Control Robotics and Machine Learning Laboratory

הטכניון - מכון טכנולוגי לישראל

TECHNION - ISRAEL INSTITUTE OF TECHNOLOGY

הפקולטה להנדסת חשמל

המעבדה לבקרה רובוטיקה ולמידה חישובית

דוח פרויקט : ראייה ממוחשבת

הנושא:

מובילאיי לאופניים

Bicycle Mobileye

מגישים:

רון לבדינסקי
Ron Lebedinsky

אורי גרוס
Ori Gross

מנחה:

קובי כוחיי

סמסטר: אביב 2025

שנה: 2025

תוכן עניינים

תוכן עניינים **2**

רשימת טבלאות ואיורים **4**

תקציר **5**

ABSTRACT **5**

רשימת סמלים וקיצורים **6**

מבוא **6**

רקע ומוטיבציה 6

הגדרות 6

סיכום התרומה העיקרית 8

עבודות קודמות בנושא 8

מבנה העבודה 8

תיאור כללי **9**

תיאור חומרה 9

מצלמת OAK-D Lite *9*

כרטיס בקרה Raspberry Pi 4B *10*

מערכת שמע עם יכולת חיבור Bluetooth *10*

אינטגרציה של החלקים *11*

תיאור תוכנה 12

מודל ראייה ממוחשבת -YOLO 10n *12*

אלגוריתם לזיהוי והתרעה על סכנה *12*

סקריפטים *13*

תיאור מפורט **14**

אלגוריתם לזיהוי והתרעה על סכנה 14

רציונל ועקרון פעולה *14*

מודל מתמטי *14*

מסנן ROI 14

לוגיקת מימוש *15*

מסנן קלמן 16

הגדרת וקטור המצב ווקטור המדידה *16*

שלב החיזוי (Prediction) *16*

שלב העדכון (Update/Correction) *17*
DEPLOY_MODEL.PY 17
ממשק התקשרות עם מצלמת OAK-D Lite *17*
רכיבי המעקב וההתרעה *18*
סיכום ומסקנות **19**
ניתוח תוצאות 19
אימון מודל *19*
הרצה בצורה סינטטית *20*
הרצה בזמן אמת *21*
מסקנות כלליות 22
גישות נוספות לפתרון 22
אפשרויות להמשך הפיתוח 22
נספחים **23**
אתחול והרצה 23
ספציפית להרצה על raspberry pi *23*
רשימת מקורות **24**

רשימת טבלאות ואיורים

איור 1 - תיאור בלוקים של המערכת.....9
איור 2 - מצלמת OAK D-Lite.....9
איור 3 - כרטיס בקרה Raspberry Pi 4B.....10
איור 4 - מערכת שמע Bluetooth.....10
איור 5 - המארז בתוך תוכנית CAD.....11
איור 6 - מארז מורכב ומחובר לכיסא אופניים.....11
איור 7 – לוגו Ultralytics YOLO.....12
איור 8 – כותרת וכותבי המאמר.....12
איור 9 – תקציר המאמר.....13
איור 10- המחשה של ROI.....15
איור 11 - תוצאות האימון.....19

תקציר

מטרתו העיקרית של פרויקט זה היא לצמצם את המספר המשמעותי של תאונות אופניים הנגרמות כתוצאה מחוסר יכולתו של הרוכב להבחין בכלי רכב המתקרבים מאחור. כדי לתת מענה לסוגיה זו, פיתחנו מערכת שנועדה להגביר את מודעות הרוכב לסכנות פוטנציאליות בכביש.
בהשראת מערכת "מובילאיי" למניעת התנגשויות, הנפוצה כיום בכלי רכב מודרניים, מערכת "מובילאיי לאופניים" שפיתחנו מספקת לרוכב מידע חיוני בזמן אמת על כלי רכב המתקרבים אליו מאחור. ההתרעות שהמערכת מפיקה מעניקות לרוכב זמן תגובה נוסף ומאפשרות לו לנקוט באמצעים למניעת תאונה.
ארכיטקטורת המערכת מבוססת על ראייה ממוחשבת, ומשלבת מודל YOLO מתקדם לזיהוי אובייקטים. המודל רץ על גבי מצלמת OAK-D Lite, המבצעת את העיבוד ישירות על החומרה (on-device processing). כרטיס Raspberry Pi משמש כיחידת הבקרה המרכזית, ועליו רץ אלגוריתם מתוחכם לזיהוי סכנות, אשר פותח בהתבסס על מחקר של חברת "מובילאיי". בקר זה מנהל גם מסד נתונים דינמי המתעד את זיהויי כלי הרכב, ומאפשר למערכת להעריך רמות סיכון בהתבסס על שילוב של מידע היסטורי ונתונים חדשים המתקבלים בזמן אמת. התרעות סכנה מועברות לרוכב באמצעות התקן שמע אלחוטי בטכנולוגיית Bluetooth. המערכת כולה תוכננה להיות ניידת ולפעול באופן עצמאי לחלוטין, ללא צורך בחיבור לרשת (offline).
תוצאות הפרויקט מוכיחות כי ניתן לשפר באופן משמעותי את המודעות הסביבתית של רוכב האופניים באמצעות מערכת מסוג זה. עם זאת, תהליך הפיתוח חשף אתגרים רבים אשר מהווים הזדמנויות למחקר עתידי ולשיפורים והרחבות של המערכת.

