AutoMPW

מערכת תכנון משימה אוטומטי

Automatic Mission Planning Workstation

מגישות : ברכה אסולין ואושרית וידאל

מנחה מקצועי : אלון מה-יפית

מנחה אקדמי: אבי טרייסטמן

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

תקציר

אלתא מערכות בע"מ היא חברת בת של התעשייה האווירית לישראל, ואחת החברות המובילות בעולם במערכות אלקטרוניות צבאיות בתחום המכ"ם, התקשורת והלוחמה האלקטרונית.

השם אלת"א הוא ראשי תיבות של "אלקטרוניקה תעשיה אווירית".

במסגרת פרויקט גמר זה פיתחנו מערכת שהיא שדרוג של מערכת תכנון משימה הקיימת בחברה, השינוי מתבטא במעבר מתכנון ידני לתכנון אוטומטי.

משימת צילום מורכבת ממספר חלקים: אוסף מטרות, נתיבי טיסה שמהם יצולמו המטרות ע"י הסנסור וקטעי הצילום בפועל על נתיבי הטיסה. בנוסף, לכל מטרה יש מספר רב של אילוצים שצריך לספקם בכדי שהסנסור יוכל לבצע את הצילום.

משימת הצילום נועדה לאיסוף מידע מודיעיני, למעקב ולניטור אחר מידע ביעדים ספציפיים.

במהלך הפרויקט כתבנו אלגוריתם לסיפוק אילוצים מבוסס Backtracking. האלגוריתם מנסה לשייך מטרות לנתיבי טיסה קיימים, תוך סיפוק אילוצי הסנסור, בכדי להגיע למספר נמוך ככל האפשר של נתיבי טיסה ולהתרחק כמה שיותר מהפתרון הטריוויאלי בו לכל מטרה משויך נתיב טיסה נפרד. בנוסף, נעשה שימוש בחישובים מורכבים לתיקון אילוצים שנכשלו. כל תהליך הפיתוח נוהל בשיטת Agile. ניהלנו מערכת לוגים כדי לקבל אינדיקציה על המתרחש בעת הרצת האלגוריתם.

לאחר בדיקות וקבלת תוצאות, ניתן לומר כי מטרת הפרויקט הושגה ואכן המערכת מספקת פתרון אוטומטי עבור כל אוסף של מטרות שהוגדרו ע"י המשתמש וכן פתרון זה אינו הפתרון הטריוויאלי.

Abstract

Elta Systems Ltd. is a subsidiary of IAI (Israel Aerospace Industries) and one of the leader companies in army electronic systems in Radar, communication and electronic warfare domain.

Elta's acronym is "Aerospace Electronics".

The goal of this final project is to develop a system which is an improvement of a mission planning system existing in Elta. We want to change manual mission planning to automatic.

An imaging mission includes a collection of targets, flight paths for photographing targets by the sensor and sections where the actual photography will be done. In addition, each target has a large number of constraints that must be provided for the sensor to be able to photograph.

An imaging mission is intended for gathering intelligence information and tracking information in specific destinations.

During this project, we wrote a constraints satisfaction algorithm based on "Backtracking" method. The algorithm trying to associate targets to existing flight paths in order to minimize the number of flight paths and stay as far away as possible from the trivial solution in which is for each target a separate flight path is associated. Moreover, we used complex calculations to change legs so that failed constraints will be provided.

The development process is done by agile method. We managed logs system to get an indication of the algorithm's decisions while running.

After testing and results, we can say that the project's goal is reached and indeed the system provides an automatic solution for each collection of targets defined by the user and this solution is not the trivial one.

תודות

בראש ובראשונה אנו מודות לבורא עולם שסייע לסיים בהצלחה את פרויקט הגמר.

תודה למנחה האקדמי מר אבי טרייסטמן, על סלילת הדרך לכניסה לחברה, הליווי במהלך השנה וההכוונה לאורך כל הדרך.

תודה לאלון מה יפית, המנחה המקצועי, יזם הרעיון, שהנחה אותנו במקצועיות רבה בכל שלבי הפרויקט, על הנכונות לעזור ולהעניק כלים מתאימים לביצוע המשימות, על הסבלנות הרבה והאמונה ביכולותינו.

תודה לדותן בן אבי, ראש תחום תצוגות סייבר ושו"ב (שליטה ובקרה) על האכפתיות בהתקדמות הפרויקט והמענה לכל בקשת עזרה.

תודה לנטע לוי, אחראית פרויקטים בתוכנה, שיזמה את שילובינו בפרויקט ותודה על כל העזרה והתמיכה בכל שאלה ובקשה.

תודה לטל עטר, ראש צוות Web, על כל העזרה ושיתוף הפעולה בפיתוח צד הלקוח .

תודה לגרגורי מיקיטס, על הזמן שהקדיש בשביל להכיר לנו את הקיים בחברה.

תודה לשי הרמל וליטל פילבר, על הסיוע בכניסתנו לחברה ובהתאקלמות.

כמובן, נודה למשפחותינו היקרות, שתמכו בנו לאורך הפרויקט, על הסבלנות העזרה והאמונה בנו.

הפרויקט הזה הוא שלכם לא פחות מאשר הוא שלנו.

תוכן עניינים

טבלאות

גרפים

תמונות

1. מבוא

בפרק זה נציג את הבעיה הכללית שבעקבותיה פיתחו את המערכת הקיימת (MPW) וכן את הצורך בפיתוח AutoMPW. בנוסף, נציג מושגים כללים על הפרויקט ונסביר אותם.

1.1 בעיה כללית

צילום מגובה נעשה כבר בסוף המאה ה-19, בעיקר כאתגר טכנולוגי ולמטרות אמנותיות, וכן למכירת התצלומים לאספנים. שימוש משמעותי ראשון בצילום מגובה התחיל במלחמת העולם הראשונה, במהלכה מטוסים צילמו מגובה רב את שדה הקרב ועל ידי כך סיפקו לפיקוד הצבאי מידע מודיעיני חשוב.

בעיה ידועה בתחום הצילום מגובה היא מציאת נתיבי טיסה עבור מטרות כלשהן בתנאים מסוימים.

עפ"י הפלטפורמה בה משתמשים על מנת לצלם נקבעים האילוצים שצריך לספקם בכדי למקסם את יעילות הסנסור. אילוצים הנובעים מהפלטפורמה יכולים להיות צילום ביום או צילום בלילה, תנאי מזג אויר, גובה, זווית צילום, מהירות טיסה ועוד. אילוצים נוספים שצריכים להתייחס אליהם נובעים מתנאי השטח , אזורים מותרים לטיסה, נתיבי טיסה הכרחיים ודלק.

הבעיה איתה מתמודדים היא שיוך נתיבי טיסה המספקים את האילוצים, בהינתן המטרות שצריך לצלם ורשימת אילוצים כנ"ל.

הבעיה לעיל היא בעיית סיפוק אילוצים (CSP-Constraints Satisfaction Problem) בה צריך לשייך ערכים למשתנים תוך סיפוק האילוצים הקיימים בין המשתנים.

1.2 בעיה ספציפית ופתרונה

בחברת אלתא התמודדו עם בעיה ספציפית של צילום מטרות מגובה ע"י סנסור SAR.
סנסור SAR (Synthetic-aperture radar) הוא מכ"ם המשמש ליצירת תמונות דו ממדיות או שחזורים תלת ממדיים של עצמים. SAR מגלה תכונות פיזיקליות ע"י קרינה אלקטרומגנטית לעומת סנסורים אחרים המשתמשים בטכנולוגיה אופטית. בעקבות כך, SAR יכול "לראות" ללא תלות מזג אוויר ובתאורת יום או לילה. SAR מותקן בדרך כלל על רציף נע כמו כלי טיס או חללית.

הפתרון הקיים בחברה כרגע הוא מודול MPW (Mission Planning Workstation) שמטרותיו הן לתכנן ולנטר ביצוע משימות צילום עבור הסנסור.
המודול מאפשר תכנון של משימות צילום לפני ביצוע המשימה בפועל על ידי המל"ט. המודול אחראי על ניטור ביצוע המשימה על ידי המל"ט בזמן אמת, אימות ההתאמה של הביצוע מול התכנון והקליטה של תוצרי המשימה בצורה תקינה בתחנה הקרקעית.
המשתמש בMPW מתכנן בצורה ידנית משימת צילום ע"י יצירת אזורי מטרה (Targets) לצילום ע"י סנסור הSAR וכן יוצר נתיבי טיסה (Legs) אשר תומכים בצילום אזורי מטרות אלו בצורה מיטבית התואמת את מאפייני הסנסור. אין התייחסות במודול לאילוצים אחרים שלא נובעים

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

ממאפייני הסנסור. ה-Leg יכול לכלול מספר מטרות ומטרה שייכת לLeg אחד בלבד. קבוצה של Legs והמטרות שלהם נשמרים כמשימה (MPW Mission).
תהליך תכנון משימת הצילום זמין רק במודול MPW, מודול זה מכיל את הפריסה הנדרשת לתכנון המשימה המכילה את כל הכלים הנחוצים לכל שלבי התכנון- יצירת משימות, עריכת משימות, ניתוח משימות והעלאתם לביצוע.
כיום מערכת ה-MPW מאפשרת בניית תכנית משימה בצורה ידנית בלבד. המשתמש במערכת (המתכנן) מגדיר מטרות לצילום ונתיבי טיסה לביצוע המטרות המוגדרות בצורה ידנית. המערכת מאפשרת בדיקת תקינות של המשימה שנבחרה. בדיקת תקינות הינו תהליך שבו המערכת מוודאת שכל המטרות שהוגדרו לצילום אכן ניתנות לצילום מנתיבי הטיסה שהוגדרו ואכן הסנסור מסוגל לבצע את העבודה שהוגדרה בהתאם ליכולות הסנסור ותנאי השטח. למשל, במידה והסנסור מוגבל לצילום ממרחק מסוים, מערכת ה-MPW תוודא שאכן המרחק בין מטרת הצילום לנתיב הטיסה יהיה בהתאם למגבלות, לאורך כל זמן הטיסה שבו יתבצע הצילום בפועל. במידה ולא, מערכת ה-MPW תתריע על אי-תקינות המשימה ולא תאפשר ביצוע של המשימה בפועל.

סט האילוצים הרב גורם לכך שהתכנון הידני קשה לביצוע ודורש מיומנויות מורכבות מהמשתמש.
במקרים מסוימים, התכנון הידני לא מאפשר מיצוי מלא של יכולות הסנסור בשל הקושי לפתור מערכת מסובכת של אילוצים בצורה ידנית.
בעקבות כך נולד הצורך לבנות את מערכת AutoMPW המחשבת נתיבי טיסה עבור מטרות נתונות בצורה אוטומטית.
מטרת AutoMPW היא שהמשתמש יכניס מטרות אותן הוא רוצה והמערכת תחשב באופן אוטומטי את נתיבי הטיסה תוך שימוש במספר נמוך של Legs. (מסמך פנימי מסווג)

1.3. סיכום

ישנה בעיה אשר נפוצה בעולם כולו של צילומי שטח מגובה תוך סיפוק אילוצים .
קיימים אילוצים רבים לביצוע משימת הצילום אשר יכולים לנבוע מאספקטים שונים :

- מאפיינים פיזיקליים של הפלטפורמה בה משתמשים לצילום.
- תוואי שטח וגיאוגרפיה.
- נתיבי טיסה אפשריים.

המודול הקיים היום בחברה מאפשר תכנון משימה באופן ידני, בדיקת תקינות המשימה ונתיבי הטיסה ובעקבותיה שינוי המשימה ע״י המשתמש כנדרש.
הקושי של המשתמש במודול הקיים הוא שכל העבודה מתבצעת באופן ידני הדורש מיומנות וזמן ולפעמים אף לא מאפשר ניצול מקסימלי של יכולות הסנסור.
לכן, התעורר הצורך בפיתוח מערכת חדשה אשר תייצר משימת צילום עבור המל״ט באופן אוטומטי.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

2. סקירה ספרותית

בפרק זה נבחן את הרקע לבעיה שהזכרנו במבוא, מתוך הספרות. בנוסף, נפרט שיטות קיימות לפתרון עבור תחומים שונים הנוגעים בבעיית CSP. כמו כן, נסקור מתודולוגיות פיתוח תכנה אפשריות.

כפי שהוזכר במבוא לעיל, מערכת AutoMPW עוסקת בפתרון של בעיית סיפוק אילוצים (CSP) אשר ידועה כבעיית NP-Hard. ישנן שתי הצגות פורמליות לבעיית סיפוק אילוצים :

1. זוג $< S, \emptyset >$, כאשר S הוא שטח החיפוש, מכפלה קרטזית של n קבוצות של טווחים(domains) סופיים $S = D_1 \times \ldots \times D_n$ ו- $\emptyset$ היא פונקציה בוליאנית על S.
   פתרון של CSP הוא $s \in S$ כאשר $\emptyset(s) = true$.
   (B. G. W. Craenen et al, 2003)
2. מציאת השמה ל-n משתנים $V_1, .., V_n$ מתוך הטווחים הסופיים שלהם $D_1, .., D_n$ כך שm האילוצים $C_1, \ldots, C_m$, שנקבעו עבור המשתנים, מסופקים.
   (Stuart J., Peter Norving, 2009)

ישנם מס' סוגי אילוצים :

- Binary Constraints- אילוצים שהם קשר בינארי בין המשתנים. לדוגמא : $X < Y$.
- Unary Constraints- אילוצים שהם קשר אונארי. לדוגמא : $X < 2$.
  . (David A. Cohen et al, 2019)
- Higher-Order Constraints- אילוצים שמעורבים בהם שלושה או יותר משתנים. לדוגמא : $X \neq Y \neq Z$
- Preference- עדיפות גבוהה יותר של ערך מסוים מטווח של משתנה לעומת ערך אחר.

2.1 ביטויים לבעיית CSP

בעיית CSP מיושמת בתחומים רבים. בכל תחום בעיית CSP באה לידי ביטוי באופן שונה וכך גם פתרונה, נציג כמה מהתחומים כאן.

2.1.1 בעיית Job Shop

בעיה זו כוללת מכונות שונות שמבצעות פעולות על משימות. לכל משימה יש סדר עיבוד ספציפי שהיא צריכה לעבור דרך המכונות, כלומר משימה מורכבת מרשימה מסודרת של פעולות שכל אחת מהן מתבצעת במכונה ספציפית ובזמן עיבוד מסוים. ישנם מס' אילוצים על המכונות והמשימות : 1. אין אילוצי קדימויות בין פעולות של משימות שונות. 2. א"א לעצור פעולה שמתבצעת (אין חטיפה) וכל מכונה יכולה לטפל בו זמנית במשימה אחת בלבד. 3. כל משימה יכולה להתבצע בו זמנית רק על מכונה אחת.

זמן השלמה מקסימלי – הוא זמן הסיום המקסימלי של פעולה כלשהיא מתוך זמני הסיום של כל

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

הפעולות. בעוד שרצף המכונות עבור כל משימה ידוע, הבעיה היא למצוא עבור כל מכונה רצף משימות לביצוע כך שנקבל 'זמן השלמה מקסימלי' מינימלי מתוך כל הרצפים האפשריים.

הגדרה פורמלית של הבעיה: $V = \{0, \dots, n\}$ הוא סט הפעולות כאשר 0 ו-n הן פעולות דמי התחלתית וסופית.

M הם m המכונות ו- A הוא סט של זוגות סדורים של פעולות לפי אילוצי קדימויות של פעולות. עבור כל מכונה k מוגדר E_k שהוא סט של זוגות של פעולות המתבצעות על מכונה k . עבור כל פעולה i מוגדר זמן עיבוד p_i ומסמנים את הזמן המוקדם ביותר להתחיל פעולה בתור t_i (Adams J. et al, 1988).

המטרה: למצוא את הזמן המינימלי עבור t_n.

צריך להתקיים:

1) $t_j - t_i \geq p_i \ \forall (\mathrm{i,j}) \in \mathrm{A}$ מבטיח שרצף העיבוד של הפעולות בכל משימה תואם את מה שנקבע מראש.
2) $t_j - t_i \geq p_i \text{ or } t_i - t_j \geq p_j \ \forall \{\mathrm{i,j}\} \in E_k \ \forall \mathrm{k} \in \mathrm{M}$ מבטיח שכל מכונה מבצעת פעולה של משימה אחת בלבד בו-זמנית.
3) $t_i \geq 0 \ \forall \mathrm{i} \in \mathrm{V}$ מבטיח השלמת כל הפעולות.

את הפיתרון לבעיה זו מציגים ב- edge-weighted graph. בגרף זה כל פעולה היא קודקוד כולל שני קודקודים לפעולות הדמי 0 ו- n. בין שתי פעולות עוקבות של אותה משימה יש קשת. עבור על זוג $\{i,j\} \in E_k$ שמתבצעות על אותה מכונה יש בין הקודקודים שתי צלעות, מi לj ובכיוון ההפוך. קשת יחידה בין שני קודקודים מבטאת את האילוץ של קדימות בין פעולות ושתי קשתות בין שני קודקודים מבטאת את העובדה שכל מכונה יכולה לטפל רק בפעולה אחת בו זמנית. משקל של צלע (i,j) הוא p_i שהוא זמן העיבוד של פעולה i. כל הצלעות שיוצאות מ-0 משקלן 0. המטרה היא למצוא את סדר הפעולות על כל מכונה. כלומר, מתוך הצלעות הדו-צדדיות צריך לבחור צלע אחת כך שאין קונפליקטים של קדימויות בין הפעולות ושהמסלול בעל העלות המקסימלית בין ההתחלה לסוף הוא מינימלי(מבין כל המסלולים המקסימליים האפשריים). כמובן ש- t_i נקבע לפי המסלול המקסימלי מקודקוד 0 לקודקוד i.

