The Iby and Aladar Fleischman Faculty of Engineering
Tel Aviv University

הפקולטה להנדסה
ע"ש איבי ואלדר פליישמן
אוניברסיטת תל אביב

שערוך טווח למטרה מנתוני מיקום עצמי וזווית צילום

פרויקט מס׳ 22-1-1-2731

דו״ח סיכום

מבצעים:

נוי קירשנר 206810558

סתיו שפר 204242218

מנחים:

ד״ר גבריאל דוידוב אוניברסיטת ת״א

מקום ביצוע הפרויקט:

קמפוס אוני׳ ת״א

תוכן עניינים

רשימת קיצורים

RLS – Recursive Least Squares

EKF – Extended Kalman Filter

ZOH – Zero Order Hold

GPS – Global Positioning System

MSE – Mean Squared Error

MMSE – Minimum Mean Square Error

ECEF – Earth-Centered Earth-Fixed

ENU – East North Up

רשימת איורים

רשימת טבלאות

תקציר

מהות פרויקט זה הינו שיערוך מיקום וטווח למטרה מנתוני מיקום עצמי וזווית יחסית למטרה, כך שהשערוך מבוצע באמצעים פאסיביים ומאפשר שמירה על חשאיות.

במסגרת הפרויקט חקרנו אלגוריתמים שונים, בניהם כאלה המבוססים על מסנן קלמן, המאפשרים שערוך מיקום של גוף מסוים, או כפי שנתייחס אליו מעתה ואילך – מטרה. ביצענו השוואה בין התוצאות השונות שקיבלנו על מנת לבחון מהו האלגוריתם המתאים ביותר עבור השיערוך.

על מנת לבצע את האיכון נאספו שני סוגי נתונים:

- המיקום העצמי שלנו – נתון GPS
- הזווית בה נמצאת המטרה, שנמדדה תוך עקיבה באמצעי אופטי

נתונים אלו הוכנסו כקלט לאלגוריתמים הבאים:

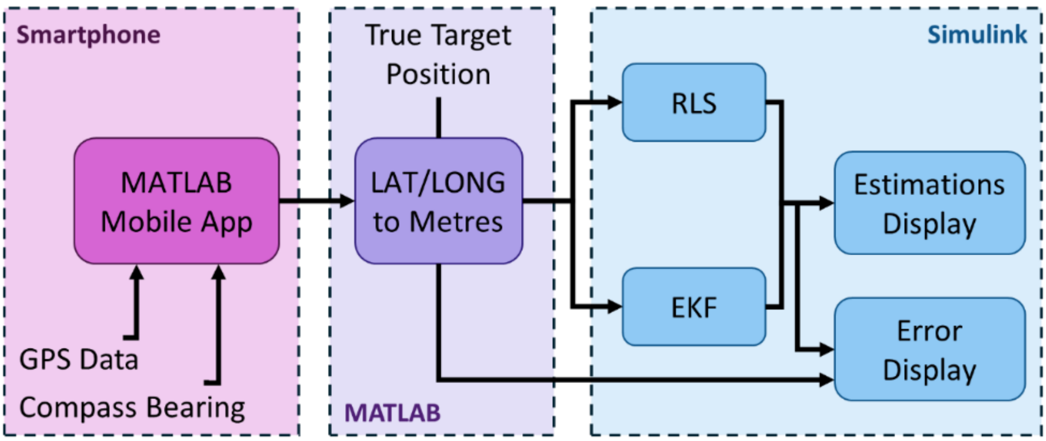
- **<u>RLS – Recursive Least Square</u>**

  אלגוריתם זה משתמש בעדכונים רקורסיביים של המשערך בהתאם לרמת הסמך של תוצאות האיטרציה הקודמת, ושואף להביא למינימום את סכום ריבועי השגיאה, כלומר הוא אופטימלי במובן MSE.
- **<u>EKF – Extended Kalman Filter</u>**

  אלגוריתם זה מעדכן גם הוא באופן רקורסיבי את המשערך, אך הוא עושה זאת בהתאם לרמת הסמך של הסנסורים שמשתתפים בעדכון. הוא שואף להביא למינימום את השגיאה הריבועית הממוצעת כך שהוא אופטימלי במובן MMSE.

מימוש כלל המודל, והאלגוריתמים בליבו, בוצע בסביבת Simulink ממשפחת MathWorks, כך שחלקיו הפנימיים של כל בלוק מומשו ב-MATLAB.

בשלב הראשון ביצענו סימולציות – הכנסנו לאלגוריתמים נתונים שיצרנו באופן מדומה, ובחנו את תגובותיהם.
בשלב השני עברנו לביצוע מעשי – ביצענו סדרת ניסויים תוך איסוף נתונים. את נתונים אלו, לאחר המרתם והתאמתם, הזנו לכניסת המודל לקבלת תוצאות אמת. עיבוד הנתונים והשוואה בין האלגוריתמים הראו ששני האלגוריתמים – RLS ו-EKF סיפקו תוצאות שעמדו ביעדי הפרויקט. עם זאת, אלגוריתם ה-EKF סיפק התכנסות מהירה יותר שגיאת מצב מתמיד קטנה יותר, כך שבשה״כ הוא סיפק שערוך מדויק יותר. דיאגרמת המערכת מתוארת באיור 1.



איור 1: דיאגרמת בלוקים כללית

1. הקדמה

המוטיבציה שעמדה מאחורי פרויקט זה מקורה בעולמות הצבאיים, כאשר התרחיש עליו התבססנו היה: כלי שיט צבאי אויב, מפליג בחשאיות בים ואינו חושף את מיקומו בפני ספינות כוחותינו המפליגות באזור. חוסר ידיעת מיקום האויב מקשה עלינו לבנות את התמונה הימית או אף לבצע ביות נשקים. מאידך, כלי שיט כוחותינו נזהרים מלהפעיל אמצעי גילוי שמשדרים כלפי חוץ, מפני ששידור כזה מנדב מידע לאויב. אם כן, **מטרת הפרויקט המרכזית היא מציאת פתרון פאסיבי לאיכון מטרות**, כאלטרנטיבה לאמצעי גילוי אקטיביים, כמו מכ״מים וסונארים על מגוון סוגיהם.

שיערוך מיקום המטרה יבוצע על ידי נתוני מיקום עצמי של הכלי המודד ומדידת זווית יחסית למטרה. נתונים אלו הוזנו למספר שיטות שיערוך שונות, אלגוריתם RLS ואלגוריתם EKF, כאשר בפרויקט השוונו בין תוצאות האלגוריתם השונים ובחנו את טיבם בפתרון שיערוך המיקום.

את תוצאות האלגוריתמים נוכל לבחון על ידי סימולציה שמדמה את מיקום המטרה ומיקומנו במרחב ולאחר מכן, התוצאות יבחנו בשטח על ידי מדידות אמתיות של מיקום עצמי וזווית למטרה במרחב.

להלן המדדים הכמותיים בהם שאפנו לעמוד:

שגיאה מקסימלית במיקום: 10% מטווח המטרה

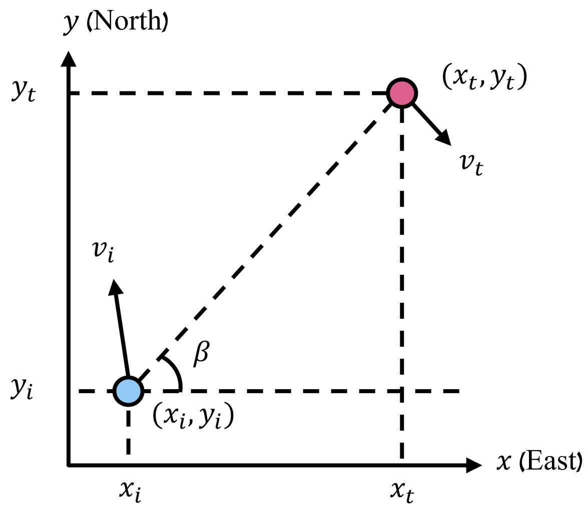
זמן התכנסות מקסימלי לפתרון: 20 שניות

מספר מדידות מקסימלי להתכנסות: 20/dt

2. רקע תיאורטי

2.1. תרחיש כללי

תחילה נתאר באופן גרפי איכותי את התרחיש הכללי הנדון:



איור 2: תיאור איכותי של התרחיש הכללי

מערכת הצירים היא מערכת לוקאלית שיחידותיה הן מטרים.

בכחול – הכלי המודד: מיקומו – (x_i, y_i) ; מהירותו – v_i

באדום – המטרה: מיקומה – (x_t, y_t) ; בהינתן שמדובר במטרה דינאמית, מהירותה – v_t

β – הזווית בין ציר ה-x לקו המחבר בין המודד למטרה

מהתבוננות בסכמה שבאיור 2 ניתן להגיד שעבור כל זמן k מתקיים:

(1)

$$\tan\beta = \frac{y_t - y_i}{x_t - x_i}$$

הצבת הקשר $\tan\beta = \frac{\sin\beta}{\cos\beta}$ תיתן:

(2)

$$\frac{sin\beta}{cos\beta} = \frac{y_t - y_i}{x_t - x_i}$$

ולאחר הכפלה במכנה והעברת אגפים נקבל:

(3)

$$(x_t - x_i)sin\beta = (y_t - y_i)cos\beta$$

(4)

$$y_i\cos\beta - x_i\sin\beta = y_t\cos\beta - x_t\sin\beta$$

נצלול כעת לעומקם של האלגוריתמים תוך שימוש בפיתוח הנ"ל.

2.2. אלגוריתם Recursive Least Square – RLS

2.2.1. RLS – מטרה סטטית

אלגוריתם זה מבצע שיערוך של מערכות לינאריות ע"י עדכון של מקדמי המסנן באופן רקורסיבי על מנת למזער את סכום ריבועי השגיאה, ולכן הוא אופטימלי במובן MSE. בשלב הראשון הוא מבצע חיזוי של מצב המערכת, במקרה שלנו – מיקום המטרה, ולאחר מכן מבצע עדכון בו הוא ממשקל את המצב הנוכחי ואת המידע החדש המגיע מהסנסורים עפ"י רמת המהימנות שלהם.