Abstract

The primary objective of this project is to mitigate the significant number of bicycle accidents that occur due to the cyclist's inability to detect vehicles approaching from the rear. To address this, we have developed a system designed to heighten a cyclist's awareness of potential road hazards. Drawing inspiration from the commonly used Mobileye collision avoidance system found in modern automobiles, our "Mobileye for Bicycles" provides the rider with crucial, real-time information about vehicles approaching from behind. The alerts generated by the system afford the cyclist additional time to react and implement measures to prevent an accident.
The system's architecture is based on computer vision, integrating a YOLO model for object detection. This model is deployed on an OAK-D Lite camera, which handles on-device processing. A Raspberry Pi serves as the central control unit, running a sophisticated danger identification algorithm adapted from research by Mobileye. This controller also manages a dynamic database that logs vehicle detections, enabling the system to assess threats based on both historical and incoming data. Danger alerts are communicated to the cyclist through a wireless Bluetooth audio device. The entire system is designed to be portable and operate in a fully offline environment.
The results of this project confirm that it is possible to significantly improve a cyclist's situational awareness through such a system. Nevertheless, the development process has identified numerous challenges that present opportunities for future research and system enhancement.

רשימת סמלים וקיצורים

YOLO	You Only Look Once
OID	Open Images Dataset
DP	Deep Learning
ROI	Region Of Interest
CAD	Computer Assisted Design
VPU	Vision Processing Unit
TTC	Time To Contact
IOU	Intersection over Union
MAP	Mean Average Precision

מבוא

רקע ומוטיבציה

בטיחות בדרכים היא אתגר מתמשך, ובשנים האחרונות אנו עדים למהפכה טכנולוגית בתחום בטיחות הרכב. מערכות עזר מתקדמות לנהג כדוגמת "מובילאיי" הישראלית, הפכו לסטנדרט בכלי רכב מודרניים והוכיחו את יעילותן בהפחתה משמעותית של תאונות דרכים מדי יום. מערכות אלו מעניקות לנהג "עיניים נוספות" ומספקות התרעות קריטיות בזמן אמת המאפשרות מניעת התנגשויות.

בניגוד לנהגים, המוגנים במעטפת מתכת ובטכנולוגיות מצילות חיים, רוכבי האופניים נותרים חשופים ופגיעים במיוחד במרחב התחבורתי. עשרות רוכבי אופניים נהרגים ומאות נפצעים קשה בכבישי ישראל מדי שנה, כאשר איום משמעותי ומתמיד הוא כלי רכב המתקרבים מאחור, מחוץ לשדה הראייה של הרוכב. בעוד שהנהג מקבל התרעות על סכנות, לרוכב האופניים חסר מנגנון דומה שיגשר על פער המודעות הקריטי הזה.

הפער הטכנולוגי הקיים, בין ההגנה הגבוהה הניתנת לנהגים לבין חוסר ההגנה המאפיין רוכבי אופניים, הוא המניע המרכזי לפרויקט זה. מטרתו היא לפתח מערכת התרעה ייעודית שתספק לרוכבים את אותה יכולת קריטית הקיימת במכוניות: זיהוי מוקדם של סכנה מתקרבת מאחור, ובכך להעניק להם את זמן התגובה היקר שעשוי למנוע את התאונה הבאה ולהציל חיים.

הגדרות

- **ראייה ממוחשבת (Computer Vision):** תחום במדעי המחשב ובבינה מלאכותית שמטרתו לאפשר למכונות "לראות" ולהבין מידע ויזואלי, כגון תמונות וסרטונים. המערכות בתחום זה מנתחות פיקסלים בתמונה כדי לזהות אובייקטים, דפוסים ומאפיינים, בדומה לאופן שבו הראייה האנושית פועלת.
- **למידה עמוקה (DP):** תת-תחום של למידת מכונה המבוסס על רשתות נוירונים מלאכותיות עם שכבות רבות (ומכאן השם "עמוקה"). בפרויקט זה, נעשה שימוש במודל מבוסס למידה עמוקה לצורך משימת זיהוי האובייקטים.
- **זיהוי אובייקטים (Object Detection):** משימה מרכזית בראייה ממוחשבת, שבה המטרה היא לא רק לסווג אובייקטים בתמונה (למשל, "קיים רכב"), אלא גם לקבוע את מיקומם המדויק. התוצאה לרוב מוצגת באמצעות "תיבה תוחמת" (Bounding Box) - מלבן המקיף את האובייקט שזוהה.
- **תיבה תוחמת (Bounding Box):** הפלט של מודל זיהוי אובייקטים. זוהי מסגרת מלבנית המקיפה אובייקט שזוהה בתמונה, ומוגדרת בדרך כלל על ידי קואורדינטות הפינה השמאלית-עליונה שלה, יחד עם רוחבה וגובהה.