Critical path- המסלול המקסימלי בין קודקוד ההתחלה לסוף. כל צלע על המסלול הקריטי נחשבת צלע קריטית. על מנת לשפר את הפיתרון ולקבל את המסלול הארוך ביותר המינימלי יש לשנות את סדר המשימות של מכונה במסלולים הקריטיים.

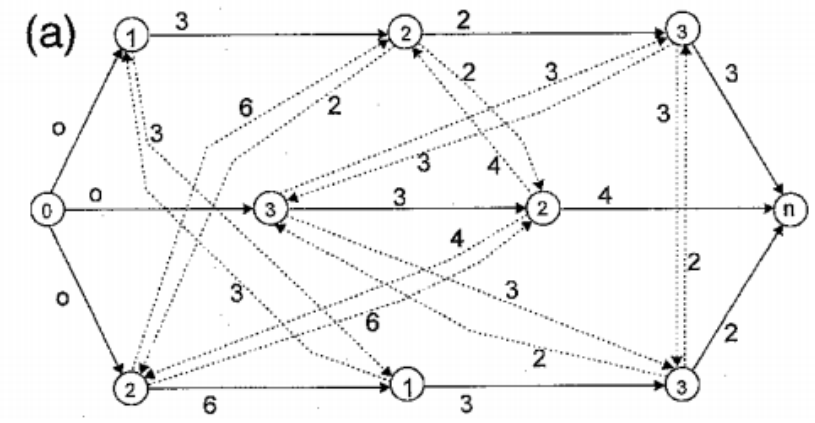
דוגמא:

	J1	J2	J3
M1	3	--	3
M2	2	4	6
M3	2	3	2

טבלה 1: זמני עיבוד של שלוש משימות ע"י שלוש מכונות

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

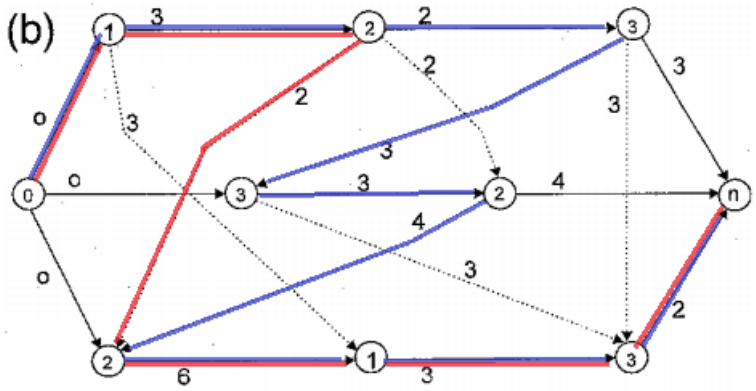
הגרף עבור המופע המוצג בטבלה שלעיל:



גרף 1: תיאור המוצג בטבלה מספר 1

המסלול העליון מציג את רצף הפעולות של משימה J_1, האמצעי של משימה J_2 והתחתון של משימה J_3.

נבחר עבור מכונה M_1 את רצף המשימות: $J_1 \rightarrow J_3$ עבור מכונות M_2 ו-M_3 את הרצף: $J_1 \rightarrow J_2 \rightarrow J_3$ ונקבל את המסלול הארוך ביותר במשקל 26 (צבע כחול), כפי שניתן לראות באיור הבא.



גרף 2: המסלולים המתקבלים עבור הרצף לפני ואחרי שינוי.

אם נשנה את הרצף של מכונה M_2 ל: $J_1 \rightarrow J_3 \rightarrow J_2$ נקבל שהמסלול הארוך ביותר אורכו 16(בצבע אדום).

יוצא, שעל ידי היפוך צלע במסלול הקריטי מזערנו את אורך המסלול הקריטי.

המעבר על כל האופציות נעשה ע"י תכנות דינמי.

(J. Btazewicz et al, 1996)

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

2.1.2. תכנון משימה מבוססת סוכן

Situation Aware UAV Mission Route Planning

נגדיר את תכנון מסלולי המשימה כמציאת סט של נקודות (waypoints) הטובות ביותר עבור המל"ט, שיגביר את ההסתברות להצלחה במשימתו. תכנון זה יכול להתבצע בזמן המשימה או לפני המשימה, תוך התייחסות להגבלות יעדים, לשטחי איום, לדלק, ולחלל האווירי. סביבת התכנון היא דינאמית לכן נחשב את המסלול בכל פעם מחדש כאשר ישנו שינוי- ערכי הצמתים יקבעו עפ"י המצב הנתון. אלגוריתם החיפוש שבו משתמשים הוא A*. המסלול מורכב מנקודות התחלה וסוף ובמיקום מסוים בין נקודות אלו יהיה מספר אפשרויות של נסיעה או אפשרות יחידה, הבעיה מתעוררת כאשר ישנן כמה אפשרויות ונרצה לקחת את האפשרות הטובה ביותר מתוכן.

"מתכנן מסלול משימה של מל"ט" הוא כלי שמבצע חיפוש כדי למצוא את מסלול הטיסה הטוב ביותר מנקודת ההתחלה דרך שטח מוגן לנקודת יעד או לקבוצה של נקודות יעד. נקודת ההתחלה יכולה להיות ההתחלה של ההמראה ממש וכן יכולה להיות כל נקודה באמצע המסלול שהיא מעבר. משימת המל"ט יכולה להיות מורכבת משתי נקודות בלבד, התחלה (המראה) יעד(מטרה) וחזרה וכן גם יכולה להיות מורכבת מסדרת נקודות. ההצלחה נמדדת ביכולת להימנע מאיומים אשר יכולים להיות מכ"ם וטילים (Paterson, 1997).

סוכן הוא כל דבר שיכול לקלוט את הסביבה שלו באמצעות חיישנים ולפעול עליה באמצעות המידע שהחיישן נותן. הסוכן מקבל מספר נתון של פעולות המובילות למצבים של ערכים ידועים, ומחליט מה לעשות ע"י בחינת פתרונות אלו ובחירת הרצף הטוב ביותר (Stuart J., Peter Norving, 2009).

התהליך מתבצע בהדרגה ע"י 1. תפיסת גורמים קריטיים בסביבה. 2. משמעות הגורמים ביחס ליעדים 3. חיזוי העתיד.

הפתרון המוצע משתמש באלגוריתם חיפוש A*. (P. Hart et al , 1968) אלגוריתם זה הוא הנפוץ ביותר לבעיות בתכנון מסלול בגלל הביצועים הטובים שלו ויישומם על מגוון רחב של בעיות ניתוב. (Sezer, 2000) ל-A* יש פונקציית הערכה $f(n)$ המורכבת מעלות בפועל $g(n)$ שהיא העלות האמיתית מצומת ההתחלה לצומת החיפוש הנוכחי + העלות היוריסטית $h(n)$ שהיא העלות המשוערת של מעבר מצומת נוכחי לצומת המטרה. $f(n) = g(n) + h(n)$

הבעיה שמוצגת היא בעיית חיפוש עבור הרבה קריטריונים (Gudaitis, 1994) ויש לשלב את הקריטריונים לפונקציה אובייקטיבית (פונקציית אופטימיזציה- קיים בתחום הפונקציה איבר כך שערך הפונקציה עבורו גדול/קטן ,בהתאם לבעיית מקסימום/מינימום, מכל ערכי הפונקציה של שאר האיברים ששייכים לתחום הפונקציה) אחת ולכן נרצה לבצע אופטימיזציה של כל אחד מהקריטריונים, (B.S. Steward et al, 1991) בבעיה זו הקריטריונים הם גילוי של המכ"ם וצריכת דלק לכן פונקציית העלות היא:

$$C = W_{fc} \times C_{fc} + W_{rd} \times C_{rd}$$

כאשר : C- העלות הכוללת.

C_{fc} -עלות צריכת הדלק.

C_{rd} - עלות גילוי המכ"ם.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

W_{fc} - שקלול צריכת הדלק.

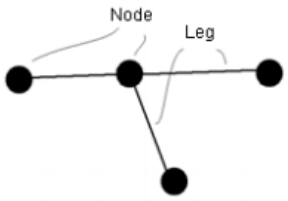
W_{rd} – שקלול גילוי המכ״ם.

C משמש כ $g(n)$ שהוא העלות בפועל.

העלות היוריסטית מחושבת ע״י המרחק הגאומטרי מהצומת הנוכחי לצומת המטרה. זה מונוטוני וקל לחישוב (Gudaitis, 1994).

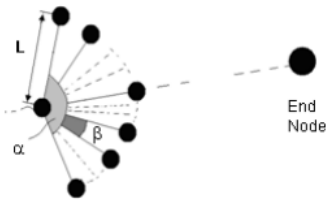
בזמן החיפוש נקבע את הצמתים העוקבים ע״י מדידת מרחק ואזימוט (זווית מהצפון לנקודת הנבדקת). עבור כל צומת ברשת, מודל הUAV מחשב מסלול מתאים לצומת זה בהתאם לתנאי השטח, הנחיית הפלטפורמה, מאפייני הניווט ומגבלות ביצועי הטיסה. מודל זה אחראי על חישוב של צריכת הדלק של המסלול המחושב ולאחר מכן נבדקות נקודות מסלול זה כנגד איומים. מודל האיום משתמש בתכונות השטח והאיום כדי לחשב את עלות האיום של המסלול לאותו הצומת.

צומת מייצג מיקום גאוגרפי בו המל״ט יתמרן או יתקדם באותו אזימוט, הצלע (Leg) מייצגת את הדרך בין שני צמתים. בסוף תהליך החיפוש, המסלול יהיה כסדרת צמתים.



גרף 3 : הדגמה של Leg , node

בתחילה נקבעת נקודת ההתחלה וכן נקבעים צמתים חדשים הניתנים לטיסה מצומת ההתחלה. לאחר מכן באמצעות אלגוריתם A* , נבחר אחד הצמתים החדשים מהצומת ההתחלתי, וממנה נקבעים צמתים חדשים. תהליך החיפוש ממשיך כך שמחפשים צמתים חדשים מהצמתים שנבחרו עד הגעה לצומת הסיום המייצג נקודת סיום. בעת בחירת הצומת הבא ניקח בחשבון את אורך הצלע, זווית α - הזווית היוצאת מהנקודה הצפונית ביותר ממני לנקודה הדרומית ביותר וכן את זווית β - הזווית שנוצרת בעקבות חלוקת החיפוש.



גרף 4 : הדגמת זוויות אלפא וביתא

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בנוסף, במהלך ביצוע החיפוש נלקחים בחשבון:

1. זווית הסיבוב המינימלית - מגבלת תמרון של המל״ט, לא ניתן להיות בזוויות חדות מהזווית המינימלית.
2. אורך צלע מינימלי – המרחק המינימלי הדרוש למל״ט לצורך ביצוע שני סיבובים רצופים.
3. רדיוס- רדיוס המעגל עליו מתבצע התמרון.

צמתי המשנה מחושבים באמצעות אסטרטגיות תמרון ואילוצי ביצועים כמו קצב סיבוב מרבי או קצב טיפוס מותר. מודל האיום מייצג מכ״מים במיקום גיאוגרפי שיכולים לגלות את המל״טים במרחק קטן יותר מרדיוס הזיהוי שלהם, רדיוס איתור תלוי במאפייני הרדאר ו- RCS (חתך הרדאר) של המל״ט. המודל שלנו מאפשר להגדיר RCS בהתאם לאזימוט המכ״ם היחסי. עלות צריכת הדלק מחושבת באמצעות טבלאות המראה של מנוע הרכב ואז בין שני תתי צמתים עוקבים נבדוק עפ״י הטבלה. כדי לקבוע את המשקלים המתאימים לקריטריונים ישנן שתי דרכים: האחת, צורכת יותר דלק אך בעלת פחות איומים והשנייה צורכת פחות דלק אך יותר חשופה לזיהוי רדאר ובכך חשופה ליותר איומים.

איסוף המידע:

1. אזור פעולה הינו מלבן אשר נקבע ע״י הנקודה הנוכחית והנקודה הבאה. על מנת לאסוף מידע על אזורי איום דוגמים פסים עפ״י היחס בין הRange (מרחק בין הנקודה ההתחלתית לנקודה האחרונה) ל-Width .

$$Width = (RangeRatio - 1) \times WidthCnst$$

כך ש: $widthCnst$ – קבוע שנקבע עפ״י הטווח המקסימאלי של המל״ט

$RangeRatio = \frac{MaxRange}{Range}$ (יחס הטווח)

2. ככל שיחס הטווח נעשה גדול יותר הטווח המקסימלי של המל״ט גדל וכך הוא יכול יותר לתמרן. מכיוון שאנו רוצים שהזווית α תגדל במהירות כאשר יחס הטווח קטן, ולעומת זאת תגדל באיטיות כאשר יחסי הטווח נמצא בערכים גדולים לכן הקשר בין יחס הטווח לזווית α הוא לוגריתמי.
3. שקלול עלות איתור הרדאר נקבע ביחס למצב האיומים באזור המבצע עפ״י קווי הדגימה (סעיף 1). הממוצע של אחוזי הגילוי של קווי הדגימה ע״י הרדאר מייצג את מצב האיום, אם יש יותר איומים, איתור הרדאר חשוב יותר, וצריך לשקול זאת ביתר רצינות ע״י הסוכן.

זווית חלוקת החיפוש β :

אם ישנם הבדלים גדולים בין עלויות איתור הרדאר של קווי הדגימה , צריך לבצע חיפוש באופן עדין יותר מכיוון שיש קשר בין עוצמת החיפוש לפתרון טוב יותר. אם שונות קווי הדגימה ע״י הרדאר גדולה אז רצוי יהיה לחפש מסלול חלופי. במקרה הפוך, הצעת אלטרנטיבה לא תעזור וזה ייקח זמן רב. לכן ניצור קשר בין זווית חלוקת החיפוש לבין שונות נראות של קווי הדגימה ע״י הרדאר. זווית חלוקת החיפוש נקבעת ע״י אלפא ומספר חלוקה (n) על מנת להשיג סימטריה ביצירת צלעות, מספר זה לא משתנה.

$\beta = \frac{\alpha}{2n}$ כך ש $2n$ הוא המספר הכולל של α (K. Tulum et al, 2009).

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בעיה זו נראית מאוד דומה לבעיה שלנו, אך ישנם מספר הבדלים משמעותיים:

1. בבעיה שלנו אנו לא מחפשים מסלול טיסה אלא נתיבי טיסה שמכסים מטרות שהוגדרו מראש. נתיבי טיסה אלו יהוו חלק ממסלול הטיסה של המל"ט שייקבע ע"י גורמים אחרים.
2. אנו מתמקדים בעיקר באילוצי סנסור וללא התחשבות בצריכת הדלק.
3. אנו לא מחפשים מסלול קצר ביותר אלא מסלול המספק את האילוצים של הסנסור תוך ניסיון ליצור כמה שפחות Legs.

2.1.3. בעיית N-Queens

בעיית N-מלכות היא הכללה של בעיית 8 מלכות. מטרת הבעיה היא לסדר N מלכות כך ששום מלכה לא תוכל לאיים על מלכה אחרת, בפועל- בכל טור, שורה, אלכסון (לשני כיוונים) חייבים להכיל מלכה אחת בלבד. בעיה זו מסווגת כ-NP-Hard, היא קשה לפחות כמו הבעיות הקשות ביותר ב-NP (Z. Wang et al, 2013).

Backtracking – אלגוריתם למציאת כל (או חלק) הפתרונות לבעיה חישובית כלשהי שבונה מערך פתרונות ומשליך כל איבר מהמערך שהוא פתרון חלקי ולא חוקי. הרעיון של האלגוריתם זה ניסוי וטעיה, ולכן ככל שיש יותר שגיאות זה פחות יעיל. צמצום מספר הניסיונות מקטין את משך הריצה של האלגוריתם.

שיפור היעילות יעשה כך שנסיר את התאים הלא רלוונטיים לפני שהאלגוריתם מנסה ליצור פתרון באמצעות אותם תאים.

הרעיון הכללי של האלגוריתם לפני שיפור, עבור בעיית N מלכות:

1. הצבת המלכה הראשונה וסימון התאים האחרים בנתיב ההתקפה של מלכה זו.
(נתיב התקפה – מי שבאותה שורה, עמודה ואלכסון של מלכה זו)
2. בדיקת השורה הבאה עד שמגיעים לתא לא מאוים ומציבים שמה את המלכה הבאה.

עבור לוחות בגודל 12 ומעלה מספר הניסויים גדל באופן דרמטי. בעיה זו נפתרת ע"י מחיקת התאים המאוימים לפני הצבת המלכה.

נמספר את התאים של המטריצה החל מהשורה הראשונה מימין עד השורה האחרונה משמאל. נגדיר איבר במערך האפשרויות לפתרון כך: מיקום במטריצה (עפ"י המספור), שורה, מערך של תאים המאוימים ע"י מלכה זו. ברגע שהאלגוריתם יבחר את המלכה הראשונה נשמור את האופציות החלופיות ונמחק את מי שמאוים (מי שבמערך האיומים של אותה מלכה), ע"י כך נקטין את מספר הניסויים. לאחר מכן, האלגוריתם ימשיך להציב את המלכות בתאים השונים הזמינים. לבסוף, נבדוק האם התוצאה תואמת את אורך הלוח, אם לא נחזור צעד אחד אחורה ונשחזר את התאים שנמחקו ונציב את המלכה במקום הזמין הבא כל עוד זה באותה שורה. אחרת, נלך עוד אחורה (יכול להיות מקרים בהם נלך כמה צעדים אחורה כדי למצוא פתרון מוצלח) (Serkan Güldal et al, 2016).