נגדיר, עבור זמן כלשהו k:

$X_k = \begin{pmatrix} x_{t,k} \\ y_{t,k} \end{pmatrix}$ – וקטור המצב (מיקום המטרה) בזמן k

$$\Phi_k = \begin{pmatrix} -\sin\beta_k \\ \cos\beta_k \end{pmatrix}$$

(5)

$$z_k = y_{i,k}\cos\beta_k - x_{i,k}\sin\beta_k$$

אז ניתן לכתוב את משוואה (5) באופן הבא:

(6)

$$z_k = \Phi_k^T X_k$$

בנוסף, ניתן להגדיר את השגיאה, שהיא ההפרש בין השערוך שבוצע בזמן k-1 למיקום שחושב עפ"י המדידות בזמן הנוכחי, k:

(7)

$$e_k = z_k - \Phi_{\mathrm{k}}^T X_{k-1}$$

בשלב העדכון בזמן k האלגוריתם כאמור מבצע מִשְקול של מצב המערכת (כפי שחושב באיטרציה ה-k-1), ושל הנתונים שהתקבלו באיטרציה הנוכחית. משקול זה מתבצע ע"י וקטור המשקלים g_k, ובהתאם לרמת הסמך של המשערך שהתקבל באיטרציה הקודמת. רמת הסמך מתבטאת מתמטית כשונות המשערך של כל אחד מרכיבי וקטור המצב – ככל שהשונות גדולה יותר כך הנתון פחות ודאי לכן יינתן משקל גדול יותר למדידה חדשה.

המטריצה P_k – היא מטריצת הקווריאנס של וקטור המצב, ומכילה את ערכי השונות עבור כל קלט. היא מתעדכנת באופן רקורסיבי בעזרת וקטור המשקלים g_k.

נגדיר אותם כך:

(8)

$$g_k = P_{k-1}\Phi_k \cdot \left(1 + \Phi_k^T P_{k-1}\Phi_k\right)^{-1}$$

(9)

$$P_k = P_{k-1} - g_k\Phi_k^T P_{k-1}$$

בסה״כ, עבור שערוך בזמן k נקבל:

(10)

$$\hat{X}_k = X_{k-1} + e_k g_k$$

2.2.2. RLS – מטרה דינאמית

מודל המטרה הדינאמית זהה בבסיסו למודל בו כלי השיט (המודד) נע במהירות קבועה.

אך כעת אנו מכניסים פרמטר נוסף – forgetting factor – המסומן ב-λ. פרמטר זה קובע כמה מהר האלגוריתם ״שוכח״ את השיערוך הקודם: כאשר $\lambda = 1$ האלגוריתם מתחשב במשערך הקודם באותו אופן שמתחשב בו האלגוריתם הסטטי, וכאשר $\lambda = 0$ האלגוריתם ״שוכח״ לחלוטין את המשערך הקודם ומסתמך על המדידות האחרונות בלבד.

עבור מטרה דינאמית קבענו את הפרמטר להיות מאותחל בערך $\lambda_{initial} = 0.8$.

כעת וקטור המשקלים ומטריצת הקוואריאנס יוגדרו כך:

(11)

$$\boldsymbol{g_k = P_{k-1}\Phi_k \cdot \left(\lambda + \Phi_k^T P_{k-1}\Phi_k\right)^{-1}}$$

(12)

$$\boldsymbol{P_k = \frac{1}{\lambda}\left(P_{k-1} - g_k \Phi_k^T P_{k-1}\right)}$$

הפרמטר λ מתעדכן בכל איטרציה על פי הנוסחה הבאה:

(13)

$$\lambda = \lambda_{initial} * \lambda + (1 - \lambda_{initial})$$

בסה״כ, עבור שערוך בזמן k נקבל:

(14)

$$\hat{X}_k = X_{k-1} + e_k g_k$$

2.3. אלגוריתם Extended Kalman Filter – EKF

אלגוריתם זה הינו הרחבה של מסנן קלמן, והוא אופטימלי במובן MMSE. הוא מתוכנן עבור מערכות לא לינאריות, בניגוד לאלגוריתם ה-RLS שהניח לינאריות של המערכת, אך גם הוא פועל בשני שלבים – חיזוי ועדכון, וגם הוא מבצע משקול בשלב העדכון. על מנת להבין את עקרונותיו של ה-EKF נתאר ראשית את האלגוריתם עליו הוא מבוסס.

מסנן קלמן

מסנן קלמן הוא אלגוריתם המתוכנן לשערוך מערכות דינאמיות לינאריות המבוסס על שני מודלים – מודל התהליך ומודל המדידות:

- **מודל התהליך** – מתאר את האופן בו מצב המערכת משתנה בין צעד לצעד.
- **מודל המדידות** – מתאר את האופן בו המדידות מתקשרות למצב המערכת.

במצב בו אין רעש ואין שגיאות – כל מודל לבדו יכול היה לספק לנו משערך מדויק לווקטור המצב. אך זה כמובן רק מצב היפותטי, שכן לא קיימת מערכת דינאמית שאינה מושפעת מגורמים חיצוניים אקראיים, ומאידך לא קיים חיישן מושלם שיכול לתאר את אותה מערכת ללא שגיאות בעצמו.

על כן, בשביל לקבל משערך מדיוק ככל הניתן, בהינתן שני המודלים האלו, יש להתחשב בשניהם. אופן השקלול של תוצאות שני המודלים האלו, לכדי משערך אופטימלי, הוא רעיון הליבה של מסנן קלמן, והוא מזוקק לגודל שנקרא הגבר קלמן – K.

ערכו של הגבר קלמן הוא בתחום שבין 0 ל-1 והוא נקבע בהתאם לשגיאותיהם של שני המודלים:

- עבור $K = 0$ – רמת הסמך הנמוכה ביותר של המדידות וגבוהה של התהליך, לכן נתחשב במודל התהליך בלבד ונזניח לחלוטין את המדידות.

- עבור $K = 1$ – רמת הסמך הגבוהה ביותר של המדידות ונמוכה של התהליך, לכן נזניח לחלוטין את מודל התהליך ונתחשב במודל המדידות בלבד.

אם כן, נתבונן במודלים באופן מתמטי –

נגדיר:

X_k – וקטור המצב

Z_k – וקטור המדידות

כך שמתקיים

(15)
$$\boldsymbol{X_k = F_k X_{k-1} + W_k}$$
(16)
$$Z_k = H_k X_k + V_k$$

כאשר

F_k – מטריצת התהליך, המוחלת על מצב המערכת בשלב הקודם לקבלת המצב הנוכחי

$W_k \sim N(0, Q_k)$ – רעש התהליך (נניח רעש גאוסי עם תוחלת 0 ומטריצת קוואריאנס Q_k)

H_k – מטריצת המדידות הממפה את וקטור המצב למרחב המדידות

$V_k \sim N(0, R_k)$ – רעש המדידות (נניח רעש גאוסי עם תוחלת 0 ומטריצת קוואריאנס R_k)

בדומה לאלגוריתם ה-RLS, גם מסנן קלמן משתמש במטריצת קוואריאנס P_k. רכיבי מטריצה זו מהווים אינדיקציה לרמת הסמך של כל מדידה, שכן הם מכילים את השונות של התפלגות המדידות.

בשלב החיזוי:

(17)
$$\boldsymbol{\widehat{X}_{k|k-1} = F \widehat{X}_{k-1|k-1}}$$
(18)
$$\boldsymbol{P_{k|k-1} = F P_{k-1|k-1} F_k^T + Q_k}$$

בשלב העדכון:

(19)
$$\boldsymbol{y_k = Z_k - H_k \widehat{X}_{k|k-1}}$$
(20)
$$K_k = P_{k|k-1} H_k^T \left(H_k P_{k|k-1} H_k^T + R_k\right)^{-1}$$

(21)
$$\boldsymbol{P_k = (I - K_k H_k) P_{k|k-1}}$$
(22)
$$\widehat{X}_k = \widehat{X}_{k|k-1} + K_k y_k$$

אולם, המערכת שלנו לא מקיימת את התנאי הבסיסי לנכונות מסנן קלמן והוא – הלינאריות. חישוב פונקציית ה-tan הופך אותה ללא לינארית, ועל כן אנו נאלצים לחפש את הפתרון בעזרת אלגוריתם שמתגבר על חוסר הלינאריות הזו.

2.3.1. EKF – מטרה סטטית

עקרונותיו של אלגוריתם זה זהים לעקרונות אותם תיארנו לעיל אך הוא מתוכנן עבור מערכות לא לינאריות. הוא מניח שמתקיים קשר לא לינארי בין וקטור המצב והמודלים, בצורה הבאה:

(23)
$$\boldsymbol{X_k = \widetilde{F}(X_{k-1}) + W_k}$$
(24)
$$Z_k = \widetilde{H}(X_k) + V_k$$

כאשר $\widetilde{F}, \widetilde{H}$ הן פונקציות לא לינאריות.

טרם ביצוע חישובי השערוך עצמם, האלגוריתם מבצע לינאריזציה למודלים באמצעות חישוב יעקוביאן –

$F_k = \frac{\partial \tilde{F}}{\partial x}\Big|_{\hat{X}_{k-1}}$ - יעקוביאן פונקציית התהליך

$H_k = \frac{\partial \tilde{H}}{\partial x}\Big|_{\hat{X}_{k|k-1}}$ - יעקוביאן פונקציית המדידות

נתאר כעת את כלל המשתנים :

$X_k = \begin{pmatrix} x_t \\ y_t \\ x_i \\ \dot{x}_i \\ y_i \\ \dot{y}_i \end{pmatrix}$ – וקטור המצב. מכיל את רכיבי המיקום העצמי, המהירות העצמית ומיקום המטרה.