- **אימון מודל (Model Training):** תהליך "לימוד" של מודל בינה מלאכותית, שבו מוצגות לו דוגמאות רבות מתוך מאגר נתונים מתויג (למשל, תמונות שבהן כל הרכבים סומנו מראש). במהלך האימון, המודל מתאים באופן איטרטיבי את הפרמטרים הפנימיים שלו ("המשקולות") במטרה למזער את שגיאת הזיהוי ולהגיע לרמת הדיוק הנדרשת.
- **העברת עומס חישובי (Off-loading):** פרקטיקה הנדסית שבה משימות חישוב אינטנסיביות מועברות מרכיב עיבוד מרכזי (כמו מעבד ראשי) לרכיב חומרה ייעודי או חיצוני, המסוגל לבצע אותן ביעילות רבה יותר. בפרויקט, חישובי רשת הנוירונים מועברים מה-Raspberry Pi למעבד הראייה הממוחשבת (VPU) המובנה במצלמת ה-OAK-D Lite.
- **YOLO:** שם של משפחת מודלים פופולרית לזיהוי אובייקטים בזמן אמת. ייחודו של המודל הוא ביכולתו לעבד תמונה שלמה במעבר אחד בלבד, ובכך להשיג מהירות זיהוי גבוהה החיונית ליישומים כמו שלנו, הדורשים תגובה מיידית.
- **עיבוד בזמן אמת (Real-time Processing):** יכולתה של מערכת לעבד נתונים ולהגיב אליהם באופן מיידי, ללא השהיות מורגשות. בהקשר של הפרויקט, הכוונה היא ליכולת לצלם פריים, לזהות בו סכנה, ולהתריע לרוכב – כל זאת בשבריר שנייה, מספיק מהר כדי לאפשר תגובה יעילה.
- **זמן פגיעה משוער (TTC):** מתאר את פרק הזמן שנותר עד להתנגשות אפשרית בין הרכב שבו מותקנת המערכת לבין רכב אחר שזוהה, בהנחה ששני הרכבים ימשיכו לנוע במסלולם הנוכחי ובמהירויות הנוכחיות. בפרויקט זה, הערכת ה-TTC מתבצעת בעיקר לפי קצב ההתרחבות של התיבה התוחמת (Bounding Box) של הרכב המתקרב. מהווה מדד מרכזי בהחלטה האם להפעיל התרעת סכנה.
- **מסנן קלמן (Kelman Filter):** אלגוריתם מתמטי המשמש להערכה ומעקב אחר מצבה של מערכת דינמית לאורך זמן, גם כאשר המדידות המתקבלות אינן מדויקות או כוללות רעש. בפרויקט שלנו, המסנן משמש למעקב אחר כלי הרכב שזוהו, החלקת המיקום שלהם, וחיזוי תנועתם העתידית, ובכך משפר את אמינות זיהוי הסכנה.
- **אזור עניין (ROI):** אזור מוגדר מראש בתוך התמונה שבו מתמקד ניתוח המערכת. שימוש ב-ROI מאפשר למערכת להתעלם מחלקים לא רלוונטיים בתמונה (כמו שמיים או צדי הדרך), ובכך מפחית את העומס החישובי ומשפר את דיוק הזיהוי של איומים אמיתיים.
- **OAK-D Lite:** מצלמה חכמה המשלבת חיישן צילום צבעוני עם יחידת עיבוד ראייה. רכיב זה מאפשר להריץ מודלים של בינה מלאכותית ישירות על המצלמה, ובכך לבצע off-loading מהבקר הראשי.
- **Raspberry Pi:** מחשב זעיר-ממדים ודל-הספק, המשמש בפרויקט כיחידת הבקרה המרכזית. הוא אחראי על ניהול כלל רכיבי המערכת: הפעלת המצלמה, קבלת תוצאות הזיהוי, הרצת אלגוריתם הערכת הסיכונים, ושליחת פקודת ההתרעה.
- **קובץ מודל blob.:** קובץ blob. MyriadX הוא ייצוג בינארי שעבר הידור (compilation) ואופטימיזציה של מודל רשת נוירונים. בתהליך ההמרה, מודל שפותח בסביבה סטנדרטית (כמו TensorFlow או PyTorch) מתורגם לפורמט ייעודי המותאם להרצה על חומרת האצה ספציפית. בפרויקט זה, הפורמט מאפשר למודל ה-YOLO לרוץ ביעילות מרבית על יחידת עיבוד הראייה (VPU) המובנית במצלמת ה-OAK-D Lite, תוך ניצול מיטבי של יכולותיה החישוביות.
- **Epoch:** מונח המתאר מעבר מלא אחד של **כלל** מאגר נתוני האימון דרך מודל הרשת הנוירונית. תהליך אימון של מודל מורכב בדרך כלל ממספר רב של epochs. בכל epoch, המודל "רואה" פעם אחת כל דוגמה במאגר הנתונים, ומעדכן את משקולותיו בהדרגה כדי לשפר את ביצועיו. ניתן לדמות זאת לקריאה חוזרת של ספר לימוד מספר פעמים, כאשר בכל קריאה ההבנה של החומר מעמיקה.
- **Batch:** תת-קבוצה של דגימות מתוך מאגר נתוני האימון, המוזנת למודל יחד במעבר חישובי יחיד. מכיוון שלא ניתן להזין את כל מאגר הנתונים העצום למודל בבת אחת עקב מגבלות זיכרון, מחלקים אותו ל-batches. גודל ה-batch הוא היפר-פרמטר הקובע כמה דגימות ייכללו בכל batch. המודל מעבד batch אחת, מעדכן את משקולותיו על סמך השגיאה המחושבת, ורק אז עובר לעבד את ה-batch הבאה. epoch אחד מושלם לאחר שהמודל עבר על כל האצוות המרכיבות את מאגר הנתונים.

- **חפיפה יחסית (IoU – Intersection over Union):** מדד המודד עד כמה שתי תיבות גבול (תיבת אמת ותיבת חיזוי) חופפות זו לזו. החישוב נעשה על ידי היחס בין שטח החפיפה של שתי התיבות לבין שטח האיחוד שלהן. ערך 1 משמעו חפיפה מלאה, ו־0 משמעו שאין חפיפה כלל. $IoU = \frac{\text{שטח החפיפה בין שתי התיבות}}{\text{שטח האיחוד של שתי התיבות}}$.
- **דיוק ממוצע משוקלל (mAP – Mean Average Precision):** מדד סטנדרטי לביצועי זיהוי אובייקטים. הוא משלב את דיוק הסיווג (Precision) ואת שיעור הכיסוי (Recall) ומודד את איכות החיזוי עבור כל מחלקה, ולאחר מכן מחשב ממוצע על פני כל המחלקות. בתחום הראייה הממוחשבת מקובל לדווח על mAP50 (כאשר נדרשת חפיפה של לפחות 50%, כלומר $IoU \geq 0.5$) ועל mAP50-95 (כאשר מחשבים ממוצע על פני ספי חפיפה מ-50% עד ל-95%).