עד כה סקרנו מס' תחומים המיישמים את בעיית *CSP* והצגנו פתרונות.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

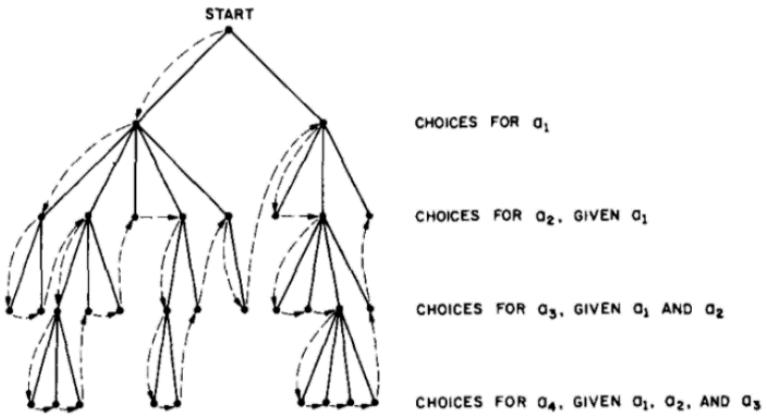
2.2. Backtracking

כאשר מדובר על בעיות *NP-Hard* אשר לא ידועה להם שיטת פתרון יעילה ייתכן שיהיה צורך לבדק את כל האפשרויות בכדי לקבוע את הפתרון. בחינה ממצה של כל הפתרונות נקראת *'Exhaustive search'* או *'Direct search'*. טכניקה מפורסמת לארגון החיפוש היא *"Backtrack"*, מנסים שוב ושוב להרחיב פתרון חלקי. בכל שלב בחיפוש, אם הפתרון הנוכחי החלקי לא מספק את כל האילוצים אז חוזרים שלב אחורה (*"backtrack"*) ומנסים שוב.

נניח שהפתרון לבעיה הוא ווקטור $(a_1, a_2, \dots)$ באורך לא קבוע. ווקטור זה מספק את כל האילוצים הקיימים על משתני הבעיה. כל a_i הוא איבר בקבוצה סופית של איברים A_i. הפתרונות האפשריים הם $A_1 \times A_2 \times \dots \times A_i \; for \; i = 1,2, \dots$. בהתחלה מתחילים עם ווקטור ריק בתור הפתרון החלקי שלנו, עפ"י האילוצים הקיימים אנחנו יכולים לדעת מהם האיברים מתוך A_1 שמועמדים להיות a_1. נקרא לתת קבוצה של האיברים המועמדים לפתרון S_1. נבחר את האיבר הקטן ביותר מ S_1 בתור a_1 וכעת יש לנו פתרון חלקי (a_1). באופן כללי, ע"י האילוצים אנו יודעים מהי הקבוצה S_k מתוך הקבוצה A_k כלומר, מיהם המועמדים להרחבת הפתרון החלקי $(a_1, a_2, \dots, a_{k-1})$ לפתרון $(a_1, a_2, \dots, a_{k-1}, a_k)$.

אם $S_k = \emptyset$ אזי חוזרים צעד אחד אחורה ובוחרים איבר אחר בתור a_{k-1}. אם אין אפשרויות נוספות עבור a_{k-1} אז חוזרים עוד צעד אחורה ומנסים אפשרות נוספת לa_{k-2}, וכן הלאה.

נוח להציג את התהליך הנ"ל בעץ באופן הבא: שורש העץ מייצג את הווקטור הריק. הבנים שלו הם האפשרויות של a_1 ובאופן כללי הצמתים ברמה הk-ית הם האפשרויות של a_k כאשר הבחירות עבור $a_1, a_2, \dots, a_{k-1}$ כבר נעשו.



גרף 5: הדגמת תהליך Backtracking

טכניקות תכנות:

הרבה יישומים של *Backtracking* לא דורשים הרבה מקום בזיכרון ולכן זה לא סביר להגדיל את דרישות הזיכרון בשביל להקטין את זמן הריצה. רצוי להשתמש ב*macros* (כלל או תבנית

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

שמציינים כיצד יש למפות קלט מסוים לפלט חלופי עפ״י פרוצדורה מוגדרת) על מנת שחלק מהעבודה יתבצע פעם אחת בזמן אסמבלי ולא הרבה פעמים בזמן ריצה.

לדוגמא אם ידוע שגודל ווקטור הפתרון הוא n נכתוב את ה*macro* הבא שנקרא $CODE_i$:

```
      compute S_i
L_i:  if S_i = ∅ then goto L_{i-1}
      a_i ← minimum element in S_i
      S_i ← S_i − {a_i}
```

ונכתוב את התוכנית הראשית :

```
      CODE_1
      CODE_2
        ⋮
      CODE_n
      record (a_1 , . . . , a_n ) as a solution
      goto L_n
L_0:  stop—all solutions have been found
```

מלבד זאת ש*macros* מספקים הגדלת מהירות התוכנית (כלומר, עיבוד מהיר יותר של הצמתים בעץ), הם מקטינים את מספר הצמתים בעץ וממילא מקטינים את זמן הריצה של התוכנית.

ישנן 4 מתודות להקטנת מספר הצמתים :

1. *Preclusion*. טכניקה כללית בה *backtrack* מתבצע רק כאשר הפתרון החלקי הנוכחי לא מספק שום פתרון.
2. *Branch merging*. כאשר זה אפשרי לא בודקים ענפים של העץ אשר זהים במבנה שלהם לענפים אחרים שכבר נבדקו.
3. *Search rearrangement*. באופן כללי, צמתים בעלי רמה נמוכה נבדקים מוקדם בחיפוש ואילו צמתים בעלי רמה גבוהה נבדקים מאוחר יותר. כיון שההתקדמות על העץ נעשית לעיתים קרובות, רצוי שנבדוק מעט צמתים בכל שלב. לכן כאשר עומדים מול מספר אופציות להרחבת הפתרון החלקי נבחר את האופציה שמציעה הכי פחות אלטרנטיבות .
4. *Branch and bound*. משתמשים בטכניקה זו כאשר מחפשים פתרון בעל עלות מינימלית. כאשר נמצא פתרון מוחקים את כל הפתרונות החלקיים שעלותם גבוהה מעלות הפתרון שנמצא(בהנחה שהעלות מצטברת). שיטה זו יעילה כאשר מוצאים פתרון טוב בשבלים המוקדמים של החיפוש לכן הכי טוב לסדר את החיפוש כך שפתרונות טובים יימצאו בהתחלה.

עבור שתי הטכניקות הראשונות אפשר להביא כדוגמא את בעיית N מלכות המוזכרת לעיל. מכיוון שבכל עמודה יכולה להיות אך ורק מלכה אחת פתרון של הבעיה הוא ווקטור $(a_1, a_2, \dots, a_n)$ כך ש a_i מייצג את מספר השורה של המלכה בעמודה הi-ית. נדגים את טכניקת *Preclusion* : אנו לא מתחשבים בכל השיבוצים של המלכות אלא רק באלה שיש בכל עמודה רק מלכה אחת. כיוון שכל שאר השיבוצים לא חוקיים הם נמחקים אוטומטית מהאפשרויות.

שתי פתרונות נקראים שקולים אם אפשר לקבל פתרון אחד מתוך השני ע״י שיקוף או ע״י סדרת סיבובים של 90°. אם נמצא את כל הפתרונות הלא שקולים נוכל למצוא בקלות את שאר הפתרונות.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

כאשר יש מלכה באחת מפינות הלוח לא ניתן שיהיו מלכות בשאר הפינות, לפי אילוצי הבעיה. לכן מכל פתרון בו יש מלכה במשבצת *(1,1)* נוכל לקבל את שאר הפתרונות בהם המשבצת *(1,1)* ריקה ע"י סיבובים של $90°$ או/ו שיקוף. לכן כאשר $a_1 \geq 2$ נקבל את כל הפתרונות הלא שקולים. יתר על כן, אם $a_1 > \left\lceil \frac{n}{2} \right\rceil$ ניתן להשיג על ידי שיקוף פתרון נוסף בו $a_1 \leq \left\lceil \frac{n}{2} \right\rceil$. לכן נתן להגביל את מציאת כל הפתרונות הלא שקולים כאשר $2 \leq a_1 \leq \left\lceil \frac{n}{2} \right\rceil$. כלומר, ניתן להתעלם מפתרונות שהם איזומורפיים לפתרונות אחרים שכבר התקבלו שזה שימוש בטכניקת *Branch merging*.

הוספת הבדיקות לעיל באלגוריתם *backtracking* הכללי "יקרה" כיוון שנצטרך לבדוק אותם כל פעם בלולאה הפנימית למרות שנדיר שהן יצליחו. אולם, בגישת ה*macro* המתוארת לעיל הפקודות של בדיקות אלו לא ייכללו ב$CODE$ עבור $i > 2$ במקום שאינן נחוצות. עיקרון של שיטה זו הוא לכתוב תכניות נפרדות(אם כי הן דומות) עבור כל רמה בעץ. כאשר השתמשו בטכניקה זו המקרה של 14×14 רץ תוך 5 דקות והמקרה של 15×15 רץ תוך 25 דקות, פי 5 מהר יותר מאשר ללא שימוש בטכניקות.

נראה דוגמא נוספת בה אורך הפתרון לא ידוע. הבעיה היא מציאת *difference-preserving codes* אופטימלי. *difference-preserving code* של גבול t (*DP-t*) הוא רצף של מילות קוד בינאריות $\{a_1, \ldots, a_k\}$, כל אחת מהן באורך n אשר מקיימות את האילוצים הבאים:

i. $\text{for } |i-j| \leq t,\ H(a_i, a_j) = |i-j|;$

ii. $for\ |i-j| > t,\ H(a_i, a_j) > t;$

כאשר $H(x,y)$ היא *Hamming distance* בין מילות הקוד x ו-y. *Hamming distance* הוא מספר הביטים השונים בין שתי מילות קוד בינאריות. באופן פרטי, אנו רוצים למצוא את הקוד המקסימלי של *DP-1* עם מילים באורך n. באלגוריתם הכללי של *backtracking* ננסה להרחיב את הקוד הקיים על ידי הוספת מילות קוד שעדיין לא מופיעות בו. *Backtracking* מתבצע כאשר לא ניתן להרחיב יותר את הקוד הנוכחי. כאשר מרחיבים את הקוד הכרחי לנסות רק את n המילים סמוכות למילה האחרונה שנוספה (n המילים ששונות מהמילה האחרונה בביט אחד) אחרת, המילה תפר את האילוץ הראשון.

נשתמש ב*macro* בשביל לייצר פונקציה עבור כל אחת מ2^n מילות הקוד האפשריות באורך n. למשל, אם W היא מילת קוד ו $W_1, W_2, \ldots, W_n$ הן n המילים הסמוכות ה*macro* ייצר את הפונקציה הבאה:

```
add W to the code
 if this code is longer than the previously generated longest code
   then record it
if W1 can be added then call W1
if W2 can be added then call W2
          ⋮
 if Wn can be added then call Wn
remove W from the code
return
```


שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

אפשר להשתמש כאן בטכניקת Branch and bound על מנת להקטין את הגודל של עץ החיפוש. בכל צומת אנו יודעים כמה מילות קוד כבר לא חוקיות כיוון שהם מדי קרובות למילים קודמות. זה נותן גבול (bound) עליון למספר מילות הקוד שיכולות להתווסף. אם נוסיף את הגבול העליון הזה לאורך הנוכחי ועדיין לא נקבל קוד ארוך יותר מהקודים שנמצאו עד כה אנו יכולים מיד לבצע backtracking כיוון שלא נקבל פתרון אופטימלי על ידי הרחבת קוד זה. מציאת פתרון אופטימלי עבור 6 bit DP-1 בשימוש בטכניקת bracnch and bound הקטינה את זמן הריצה מ25 שניות ל3 שניות.

2.3. סיכום

בפרק זה סקרנו את האלגוריתמים הקיימים בתחום אליו הפרויקט שלנו שייך ולאחר חשיבה הסקנו שהאלגוריתם המתאים ביותר לכלל הבעיה הוא Backtracking. זאת מכיוון שהמערכת מכילה המון אילוצים ותלויות ביניהם ובנוסף היא כוללת מספר רב של ווריאנטים הגורמים לסיבוך החישוב לכן מימוש ע״י Backtracking יהיה פשוט יותר.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

3. תכנון

בפרק זה נציג את דרישות הלקוח והעיצוב (design) של מערכת AutoMPW.

3.1 דרישות לקוח

המערכת הינה מערכת תכנון משימה אוטומטי העונה על הדרישות הבאות:

המערכת מכילה צד לקוח וצד שרת המתקשרים ביניהם ע"י פרוטוקול http .

דרישות פונקציונליות:

1. בניית מנוע לחישוב תכנית משימה על בסיס קלט נתונים שהוא אוסף מטרות שהמשתמש מגדיר.
2. מימוש אלגוריתם לחישוב כיסוי אזור ע"י מטרות מסוג מסוים עם אחוז חפיפה משתנה בין המטרות.
3. הגדרה דינאמית של אילוצים וטעינת טווח הערכים של המשתנים מקובץ קונפיגורציה.
4. קליטת מטרות מסוגים שונים ושטח לכיסוי מהמשתמש.
5. אפשרות לבחור אחוז חפיפה של המטרות וסוג מטרות שיכסו את השטח (פוליגון).
6. כאשר נקלטת מטרה חדשה או כאשר מטרה קיימת נבחרת ייפתח חלון בו תינתן האפשרות לערוך את מאפייני המטרה השונים.
7. כאשר מטרה נבחרת וכבר שויך לה קטע על נתיב טיסה ממנו מצלמים אותה יצויר על המסך קטע שיחבר בין המטרה לקטע הצילום.
8. הצגת רשימת המשימות השמורות בבסיס הנתונים ובצורה היררכית את המטרות, נתיבי הטיסה והאזורים שכל משימה מכילה.
9. אפשרות לסנן תוצאות של משימות מהרשימה לפי תאריך של יצירת המשימה.
10. המערכת תכלול רכיב מפה המתפרש על כל המסך.
11. דקירת נתיב טיסה כלשהו על המפה.
12. סרגל למדידת מרחק בין נקודות על המפה.
13. שמירת משימות בקבצי json.
14. במידה ויש שגיאת שרת במערכת תופיע הודעת שגיאה למשתמש.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

דרישות לא פונקציונליות:

1. המערכת תהייה אמינה, כלומר הקטנת הסיכויים של כניסתה למצבים קריטיים.
2. המנוע החישובי והשרת יתמכו במספר פלטפורמות.
3. עלויות מינימליות ע"י שימוש ברכיבים הקיימים בחברה.

3.2. מטרת המערכת

המטרה העיקרית של המערכת היא למצוא פתרון אוטומטי הכולל מערך נתיבי טיסה עבור סט של מטרות ואילוצים. בנוסף, המשימה שהמערכת תחזיר כפלט תכלול מס' Legs נמוך ממס' ה-Targets שהתקבלו מהמשתמש. זאת אומרת, שלא יוחזר הפתרון הטריוויאלי בו לכל Target משויך Leg נפרד.

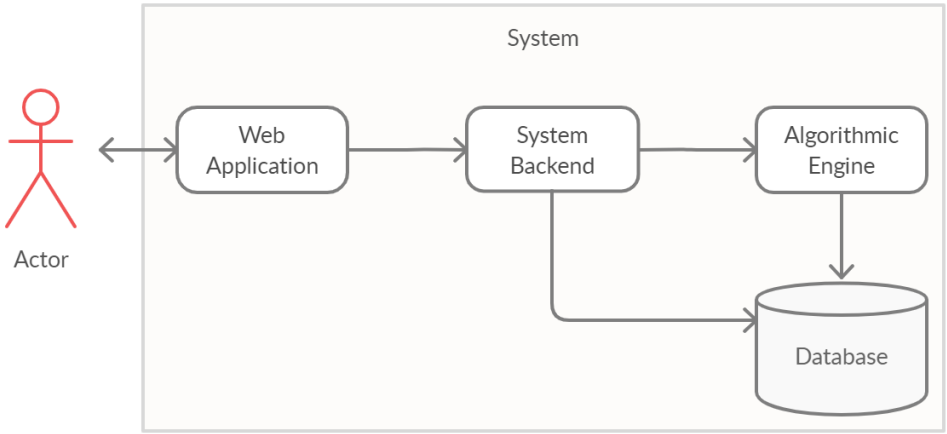
המטרה המשנית של הפרויקט היא יצירת תצוגות Web עבור המשתמש.