$Z_k = \begin{pmatrix} x_i \\ y_i \\ 0 \end{pmatrix}$ – וקטור המדידות

את מודל המדידות נוכל לכתוב כך שמטריצת היעקוביאן היא

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \sin\beta & -\cos\beta & -\sin\beta & 0 & \cos\beta & 0 \end{pmatrix}$$

ואז מתקיים

(25)

$$Z_k = \begin{pmatrix} x_i \\ y_i \\ 0 \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \sin\beta & -\cos\beta & -\sin\beta & 0 & \cos\beta & 0 \end{pmatrix}}_{H_k} \cdot \begin{pmatrix} x_t \\ y_t \\ x_i \\ \dot{x}_i \\ y_i \\ \dot{y}_i \end{pmatrix} = H_k X_k$$

את מודל התהליך נוכל לכתוב כך שמטריצת היעקוביאן היא

$$F = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

ואז מתקיים

(26)

$$\boldsymbol{\hat{X}_k} = \begin{pmatrix} \boldsymbol{x_t} \\ \boldsymbol{y_t} \\ \boldsymbol{x_i} \\ \boldsymbol{\dot{x}_i} \\ \boldsymbol{y_i} \\ \boldsymbol{\dot{y}_i} \end{pmatrix}_{\boldsymbol{k}} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \boldsymbol{\Delta t} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \boldsymbol{\Delta t} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{x_t} \\ \boldsymbol{y_t} \\ \boldsymbol{x_i} \\ \boldsymbol{\dot{x}_i} \\ \boldsymbol{y_i} \\ \boldsymbol{\dot{y}_i} \end{pmatrix}_{\boldsymbol{k-1}} = \boldsymbol{F_k X_{k-1}}$$

מנקודה זו ואילך, ניתן להשתמש במשוואות המודל של מסנן קלמן משום ששני המודלים שלנו – המדידות והתהליך – שניהם לינארים.

2.3.2. <u>EKF – מטרה דינאמית</u>

גם עבור אלגוריתם זה נדרשת הרחבה לוקטור המצב ולמטריצת המקדמים.

נגדיר :

H_{dyn} – מטריצת יעקוביאן המדידות למטרה דינאמית

F_{dyn} – מטריצת יעקוביאן התהליך למטרה דינאמית

אחרי העדכון נקבל:

(27)

$$\hat{X}_{k,dynamic} = \begin{pmatrix} x_t \\ \dot{x}_t \\ y_t \\ \dot{y}_t \\ x_i \\ \dot{x}_i \\ y_i \\ \dot{y}_i \end{pmatrix}_k = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}}_{F_{dyn}} \cdot \begin{pmatrix} x_t \\ \dot{x}_t \\ y_t \\ \dot{y}_t \\ x_i \\ \dot{x}_i \\ y_i \\ \dot{y}_i \end{pmatrix}_{k-1} = F_{k,dyn} X_{k-1}$$

(28)

$$Z_{k,dynamic} = \begin{pmatrix} x_i \\ y_i \\ 0 \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \sin\beta & 0 & -\cos\beta & 0 & -\sin\beta & 0 & \cos\beta & 0 \end{pmatrix}}_{H_{k,dyn}} \cdot \begin{pmatrix} x_t \\ \dot{x}_t \\ y_t \\ \dot{y}_t \\ x_i \\ \dot{x}_i \\ y_i \\ \dot{y}_i \end{pmatrix} = H_k X_k$$

בהינתן משוואות מטריציות אלה ניתן להחיל את האלגוריתם עליהן.

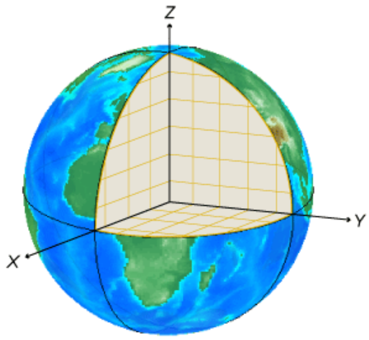
2.4. המרת נתון GPS למטרים במערכת קואורדינטות לוקלית

על מנת לחשב את כיוון ומרחק המטרה מאיתנו נדרש מאיתנו לעבוד במערכת קואורדינטות לוקאלית כך שאת מרחק המטרה מאיתנו נוכל להביע ביחידות של מטרים. לצורך כך, ביצענו המרה של נתוני ה-GPS ממערכת קואורדינטות גאודזית למערכת ECEF, ולאחר מכן מ-ECEF למערכת לוקאלית ENU.

מערכת קואורדינטות גאודזית מתארת כל נקודה על פני כדה"א ע"י קו רוחב (latitude), קו אורך (longtitude) וגובה (altitude). נגדיר:

- ϕ - הזווית ביחס לקו המשווה, מתארת את המיקום על ציר ה-latitude.
- λ – הזווית ביחס למרידיאן הראשי, מתארת את המיקום על ציר ה-longtitude.
- h – הגובה מעל האליפסואיד.

מערכת הקואורדינטות (Earth-Centered, Earth-Fixed) **ECEF** היא מערכת קואורדינטות קרטזית שראשיתה במרכז כדה"א, כפי שניתן לראות באיור הבא.



איור 3: מערכת קואורדינטות ECEF

לצורך המרה ממערכת גאודזית למערכת ECEF יש להתחשב במידות כדה״א ובמודל מסוים של אליפסואיד המתאר אותו. ממודל זה נגזרים 3 ערכים : a – חצי הציר הראשי של האליפסואיד ; e – אקסצנטריות האליפסואיד ; N – רדיוס העקמומיות באנך הראשי.

בהינתן הנ״ל נוסחת ההמרה תהיה :

(29)

$$N = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}}$$

(30)

$$X = (N + h)\cos\phi\cos\lambda$$

(31)

$$Y = (N + h)\cos\phi\sin\lambda$$

(32)

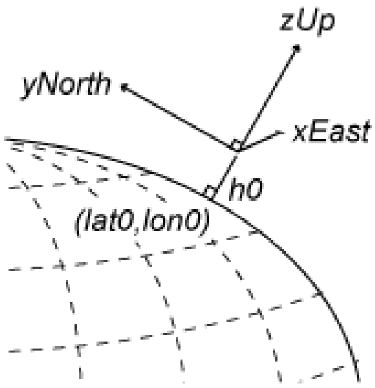
$$Z = \big((1 - e^2)N + h\big)\sin\phi$$

המודל הגיאומטרי הסטנדרטי לצורת כדה״א, בו השתמשנו, הוא wgs84 והוא מכיל בתוכו את הגדלים a ו-e, ומהם נגזר הגודל N.

על מנת לעבור שנוכל להביע את מיקום המטרה במטרים ביחס למיקומנו ביצעו המרה נוספת ל-ENU.

ENU (East North Up) היא מערכת קואורדינטות לוקאלית בעלת מישור שמשיק לכדה״א. ראשיתה נמצאת בנקודת ייחוס מסוימת על פני כדה״א, וציריה מקיימים את האוריינטציה הבאה :

- ציר ה-E מצביע לכיוון מזרח
- ציר ה-N מצביע לכיוון צפון
- ציר ה-U מצביע כלפי מעלה, אנכית לפני כדה״א



איור 4: מערכת קואורדינטות ENU

בהינתן :

- ϕ_0 – זווית ה-Lat של קואורדינטת הייחוס במערכת גאודזית
- λ_0 – זווית ה-Long של קואורדינטת הייחוס במערכת גאודזית
- h_0 – גובה קואורדינטת הייחוס במערכת גאודזית
- (X_0, Y_0, Z_0) – קואורדינטת הייחוס (הראשית) במערכת ECEF
- (X, Y, Z) – קואורדינטת המיקום אותה אנו רוצים להמיר ל-ENU, במערכת ECEF

והגדלים $\Delta X, \Delta Y, \Delta Z$ המוגדרים כך:

$$\Delta X = X - X_0 \tag{33}$$

$$\Delta Y = Y - Y_0 \tag{34}$$

$$\Delta Z = Z - Z_0 \tag{35}$$

מחושבת מטריצת סיבוב באופן הבא –

$$R = \begin{pmatrix} -\sin\lambda_0 & \cos\lambda_0 & 0 \\ -\sin\phi_0\cos\lambda_0 & -\sin\phi_0\sin\lambda_0 & \cos\phi_0 \\ \cos\phi_0\cos\lambda_0 & \cos\phi_0\sin\lambda_0 & \sin\phi_0 \end{pmatrix} \tag{36}$$

ומופעלת כך שמתקיים

$$(E \quad N \quad U) = R\begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix} \tag{37}$$

3. תכנון וסימולציה

3.1. יישום

עיקרו של השלב הראשון בפרויקט הוא יישום סימולציה בסביבת Simulink שאפשרה לנו להריץ תרחישים שונים וכך לבחון את נכונות האלגוריתמים ואת יעילותם בטרם ביצוע ניסויים חיים.

3.1.1. נתון מיקום עצמי

בלוק המדמה קבלת נתוני מיקום עצמי אשר מתעדכן עם תנועת הכלי, לשם כך נגדיר:

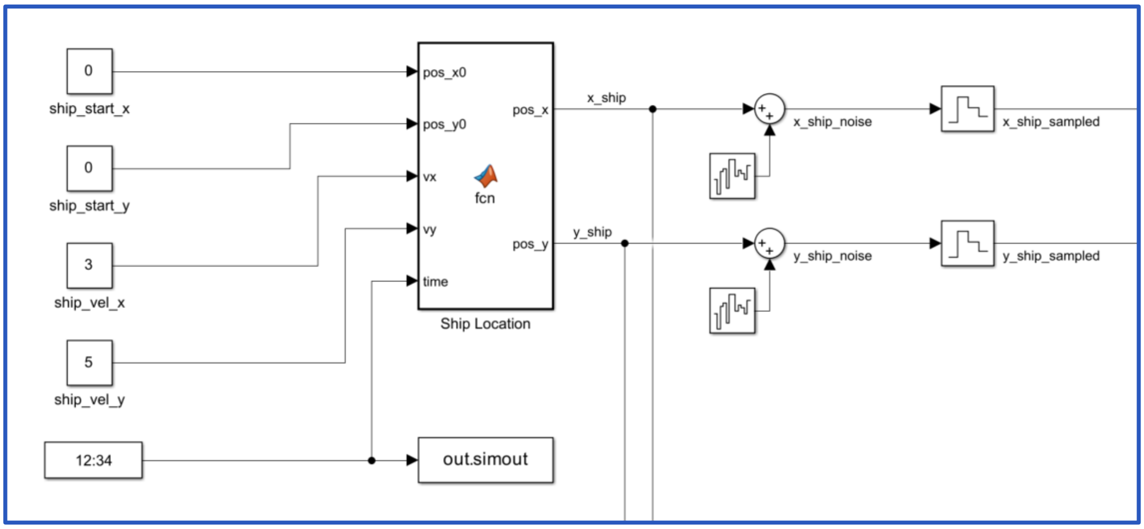
- x_0 – המיקום ההתחלתי שלנו בציר x
- v_x – מהירות בציר x
- y_0 – המיקום ההתחלתי שלנו בציר y
- v_y – מהירות בציר y
- t – זמן שעון הסימולציה