סיכום התרומה העיקרית

הפרויקט שפותח מהווה שילוב בין תחום הראייה הממוחשבת ובין אלגוריתמי בטיחות לרוכבי אופניים, במטרה לספק מערכת התרעה חכמה ואוטונומית בזמן אמת. בצד הראייה הממוחשבת, נעשה שימוש במודל ממשפחת YOLO, אשר אומן על בסיס מאגר נתונים מתויג מתוך OID, עם התמקדות בזיהוי רכבים. המודל הוטמע והורץ על מצלמת OAK-D Lite, המאפשרת ביצועי עיבוד תמונה בזמן אמת תוך שמירה על ניידות ויעילות אנרגטית. מעבר לכך, פותח אלגוריתם ייעודי לזיהוי מצבי סכנה, אשר מתבסס על עקרונות מתוך מאמר של חברת Mobileye. האלגוריתם כולל שילוב בין מסנן קלמן (Kalman Filter) לניבוי תנועה עתידית של אובייקטים, לבין מסנן ROI לצמצום רעשים וזיהוי ממוקד של אזורים רלוונטיים בלבד. כלל המערכת פועלת באופן עצמאי על גבי כרטיס בקרה מסוג Raspberry Pi 4B, המחובר לספק כוח ומריץ את המודל והאלגוריתם ברצף. בעת זיהוי מצב המסווג כסכנה מתקרבת, המערכת משדרת התראות קוליות בזמן אמת לרוכב באמצעות אוזניות Bluetooth, במטרה לאפשר תגובה מהירה ומניעת תאונה פוטנציאלית. בכך, תרומת הפרויקט היא בהוכחת היתכנות למערכת התרעה אינטגרטיבית, ניידת וזולה יחסית, המבוססת על טכנולוגיות מתקדמות של ראייה ממוחשבת ובינה מלאכותית, אשר יכולה בעתיד להשתלב ככלי עזר משמעותי לשיפור בטיחות רוכבי אופניים בכבישים עירוניים ובין-עירוניים.

להשלמת הפתרון, תוכנן ויוצר מארז ייעודי באמצעות כלי תכנון CAD והדפסת תלת-ממד. המארז מכיל את הבקר והמצלמה באופן יציב ומוגן, וכולל מנגנון המאפשר חיבור פשוט ומאובטח למושב של האופניים.

עבודות קודמות בנושא

בחברת Velo פותח מוצר בשם Copilot שנועד לשמש כמערכת התרעה חכמה לבטיחות רוכבי אופניים. דומה ברעיון - מדובר במתקן המותקן בגב האופניים המשלב מצלמה עם עיבוד בינה מלאכותית, במטרה לספק מעקב מדויק אחר הסביבה האחורית של הרוכב ותגובה מהירה להתרעות בטיחותיות בזמן אמת.

מבנה העבודה

דוח זה מחולק לארבעה פרקים מרכזיים. הפרק הנוכחי, המבוא, מציג את הרקע והצורך בפרויקט, מגדיר מונחי יסוד, מסכם את תרומתה העיקרית של העבודה וסוקר עבודות קודמות בתחום.

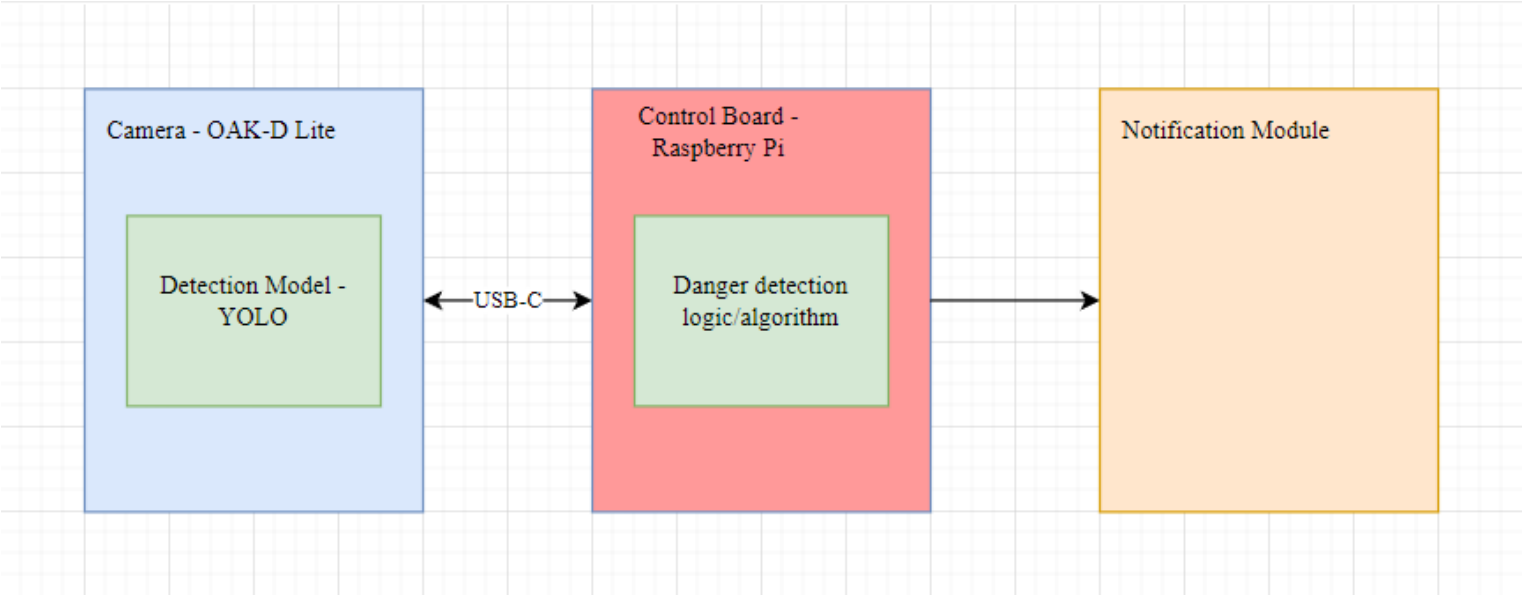
הפרק השני, תיאור כללי, מציג את ארכיטקטורת המערכת. הוא מפרט את רכיבי החומרה שנבחרו, ובראשם מצלמת ה-OAK-D Lite והבקר Raspberry Pi. את רכיבי התוכנה המרכזיים הכוללים את מודל ה-YOLO ואלגוריתם זיהוי הסכנה, ואת האופן שבו הם משתלבים פיזית ותוכנתית ליצירת הפתרון השלם.