3.3. Main System Flows

1. **קליטת נתונים**- המשתמש בונה מטרות או/ו אזורים לצילום.
2. **כיסוי שטח-** במידה והמשתמש דקר אזור ולחץ על כפתור "coverage" אז מופעל אלגוריתם המחשב כיסוי של האזור הנבחר ע"י סוג מסוים של מטרות שהמשתמש בחר וכן אחוז חפיפה בין המטרות שיכסו את האזור. בשביל לקבל תמונה של כל השטח בצורה מדויקת זקוקים לכך שתהיה חפיפה בין המטרות המכסות את האזור.
3. **חישוב משימה**- כאשר המשתמש לוחץ על כפתור "solution" מופעל אלגוריתם לחישוב משימה המשייך נתיבי טיסה עבור המטרות תוך סיפוק האילוצים במערכת. במהלך הפתרון כל השינויים מתבצעים אך ורק על נתיבי הטיסה ואילו המטרות נשארות כפי שהמשתמש מגדיר.
4. **הצגה**- הפתרון יופיע בממשק משתמש מבוסס WEB.
5. **שמירה**- השמירה מתבצעת לקובץ Json וכל המשימות שנשמרו עד כה מופיעות ברשימה בצד המסך, וניתן להציגן שוב.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

3.4. מבט על של המערכת



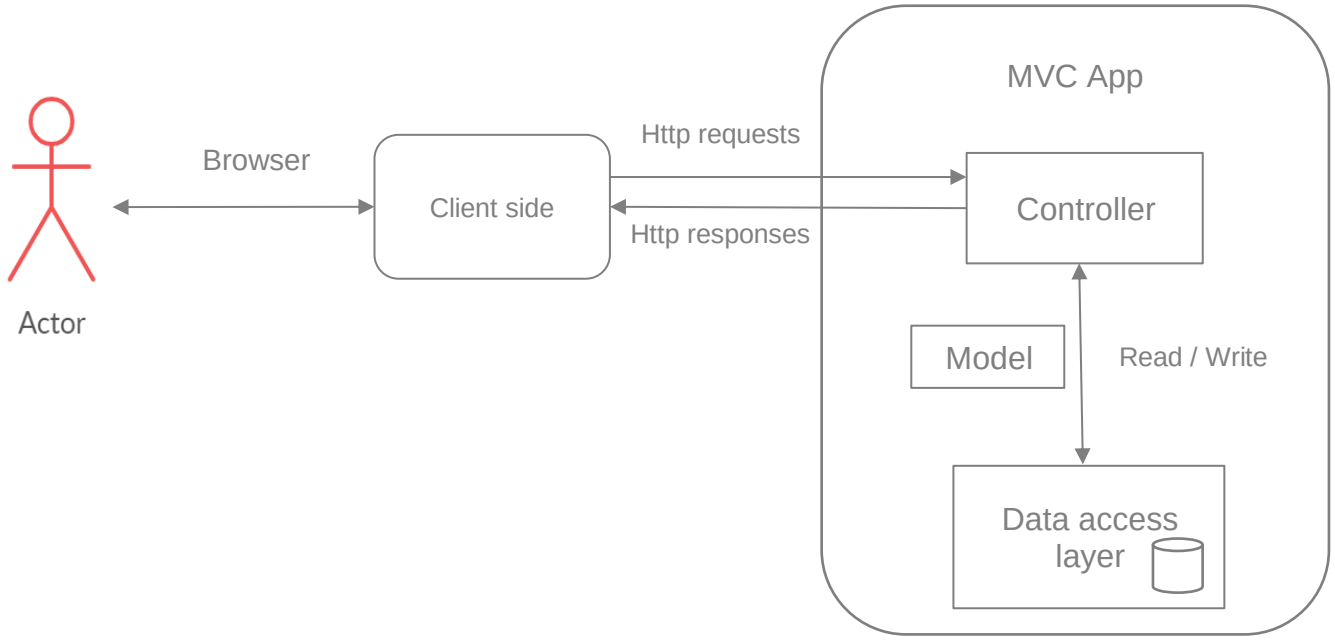
גרף 6 : מבנה המערכת

Algorithmic Engine הוא המנוע החישובי המכיל את כל החישובים בניהם אלגוריתמים לחישוב משימה, לכיסוי שטח ולתיקון אילוצים שנכשלו המשתמשים בחישובים מתמטיים וגאומטריים.
הEngine מכיר אך ורק את הישויות במערכת ללא תלות בצד שרת וצד הלקוח.
System Backend מכיר ופונה אל הEngine ותוצאות החישוב חוזרות בhttp response לצד לקוח.

שיקולים טכנולוגיים בפיתוח צד שרת

עבור צד השרת התלבטנו בין פיתוח בNode.js לבין פיתוח בAsp.net core.
הפיתוח ב Asp.net core הוא בC# והמנוע הוא ספרייה של C# ולכן הגישה למנוע מצד השרת קלה יותר כאשר הוא בAsp.net core מאשר הגישה מNode.js. בנוסף, השימוש בAsp.net core פשוט יותר ומכיוון שהחלק העיקרי של הפרויקט מתמקד באלגוריתם לחישוב המשימה שדרש את רוב זמן הפיתוח לכן בחרנו עבור צד השרת בטכנולוגיה פשוטה יותר ללמידה ולשימוש.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il



גרף 7: ארכיטקטורת MVC

ארכיטקטורת MVC בנויה משלושה רכיבים:

1. Model- ייצוג וניהול של הנתונים עליהם האפליקציה פועלת
2. View- ממשק משתמש, הview מציג נתונים מהmodel למשתמש וגם מאפשרת לו לשנות את הנתונים.
3. Controller - מטפל בבקשות URL נכנסות. הcontroller מכיל public methods שנקראים Action methods. הcontroller ע"י action method מטפל בבקשות הנכנסות מהדפדפן, שולף נתונים נחוצים מהmodel ומחזיר תגובות מתאימות.

צד השרת בנוי עפ"י ארכיטקטורת MVC המפורטת לעיל. אצלנו במערכת בנינו את הmodel כרכיב נפרד מצד השרת כיוון שגם המנוע החישובי פונה אליו. מבחינת הdesign מתאים יותר שהmodel יהיה בנפרד כדי שנקבל צימוד נמוך ולכידות גבוהה (loose coupling high cohesion).

צד הלקוח הוא אפליקציית Web המשתמשת בספריות Bootstrap ו Material Design על מנת ליצור ממשק משתמש נוח ומודרני. בצד הלקוח יש שימוש ברכיב מפה MapManager כמעטפת ל Cesium הפותח באלתא מערכות בע"מ. הGUI של מערכת MPW מבוסס WPF וחלק ממטרת שדרוג המערכת היה להעביר אותה לאפליקציית Web. החלטה זו נבעה משיקולי חדשנות ורלוונטיות לקיים בשוק.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

3.5. תכנון האלגוריתם לחישוב משימה

במהלך שלב התכנון של המערכת ערכנו חקר ספרות נרחב אודות אלגוריתמים קיימים בשוק הפותרים את בעיית סיפוק אילוצים.

בחרנו לפעול בשיטת backtracking כיוון שהמערכת מכילה המון אילוצים ותלויות ביניהם ובנוסף היא כוללת מספר רב של וריאנטים הגורמים לסיבוך החישוב.

3.6. סיכום

בפרק זה הצגנו את תכנון המערכת עפ״י דרישות הלקוח והגדרנו את העיצוב של המערכת בצורה נכונה כדי שנגיע לתוצאות טובות במימוש.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4. מימוש

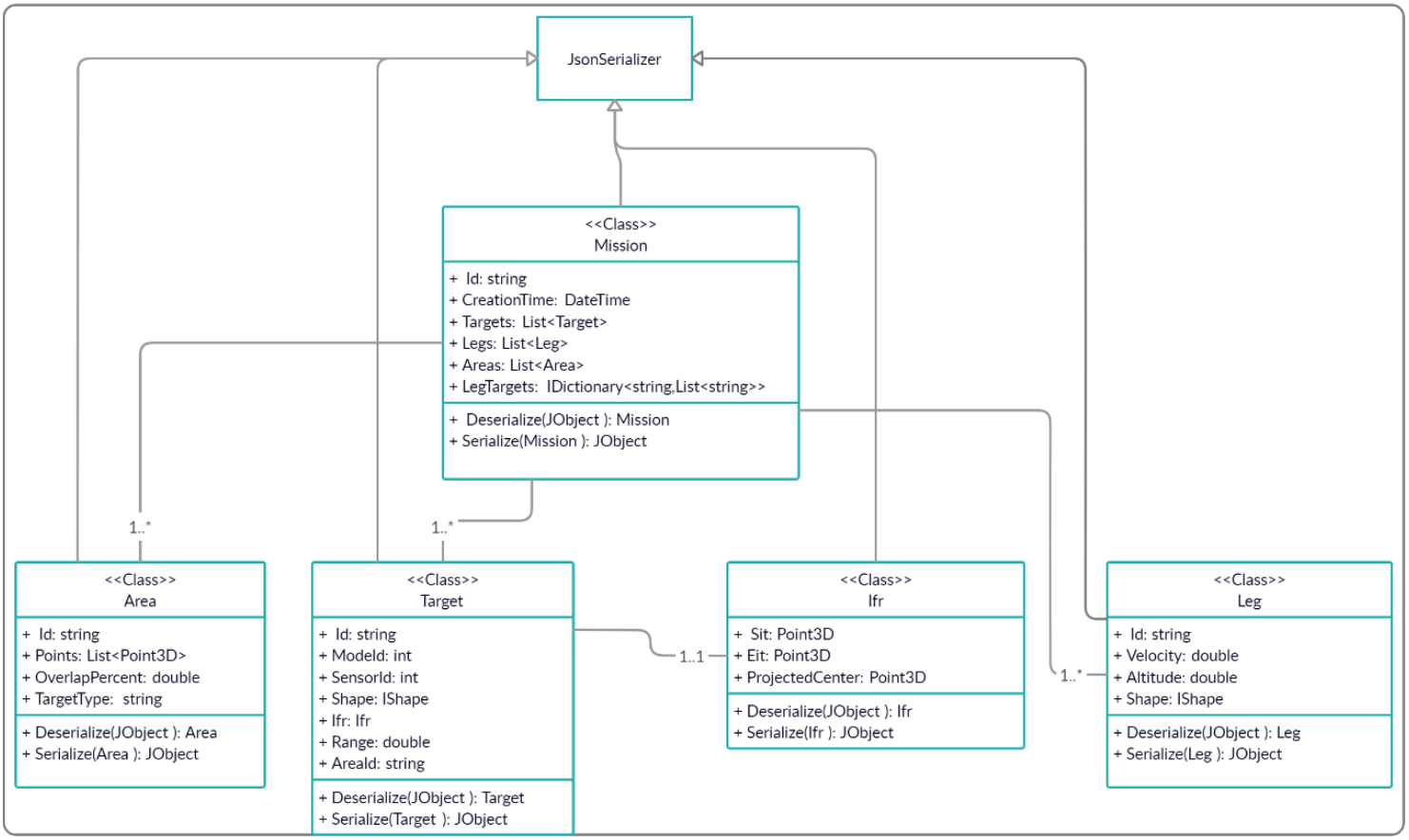
בפרק זה נציג את מימוש המערכת בפועל, את הטכנולוגיות שהשתמשנו בהן לפיתוח המערכת וכן את הבדיקות שביצענו.

4.1 תרשימי UML

4.1.1 Data Model UML

החבילה DataModel מכילה מס' ספריות.

התרשים הבא מתאר את הישויות המערכת (AutoMPW.Entities).



גרף 8: Entities UML

נציג בקצרה כל אחת מהמחלקות:

Leg- מחלקה המגדירה נתיב טיסה. לכל נתיב טיסה יש: מס' סידורי, מהירות הטיסה, גובה הטיסה על הנתיב והגדרתו של נתיב הטיסה (קו ישר בעל רשימת נקודות).

Ifr – מחלקה המייצגת את קטע צילום המטרה בפועל על נתיב הטיסה. לכל Ifr יש 3 נקודות במרחב המייצגות את תחילת (Sit), אמצע (projected Center) וסוף (Eit) קטע הצילום.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

Area- המחלקה מייצגת פוליגון כלשהו לצילום. לכל אזור יש: מס' סידורי, רשימת נקודות המגדירות את הפוליגון, אחוז חפיפה בין המטרות שיכסו את הפוליגון, סוג המטרות שיכסו את השטח.

Target – מחלקה המייצגת מטרה כלשהי לצילום. לכל מטרה יש מס' סידורי, מס' סידורי של הסנסור, מס' סידורי של Mode שבו הסנסור נמצא, צורת המטרה (מקבילית המוגדרת ע"י נקודת מרכז, אורך, רוחב, Azimuth - זווית מהצפון לנקודת מרכז המטרה), Ifr, מרחק ממרכז Ifr ו- Id של אזור אליו משויכת המטרה (אם מטרה זו לא חלק מכיסוי אזור ערכו Null). גודל המטרה נקבע ע"י הSensorId, וModeId עפ"י שימוש בקובץ קונפיגורציה. בנוסף מטרה כוללת:

Mission- מחלקה המייצגת משימה. לכל משימה יש מס' סידורי, זמן יצירתה, רשימת נתיבי טיסה, רשימת מטרות, רשימת אזורים, ומבנה נתונים המחזיק לכל Id של נתיב טיסה רשימת Id1 של מטרות המשויכות לנתיב טיסה זה.

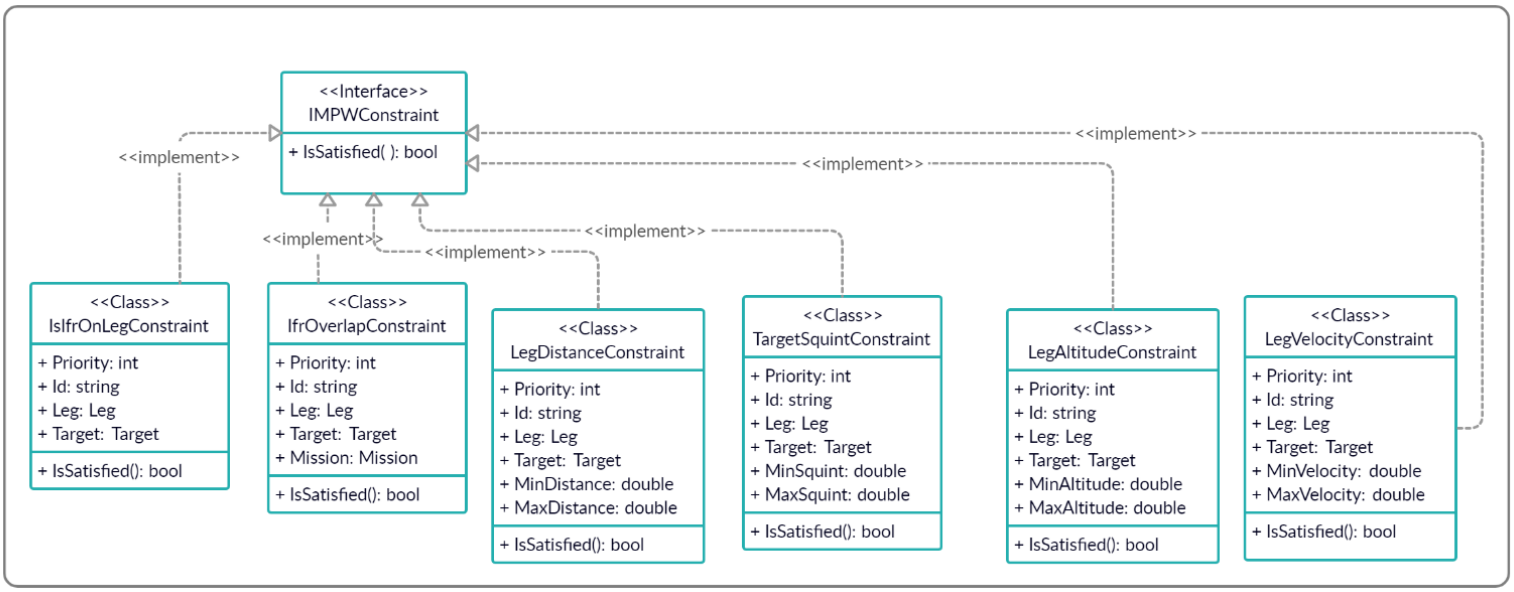
כל המחלקות הנ"ל יורשות מJsonSerializer וממשות את המתודות:

- Deserialize- המרה של משתנים מפורמט של קובץ המאחסן נתונים לאובייקט.
- Serialize- המרה של אובייקט לפורמט של קובץ המאחסן נתונים.

אצלנו הנתונים נשמרים בפורמט Json.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

תרשים זה כולל את אילוצי המערכת (AutoMPW.Constraints).



הספרייה מכילה ממשק המייצג אילוץ, כל אילוץ מממש את המתודה הבוליאנית IsSatisfied המחזירה true אם האילוץ מסופק אחרת, מחזירה false.

בנוסף, כל אילוץ מחזיק את הזוג מטרה-נתיב טיסה שעבורם הוא צריך להיות מסופק.

טווח הערכים שאילוץ יכול לקבל נקבע ע"י קובץ קונפיגורציה כאשר טווח הערכים משתנה בין סוגים שונים של מטרות.

כל מחלקה מחזיקה במשתנה המייצג את עדיפות האילוץ בבדיקה האם האילוצים מסופקים או לא.