נתונים אלו הוזנו למשוואות קינמטיקה כך שהתקבל מיקומנו המדומה (x_i, y_i):

$$\boldsymbol{x_i = x_0 + v_x t} \tag{38}$$

$$\boldsymbol{y_i = y_0 + v_y t} \tag{39}$$

למוצא הבלוק, שהוא המיקום העצמי שלנו, הוספנו רעש לבן, שמדמה את שגיאת המיקום כתוצאה מחוסר דיוק ה-GPS. המוצא המורעש בכל אחד מהצירים נדגם ב-ZOH על מנת לדמות דגימה בדידה כפי שתתקבל בפועל, כפי שניתן לראות באיור 5:



איור 5: דיאגרמת Simulink – חלק דימוי המיקום העצמי עבור תרחישי סימולציה

3.1.2. <u>נתון זווית למטרה</u>

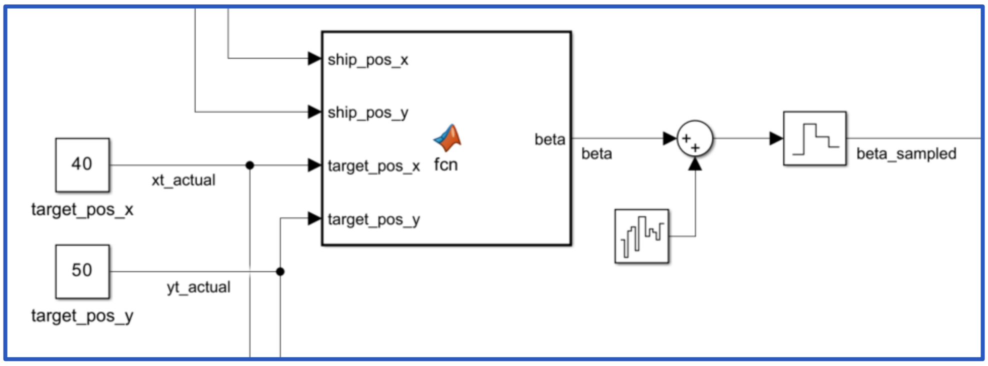
בלוק אשר מדמה את נתון הזווית היחסית למטרה, המתעדכנת עם תנועתנו. הבלוק מוזן על ידי הנתונים הבאים:

- x_i – מיקום עצמי בציר x
- y_i – מיקום עצמי בציר y
- x_{target_actual} – מיקום המטרה המדומה בציר x
- y_{target_actual} – מיקום המטרה המדומה בציר y

נתונים אלו הוזנו למשוואה הטריגונומטרית המתארת את הקשר בין המרחק למטרה והזווית, כך שהתקבלה הזווית למטרה המדומה:

$$\beta = \arctan\left(\frac{y_{target\_actual} - y_i}{x_{target\_actual} - x_i}\right) \tag{40}$$

גם למוצא בלוק זה – הזווית המדומה בנינו לבין המטרה – הוספנו רעש לבן, וגם הוא נדגם ב-ZOH.

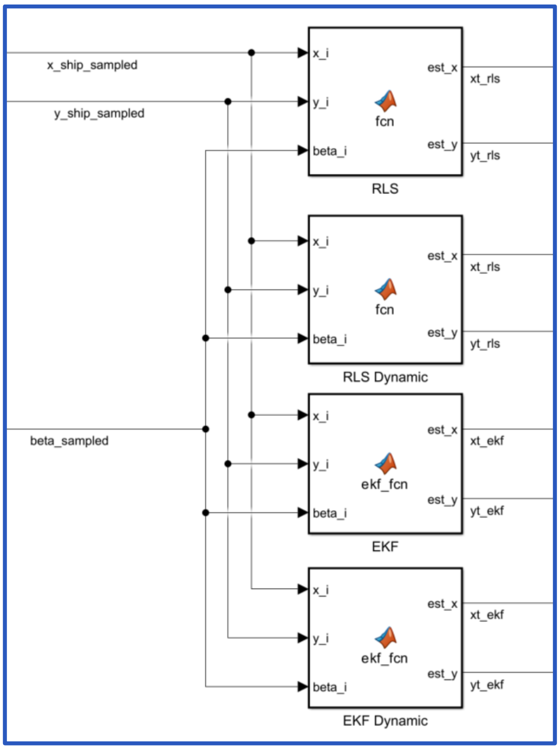


איור 6: דיאגרמת Simulink – חלק דימוי הזווית למטרה עבור תרחישי סימולציה

אם כן, בסה"כ יצרנו 3 נתונים מדומים:

- x_i – מיקום עצמי בציר x
- y_i – מיקום עצמי בציר y
- β – זווית בין המיקום העצמי למטרה

אלו הם קלטי האלגוריתמים, לכן את שלושתם הוזנו לארבעה בלוקים – עבור מטרה סטטית ועבור מטרה דינאמית – לכל אחד משני האלגוריתמים כפי שניתן לראות באיור 7:



איור 7: דיאגרמת Simulink – בלוקי מימוש האלגוריתמים

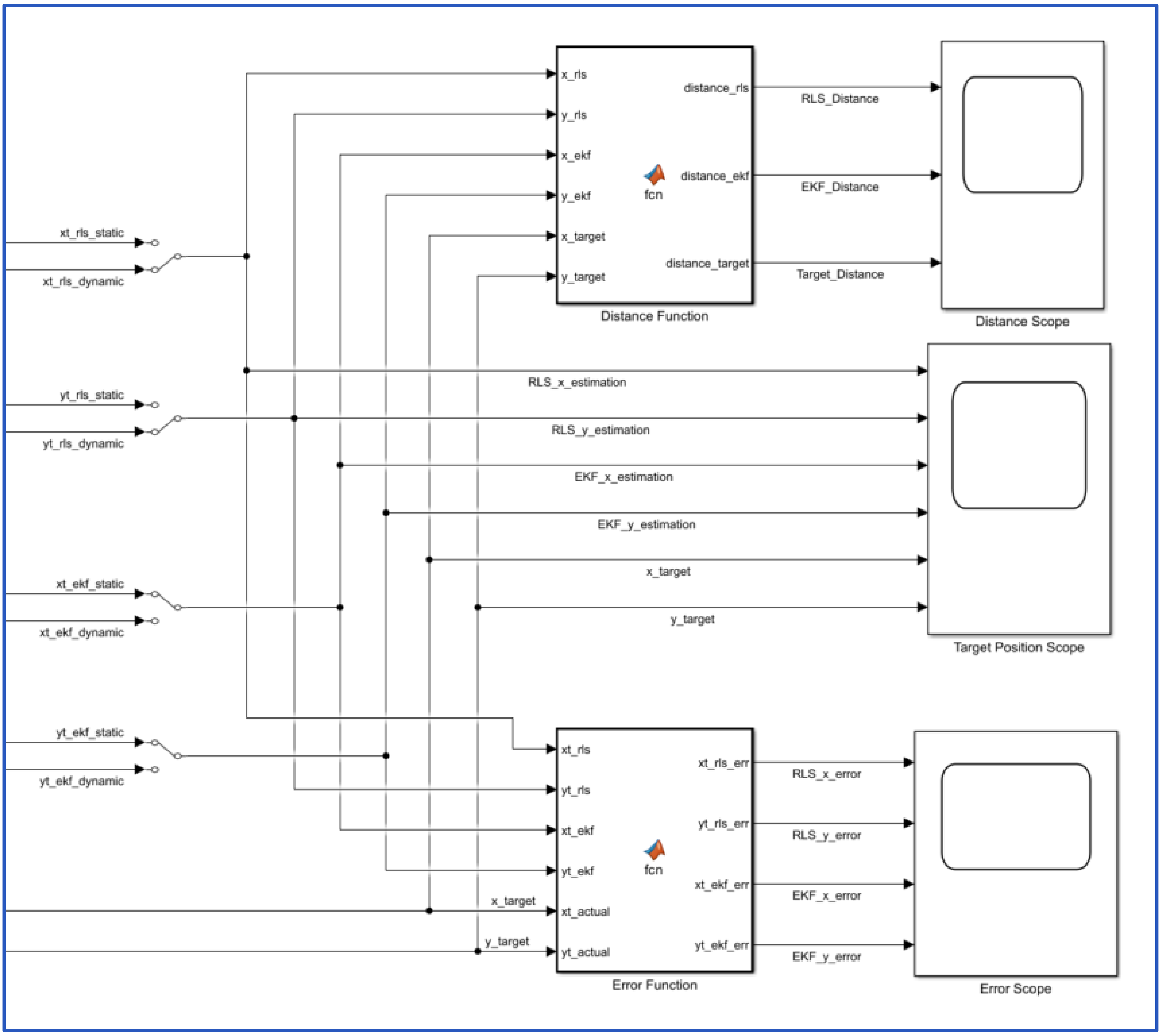
את תוצאות האלגוריתמים השונים הצגנו ב-2 סקופים בצורה הבאה:

- <u>סקופ מיקום –</u> מתאר את שערוך מרחק המטרה המדומה מהראשית בכל אחד מהצירים – x ו-y – בנפרד, ביחידות של מטר.
- <u>סקופ טווח –</u> המתאר את שערוך מרחק המטרה בערך מוחלט מהראשית, ביחידות של מטר, לפי הנוסחה הבאה למרחק:

(41)

$$d = \sqrt{x^2 + y^2}$$

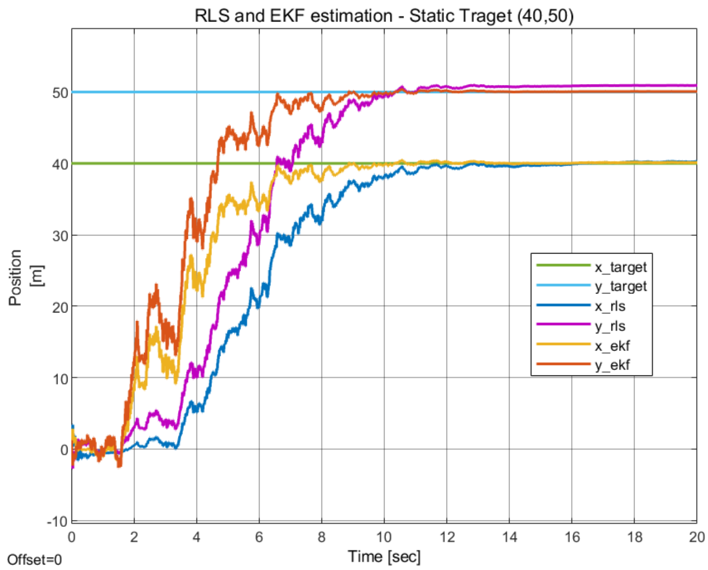
נוסף על שני הסקופים בהן הוצגו התוצאות, סקופ נוסף הציג את השגיאה בשערוך הטווח למטרה באחוזים בכל אחד מהצירים – x ו-y – ביחס למיקום האמיתי הידוע של המטרה.



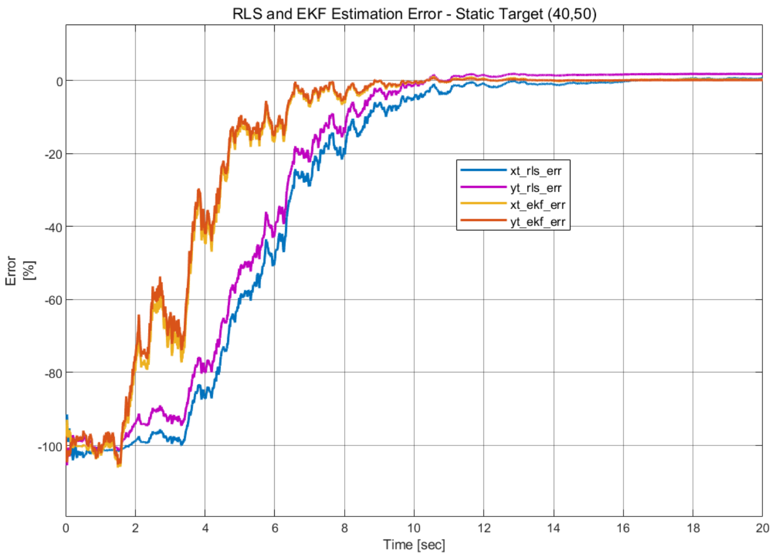
איור 8: דיאגרמת Simulink – חישוב מרחק, תצוגת תוצאות ושגיאות

3.2. תוצאות הסימולציה

3.2.1. תוצאות הסימולציה עבור מטרה סטטית:

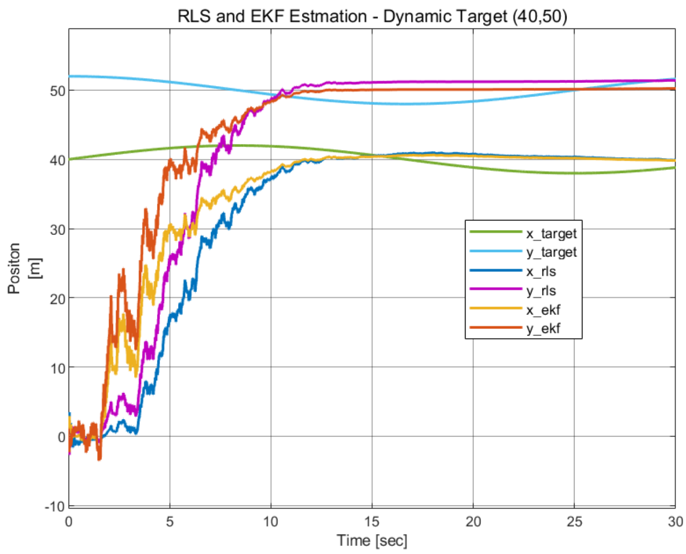


איור 9: מיקום המטרה האמיתי בציר x ובציר y, שערוך מיקום המטרה בציר x ובציר y עבור האלגוריתמים RLS ו-EKF, עבור מטרה סטטית

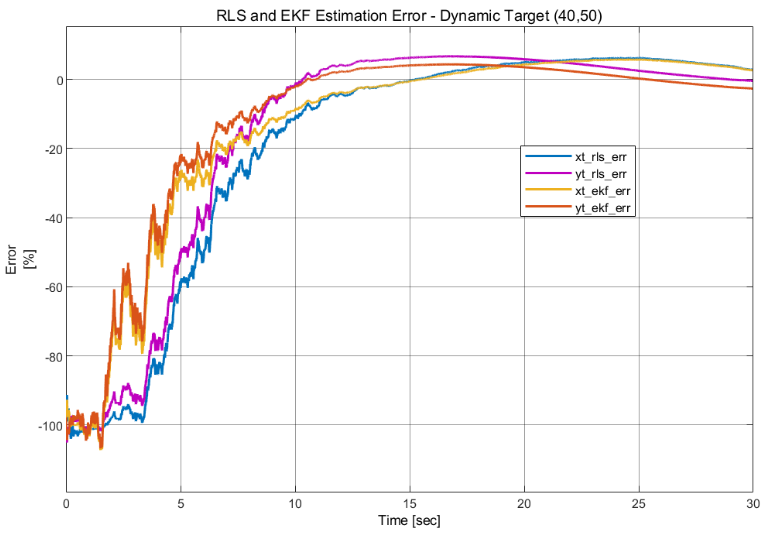


איור 10: שגיאת שערוך מיקום המטרה בציר x ובציר y עבור האלגוריתמים RLS ו-EKF, עבור מטרה סטטית

3.2.2. תוצאות הסימולציה עבור מטרה דינאמית:



איור 12: מיקום המטרה האמיתי בציר x ובציר y, שערוך מיקום המטרה בציר x ובציר y עבור האלגוריתמים RLS ו-EKF, עבור מטרה דינאמית



איור 11: שגיאת שערוך מיקום המטרה בציר x ובציר y עבור האלגוריתמים RLS ו-EKF, עבור מטרה דינאמית

3.2.3. תרחיש הסימולציה:

- המיקום ההתחלתי של הכלי המודד (0,0) במהירות של $v_x = 3\left[\frac{m}{sec}\right]$, $v_y = 5\left[\frac{m}{sec}\right]$.
- מיקום המטרה הסטטית: $(x_{target}, y_{target}) = (40{,}50)\ [m]$.
- המטרה הדינאמית נעה במעגל ברדיוס של 2 מטר כאשר מרכז הנקודה היה $(40{,}50)\ [m]$.

ניתן לראות מאיורים 9 ו-11 שתוצאות שני האלגוריתמים, הן עבור המטרה הסטטית והן עבור המטרה הדינאמית, מתכנסות לשגיאה נמוכה מ-10% תוך פחות מ-10 שניות, עם זאת ברור שאלגוריתם ה-EKF מספק תוצאות טובות יותר שמתבטאות בהתכנסות מהירה יותר לערך האמיתי, לעומת ה-RLS שמתכנס לאט יותר ושומר על שגיאת מצב מתמיד שהיא אמנם נמוכה, אך קיימת.

לסיכום ביניים, פרק הסימולציה אימת את נכונות שני האלגוריתמים וסיפק רמת סמך גבוהה לכך שהם יספקו את התוצאות הרצויות בשלב המעשי של הפרויקט.

4. מימוש

בכדי לממש בפועל את המערכת המתוארת באיור 2 היה עלינו לממש איסוף והקלטת נתוני מיקום עצמי שלנו ואת הזווית למטרה, לאורך המסלול שביצענו. המכשיר ששימש אותנו לאיסוף הנתונים היה טלפון נייד שעליו מותקנת אפליקציה אשר מקליטה את הנתונים שלאחר מכן מוזנים למודל לקבלת שערוך המיקום.

המעבר משלב הסימולציה לשלב המימוש המעשי כלל 3 חלקים עיקריים:

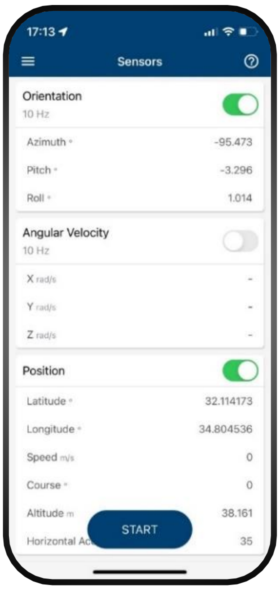
1. הקמת מערך – פיזי ותוכנתי – לאיסוף הנתונים
2. יצירת מבנה תוכנתי להתאמת הנתונים הנאספים לקבלה כקלט האלגוריתמים
3. ביצוע ניסויים בפועל לקבלת תוצאות אמת

נפרט להלן את שלושת שלבים אלו:

4.1. הקמת מערך איסוף הנתונים

4.1.1. הקמת מערך תוכנתי:

4.1.1.1. שימוש באפליקציית MATLAB Mobile לשמירת הנתונים הנקלטים בזמן הניסוי –
אפליקציה זו מאפשרת קריאת נתונים מחיישני המכשיר, הקלטתם לפרק זמן מוגדר, ושמירתם כקובץ mat אשר ניתן לייבא לסביבת MATLAB.



איור 13: מסך הסנסורים באפליקציית MATLAB Mobile, נתוני מיקום ואוריינטציה מאופשרים

4.1.1.2. קריאת הנתונים בסביבת MATLAB –
את הנתונים שהקלטנו באפליקציה טענו ל-MATLAB דרך MATLAB Drive ופתחנו אותם כך שיישמרו ב-workspace.