הפרק השלישי, תיאור מפורט, מעמיק בהיבטים האלגוריתמיים של הפרויקט. הוא מציג את המודל המתמטי לחישוב זמן עד להתנגשות (TTC), את המימוש של מסנן קלמן ומסנן אזור העניין (ROI), וכן סוקר את המבנה של סקריפט ההרצה הראשי deploy_model.py, ואת רכיבי המעקב המרכזיים.

הפרק הרביעי, סיכום ומסקנות, מנתח את תוצאות הפרויקט, החל מתוצאות אימון המודל, דרך בחינות סינתטיות בסביבה מבוקרת ועד לניסויי הרצה בזמן אמת. פרק זה מסתיים במסקנות הכלליות העולות מהעבודה, ודן בגישות חלופיות ובאפשרויות להמשך פיתוח.

בסוף הדוח מופיעים נספחים הכוללים הוראות הרצה מפורטות ורשימת מקורות.

תיאור כללי



איור 1 - תיאור בלוקים של המערכת

תיאור חומרה

מצלמת OAK-D Lite

איור 2 - מצלמת OAK D-Lite

בפרויקט זה נעשה שימוש במצלמת OAK-D Lite שנבחרה בזכות יכולתה להריץ מודלים של ראייה ממוחשבת ישירות על גביה, ובשל התמיכה בספריית Python המאפשרת אינטגרציה פשוטה ונוחה עם כרטיס הבקרה. שילוב זה הופך אותה לרכיב מתאים במיוחד עבור פיתוח מהיר של מערכות מבוססות ראייה ממוחשבת.

לצד יתרונותיה, למצלמה קיימים גם חסרונות. אחד החסרונות הבולטים הוא התחממות מהירה המחייבת הוספת מערכת קירור חיצונית לשמירה על פעולה יציבה לאורך זמן. בנוסף, על אף שהמצלמה כוללת יכולת מובנית להערכת מרחק (Depth Estimation), בפרויקט הנוכחי לא נעשה שימוש ביכולת זו מכיוון שחישוב המרחקים ובחינת מצבי הסכנה מתבצעים באמצעות האלגוריתם הרץ על גבי כרטיס הבקרה.

בהתחשב בכך, ניתן לשקול חלופות פשוטות וזולות יותר כגון שימוש במצלמה רגילה או אפילו במצלמת טלפון נייד. לחלופין, קיימת גם אפשרות להחליף את המצלמה בחיישן Lidar אשר עשוי להציע דיוק גבוה יותר במיפוי עומק ותלת-ממד, אך במחיר גבוה יותר ובמורכבויות טכניות נוספות.

כרטיס בקרה Raspberry Pi 4B

איור 3 - כרטיס בקרה Raspberry Pi 4B

בפרויקט נעשה שימוש בכרטיס בקרה Raspberry Pi 4B. זהו כרטיס בקרה יחסית חלש אשר יכול לתפקד כמו מחשב קטן. נהוג לתכנת עליו אפליקציות בעזרת Python אך מסוגל לתמוך בשפות אחרות. יש לו distru ספציפי של Linux אשר מספק גמישות רבה. כתוצאה מהעברת העומס החישובי של המודל YOLO למצלמת ה-OAK-D Lite אין צורך במעבד חזק ולכן ניתן להסתפק בדור 4B במקום 5. כמו כן, 4B יותר זול למרות שאין שיפור משמעותי מבחינת מעבד בהשוואה ל-Jetson nano.

כרטיס הבקרה אחראי על ניהול המידע המגיע מן מהמצלמה במקביל עם המידע הקיים אצלו על מנת להסיק האם יש להתריע על סכנה.

אם מוותרים על מצלמת ה-OAK-D Lite אז העומס החישובי של הרצת המודל ייפול חזרה על הכרטיס בקרה ובמקרה זה כדאי לשקול להשתמש בכרטיס בקרה עוצמתי יותר כגון Raspberry Pi 5.

הכיוון של שימוש בטלפון נייד יכול להפיק גם תוצאות טובות ונוחיות מימוש מכך שכל הרכיבים הדרושים לפרויקט כבר משולבים במארז יחיד.

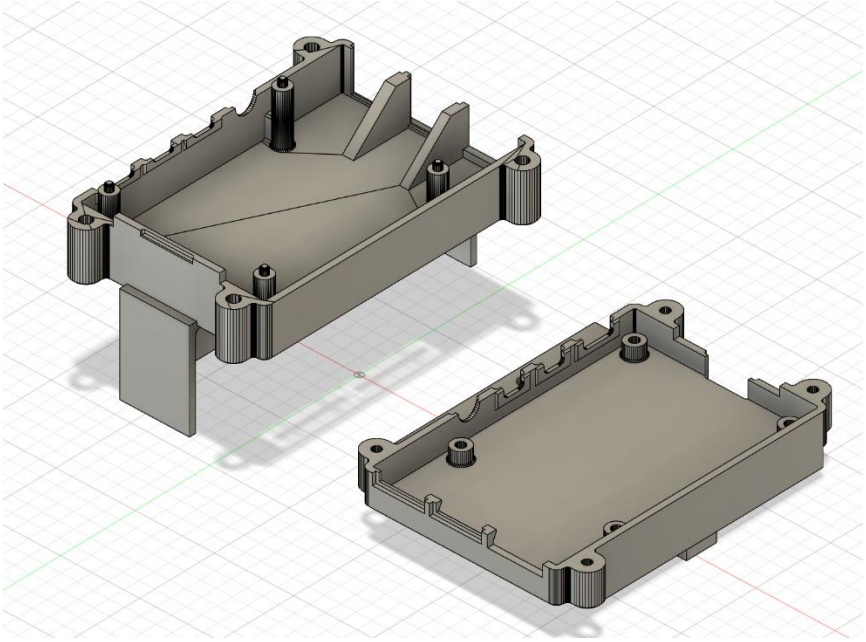
מערכת שמע עם יכולת חיבור Bluetooth

איור 4 - מערכת שמע Bluetooth

מערכת שמע ניידת עם יכולת חיבור Bluetooth למען חיבור פשוט עם כרטיס הבקרה אשר מהווה דרך התרעה לרוכב אופניים מפני סכנה.