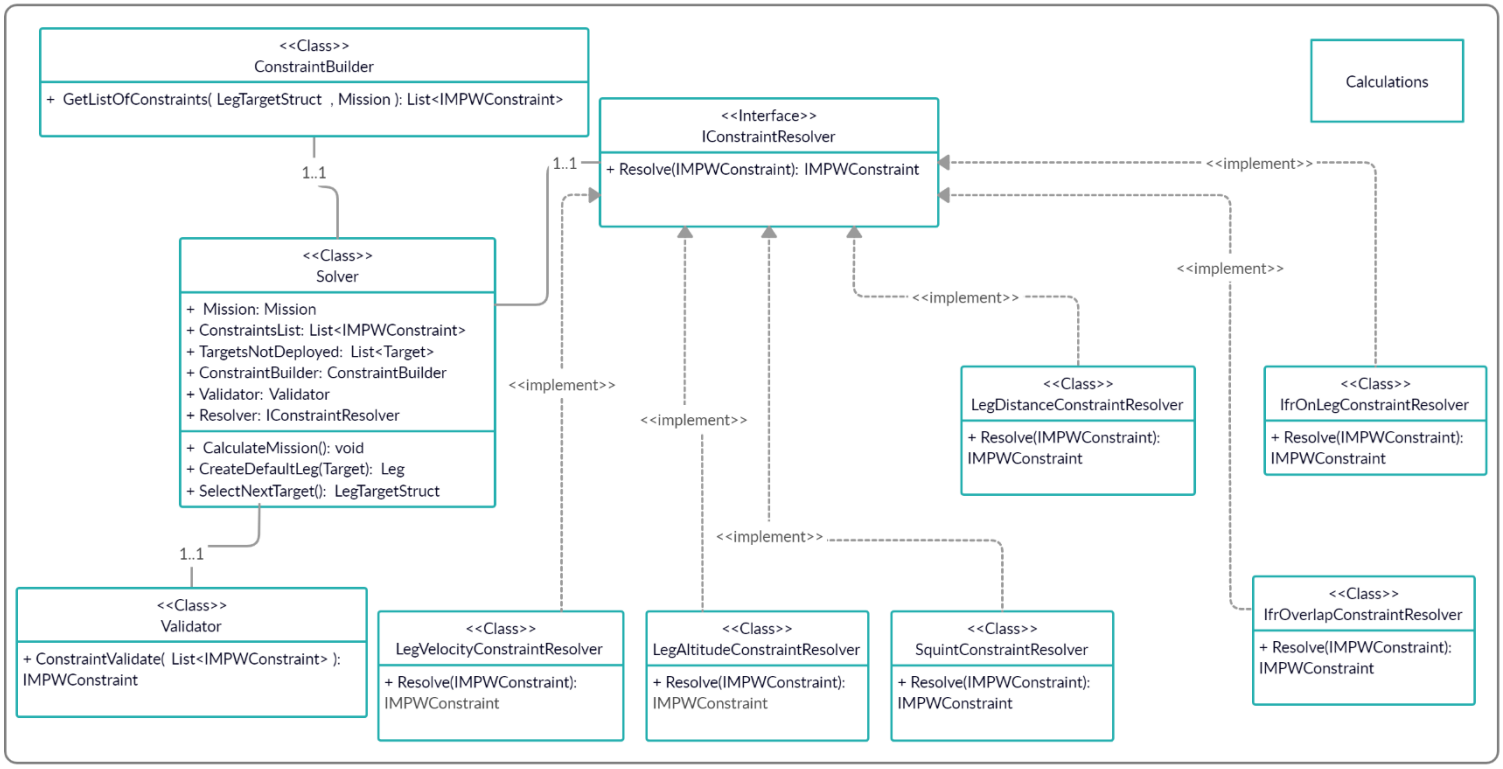
נסביר בקצרה את המשמעות של כל אילוץ:

1. IsIfrOnLeg- אילוץ זה מסופק כאשר יש למטרה קטע צילום על נתיב טיסה קיים.
2. IfrOverlap- אילוץ זה נכשל (לא מסופק) כאשר קטע הצילום עבור המטרה חופף לקטע צילום של מטרה אחרת כלשהי במשימה, עבור בדיקה זו האילוץ מכיל אובייקט של המשימה.
3. LegDistance- אילוץ זה מסופק כאשר המרחק בין מרכז המטרה למרכז קטע הצילום נמצא בטווח של המרחק האפשרי עבור מטרה מסוג זה.
4. TargetSquint- אילוץ זה מסופק כאשר זווית הצילום נמצאת בטווח האפשרי עבור מטרה מסוג זה.
5. LegAltitude- אילוץ זה מסופק כאשר גובה הטיסה עבור נתיב הטיסה שהמטרה משויכת אליו נמצא בטווח האפשרי עבור מטרה מסוג זה.
6. LegVelocity- אילוץ זה מסופק כאשר מהירות נתיב הטיסה שהמטרה משויכת אליו נמצאת בטווח האפשרי עבור מטרה מסוג זה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.1.2. Engine UML

התרשים הבא מתאר את המנוע החישובי שהוא החלק העיקרי במערכת.



Solver- המחלקה העיקרית במנוע החישובי הכוללת את האלגוריתם המרכזי המחשב משימת צילום עבור מטרות נתונות. מתוך האלגוריתם הראשי יש קריאה לפונקציות שונות ברכיבים השונים.

Constraint Builder- מחלקה זו כוללת מתודה המקבלת כקלט זוג - מטרה ונתיב טיסה. המתודה מחזירה עבורם רשימת אילוצים שצריכים להיות מסופקים במערכת.

Validator- מחלקה זו כוללת מתודה המקבלת כקלט רשימת אילוצים ומחזירה את האילוץ הראשון שנכשל אם לא קיים אחד כזה, מחזירה Null.

IConstraintResolver- ממשק זה מגדיר פתרון לאילוץ שלא סופק. הוא מכיל מתודה המקבלת אילוץ שלא סופק ומשנה ערכים בהתאם כך שהאילוץ יסופק. לכל אחד מהאילוצים במערכת יש דרך לפתרון שונה ולכן מחלקות שונות עבור אילוצים שונים יממשו את ממשק זה באופן שונה.

Calculations- מחלקה המכילה את כל החישובים הגאומטריים שחוזרים על עצמם במנוע החישובי. כל אחד מהרכיבים יכול לפנות לרכיב זה ולהשתמש בו במידת הצורך, מטרת יצירת רכיב זה הייתה למנוע חזרה בקוד .

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.2. אלגוריתם לחישוב כיסוי שטח

פונקציונליות שלא קיימת במערכת MPW ונוספה במערכת שפיתחנו היא אפשרות לדקירת אזור לצילום ע״י המשתמש. אזור הוא פוליגון כלשהו. כיוון שהסנסור יודע לצלם סוגים ספציפיים של מטרות צריך לכסות את האזור ע״י מטרות ורק אחר כך להפעיל את האלגוריתם שמשייך עבורן נתיבי טיסה. לכן כאשר המשתמש בונה אזור על המפה מופיע לו חלון בו הוא מזין את סוג המטרה על ידה האזור יכוסה וכן את אחוז החפיפה בין המטרות שיכסו את האזור.

המטרה בחפיפה בין המטרות המכסות אזור היא לשפר את היכולת לתפירת התמונות לקבלת תמונה מדויקת יותר של האזור כולו. לכן, נדרש מהמשתמש להזין גם את אחוז החפיפה.

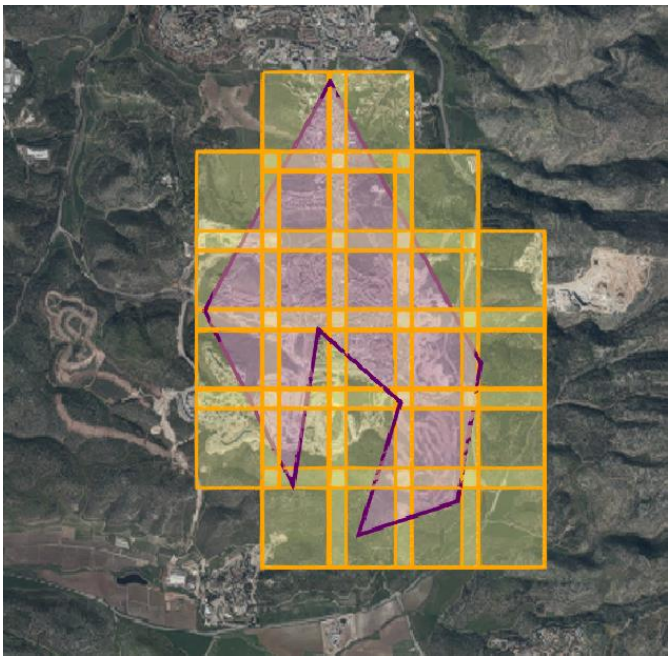
במהלך חיפושינו אחר אלגוריתם לחישוב כיסוי שטח לא מצאנו אלגוריתם שעונה בדיוק על הצורך שלנו הכולל התייחסות לאחוז חפיפה בין המטרות לכן פיתחנו אלגוריתם.

אופן המימוש:

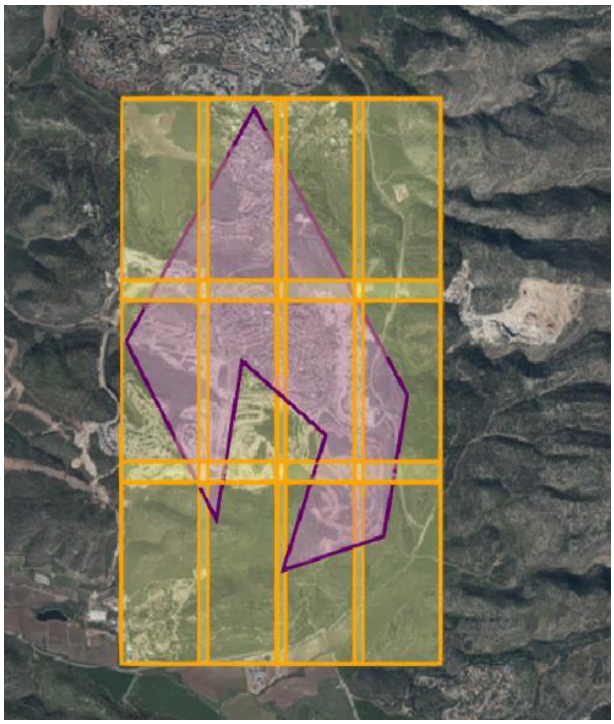
1. תיחום האזור ע״י מלבן כאשר קודקודי המלבן נקבעים לפי הקודקודים הקיצוניים של הפוליגון.
2. חלוקת המלבן למטרות עפ״י הסוג שנבחר ע״י המשתמש. בכדי לקבל חפיפה בין המטרות הפחתנו מהאורך וכן מהרוחב של המטרה (אורך ורוחב המטרה משתנה לפי סוגה) את אחוז החפיפה (להלן ״המטרה הקטנה״) וחילקנו את אורך המלבן באורך המטרה שקיבלנו וכן את רוחב המלבן ברוחב המטרה בשביל לדעת את מס׳ המטרות שהמלבן מכיל.
3. לפי מס׳ המטרות שקיבלנו חישבנו את נקודות האמצע של המטרות הקטנות.
4. בדיקה מיהן המטרות שיש להן נקודת חיתוך עם הפוליגון.
5. חזרה לגודל המקורי של המטרות וכך קיבלנו חפיפה בניהן.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בתמונות הבאות נציג אזור (סגול) ומטרות שמכסות את האזור (כתום) ובנוסף נראה שסוגים שונים של מטרות וכן אחוז חפיפה שונה בין מטרות סמוכות משפיעים על אופן הכיסוי.
בתמונה 1 ניתן לראות שאחוז החפיפה גדול יותר מאשר בתמונה 2, בחירה זו נעשית ע"י המשתמש.



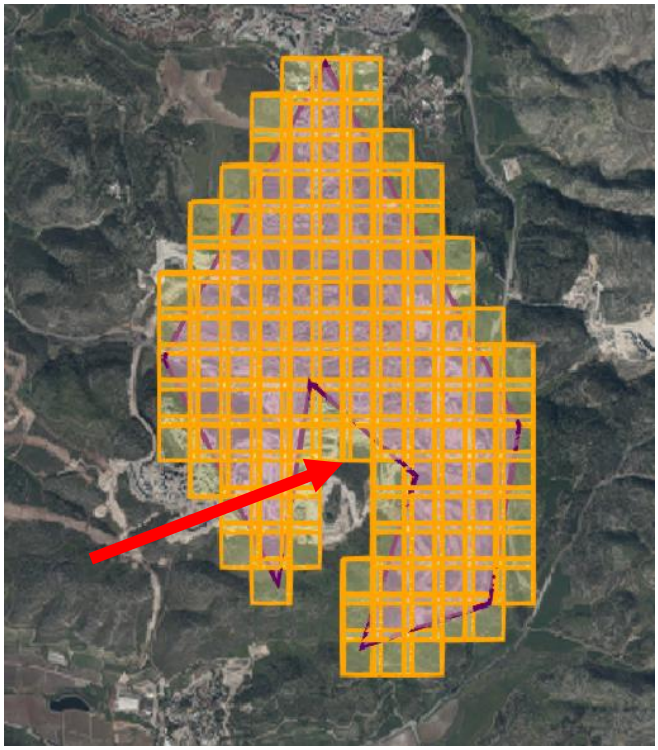
תמונה 1 : כיסוי שטח ע"י מטרות מסוג Spot_1 עם 20% חפיפה



תמונה 2 : כיסוי אזור ע"י מטרות מסוג Strip_0 עם 10% חפיפה

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בתמונה הבאה מוצג כיסוי שטח (אותו שטח כמו בתמונות לעיל) ע״י מטרה מסוג Spot_0. גודל מטרה זו קטן יותר מגודל מטרה מסוג Spot_1 ולכן נדרשות יותר מטרות עבור הכיסוי.
בנוסף, ניתן לראות שבעזרת כיסוי האזור על ידי מטרות מסוג Spot_0 המטרות מכסות שטח קטן שהוא לא חלק מהאזור אותו נדרש לכסות (מוצג עם חץ בצבע אדום) לעומת תמונה 2 בה על ידי כיסוי האזור מכוסה שטח נוסף ודי גדול. לא ניתן לוותר על מטרות אלו כי הן עדיין חופפות לאזור ומכסות חלק ממנו.



תמונה 3 : כיסוי אזור ע״י מטרות מסוג Spot_0

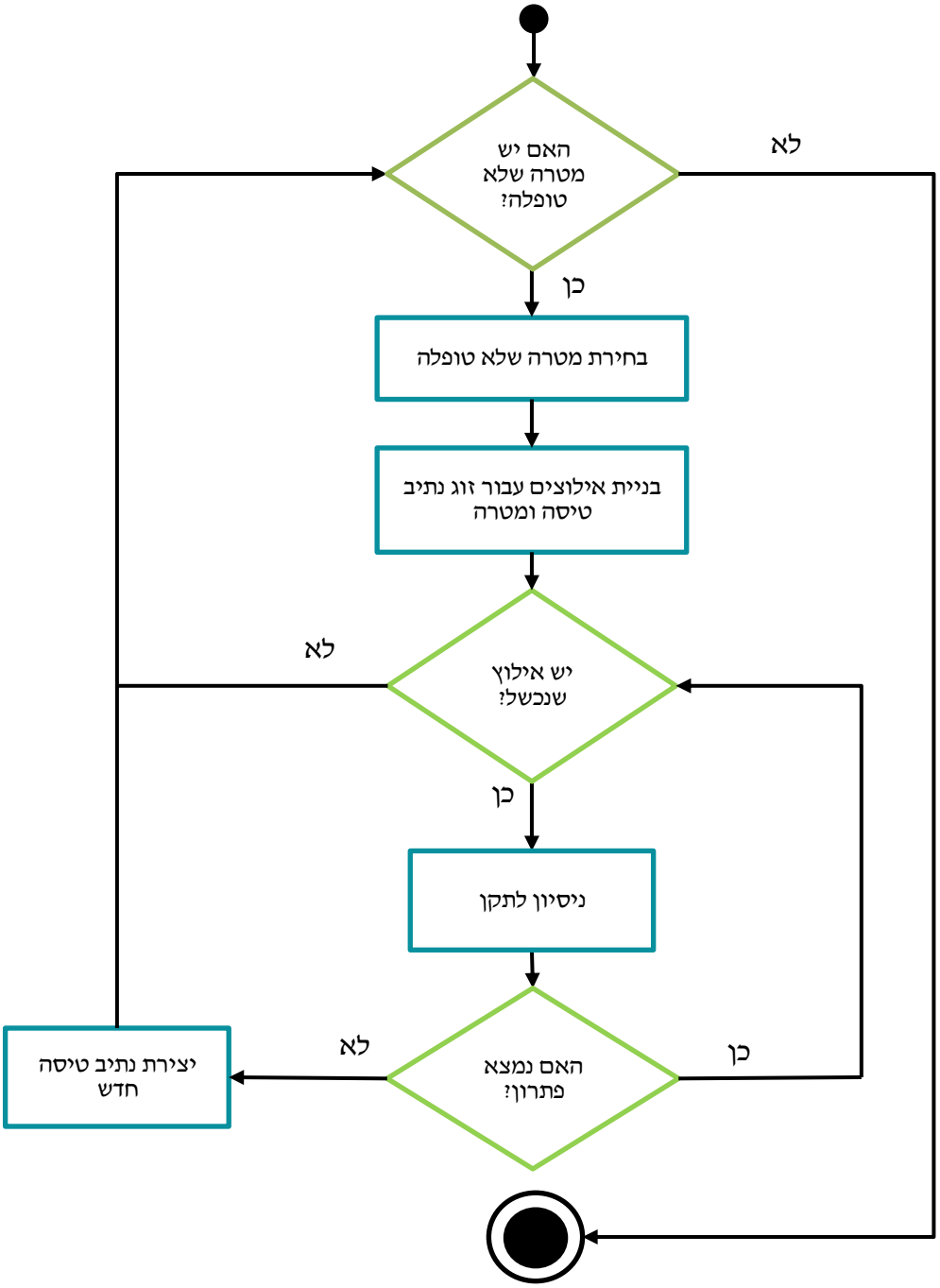
שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.3. אלגוריתם לחישוב המשימה

בפרק זה נציג את האלגוריתם לחישוב משימה, המספקת את אילוצי הסנסור, ואת התהליך של כתיבת האלגוריתם.

4.3.1. תרשים האלגוריתם

ראשית נציג את התרשים הכולל של האלגוריתם:



גרף 11: Flow Chart של האלגוריתם הראשי

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.3.2. מהלך האלגוריתם

נסביר את המהלך המתרחש עפ״י התרשים הנ״ל:

בחירת המטרה הראשונה שמטפלים בה מתבצעת באופן הבא- בוחרים את המטרה הראשונה באופן רנדומלי ויוצרים לה Default Leg המקיים את כל אילוצי הסנסור לפי סוג המטרה. משמעות הדבר היא שבשלב הראשון כל האילוצים שנבנו עבור המטרה ונתיב הטיסה הראשונים מסופקים ולכן מיד עוברים לבחירת המטרה הבאה.