4.1.2. <u>כלים פיזיים:</u>

בכדי לקבל נתון זווית יציב ומדויק למטרה, בחרנו להשתמש בשיטת עקיבה אופטית ולהיעזר באמצעי עקיבה מיוצב, המייצב הינו "OSMO Mobile 6" של חברת DJI.

איור 14: מייצב גימבל, DJI OSMO Mobile 6

4.2. <u>התאמת הנתונים הנאספים לקבלה כקלט האלגוריתמים</u>

בשלב הפרויקט הראשון, הנתונים שהוזנו לאלגוריתמים היו ביחידות המתאימות למערכת צירים לוקאלית, והוזרקו אחד אחרי השני בקצב שנקבע ע"י שעון פנימי. לעומת זאת, הנתונים שנאספו בפועל נטענים תחילה כמטריצה, שמכילה בנוסף נתונים מיותרים, הם בפורמט גאודזי המתאים למערכת צירים גלובאלית ועם חותמת זמן שונה. לאור כל זאת, נדרשות מספר התאמות על מנת שהנתונים המוקלטים יתקבלו כקלט ע"י בלוקי האלגוריתמים:

4.2.1. <u>המרה ממערכת קואורדינטות גאודזית למערכת קואורדינטות ENU</u> –

כפי שפורט בפרק הרקע התיאורטי[1], ביצענו המרה מקואורדינטות (lat, long, alt) לקואורדינטות לוקאליות (east, north, up). המעבר בוצע בעזרת פונקציות מובנות של MATLAB כאשר נקודת הראשית במערכת הוגדרה כנקודת תחילת התנועה העצמית שלנו. בנוסף, המרנו גם את מיקום המטרה האמיתי, וזאת לשם חישוב השגיאה בהמשך.

4.2.2. <u>התאמת הזווית מ-Azimuth לזווית ביחס לציר ה-East</u> –

משום שחיישן הזווית ששימש אותנו בטלפון הנייד הוא בעצם המצפן, נתון הזווית שהתקבל הוא ה-Azimuth. כלומר, הזווית שקיבלנו היא הזווית ביחס לצפון אליה הצביע הטלפון. עם זאת, הזווית הנדרשת לחישובים היא הזווית ביחס לציר ה-East, כמתואר באיור 2. ההמרה בוצעה לפי הנוסחה הבאה:

$$\beta = 90 - bearing \tag{42}$$

4.2.3. <u>התאמת טבלת הנתונים לפורמט המותאם ל-Simulink</u> –

הנתונים מוקלטים כמטריצה, בפורמט mat, שמכילה בין היתר נתונים שאין לנו בהם שימוש. על מנת שמודל ה-Simulink יוכל לעבד את נתוני הטבלה נדרש היה להמיר אותה לפורמט timetable.

4.3. <u>ביצוע ניסויים בפועל</u>

ביצענו מספר מסלולים שונים, הן עבור מטרות סטטיות והן מטרות דינאמיות, בשלושה מועדים, בניהם ביצענו הסקת מסקנות והפקת לקחים בשאיפה לשיפור התוצאות.

[1] סעיף 2.4

במשך הניסויים נתקלנו בקושי במדידת זווית מדויקת – על אף העקיבה האופטית שהתאפשרה בזכות הייצוב המכאני, מספר ניסויים הניבו תוצאות מדידות זווית שלא התכתבו עם המציאות. קיבלנו תמונה ברורה של המצב בזכות השוואה בין גרף הזווית המדודה לגרף הזווית שחושבה בחישוב נאיבי עפ״י הנוסחה

(43)

$$\beta = \arctan\left(\frac{y_{target\_actual} - y_i}{x_{target\_actual} - x_i}\right)$$

כאשר זיהינו הפרש ניכר בין הגרפים הסקנו שמדובר בשגיאת העוקב או לחלופין בשגיאת החיישן שייתכן שנבעה מקרבה למגנט על המייצב.

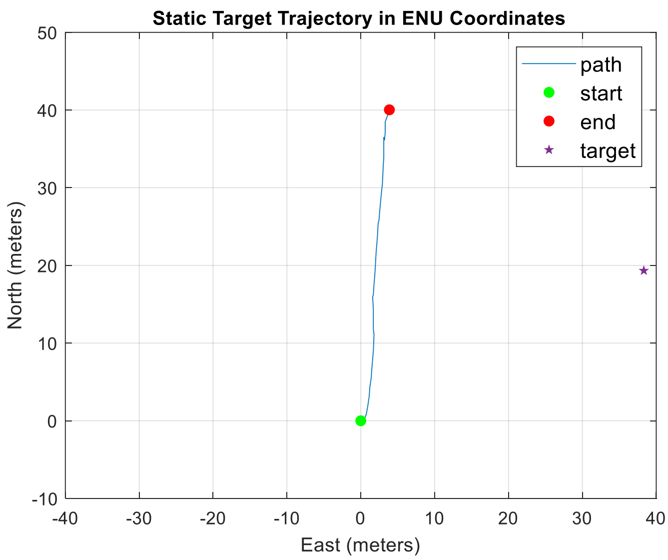
5. ניתוח תוצאות

במסגרת שלב הניסויים ביצענו 4 מסלולים לשערוך 2 מטרות סטטיות שונות ו-2 מסלולים לשערוך מטרה דינאמית. עבור כל סוג מטרה נציג מסלול מייצג שביצענו ביחס למטרה הספציפית ועבור כל אחד מהאלגוריתמים – את שערוך הטווח שלה, את השגיאה בטווח המטרה ואת מספר הדגימות הנחוץ להתכנסות הפתרון.

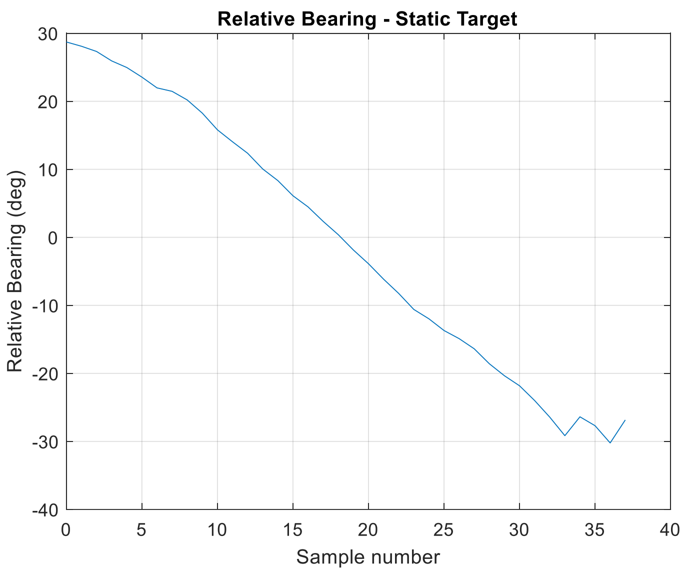
5.1. מטרה סטטית

- בוצע מסלול בקו ישר מכיוון דרום לצפון במרחק כולל של כ-40 מטרים ובמהירות של כ $1.6\ [\frac{m}{sec}]$.
- המטרה הסטטית מוקמה בנק׳ $(37,19)\ [m]$, במרחק של 41.593 מטרים מנקודת התחלת המדידות.

הגרפים הבאים מתארים את המסלול שבוצע ביחס למיקום המטרה (איור 15) ואת השתנות הזווית היחסית למטרה לאורך המסלול (איור 16).

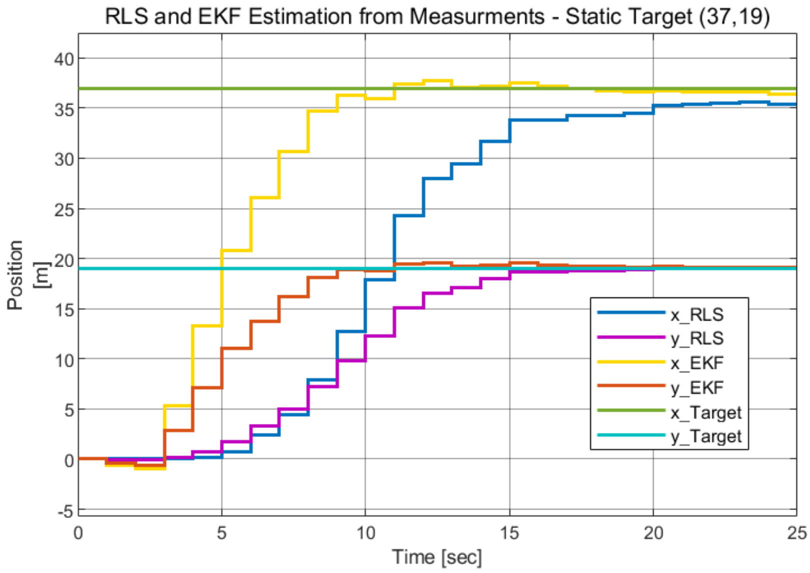


איור 15: מסלול תנועת המודד עבור ניסוי מטרה סטטית

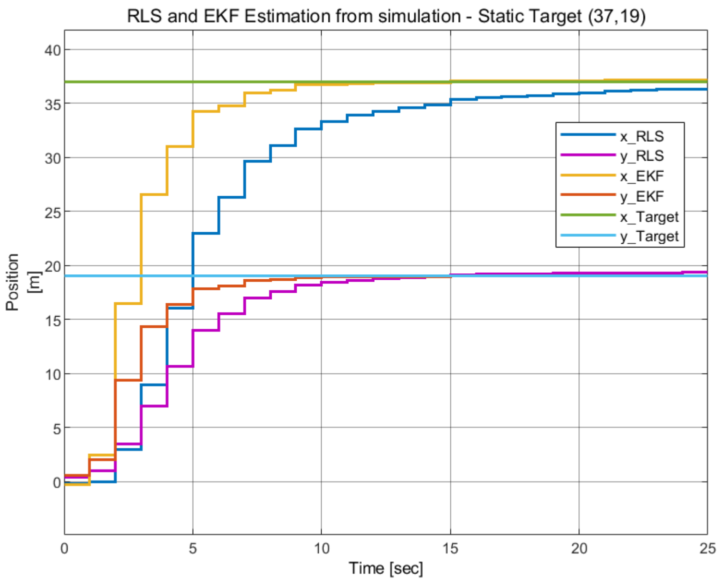


איור 16: זווית יחסית למטרה שהתקבלה ממדידות במהלך ניסוי מטרה סטטית

תוצאות המדידות שהתקבלו במוצא מודל ה-Simulink מתוארות בגרפים הבאים:



איור 17: תוצאות שערוך המיקום עבור האלגוריתמים RLS ו-EKF, עבור כל ציר בנפרד, מיקום המטרה האמיתית, עבור מטרה סטטית

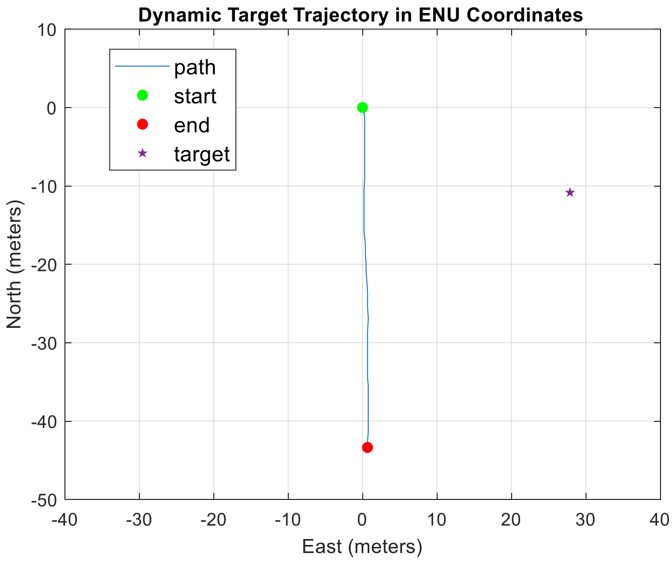


איור 18: תוצאות **סימולציית** שערוך המיקום עבור האלגוריתמים RLS ו-EKF, מיקום המטרה האמיתי, עבור מטרה סטטית

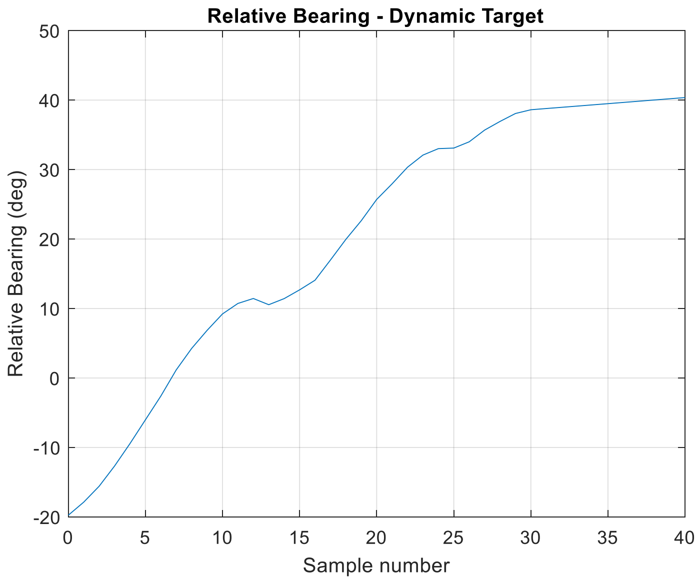
5.2. <u>מטרה דינאמית</u>

- בוצע מסלול בקו ישר מכיוון צפון לדרום במרחק כולל של כ-45 מטרים ובמהירות של כ $1.6\ [\frac{m}{sec}]$.
- המטרה ביצעה מסלול מעגלי ברדיוס של 1 מטר סביב הנקודה $(28.5,-8.5)\ [m]$, במרחק של 29.74 מטרים מנקודת התחלת המדידות.

הגרפים הבאים מתארים את המסלול שבוצע ביחס למיקום המטרה (איור 19) ואת השתנות הזווית היחסית למטרה לאורך המסלול (איור 20).

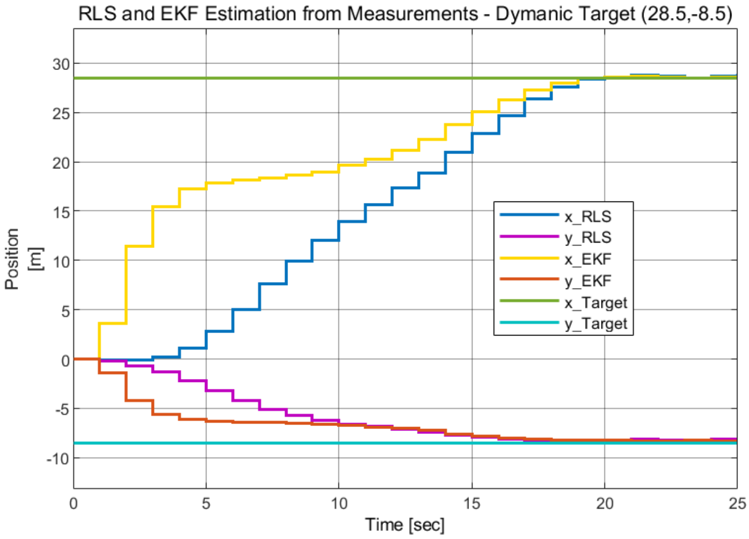


איור 19: מסלול תנועת המודד עבור ניסוי מטרה דינאמית

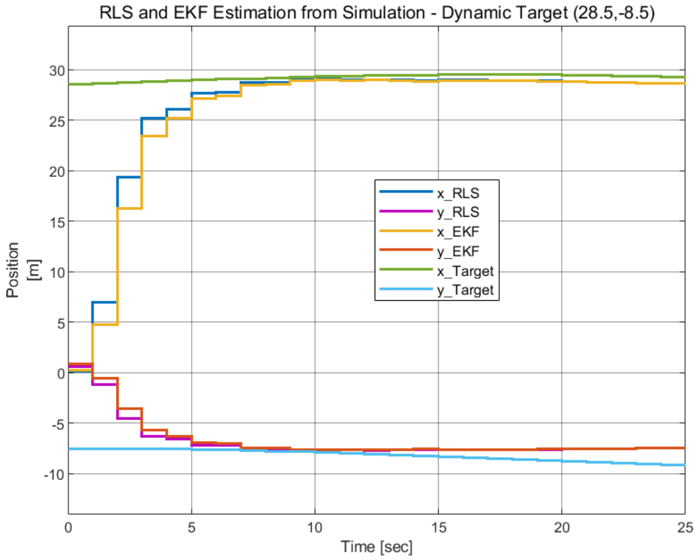


איור 20: זווית יחסית למטרה שהתקבלה ממדידות במהלך ניסוי מטרה דינאמית

תוצאות המדידות שהתקבלו במוצא מודל הSimulink מתוארות בגרפים הבאים:



איור 21: תוצאות שערוך המיקום עבור האלגוריתמים RLS ו-EKF, עבור כל ציר בנפרד, מיקום המטרה האמיתית, עבור מטרה דינאמית



איור 22: תוצאות **סימולציית** שערוך המיקום עבור האלגוריתמים RLS ו-EKF, מיקום המטרה האמיתי, עבור מטרה דינאמית

5.3. השוואות בין תוצאות הסימולציה לביצועי המערכת בשטח – מטרה סטטית

נרצה להשוות את תוצאות המדידות בשטח עבור מטרה סטטית (איור 17) אל מול תוצאות הסימולציה (איור 18) עבור תרחיש זהה.

נתעניין בעיקר ב-3 פרמטרים בכדי לקבוע את טיב שערוך הטווח למטרה עבור כל אלגוריתם

1. **זמן התכנסות** – נקבע כזמן שחלף מתחילת המדידות ועד שיערוך טווח בדיוק של 10% מהטווח האמיתי של המטרה הסטטית.
2. **שיערוך טווח** – נקבע כשיערוך הטווח שנותן כל אלגוריתם לאחר 20 דגימות, זהו זמן מספיק עבור התייצבות המשערכים.
3. **שגיאת השערוך** – השגיאה באחוזים בין ערך השערוך (פרמטר 2) של כל אלגוריתם ובין הטווח האמיתי של המטרה, ביחס לראשית הצירים.

המדידות נלקחו מתוך הגרפים הרלוונטיים ומתוכן מילאנו את הטבלה הבאה:

	פרמטרים	**סימולציה מטרה סטטית**		**מדידות בשטח מטרה סטטית**	
		RLS	**EKF**	**RLS**	**EKF**
.1	**זמן התכנסות לפתרון (עד 10% שגיאה בטווח) [sec]**	10	5	15	8
.2	**מרחק לאחר 20 דגימות [m]**	40.810	41.665	40.030	41.446
.3	**שגיאה מקסימלית במיקום [%]**	1.88	0.17	3.76	0.35

טבלה 1: השוואת תוצאות סימולציה לתוצאות אמת – מטרה סטטית

מהטבלה ניתן להסיק כי עמדנו בכלל יעדי הפרויקט עבור זמני ההתכנסות ושגיאת השערוך, נרחיב על המסקנות מתוצאות אלו בפרק הרלוונטי (פרק מס 6.1).

5.4. השוואות בין תוצאות הסימולציה לביצועי המערכת בשטח – מטרה דינאמית

נרצה להשוות את תוצאות המדידות בשטח עבור מטרה דינאמית (איור 21) אל מול תוצאות הסימולציה (איור 22) עבור תרחיש זהה.

נתעניין בעיקר ב-3 פרמטרים בכדי לקבוע את טיב שערוך הטווח למטרה עבור כל אלגוריתם:

1. **זמן התכנסות** – נקבע כזמן שחלף מתחילת המדידות ועד שיערוך טווח בדיוק של 10% מהטווח האמיתי של המטרה הסטטית.
2. **שיערוך טווח** – נקבע כשיערוך הטווח שנותן כל אלגוריתם לאחר 20 דגימות, זהו זמן מספיק עבור התייצבות המשערכים.
3. **שגיאת השערוך** – השגיאה באחוזים בין ערך השערוך (פרמטר 2) של כל אלגוריתם ובין הטווח האמיתי של המטרה, ביחס לראשית הצירים.