אינטגרציה של החלקים

על מנת שיהיה ניתן לחבר את כלל החלקים בפשטות למושב הכיסא, עוצב במהלך הפרויקט מארז שניתן להדפיס ואז לחבר למושב האופניים.



איור 5 - המארז בתוך תוכנית CAD

איור 6 - מארז מורכב ומחובר לכיסא אופניים

הכרטיס בקרה יושב בין שני החלקים המרכזיים והמצלמה מעוגנת על ידי שני ברגים. כמו כן, יש לחבר כבל USB 3.0 בין המצלמה לכרטיס הבקרה ויש לספק לכרטיס הבקרה חשמל על ידי סוללה חיצונית.

מודל ראייה ממוחשבת - YOLO 10n

איור 7 – לוגו Ultralytics YOLO

בפרויקט זה נעשה שימוש באלגוריתם הראייה הממוחשבת YOLOv10n מבית Ultralytics. האלגוריתם אומן על בסיס מאגר Open Images עם התמקדות בקטגוריות רכבים רלוונטיות לסביבת רוכב האופניים: מכוניות, משאיות, אוטובוסים, אופנועים, ואנים ואמבולנסים.

האימון בוצע על 10,000 תמונות בסט האימון ועל 2,500 תמונות נוספות בסט הווליד‌ציה, במטרה להבטיח דיוק גבוה ויכולת הכללה טובה למצבים מגוונים.

מלבד ביצועיו הטובים של מודל זה, קיימים לו יתרונות רבים: המודל נתמך הן בספריות Python והן במצלמת OAK-D Lite, מה שמקל על שילובו במערכת. המודל חינמי וזמין בקוד פתוח, מתעדכן באופן תדיר, וכל הזמן משתחררות גרסאות משופרות. את המודל המאומן ניתן להמיר לקובץ blob. שאותו המצלמה יודעת להריץ ישירות. גרסת YOLOv10n היא קטנה יחסית, עם כ־2.3 מיליון פרמטרים נלמדים בלבד. כתוצאה מכך זמן האימון קצר יחסית, ודרישות החומרה נותרות מתונות.

חלופות אפשריות הן שימוש במודלי סגמנטציה ממשפחת SAM (כגון FastSAM, MobileSAM או SAM2) המאפשרים הבנה מרחבית מפורטת יותר של הסצנה או ניתן היה לבחור בגרסאות מתקדמות או גדולות יותר של YOLO, אולם ייתכן שחלופות אלה אינן נתמכות ע"י מצלמת ה-OAK-D Lite ואז המודל יצטרך לרוץ ישירות על כרטיס הבקרה - מה שיצריך כרטיס בקרה חזק ויקר יותר. בנוסף, זמן האימון של מודלים גדולים יותר יהיה ארוך בהרבה – מה שלא בהכרח משתלם יותר מדי מבחינת ביצועים.

כמו כן, המחיר של הרצת המודל בזמן אמת הוא קריטי פה, גרסאות ה-n של YOLO מביאות תוצאות די טובות אך צפוי שהמודלי סגמנטציה יהיו כבדים מידיי.

אלגוריתם לזיהוי והתרעה על סכנה

האלגוריתם לזיהוי סכנה מבוסס על המאמר "Forward Collision Warning with a Single Camera" של ארז דגן, עופר מנו, גדעון שטיין ואמנון שעשוע מחברת Mobileye משנת 2004, אשר מהווה את אחת מאבני הדרך הראשונות בטכנולוגיה של Mobileye.

Forward Collision Warning with a Single Camera

Erez Dagan	Ofer Mano	Gideon P. Stein	Amnon Shashua
MobileEye Vision Technologies Ltd.	MobileEye Vision Technologies Ltd.	MobileEye Vision Technologies Ltd.	Hebrew University
Jerusalem, Israel	Jerusalem, Israel	Jerusalem, Israel	Jerusalem, Israel
erez.dagan@mobileye.com	ofer.mano@mobileye.com	gideon.stein@mobileye.com	shashua@cs.huji.ac.il

איור 8 – כותרת וכותבי המאמר

Abstract

The large number of rear end collisions due to driver inattention has been identified as a major automotive safety issue. Even a short advance warning can significantly reduce the number and severity of the collisions. This paper describes a vision based Forward Collision Warning (FCW) system for highway safety. The algorithm described in this paper computes time to contact (TTC) and possible collision course directly from the size and position of the vehicles in the image - which are the natural measurements for a vision based system - without having to compute a 3D representation of the scene. The use of a single low cost image sensor results in an affordable system which is simple to install. The system has been implemented on real-time hardware and has been test driven on highways. Collision avoidance tests have also been performed on test tracks.

איור 9 – תקציר המאמר

למעשה, האלגוריתם לזיהוי סכנה צריך רק אלגוריתם לראייה ממוחשבת כדי לזהות רכבים ואת קצב הגדילה של רוחב ה-bounding box של כל רכב על מנת לחשב את הזמן הפגיעה המשוער (Time-To-Contact: TTC).