מכאן והלאה:

1. בחירת המטרה הבאה לטיפול באופן הבא- מתוך נתיבי הטיסה הקיימים בוחרים את המטרה הקרובה ביותר לנתיב טיסה כלשהו מתוך המטרות שלא טופלו.
2. עבור המטרה החדשה ונתיב הטיסה הקרוב אליה ביותר, הSolver קורא לConstraint Builder שמחזיר רשימת אילוצים חדשה עבור נתיב הטיסה והמטרה, רשימה זו משורשרת לרשימת האילוצים עד כה.
3. הSolver קורא לValidator שמחזיר את האילוץ הראשון שנכשל, במידה וקיים כזה.
   - 3.1. אם לא קיים אילוץ שנכשל- חזור לשלב 1.
   - 3.2. אם קיים אילוץ שנכשל - עבור לשלב 4.
4. ניסיון לתקן את האילוץ שנכשל על ידי קריאה לResolver המתאים (לפי סוג האילוץ שנכשל). ביצוע התיקון נעשה בשיטת Backtracking לפיה נותנים ערך חדש למשתנה מסוים של הLeg ובודקים האם עבורו האילוץ מסופק. במידה ולא, מנסים עבור ערכים נוספים כל עוד הם בטווח הערכים של האילוץ. הטווח נקבע לפי סוג המטרה הספציפית.
5. בדיקה האם קיים פתרון לאילוץ שנכשל. לאחר שהאילוץ סופק בודקים האם האלגוריתם נכנס ללולאה- פינג פונג בין שני אילוצים או יותר (סיפוק של אילוץ אחד גורם לאילוץ אחר להיכשל והפוך).
   - 5.1. אם לא קיים פתרון, כלומר ישנה לולאה- יוצרים Default Leg עבור המטרה בה מטפלים וחוזרים לשלב 1.
   - 5.2. אם כן קיים פתרון כלומר, האילוץ סופק ואין לולאה שומרים את הערכים החדשים עבור הLeg הקיים לפי פתרון הResolver ובודקים את האילוצים מתחילת הרשימה כי ייתכן שכאשר בצענו תיקון על מנת שאילוץ אחד יסופק גרמנו בכך לאילוץ אחר שכבר סופק בעבר להיכשל. ייתכן שמצב זה יקרה גם עבור מטרה וLeg אחרים שאנו לא מטפלים בהם באיטרציה זו כלל.

האלגוריתם יסתיים כאשר הוא טיפל בכל המטרות ולכל מטרה האלגוריתם שייך נתיב טיסה שעל נתיב זה יש את קטע צילום המטרה בפועל (Ifr).

אסטרטגיית בחירת המטרה:

לצורך בחירת המטרה אני מבצעים מעין Clustering- חלוקה של רשימת איברים לקבוצות ע״י קיבוץ של רשימת איברים לקבוצות. אצלנו באלגוריתם בחירת המטרה הבאה לטיפול מתבצעת באופן כזה שמתוך נתיבי הטיסה הקיימים והמטרות שלא שויך להם נתיב טיסה מחפשים את הזוג מטרה - נתיב טיסה שהמרחק ביניהם הוא הקצר ביותר מבין כל המרחקים בין הזוגות

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

האפשריים של נתיבי טיסה ומטרות שלא טופלו. קיבוץ המטרות לקבוצות מתבצע תוך כדי החישובים של נתיבי הטיסה.

אופן ביצוע הClustering

המטרה הראשונה נבחרת רנדומאלית ומיוצר עבורה Default Leg המספק את כל אילוצי המערכת. לאחר מכן נבחרת המטרה הקרובה ביותר לנתיב טיסה זה. מטרה זו משתייכת לאותו Cluster של המטרה הראשונה, זאת בתנאי שיתאפשר לצלם מנתיב הטיסה הקיים את המטרה תוך סיפוק האילוצים. וכן הלאה ממשיכים באותו האופן עד אשר מסיימים עם הCluster הראשון. איך יודעים מתי עוברים לCluster הבא? כאשר האלגוריתם מנסה לשייך מטרה חדשה לLeg קיים ולא ניתן למצוא פתרון (פינג פונג בין שני אילוצים או יותר) אז הוא מייצר leg חדש עבור מטרה זו וכעת כל המטרות המשויכות לאותו נתיב טיסה מייצגות Cluster. בנוסף, הגדרנו מרחק מקסימלי כך שכאשר המטרה הקרובה ביותר לנתיב טיסה כלשהו רחוקה יותר ממרחק זה נפתח Cluster חדש.

4.3.3. סיבוכיות האלגוריתם

קלט – n מס' המטרות וm מס' האילוצים במערכת (עבור כל מטרה).

הסבר – עבור כל מטרה עוברים על כל האילוצים שלה ובודקים האם הם מסופקים. במידה ויש אילוץ שלא מסופק מתקנים אותו ובלולאה פנימית חוזרים ובודקים את כל האילוצים עד כה. במהלך החישוב מצטברים עוד ועוד אילוצים עד לm*n אילוצים. לכן, במקרה הגרוע אם עבור כל אחת מn המטרות ועבור כל אחד מהאילוצים שלה נעבור על כל האילוצים הקיימים עד כה (m*i אילוצים כאשר i הוא מס' האילוץ) מחדש נקבל את הסיבוכיות הבאה:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} m*i*j = \sum_{i=1}^{n} m*i*\sum_{j=1}^{m} j = \sum_{i=1}^{n} m*i*\left[(1+m)*\frac{m}{2}\right]$$

$$= \theta(m^3)*\sum_{i=0}^{n} i = \theta(m^3n^2)$$

4.3.4. אתגרים בכתיבת האלגוריתם

במהלך כתיבת האלגוריתם נתקלנו במספר אתגרים. אתגרים אלו באו לידי ביטוי במישור לימוד החומר וכן במישור כתיבת הקוד:

1. <u>האתגר</u>: בחירת האלגוריתם

<u>ההתמודדות</u>: ראשית הגדרנו את הבעיה איתה אנו מתמודדות. לאחר מכן בכדי למצוא את הפתרון הטוב ביותר לבעיה זו ביצענו חקר מעמיק בספרות הקיימת שכלל השוואות בין אלגוריתמים שונים. לבסוף בחרנו את האלגוריתם המתאים ביותר למאפייני הבעיה שלנו. האלגוריתם שבחרנו לפעול על פיו הוא Backtracking מהשיקולים שהוזכרו לעיל בתכנון.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

2. <u>האתגר</u>: האלגוריתם נכנס לLoop, סדרה של אילוצים שנכשלו חוזרת על עצמה שוב ושוב. ישנו אילוץ שאינו מסופק ואנו מתקנים אותו ובעקבות כך אילוץ אחר משתנה והוא כבר אינו מספק את מה שהוא נדרש אז האלגוריתם יספק גם אותו, ובעקבותיו האילוץ הראשוני שוב לא מסופק. ובכך בעצם נוצר פינג פונג של

אילוץ 1 לא מסופק ← תיקון ← אילוץ 2 לא מסופק ← תיקון וחוזר חלילה.

<u>ההתמודדות</u>:

זיהוי סדרת האילוצים הנכשלים שחוזרת על עצמה פעמיים, ופתירת בעיה זו ע"י יצירת נתיב טיסה חדש למטרה בה מטפלים.

איטרציה באלגוריתם נחשבת כאשר קוראים לValidator והוא מחזיר אילוץ שנכשל.

האלגוריתם שומר כל איטרציה ברשימה המכילה אובייקטים מסוג Iteration (כל אובייקט מכיל את רשימת האילוצים עד האילוץ שנכשל ותוקן ושדה של האילוץ האחרון שנכשל לפני תיקונו).

הפונקציה loop נועצת מצביע לאיטרציה האחרונה ועם מצביע אחר מתקדמת מהסוף להתחלה עד אשר מוצאת איטרציה הזהה בדיוק לאיטרציה האחרונה (אותו אילוץ נכשל עבור אותו Leg ואותו Target). מסמנים את מיקום האיטרציה הראשונה שזהה לאחרונה על ידי אינדקס start index.

כעת, הפונקציה בודקת לאחור אם כל שני אילוצים מקבילים שווים (מהאיטרציה האחרונה עד start index) אם כן, אז הפונקציה מחזירה true ונייצר נתיב טיסה חדש עבור המטרה. אחרת נחזיר שאין לולאה ונמשיך לנסות לתקן את האילוצים.

פונקציה זאת נקראת אחרי כל פתרון אילוץ (אחר כל קריאה לResolver).

3. <u>האתגר</u>: חישובים גאודזיים מסובכים.

<u>ההתמודדות</u>: פתרנו חלק מהחישובים במישור ולא במרחב מה שגרם לסטייה בתוצאות החישובים. לאחר בדיקה מקיפה, הגענו למסקנה שסטייה זו לא משפיעה כלל על ביצוע האלגוריתם ועל הדיוק בתוצאה. בנוסף השתמשנו בכלים מתמטיים לביצוע חישובים מורכבים וכן בחלק מהחישובים נעשה שימוש בקבצי third-party של החברה.

4. <u>האתגר</u>: מציאת שגיאות לוגיות בפלט שהאלגוריתם נותן.

<u>ההתמודדות</u>: כיוון שהאלגוריתם לחישוב משימה מורכב מאוד וקיים קושי בהבנת שגיאות אך ורק ע"י Debugging לכן הכנסנו בנוסף לכל אורך האלגוריתם כתיבת Logs לקבצים לפי המתרחש בהרצה של האלגוריתם. הכלי (open source) שהשתמשנו בו להצגת הLogs בצורה מסודרת וידידותית הוא ELK Stack. פתרון זה הוביל אותנו למציאת מקור הבעיה ע"י מציאת Log המצביע עליה בקובץ הLogs וזיהוי מקום התרחשותה בקוד.

5. <u>האתגר</u>: במהלך הפיתוח קיבלנו תוצאות שונות מהמצופה על המפה בצד הלקוח.

<u>ההתמודדות</u>: לאחר בדיקה מעמיקה וביצוע Unit Tests עפ"י מתודולוגיית Agile הבנו שמקור הבעיה הוא בהמרת ערכי הנתונים ביחידות של מטרים למעלות ובכך קיבלנו מיקומים שונים על המפה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.4. מימוש צד לקוח

את מימוש צד הלקוח חילקנו למספר קבצים.

1. קבצי JavaScript המחולקים לשניים:

- קובץ המכיל את כל המחלקות של הישויות במערכת בהתאמה למחלקות ב DataModel.Entities. הפעולות המוגדרות על אובייקט הן מחיקה, עדכון והוספה. עבור כל פעולה המתבצעת בצד לקוח נשלחת בקשת http לעדכון הנתונים בצד שרת.

- קובץ המכיל את הטיפול באירועים המתרחשים ע"י המשתמש. הקובץ מחולק לפי סוגי האירועים ולפי השתייכותם לסוגי אובייקטים. לפי הצורך נשלחות בקשות http לשרת לביצוע חישובים כלשהם.

2. קובץ HTML- מגדיר את מיקומי האלמנטים בתצוגה.

3. קבצי CSS- מגדירים את עיצוב האלמנטים בתצוגה.

4.5. תהליך פיתוח

4.5.1. מתודולוגיית פיתוח

במהלך בחירתנו במתודולוגיית הפיתוח בדקנו מספר מודלים קיימים:

- מודל מפל המים - מורכב משלבים מוגדרים היטב, בצורה קשוחה. השלבים מבוצעים אחד אחרי השני, ובכל שלב יש מיקוד במשימה עיקרית אחת בלבד. המודל אינו מאפשר למפתחים לחזור לשלבים קודמים ואם ישנה בעיה מתחילים את כל התהליך מההתחלה (Eric Conrad, 2014).

- מודל הספירלה - המודל מורכב מארבעה שלבים החוזרים על עצמם בצורה איטרטיבית: תכנון הגדרת מטרות, ניתוח סיכונים, פיתוח המוצר כולל ביצוע בדיקות והערכת הלקוח את תוצאות הפיתוח ותכנון השלב הבא (W.Boehm, 2017).

- מודל V- בכל שלב בפיתוח התוכנה במודל V ישנו שלב במקביל שבו מתבצעות בדיקות עבור חלק זה. מודל זה אינו מודל בדיקות אלא מודל פיתוח, אך עם זאת, מודל זה מדגים איך בדיקות התוכנה יכולות להשתלב בשלבי מחזור חיי הפיתוח.

- מתודולוגיית Agile- פיתוח באיטרציות מוגדרות מראש מבחינת משך האיטרציה, התוכן ובסיום כל איטרציה ביצוע בדיקות. בנוסף Agile מחייבת תקשורת פנים אל פנים, אשר מציינת עבודה במקום קרוב, הקלה על צוותים לקבל החלטות ולפעול על פיהן באופן מיידי במקום לחכות. Agile מחייב גם שיתופי פעולה קרובים עם לקוחות (Casteren, 2017).

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

מתודולוגיית הפיתוח בפרויקט זה הייתה Agile. חילקנו את זמן הפיתוח לאיטרציות כאשר משך כל איטרציה הוא כשבועיים. הגדרת משך האיטרציה, תכולתה והבדיקות בסוף האיטרציה ע"י GitLab. בכלי זה כל איטרציה מוגדרת כmilestone המכיל issues. כל issue הוא משימה כלשהי לביצוע, טיפול בבאג או הוספת feature. ניתן לתת משקלים לissues וכן לתייג אותם לפי נושאים שונים. כאשר התחלנו לעבוד על issue יצרנו branch בשרת המרוחק שניתן לעבוד עליו מקומית בסביבת הפיתוח. במהלך הקידוד ביצענו commit עם הערות לפי הצורך ובסיום הפיתוח של אותו issue ביצענו push לשרת המרוחק וסגרנו את הissue. כמובן שכל issue המגדיר הוספת פונקציונליות כלשהי למערכת מכיל גם בדיקות בסיומו לפני סגירתו.

4.5.2. ניהול סיכונים

במהלך הגדרת התוכן של כל איטרציה זיהינו את החלקים העלולים להתרחש שלא עפ"י התכנון וכתוצאה מכך עלול להיגרם עיכוב בזמן הפיתוח, תיעדפנו את הפיתוח שלהם וקבענו איך נתמודד במידה ואכן הסיכון יתרחש.

הפיתוח של המערכת נעשה בצורה רוחבית כלומר בכל שלב נגענו בכל אחד מחלקי הפרויקט וכך יכולנו לזהות חלקים שדורשים זמן פיתוח גדול יותר ולתעדף אותם על פני חלקים אחרים.

4.5.3. תהליכי CI/CD

בעזרת ר' צוות DevOps בחברה הגדרנו תהליך אינטגרציה אוטומטי וקבוע, ע"י שימוש בJenkins, בשביל לקמפל, לארוז ולבדוק את האפליקציה. בעקבות התהליך האוטומטי עשינו commit לקוד לעיתים קרובות יותר וזה הוביל לשיתוף פעולה טוב יותר ותוכנה איכותית.

4.6. בדיקות

כאמור לעיל, לאורך כל הפיתוח ביצענו בדיקות.

4.6.1. Unit Test

בסוג בדיקות זה בודקים את היחידות (פונקציות) במערכת, שכל אחת ממלאת את תפקידה בנפרד. עבור כל פרויקט (ספרייה) יצרנו במקביל פרויקט של MSTest. עבור כל מחלקה יצרנו במקביל אליה מחלקת Test ועבור כל פונקציה של המחלקה יצרנו פונקציה שבודקת אותה.

חילקנו את המקרים האפשריים למחלקות שקילות (Equivalence partitioning). מחלקת שקילות מקבצת את כל הקלטים שתגובת המערכת אליהם שקולה. עבור כל מחלקת שקילות כתבנו לפחות מקרה בדיקה אחד המכסה את כולה. בנוסף, כתבנו בדיקות עבור מקרי קצה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.6.2. System Test

בסוג בדיקות זה בודקים את פעולת המערכת כולה שאכן כל הרכיבים עובדים יחד בצורה נכונה. בבדיקות אלו לקחנו שלוש משימות שנבדקו במערכת MPW. ציפינו בבדיקות אלו לקבל פתרון עבור כל אוסף של מטרות (שהמערכת לא תכנס ללולאה אינסופית או תתקע) וכן שמס' הLegs יהיה קטן ממס' הTargets, כלומר שלא נקבל את הפתרון הטריוויאלי- שלכל Target משויך Leg נפרד. כל הבדיקות עברו בהצלחה. כדי להדגיש את יכולות המערכת AutoMPW שפיתחנו נתונה ההשוואה בטבלה הבאה- עבור כל משימה השוואנו את מס' נתיבי הטיסה במערכת MPW כאשר התכנון הוא ידני לעומת מס' נתיבי הטיסה במערכת שלנו- AutoMPW כאשר התכנון אוטומטי.