המדידות נלקחו מתוך הגרפים הרלוונטיים ומתוכן מילאנו את הטבלה הבאה:

	פרמטרים	סימולציה מטרה דינאמית		מדידות בשטח מטרה דינאמית	
		RLS	EKF	RLS	EKF
.1	**זמן התכנסות לפתרון (עד 10% שגיאה בטווח) [sec]**	5	5	17	16
.2	**מרחק לאחר 20 דגימות [m]**	29.754	29.754	29.741	29.741
.3	**שגיאה מקסימלית במיקום [%]**	3.21	3.21	3.26	3.26

טבלה 2: השוואת תוצאות סימולציה לתוצאות אמת – מטרה דינאמית

מהטבלה ניתן להסיק כי עמדנו בכלל יעדי הפרויקט עבור זמני ההתכנסות ושגיאת השערוך, נרחיב על המסקנות מתוצאות אלו בסעיף 7.1.

6. סיכום, מסקנות והצעות להמשך

6.1. סיכום ובחינת התוצאות

בפרויקט זה חקרנו ויישמנו אלגוריתמים שנמצאים בליבת הטכנולוגיה בעזרתם ביצענו שיערוך מיקום למטרות סטטיות ודינאמיות באמצעים פאסיביים שאינם טריוויאליים. בנינו מודלים מתמטיים המתארים את האלגוריתמים הללו ובעזרתם ביצענו סימולציות שתוצאותיהן הוכיחו היתכנות לטובת המשך הפרויקט. לאחר מכן, ביצענו סדרת ניסויי שטח בהם עקבנו עקיבה אופטית אחרי מטרות שונות במסלולי תנועה עצמית שונים, ואת הנתונים שאספנו הזנו כקלט לאלגוריתמי השערוך. השוונו בין תוצאות המשערכים ובהתבוננות בתוצאות שקיבלנו, המוצגות לעיל – ניתן לראות שהשלמנו את מטרת הפרויקט ועמדנו בכלל המדדים שהגדרנו בתחילתו:

	פרמטרים	מטרה סטטית		מטרה דינאמית		עמידה ביעד
		RLS	EKF	RLS	EKF	
.1	**זמן התכנסות (עד 10% שגיאה בטווח) [sec]**	15	8	17	16	✓
.2	**מרחק לאחר 20 דגימות [m]**	40.030	41.446	29.741	29.741	✓
.3	**שגיאה בשערוך הטווח למטרה [%]**	3.76	0.35	3.26	3.26	✓

טבלה 3: השוואת תוצאות למדדים הכמותיים

כצפוי, תוצאות האמת נפלו מתוצאות הסימולציות, וזאת משום שבביצוע ניסוי ואיסוף מדידות בפועל אנחנו חשופים להרבה גורמים מרעישים שמגדילים באופן משמעותי את רמת השגיאה, בעוד שהסימולציה חשופה לרעש שאמנם הוא אקראי, אך ניתן להגדירו כ"מבוקר".

6.2. מסקנות

6.2.1. עבור מטרה סטטית קיימת עדיפות משמעותית למשערך ה-EKF: מהשוואה בין שגיאת המצב המתמיד של אלגוריתם ה-EKF – 0.35%, לעומת השגיאה שהתקבלה עבור אלגוריתם ה-RLS – 3.76%, בנוסף זמן

להתכנסות ה-EKF – 8 שניות, לעומת 15 שניות ב-RLS, ניכר שישנה עדיפות משמעותית לאלגוריתם ה-EKF בשערוך מיקום של מטרה סטטית.

6.2.2. עבור מטרה דינאמית קיימת עדיפות ל-EKF, אולם לא מובהקת: מהתוצאות שהתקבלו עבור המטרה הדינאמית אנחנו מבינים שאלגוריתם ה-RLS שכולל את פקטור השכחה מביא לשיפור משמעותי לעומת ה-RLS ללא הפקטור. לעומת זאת, ה-EKF המותאם למטרה דינאמית לא מציג שיפור משמעותי, ככל הנראה בגלל פרמטרים לא מיטביים באתחול מטריצות הקוואריאנס.

6.2.3. בתחילת שלב א' של הפרויקט הבנו שכלל האלגוריתמים שחקרנו לא מספקים התכנסות עבור מטרות דינאמיות שמסלולן אקראי. לכן התמקדנו בתנועה מעגלית סביב נ.צ. קבוע.

6.2.4. השימוש במייצב היה טוב וגרם לשגיאות קטנות ביחס למצופה: המייצב בו השתמשנו מכיל בתוכו מגנט אליו מתחבר הטלפון. מהעובדה שהתוצאות שקיבלנו התכנסו ברמה דומה מאוד לסימולציות הסקנו שהשפעת המגנט כגורם אפשרי לשגיאה הייתה מינורית.

6.2.5. אפליקציית MATLAB Mobile הוכחה כמתאימה במיוחד למטרת הניסויים. השימוש בה אפשר לנו לנצל משאבים קיימים, מה שהביא לעבודה יעילה והתממשקות נוחה ומהירה לסנסורים. באופן דומה, שימוש בפונקציות מובנות של MATLAB, בדגש על המרת יחידות והתאמת הנתונים.

6.3. הצעות להמשך

6.3.1. ביצוע שיערוך בזמן אמת: עיבוד הנתונים שאנחנו ביצענו התבצע ב-offline, לאחר ביצוע הניסוי ולא במהלכו. ביצוע העיבוד בזמן אמת (real-time) יאפשר שימוש באלגוריתם בכלים ניידים, בתרחישים מציאותיים. לשם מימוש זה נדרשות מספר התאמות בדיאגרמת ה-Simulink.

6.3.2. שיפור תנאי התחלה: אתחול מטריצות הקוואריאנס והרעש באופן שמותאם ספציפית לרכיבי המערכת יכול לשפר משמעותית את קצבי ההתכנסות ותוצאות השערוך. אנחנו ביצענו אתחול משוערך למטריצות אלה אולם ניתן לחקור את מערכת המדידה יותר לעומק ולהסיק ערכים מדויקים יותר שמתארים את המצב באופן מדויק יותר.

6.3.3. ביצוע סטטיסטיקה על סדרות ניסויים: בביצוע סדרות של מספר ניסויים עם מערכים זהים (מסלול מס' 1 ומטרה מס' 1 – ביצוע מספר פעמים, מסלול מס' 2 ומטרה מס' 2 – ביצוע מספר פעמים וכן הלאה) ניתן לאסוף די נתונים כך שיתאפשר ביצוע סטטיסטיקה עבור תוצאות של כל מסלול, וכן עבור התוצאות הכלליות של השערוך. ביצוע סטטיסטיקה כמובן נותן לנו תוצאות ברמת סמך טובה יותר שכן הן מבוססות על מספר גדול יותר של דגימות.

6.3.4. שיפור מיקום ביצוע הניסויים: ברמה הפרקטית, עקיבה אופטית אחרי מטרה רחוקה נתונה להפרעות רבות באזור הומה אדם. על מנת לייעל את תהליך איסוף הנתונים מומלץ לבצע את הניסויים במקום עם מינימום הפרעות דינאמיות. בנוסף, היות שנתון הזווית מתקבל ממצפן, מומלץ למזער קרבה להפרעות מגנטיות.

7. תיעוד הפרויקט

הפרויקט תועד במלואו ב-GitHub תחת הקישור הבא:
https://github.com/NoyKirshner/Target-Localization-Final-Project

התיעוד כולל את הרכיבים הבאים:

1. שני קבצי מידע גולמי, אחד עבור מדידות מטרה סטטית והשני עבור מדידות מטרה דינאמית:
   - sensorlog_20240704_142451_static_target.mat – עבור מדידות מטרה סטטית
   - sensorlog_20240710_172209_dynamic_target.mat – עבור מדידות מטרה דינאמית
2. קובץ MATLAB: sensorlog_to_simulink.m עבור הכנת המידע הגולמי מהסנסורים לפורמט מתאים עבור מודל ה-Simulink
3. מודל Simulink עבור תרחישי סימולציה: Ship_location_simulation_model_final.slx
4. מודל Simulink עבור נתוני מדידות: ship_location_measurements_model_final.slx
5. קובץ ReadMe הכולל הסבר מפורט על כלל מרכיבי הפרויקט ואופן הרצתם

8. רשימת מקורות

[1] Gabriel A. Terejanu, “Extended Kalman Filter Tutorial”, 2009.

[2] Vincent J. Aidala, “Kalman Filter Behavior in Bearings-only Tracking Applications”, 1979.

[3] Simon J. Julier, Jeffrey K. Uhlmann, “A New Extension of the Kalman Filter to Nonlinear Systems”, Proceedings of the SPIE, Volume 3068, p. 182-193, July 1997.

[4] Matthew B. Rhudy, Roger A. Salguero, Keaton Holappa, “A Kalman Filtering Tutorial for Undergraduate Students”, International Journal of Computer Science & Engineering Survey (IJCSES) Vol.8, No.1, February 2017.

[5] Imanol Iriarte Arrese, "Implementation and Testing of Sigma-Point Kalman Filters in Simulink for Nonlinear Estimation", 2016.

[6] MathWorks, “Comparison of 3-D Coordinate Systems”. https://www.mathworks.com/help/map/choose-a-3-d-coordinate-system.html

[7] MathWorks, “geodetic2ecef. Transform geodetic coordinates to geocentric Earth-centered Earth-fixed”, https://www.mathworks.com/help/map/ref/geodetic2ecef.html

[8] MathWorks, “ecef2enu. Transform geocentric Earth-centered Earth-fixed coordinates to local east-north-up”, https://www.mathworks.com/help/map/ref/ecef2enu.html

[9] “DJI, Osmo Mobile 6, Specs”, https://www.dji.com/uk/osmo-mobile-6/specs