המערכת משתמש בשני מסננים בנוסף:

- מסנן ROI – מסנן זה נועד להתמקד רק באובייקטים הרלוונטיים למסלול הנסיעה. לעיתים רכבים חונים בצידי הדרך או נוסעים בנתיב הנגדי. במקרים אלו, ה-bounding box שלהם מתרחב בקצב מהיר מאוד, משום שהם נכנסים אל תוך התמונה בבת אחת, אף על פי שבפועל אינם מהווים איום ממשי על הנוסע. מסנן ה-ROI מזהה מצבים כאלו ומסנן אותם, ובכך מצמצם את מספר ההתרעות השגויות.
- מסנן קלמן – מדידות המיקום והגודל של ה-bounding box שמופקות על ידי מודל הזיהוי עשויות להכיל רעש ותנודות אקראיות. מסנן הקלמן משמש להחלקת המידע, מעקב רציף אחרי האובייקטים, וחיזוי המיקום העתידי שלהם. המסנן מניח שהמערכת הדינמית (תנועת הרכב שזוהה) מתפתחת באופן חלק, ומשלב בין המדידות בפועל לבין תחזית המבוססת על המודל. כך הוא מספק הערכה יציבה ואמינה יותר של תנועת כלי הרכב, ובסופו של דבר משפר את איכות קבלת ההחלטות של מערכת ההתרעה.

אלגוריתם הזיהוי וההתרעה מוגדר בקובץ **vehicle_tracker.py** ומכיל שלוש מחלקות:

- **TrackedVehicle:** מייצגת מידע על רכב במעקב, למשל גבולות ה-bounding box שלו, זמן הפגיעה המשוער שלו, היסטוריית רוחב לחישוב ה-ttc ועוד.
- **WarningStateManager:** מנגנון ההתרעה – מרכז את מצב ההתרעה הגלובלי על פני כלל הרכבים ומפעיל thread נפרד להשמעת התרעה קולית כאשר קיימת סכנה פעילה.
- **VehicleTracker:** מערכת המנהלת את כל הרכבים במעקב ואת מנגנון ההתרעה.

סקריפטים

- **downloader.py:** הורדה של מאגר האימון "open-images-v7". ניתן להחליט על פרמטרים כגון: גדול סט האימון וסט הוולידציה, המחלקות (classes) אשר תמונות המכילות אותן יורדו ותיקיית היעד של התמונות.
- **train_yolo.py:** אימון של מודל ממשפחת YOLO על מאגר תמונות. ניתן להחליט על פרמטרים כגון: המודל (איזו גרסה של YOLO), כמות ה-Epochs, גודל ה-Batch ומאגר המידע.
- **convert_model_to_blob.py:** המרה של מודל YOLO מאומן מקובץ pt. לקובץ blob. אותו מצלמת ה-OAK-D Lite יכולה

להריץ.

- **deploy_model.py:** העלאה של מודל YOLO מאומן בפורמט blob. אל מצלמת ה-OAK-D Lite והרצת המערכת.

תיאור מפורט

אלגוריתם לזיהוי והתרעה על סכנה

אחד הרכיבים המרכזיים במערכת ההתרעה הוא האלגוריתם לחישוב "זמן עד להתנגשות" (Time-to-Collision). מטרת האלגוריתם היא להעריך תוך כמה זמן, בהינתן המצב הנוכחי, צפויה להתרחש התנגשות עם רכב מתקרב. לצורך כך, פותח מודל המבוסס על עקרונות של ראייה ממוחשבת, אשר רותם את המידע הוויזואלי הגלום בשינוי גודל הרכב בתמונה לאורך זמן.

רציונל ועקרון פעולה

מערכת המבוססת על מצלמה יחידה מתקשה למדוד באופן ישיר ומדויק מרחק ומהירות מוחלטים. לעומת זאת, מדידה של שינויים יחסיים, ובפרט שינוי בגודלו הנצפה של אובייקט, היא משימה טבעית ואמינה עבורה.

העיקרון המנחה את האלגוריתם הוא שכאשר רכב מתקרב אל המצלמה במהירות יחסית קבועה, גודלו הנראה בתמונה (הרוחב או הגובה שלו בפיקסלים) יגדל בקצב שניתן למדוד. האלגוריתם מנצל קשר זה כדי לחשב את ה-TTC, מבלי להזדקק לחישוב המרחק או המהירות בערכים מוחלטים.

מודל מתמטי

האלגוריתם מבוסס על שני שלבים עיקריים:

1. חישוב "פקטור שינוי הגודל" (Scale-Change Factor):

בשלב הראשון, המערכת מודדת את רוחב התיבה התוחמת של הרכב בשתי נקודות זמן שונות ומחשבת את היחס ביניהן. יחס זה מוגדר כפקטור שינוי הגודל, S.

$$S = \frac{w_1}{w_0}$$

כאשר:

- w_0 – רוחב הרכב במדידה הקודמת.
- w_1 – רוחב הרכב במדידה הנוכחית.

2. חישוב ה-TCC:

בשלב השני, ה-TTC מחושב על בסיס פקטור שינוי הגודל (S) ופרק הזמן (Δt) שחלף בין שתי המדידות.

$$T_{TTC} = \frac{\Delta t}{S - 1}$$

כאשר:

- T_{TTC} – הזמן עד להתנגשות המוערך.
- Δt – פרק הזמן בין המדידה של w_0 ל-w_1.
- S – פקטור שינוי הגודל שחושב בשלב הקודם.

מודל זה מספק הערכה יעילה ומהירה של הזמן עד להתנגשות, והוא יעיל במיוחד בתרחישים של התקרבות במהירות יחסית קבועה.

מסנן ROI

הרציונל מאחורי המסנן הוא ההנחה שכלי רכב המהווים סכנת התנגשות ישירה יופיעו לרוב במרכז שדה הראייה של המצלמה, ישירות מאחורי

הרוכב. כלי רכב המופיעים ונעים באופן קבוע בצדי התמונה הקיצוניים, ככל הנראה אינם נמצאים על אותו נתיב נסיעה ולכן מהווים סיכון נמוך יותר. המסנן מגדיר "מסדרון" וירטואלי במרכז התמונה, ורק כלי רכב הנמצאים בתוך מסדרון זה נלקחים בחשבון לצורך הפעלת התרעת סכנה.

לוגיקת מימוש

מוגדר אזור עניין אופקי באמצעות שני קבועים:

$$ROI_\min = \alpha$$

$$ROI_\max = 1 - \alpha$$

כאשר $0 < \alpha < 1$.