להלן התוצאות:

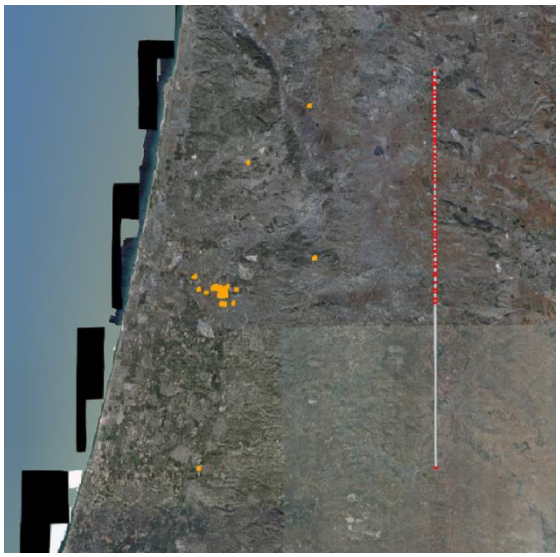
ID של משימה	מס' Targets	Legs במערכת MPW	Legs במערכת AutoMPW	זמן חישוב (sec)
1	50	13	1	27
2	27	24	4	133
3	15	15	1	40

טבלה 2: תוצאות system test

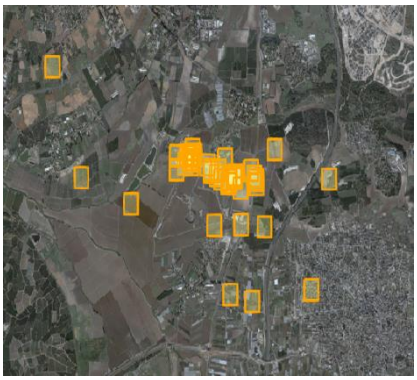
ראשית, ניתן לשים לב שמס' נתיבי הטיסה קטן משמעות במערכת שלנו כאשר התכנון הוא אוטומטי. בהערת אגב נאמר שכאשר נרחיב את אילוצי המערכת להיות לא רק מסוג אילוצים שנובעים מהמאפיינים הפיזיקליים של הסנסור מס' נתיבי הטיסה שנקבל יהיה גדול יותר אך עדיין נמוך ממס' נתיבי הטיסה בתכנון הידני.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

הצגת משימה מס' 1



תמונה 4: משימה system test 1



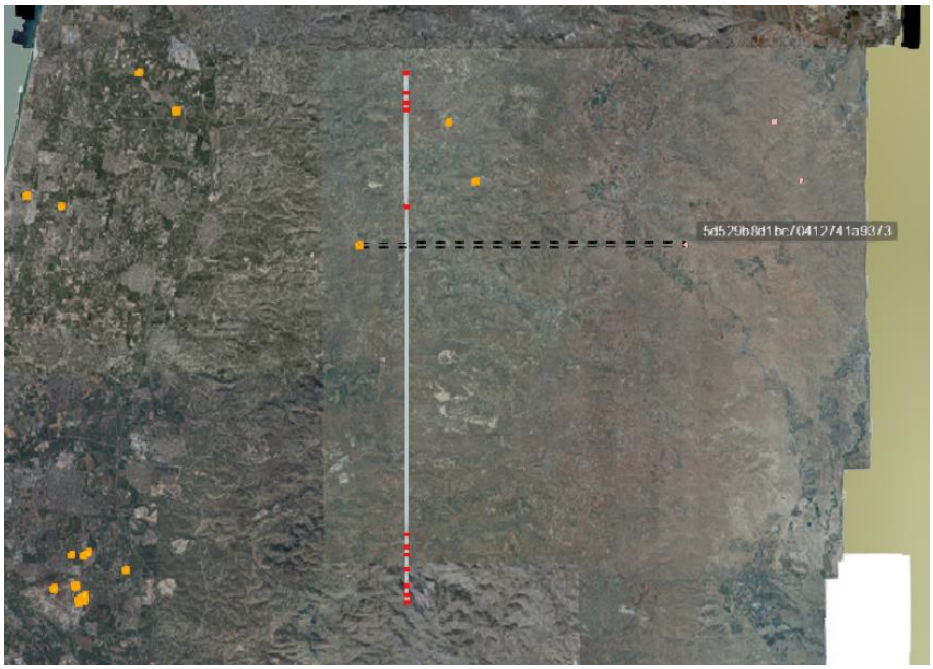
תמונה 5: zoom in של משימה 1

הנקודות הכתומות אלו המטרות (Targets), הפס הלבן זהו נתיב הטיסה (Leg) שהמערכת חישבה עבור המטרות והקטעים האדומים על נתיב הטיסה אלו קטעי הצילום בפועל של המטרות (Ifrs), לכל מטרה משויך קטע כזה שממנו הסנסור על כלי הטיס מצלם אותה.

בתמונה 2 רואים חלק מהמטרות מקרוב. כפי שניתן לשים לב המטרות חופפות אחת לשנייה כלומר זהו לא מקרה טריוויאלי כלל.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

הצגת משימה מס' 2



תמונה 6: משימה system test 2

כאן ניתן לראות עוד פונקציונליות שקיימת במערכת- לאחר חישוב המשימה ושיוך Ifr לכל אחת מהמטרות, כאשר בוחרים מטרה יש קטע על המסך (קווים מקווקוים שחורים) המשייך את המטרה לקטע הצילום ובנוסף יש תווית ליד קטע הצילום עם הId של המטרה המשויכת לקטע זה.

כפי שניתן לראות בטבלה לעיל, עבור מקרה זה האלגוריתם החזיר ארבעה נתיבי טיסה. נסביר מדוע האלגוריתם יצר למטרה הנבחרת בתמונה לעיל נתיב טיסה נפרד עליו יש Ifr ולא הגדיר קטע Ifr על הLeg הארוך והמרכזי: כפי שהסברנו לעיל אחד מאילוצי הסנסור הוא המרחק בין מרכז המטרה למרכז הIfr, למרחק זה יש טווח מסוים עבור כל סוג מטרה. במשימה לעיל, בשביל לשייך את המטרה המסומנת לLeg הארוך והמרכזי היה צריך להרחיק אותו כי המרחק הנוכחי קטן מהמרחק המינימלי האפשרי ובכך המרחק בין הLeg לשאר המטרות גדל לערך הגדול מהמרחק המקסימלי האפשרי עבורם ז"א תיקון של אילוץ אחד שנכשל גרם לאילוץ אחר שכבר סופק, להיכשל. האלגוריתם נכנס לloop כפי שהוסבר לעיל ופתר את הבעיה ע"י יצירת default leg למטרה המסומנת.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

הצגת משימה מס׳ 3

תמונה 7 : משימה 3 system test

ערכנו בדיקה נוספת על מנת לוודא שאכן מס׳ הLegs שנוצרים גדל אך ורק בעקבות אילוצי הסנסור. עבור הבדיקה בנינו שתי מטרות חופפות, פעם אחת קבענו בקובץ הקונפיגורציה טווח קטן של ערכי זווית הצילום ופעם נוספת טווח גדול יותר. ציפינו לקבל עבור הטווח הגדול יותר מס׳ קטן של legs יחסית לפתרון עבור הטווח הקטן יותר. ואכן הבדיקה עברה בהצלחה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

4.7. שפות, כלים וטכנולוגיות

4.7.1. שפות תוכנה

צד לקוח:

- ✓ HTML5
- ✓ JavaScript
- ✓ CSS

צד שרת:

- ✓ C#

4.7.2. כלים וטכנולוגיות

נציג את רשימת הטכנולוגיות שהשתמשנו בהן במהלך פיתוח הפרויקט:

- ✓ Bootstrap – ספריית CSS פופולרית ביותר לפיתוח אתרים אינטראקטיביים.
- ✓ Material Design – מערכת עיצוב שפותחה על ידי Google כדי לעזור לצוותי פיתוח לבנות חוויות דיגיטליות באיכות גבוהה עבור web בין היתר.
- ✓ ASP.NET – cross platform framework לפיתוח web ושרתים ב#C.
- ✓ Jenkins – שרת אוטומציה שהוא open source המאפשר למתכנתים לבנות, לבדוק ולשחרר את התוכנה שלהם באופן אמין. הוא מאפשר להגדיר תהליכי CI/CD בצורה פשוטה כמעט לכל שילוב של שפות ומקורות קוד שונים ע"י צינורות עיבוד נתונים (pipelines). אנחנו הגדרנו את תהליכי CI/CD באמצעות Jenkins.
- ✓ GitLab –כלי מחזור חיים מבוסס DevOps. מספק ניהול של Git-repositories, מעקב אחר Issues ויכולות לביצוע Continuous Integration וContinuous Deployment.
- ✓ ELK Stack - "ELK" זה ראשי תיבות של שלושה פרויקטים open source:
Elasticsearch, Logstash, Kibana. Elasticsearch הוא מנוע חיפוש ואנליזה. Logstash הוא צינור עיבוד נתונים (pipeline) בצד שרת שמזין נתונים ממקורות מרובים בו זמנית ושולח אותם לElasticsearch. Kibana מאפשר למשתמשים לראות ויזואלית נתונים באמצעות תרשימים וגרפים בElasticsearch .
בפרויקט שלנו כתבנו Logs לקבצים שונים לכל אורך הפתרון. ע"י log4net הגדרנו תבנית לכל הודעת log הכוללת: תאריך ושעה של כתיבתה, רמת הlog (error, debug ועוד),

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

תוכן ההודעה, Id של Leg שההודעה קשורה אליו וכן Id של Target שההודעה קשורה אליו. אצלנו, השימוש בELK Stack הוא הצגת הLogs בצורה ידידותית למשתמש וביצוע אנליזה על הנתונים. כל Log מחולק לשדות בתצוגה לפי התבנית שהגדרנו בקובץ הקונפיגורציה של log4net.

4.8 סיכום

בפרק זה הצגנו את מימוש המערכת בצורה ויזואלית ע"י תרשימי UML ו Flow Chart, את האתגרים שהתמודדנו איתם במהלך המימוש, את הבדיקות שערכנו וכן את הטכנולוגיות שהשתמשנו בהם במהלך המימוש.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

5. דיון ומסקנות

לאורך הדרך נעשו מאמצים רבים לפיתוח האלגוריתם לפתרון הבעיה, תוך שימוש בטכנולוגיות מתקדמות, ממשק ידידותי למשתמש, טכניקות וידע שנרכשו במהלך הפיתוח.
תהליך הפיתוח היה ארוך ודרש למידה עצמית מרובה והתייעצות עם צוות מומחה בנושאים מסוימים הבקיאים בתחומים השונים של הפרויקט.
אחת הסיבות העיקריות שבחרנו בפרויקט זה היא האתגר הגדול שהפרויקט הציב בפנינו.

5.1 מהלך העבודה

בשלב הראשון בעבודה על הפרויקט התמקדנו בחיפוש טכנולוגיות שיתאימו לדרישות תוך התחשבות במורכבותו של הפרויקט הכולל צד לקוח וצד שרת שצריכים לתקשר ביניהם. בנוסף לטכנולוגיות ערכנו סקר ספרות נרחב בחיפוש אחר אלגוריתמים הקיימים בשוק לפתרון בעיית CSP. כמו כן, השקענו מחשבה רבה יחד עם המנחה לתכנון הDesign של המערכת כדי שהמערכת תהיה סגורה לשינויים ופתוחה להרחבות עפ״י עקרון OCP.
אחרי כן, בחרנו במתודולוגיית הפיתוח Agile והשתמשנו בכלי DevOps לניהול התצורה והאיטרציות של מהלך הפיתוח. הכלים שנבחרו הם Jenkins ו GitLab.
בעזרת כלים אלו, ניהלנו את תהליך הפיתוח אשר תרם לעבודה מסודרת ועמידה בזמנים.
הפרויקט כולל חלקים נפרדים העומדים בפני עצמם, אך כל אחד מהווה חלק אינטגרלי במערכת.
במבט לאחור מהלך העבודה שלנו היה נכון והביא אותנו לפתרון מסודר אשר תמך בכל הרחבה שהיינו צריכות להוסיף.

5.2 אתגרים

בתחילת הפרויקט החשש העיקרי היה על האופי המחקרי של הפרויקט הדורש פיתוח אלגוריתם לבעיה מורכבת.
פיתוח אלגוריתמים הוא משימה בה לא ניתן לדעת בוודאות מראש האם אכן תהיה תוצאה סופית ומה יהיה טיבה.
במשך פיתוח הפרויקט, נתקלנו בהתלבטויות ובאתגרים שהיה צורך להתמודד איתם על ידי לימוד מעמיק יותר של החומר והתייעצות עם המנחים. ההתמודדות מול האתגרים סייעה רבות לקידום הפרויקט, תרמה להרחבת הידע ולניסיון רב בתחום.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

להלן האתגרים העיקריים:

- למידה עצמית ומרובה- פיתוח אלגוריתם לבעיית סיפוק האילוצים דרש לימוד יסודי ומעמיק של תחום זה והיכרות עם בעיות CSP נוספות ופתרונן. חיפוש מקורות והלימוד נעשו באופן עצמאי.
- טכנולוגיות וסביבת פיתוח- אתגר נוסף היה להתרגל לפיתוח בWeb אשר לא נעשה בו שימוש מקדים, ללמוד מהי הצורה הנכונה לעבודה בWeb וכן לדעת איך לתקשר בין צד השרת לצד הלקוח ובאיזה פרוטוקול תקשורת להשתמש. כמו כן באפליקציית הWeb נעשה שימוש ברכיב מפה שפותח ע"י צוות אחר בחברה. השימוש ברכיב זה דרש הכרת ה API של הרכיב והתאמת הפונקציונאליות הקיימת בו לדרישות ולצרכים של הפרויקט שלנו.
- שינויים בדרישות- לפני העבודה המעשית על הפרויקט, התקבלו דרישות מסוימות שהפרויקט צריך לעמוד בהן. לאורך הפיתוח התבצעו הרחבות בדרישות בצד הלקוח מתוך ראיית הצורך להנגיש למשתמש את השימוש במערכת. הרחבות אלו גרמו גם לשינויים בצד השרת. נוצרו עיכובים בתהליך שתוכנן אך בעקבות כך הגענו לתוצאות טובות יותר.

5.3 פיתוחים עתידיים

לאחר פיתוח מערכת תכנון משימה אוטומטי וההישגים שהגענו אליהם, ראינו צורך להמשך פיתוח במספר מישורים:

- הרחבת המערכת לכלל האילוצים- הוספת אילוצי נתיבים (מלכתחילה יש נתיבים שהמשתמש קובע שהמטוס צריך לטוס דרכם), חסכון בדלק ובזמן – דלק כתלות בגובה וזמן כתלות במהירות.
- שיפור ביצועי זמן וייעול הפתרון – נרצה לקבל מס' קטן יותר של Legs ע"י שיפור אלגוריתם הClustering. לפני הפעלת האלגוריתם לחישוב משימה נזמן את האלגוריתם Agglomerative Clustering שיחלק את המטרות לקבוצות. בתחילה כל מטרה תהייה Cluster בפני עצמה, נגדיר מרחק באופן הבא: מבין כל הזוגות האפשריים עבור שני Clusters נחשב את המרחק בין 2 נקודות ונבחר או את המרחק המינימלי או המקסימלי או הממוצע. ונמשיך לאחד בין Clusters שונים עד מרחק מסוים שנחליט שממנו והלאה מפסיקים לאחד קבוצות. בעקבות חלוקה לClusters מראש נמנע מהאלגוריתם במקרים רבים להיכנס לloop (כי ממילא מטרות במרחקים גדולים יהיו בclusters שונים) . עבור כל Cluster נקרא לאלגוריתם לחישוב המשימה בתהליכון נפרד ובכך נשפר את זמני הריצה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

5.4 רווחים אישיים מהפרויקט

- במהלך הפרויקט רכשנו מיומנויות רבות ביניהן: התמודדות עם שפות, טכנולוגיות, שיטות חדשות וכלים לתהליך פיתוח נכון. מיומנויות אלו יסייעו לנו בעבודה בתור מהנדסות.
- התמחינו בפיתוח אלגוריתמים, פיתוח אפליקציית Web ותקשורת בין צד שרת לצד לקוח.
- ביצענו פעמיים בשבוע ישיבה עם המנחה המקצועי לתכנון ההמשך, בדקנו מה נעשה עד כה והאם ישנה עמידה בזמנים. פגישות אלו תרמו לניסיון שלנו בתיאום ציפיות ובהתאמת הביצועים לדרישות.
- שיתפנו פעולה עם צוותים אחרים המתמקצעים בתחומים הקשורים אלינו ובכך צברנו ניסיון בעבודת צוות וביצוע שיתופי פעולה.
- פיתחנו מיומנויות בשיטות עבודה אמיתיות בתעשייה.

עם זאת, ההישג העיקרי עבורנו הוא ביצוע המוטל עלינו וקבלת תוצאות טובות ורצויות. בנוסף, הפרויקט הועבר לאינטגרציה עם גרסת web הקיימת בתחום.

5.5 סיכום

בפרק זה דנו במהלך העבודה, באתגרים בהם נתקלנו במהלך הפרויקט, הסקנו מסקנות ומתוכן הגענו לפיתוח עתידי.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

6. סיכום כללי

בדו״ח זה הוצג מהלך העבודה של פרויקט הגמר אשר נעשה במסגרת לימודי הנדסת תכנה במרכז האקדמי לב.

הפרויקט בוצע בחברת אלתא מערכות בע״מ ועסק בתכנון משימה אוטומטי. שיוך נתיבי טיסה למטרות שהמשתמש מגדיר תוך התייחסות לאילוצי הסנסור. העבודה על הפרויקט כללה למידה, גיבוש רעיונות וכיוונים לפיתוח ומחקר, עבודה בצוות ובשיתוף פעולה, ניתוח בעיות ופתרונן.

הפרויקט הוכיח את מסוגלותנו להתמודד עם אתגרים ולעבוד בלחץ של זמנים. בנוסף, למדנו דברים חדשים שתרמו להצלחת הפרויקט. לאחר שנה של פיתוח ניתן לומר שהפרויקט משיג את מטרתו ועונה על הדרישות שהוגדרו.

הפרויקט הסתיים לשביעות רצונה של החברה ושל התחום אליו הפרויקט שייך. כאמור הוא הועבר לאינטגרציה עם גרסת Web קיימת.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

7. נספחים

7.1. תצלומים מהפרויקט

בתמונה הבאה מוצג מסך האפליקציה. כפי שהוזכר לעיל רכיב המפה מכסה את כל המסך כאשר המשתמש יכול לנוע על המפה בעזרת העכבר ולהגיע לרזולוציות קטנות יותר. הכפתורים הנמצאים בצד ימין של המסך מאפשרים לצייר רכיבים שונים על המפה:

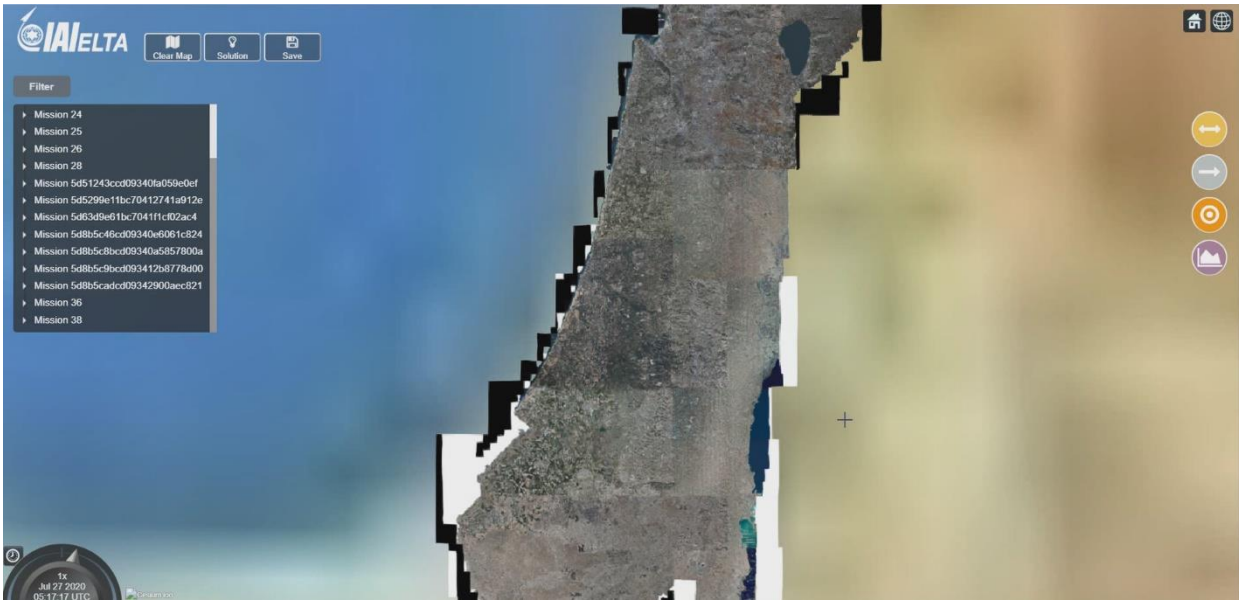
- הכפתור הצהוב – דקירת נקודות על המפה שהמשתמש רוצה לדעת את המרחק ביניהם, המרחק מחושב מידית ומוצג בתווית הצמודה לסרגל המדידה.
- הכפתור האפור- יצירת Leg.
- הכפתור הכתום- בניית Target.
- הכפתור הסגול- בניית Area.

