות, זאת, תוך סימלוץ סביבת האיומים ותנועת הרובוט במעבדה - ע"י המערכת שבניתי.

- לאחר מכן, ניתן להרחיב זאת למספר רובוטים, ולעסוק בפרט ב-Formation - משימה משותפת של מספר רובוטים בסביבה עוינת, בה הם מאוימים.

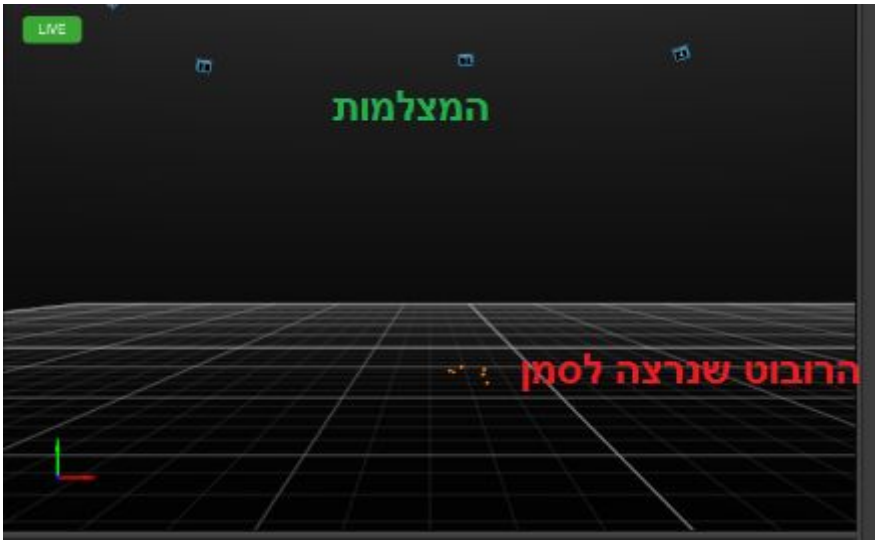
- הרחבת הפרויקט למסגרת הצבאית/ביטחונית. ניתן להסתכל על כל איום כמוקש, ולהשתמש במערכת שבניתי על מנת לסמלץ את תנועת הרובוטים בקבוצה שמבצעת משימת פטרול בשטח צבאי, כך שהאיומים הם המוקשים וכל מוקש פוגע ברובוט בהסתברות מסויימת.

מדריך למשתמש - הוראות התקנה/תחזוקה/הרצה

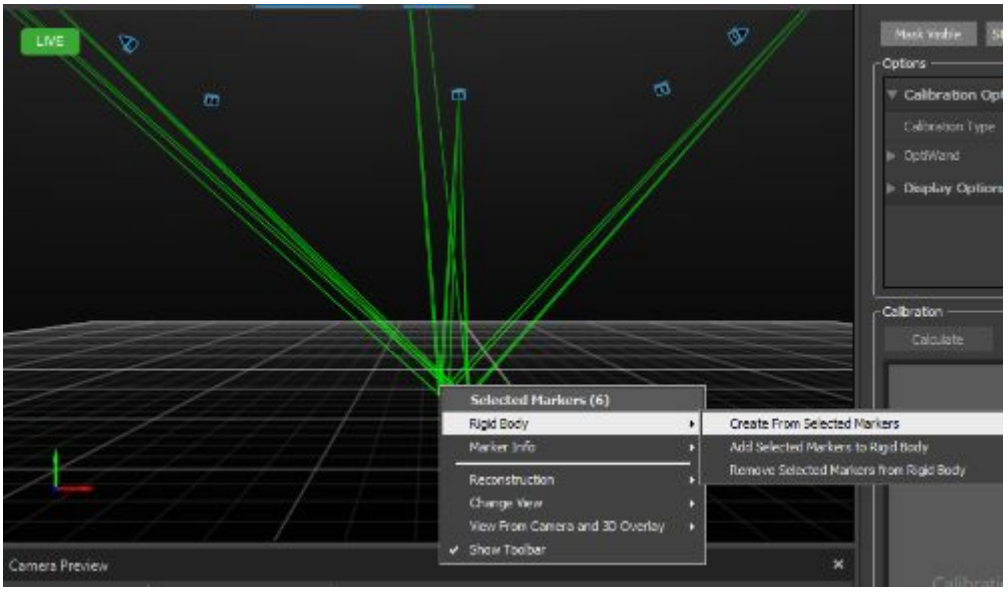
עבור ה- Hamster-ים במעבדה:

ראשית, חיבור הכבל של ה-optitrack במעבדה ואז יש לפתוח את תוכנת Motive. יש לפתוח ב-Movit את הקובץ הגדרות שניתן להורדה בלינק הבא:
https://drive.google.com/file/d/1XiF8fgbNTJXvY2toFI8YOETLBxPU4R8c/view?usp=sharing

Motive, ניתן לראות את קרני האור שחוזרות ממחזירי האור שנמצאים על ה-Hamster:



יש לסמן את הרובוט:



כעת, יש לוודא את הנתונים הבאים ב stremming -> File (על מנת שנוכל לקבל את המידע מחיישני ה-optitrack):

לאחר מכן:
במחשב הלוקאלי (לא חייבים במחשב של ה-optitrack) צריך להתקין שני package-ים על מנת שיהיה ניתן להתחבר למידע שה-optitrack מפרסם.

- download packages from here
- copy-paste them to catkin_ws

לאחר מכן, סנכרון עם מערכת ה-optitrack (**מהמחשב הלוקאלי**, לא צריך באחד מהרובוטים, באופן הבא):

- cd catkin_ws
- source ./devel/setup.bash && catkin_make
- export ROS_IP={your_local_ip in lab wifi).
- export ROS_MASTER_URI=http://10.0.2.X:11311 (where 10.0.2.X is the IP address of the master node, which contains the monitor, as mention later).
- Before running the next line, verify that the master node is running, see (*).
- If rviz was not installed, you have to install it (see http://wiki.ros.org/rviz).
- roslaunch optitrack optitrack_pipeline.launch (to listen to optitrack's data)

אתחול אחד מהרובוטים כרובוט ה-master:

- ssh pi@10.0.2.X (where X is the hamster id (1-9), pi is the user)
- cd hamster_ws
- source ./devel/setup.bash && catkin_make (compile the packages)
- roslaunch hamster_launch single_board.launch (this runs the master node) (*)
- roslaunch ros_pam monitor_server.launch

- roslaunch ros_pam move_robot_only.launch robot_name:=agentX

לאחר מכן - עבור כל אחד מהרובוטים הבאים:

- ssh pi@10.0.2.Y (Y is the hamster id (1-9)).
- export ROS_MASTER_URI=http://10.0.2.X:11311 (where 10.0.2.X is the IP address of the master node (define 10.0.2.X as master for this robot).
- cd hamster_ws
- source ./devel/setup.bash && catkin_make (compile the packages)
- roslaunch ros_pam move_robot_only.launch robot_name:=agentY

חשוב לוודא שה-identifier בקובץ optitrack_listeners זהה לזה שהוגדר במחשב שבו נמצא ה-optitrack (ניתן לראות זאת ב- rviz).

מצ"ב התמונה מה-rviz שבה ניתן לראות לאיזה מיקומי רובוטים אנו מאזינים. לדוגמה, כאן בתמונה אנו עוקבים אחרי bstick.