תפקידי הכפתורים הנמצאים בצד שמאל בפינה העליונה (משמאל לימין):

- Clear Map- מחיקת כל הרכיבים המצוירים על המפה.
- Solution- חישוב משימה עבור מטרות/אזורים המצוירים על המפה.
- Save- שמירת המשימה המוצגת על המפה בבסיס הנתונים.

מצד שמאל מופיעה רשימת כל המשימות השמורות בבסיס הנתונים. כל משימה מוצגת בצורה היררכית וניתן להציג את המטרות, נתיבי הטיסה והאזורים שהיא מכילה. ניתן לסנן את רשימת המשימות לפי תאריך יצירתן ע"י לחיצה על כפתור filter (המופיע מעל רשימת המשימות) ובחירת תאריך. בעת בחירה של משימה היא תופיע מיד על המפה והמסך יעבור להיות בפוקוס עליה.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il



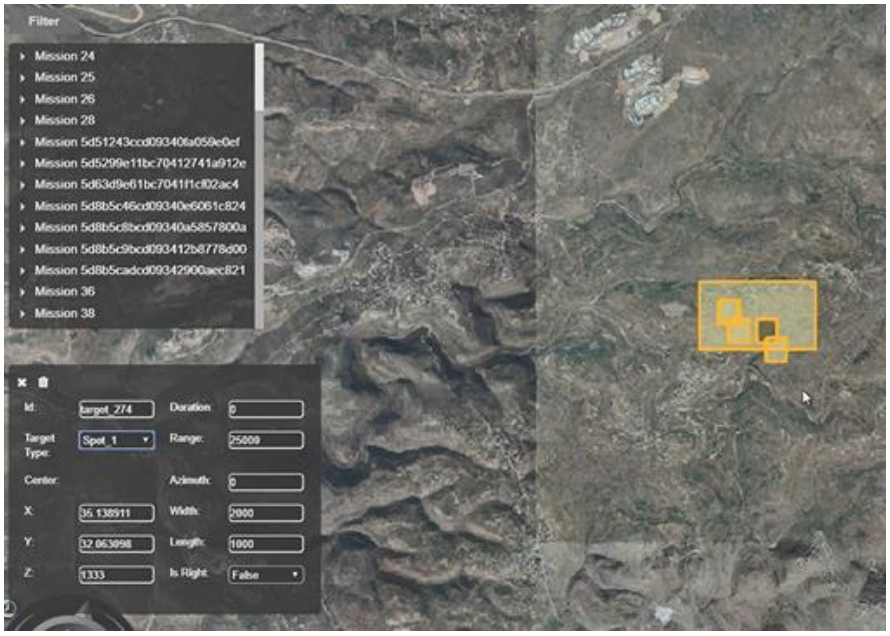
תמונה 8 : מסך האפליקציה

בצילום הבא רואים אזור (סגול) אותו נרצה לכסות. ניתן לראות בצד שמאל של המסך שישנו חלון שבו המשתמש יכול לקבוע את אחוז החפיפה של המטרות שיכסו את האזור. בנוסף, המשתמש יכול לבחור את סוג המטרה איתה הוא יכסה. מה שמופיע בתמונה הוא תוצאת חישוב כיסוי השטח לאחר שהמשתמש לחץ על כפתור "coverage".

תמונה 9 : כיסוי שטח

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בצילום הבא , ניתן לראות מטרות מסוגים שונים במערכת ע״י השוני בגודליהן וכן מצד ימין ישנו חלון המאפשר עריכה של מאפייני המטרה שהפוקוס עליה. לא ניתן לערוך אורך ורוחב של מטרה.



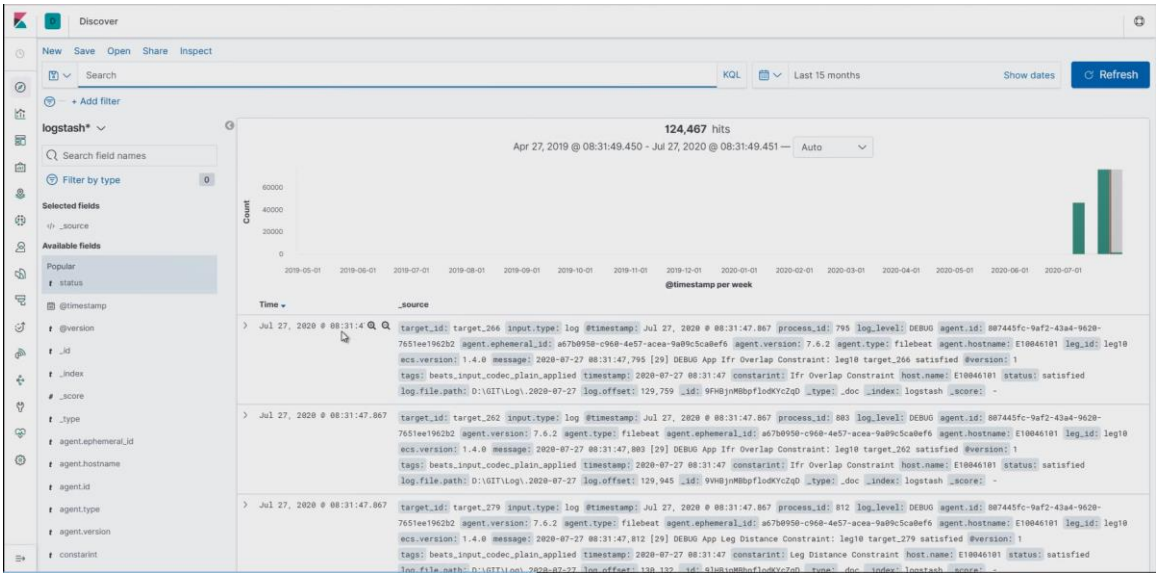
תמונה 10 : מטרות

לאחר דקירת מטרות/אזורים על המפה ולחיצה על כפתור ״solution״ נקבל : נתיבי טיסה (בלבן) המשויכים למטרות (כתום) וקטעי הצילום בפועל על נתיב הטיסה (אדום).

תמונה 11 : הצגת המשימה שחושבה

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

בצילום הבא , נראה רשימת Logs המוצגים בKibana , המתעדכנים בכל זמן הרצת האלגוריתם.
ניתן לפלטר לפי סוגי שדות ובכך לקבל אינדיקציה רחבה על הדרך שבה האלגוריתם הגיע לפתרון.



תמונה 12 : רשימת Logs

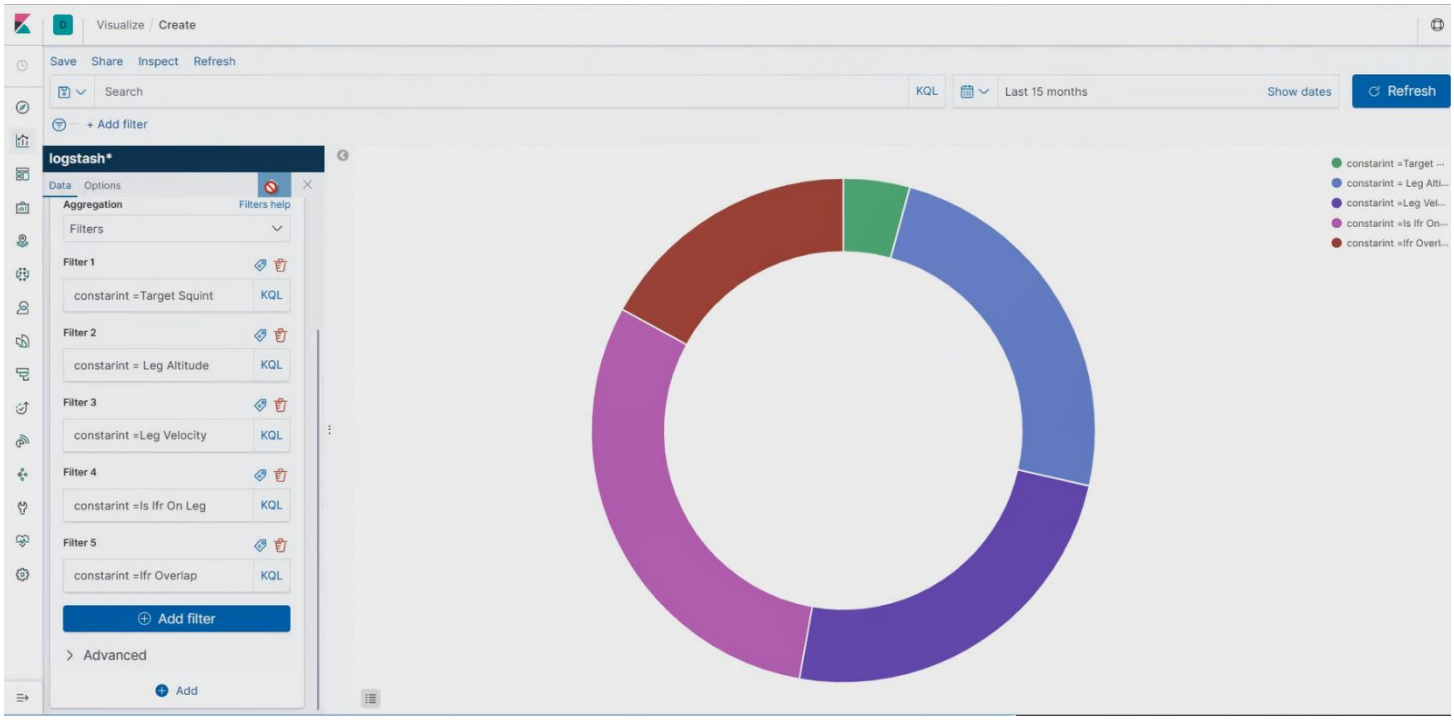
כאן נראה את כל השדות של Log יחיד.

t	_id	9FHBjnMBbpflodKYcZqD
t	_index	logstash
#	_score	-
t	_type	_doc
t	agent.ephemeral_id	a67b0950-c960-4e57-acea-9a09c5ca0ef6
t	agent.hostname	E10046101
t	agent.id	807445fc-9af2-43a4-9620-7651ee1962b2
t	agent.type	filebeat
t	agent.version	7.6.2
t	constarint	Ifr Overlap Constraint
t	ecs.version	1.4.0
t	host.name	E10046101
t	input.type	log
t	leg_id	leg10
t	log.file.path	D:\GIT\Log\.2020-07-27
#	log.offset	129,759
t	log_level	DEBUG
t	message	2020-07-27 08:31:47,795 [29] DEBUG App Ifr Overlap Constraint: leg10 target_266 satisfied
t	process_id	795
t	status	satisfied
t	tags	beats_input_codec_plain_applied
t	target_id	target_266
t	timestamp	2020-07-27 08:31:47

תמונה 13 : שדות של Log בודד

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

כעת, נראה שישנה אפשרות להציג גרפים עבור הLogs עפ״י filter שנבחר כמו בדוגמא הבאה שבה רואים את מספר הפעמים שכל אילוץ נבדק האם הוא מסופק או לא ביחס למספר הפעמים ששאר האילוצים נבדקו. כמובן שניתן לשנות משמאל את הסינון וכן להציג את הנתונים בדרכים שונות.



תמונה 14 : הצגת מס׳ הפעמים שכל אילוץ נבדק בעזרת דיאגרמת פאי

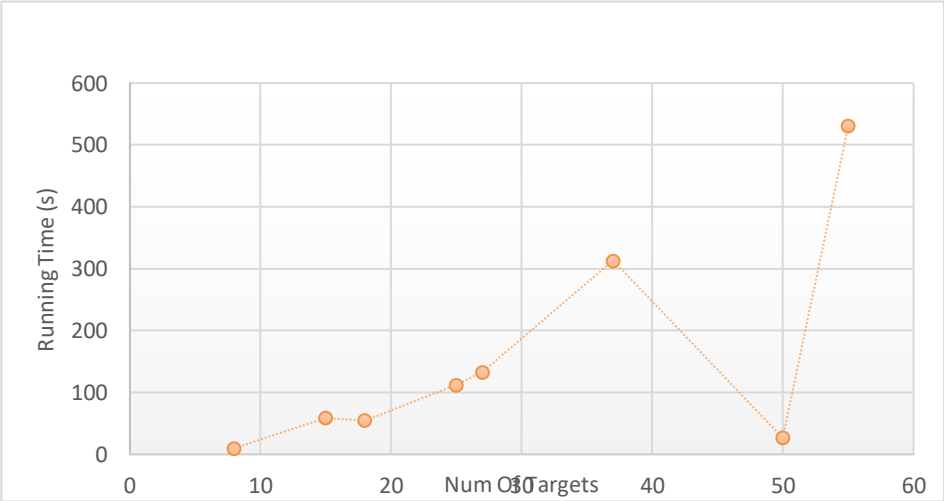
שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

7.2. בדיקות ביצועים

בטבלה לעיל מוצגות תוצאות של בדיקות ביצועים שערכנו במערכת. בכל קלט הגדלנו את מס' המטרות. ניתן לראות שככל שמספר המטרות עולה זמן הריצה גדל. כמו כן, ניתן לראות שככל שהקלט מכיל מס' גדול יותר של אזורים זמן הריצה גדל, זאת כמובן בעקבות החפיפה בין המטרות שמכסות אזור כלשהו שמקשה על מציאת פתרון.

מס' משימה	מס' מטרות כולל	מס' אזורים המכוסים ע"י חלק מהמטרות	מס' נתיבי טיסה שהתקבלו	משך זמן חישוב (s)
1	8	2	1	10
2	15	2	2	59
3	18		1	55
4	25	2	1	112
5	27		4	133
6	37	1	1	313
7	50		1	27
8	55	3	1	531

טבלה 3 : ביצועי זמן כתלות במספר המטרות



גרף 12 : זמן ריצה כתלות במס' המטרות

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il

8. ביבליוגרפיה

1. Adams J., Balas E., and Zawack D .(1988) .The shifting bottleneck procedure for job shop scheduling .*Management Science 34*.391-401 ,
2. B. G. W. Craenen, A. E. Eiben, and J. I. van Hemert. (2003). Comparing Evolutionary Algorithms on Binary. *IEEE Transactions On Enolutionary Computation*.
3. B.S. Steward, C' W .(1991) .'Multiobjective A .*Journal of the ACM, 38.(4)*
4. Casteren, W' v .(2017 2) .'The Waterfall Model and the Agile Methodologies : A comparison by project characteristics.
5. David A. Cohen, Martin C. Cooper, Peter G. Jeavons, Stanislav Zivny. (2019). Binary Constraint Satisfaction Problems Defined by Excluded Topological Minors. *Information and Computation*.
6. Eric Conrad, S' M .(2014) .'*Eleventh Hour CISSP* .
7. Gudaitis, M .(1994) .'Multicriteria Mission Route Planning Using a Parallel A* Search .*M.Sc. Thesis, Air Force Institute of Technology, Air University*.
8. J. Btazewicz, W. Domschke and E. Pesch .(1996) .The job shop scheduling problem: Conventional .*European Journal of Operational Research 93*.1-33 ,
9. James R. Bitner and Edward M. Reingold .(1975) .Backtrack Programming Techniques .*Communications of the ACM*.
10. K. Tulum, U. Durak , S. Kemal İder .(2009) .Situation Aware UAV Mission Route Planning .*IEEE Xplore*.
11. Kent B., M' B .(2001) .'*Manifesto for Agile Software Development* .
12. P. Hart, N. Nilsson, and B. Raphael .(1968) .A formal basis for the heuristic determination of minimum cost paths .*IEEE Transactions of Systems Science and Cybernetics*.
13. Paterson, J .(1997) .'Measuring Low Observable Technology's .*AIAA*.
14. Serkan Güldal , Veronica Baugh , Saleh Allehaibi .(2016) .N-Queens Solving Algorithm by Sets and Backtracking .*IEEE Xplore*.
15. Sezer, E .(2000) .'Mission Route Planning with Multiple Aircraft and Targets Using Parallel A* Algorithm .*M.Sc. Thesis, Air Force Institute of Technology*.
16. Stuart J., Peter Norving .(2009) .*Artificial Intelligence : A Modern Approach .(6.1-6.4)*
17. W.Boehm, B .(2017 6 27) .'A Spiral Model of Software Development and Enhancement.
18. Z. Wang, D. Huang, J. Tan, T. Liu, K. Zhao and L. Li .(2013) .A parallel algorithm for solving the n-queens problem based on inspired computational model .*BioSystems, pp. 22-29*.

שד' יצחק הנשיא 100, ת.ד 330, אשדוד טלפון: 08-8623400/1 www.iai.co.il