במהלך פיתוח וניסויים על פני הרצה של המודל עם ערכים שונים של α התגלה כי $\alpha \cong 0.15$ הפיק תוצאות טובות.

הערכים ROI_min ו-ROI_max מייצגים את גבולות אזור העניין בקואורדינטות מנורמלות (בין 0 ל-1), כאשר 0 הוא הקצה השמאלי של התמונה ו-1 הוא הקצה הימני.

עבור על זיהוי חדש של רכב, המערכת מבצעת את הבדיקה הלוגית הבאה:

1. חישוב המרכז האופקי של הרכב:

נגדיר את cx_norm כקואורדינטה האופקית המנורמלת של מרכז התיבה התוחמת של הרכב שזוהה: $cx_norm = \frac{x_min + x_max}{2}$

כאשר:

- x_min ו-x_max הן הקואורדינטות האופקיות המנורמלות של התיבה התוחמת.

2. בדיקה לוגית:

המערכת בודקת האם מרכז הרכב נמצא **מחוץ** לאזור העניין המוגדר. רכב יסומן להתעלמות (ignore_for_warning) אם מתקיים התנאי הבא: $cx_norm > ROI_MAX \;||\; cx_norm < ROI_MIN$

חשוב לציין כי דגל ה-ignore_for_warning הוא "דביק" (sticky). כלומר, אם רכב כלשהו זוהה אפילו פעם אחת מחוץ לאזור העניין, הוא יסומן באופן קבוע להתעלמות ולא יוכל להפעיל התרעה, גם אם ייכנס בהמשך לאזור העניין. הדבר נועד למנוע התראות מרכבים החותכים את נתיב הרכיבה באופן שאינו מסכן את הרוכב.

השימוש בדגל בא לידי ביטוי בקוד בבדיקה לפני החלטה של התרעה על הרכב.

איור 10- המחשה של ROI

מסנן קלמן

הגדרת וקטור המצב ווקטור המדידה

ראשית, נגדיר את המשתנים המרכזיים:

- **וקטור המצב x_k (State Vector):** וקטור זה מתאר את כל המאפיינים של הרכב שאנו עוקבים אחריהם ברגע k. וקטור המצב מכיל 8 משתנים:

$$x_k = \begin{bmatrix} x \\ y \\ w \\ h \\ v_x \\ v_y \\ v_w \\ v_h \end{bmatrix}$$

כאשר:

  - x, y - קואורדינטות מרכז התיבה התוחמת.
  - w, h - רוחב וגובה התיבה התוחמת.
  - v_x, v_y - המהירות (שינוי המיקום) בציר x ו-y .
  - v_w, v_h - קצב השינוי של הרוחב והגובה.

- **וקטור המדידה z_k (Measurement Vector):** וקטור זה מכיל את הנתונים הנמדדים בפועל מהסביבה בכל רגע k. נתונים אלו המתקבלים ממודל ה-YOLO (תיבה התוחמת):

$$z_k = \begin{bmatrix} x_{meas} \\ y_{meas} \\ w_{meas} \\ h_{meas} \end{bmatrix}$$

שלב החיזוי (Prediction)

בשלב זה, המערכת חוזה את המצב הבא של הרכב על סמך המצב הקודם, עוד לפני קבלת מדידה חדשה מהמצלמה.

המשוואות לשלב זה הן:

- **חיזוי המצב הבא:** $\hat{x}_k^- = A\hat{x}_{k-1}$
- **חיזוי שגיאת השערוך:** $P_k^- = AP_{k-1}A^T + Q$

הסבר המשתנים:

  - $\hat{x}_k^-$: המצב **החזוי** של הרכב בזמן k.
  - $\hat{x}_{k-1}$: המצב **הסופי** (המתוקן) של הרכב בזמן הקודם k-1.
  - A: מטריצת מעבר המצבים (State Transition Matrix). היא מתארת כיצד מצב המערכת צפוי להשתנות מזמן k-1 לזמן k בהנחת מודל תנועה מסוים.
  - P_k^-: מטריצת השונות המשותפת (Covariance) של השגיאה **החזויה**. היא מייצגת את אי-הוודאות לגבי החיזוי.
  - P_{k-1}: מטריצת השונות המשותפת של השגיאה בזמן הקודם k-1.
  - Q: מטריצת הרעש של התהליך (Process Noice Covariance). היא מייצגת את אי-הוודאות במודל התנועה עצמו (למשל, רכב עשוי להאיץ או להאט באופן בלתי צפוי).

שלב העדכון (Update/Correction)

בשלב זה, המערכת מקבלת מדידה חדשה z_k ממודל ה-YOLO ומשתמשת בה כדי לתקן את החיזוי. זה מתבצע על ידי הקריאה לפונקציה self.kalman.correct(measurement). המשוואות לשלב זה הן:

- **חישוב הגבר קלמן (Kalman Gain):** $K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}$
- **עדכון המצב עם המדידה החדשה (Updated State Estimate):** $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
- **עדכון שגיאת השערוך (Updated Error Covariance):** $P_k = (I - K_k H)P_k^-$

הסבר המשתנים:

- K_k: הגבר קלמן (Kalman Gain): זהו "המשקל" שהמסנן נותן למדידה החדשה לעומת החיזוי. אם אי-הוודאות במדידה נמוכה, K_k יהיה גבוה, והמערכת "תסמוך" יותר על המדידה.
- H: מטריצת המדידה (Measurement Matrix). היא ממפה את וקטור המצב (בעל 8 ממדים) למרחב המדידה (בעל 4 ממדים).
- R: מטריצת הרעש של המדידה (Measurement Noise Covariance). היא מייצגת את אי-הוודאות במדידה המתקבלת ממודל ה-YOLO.
- $\hat{x}_k$: המצב **הסופי והמתוקן** של הרכב בזמן , לאחר שילוב המדידה החדשה. זהו הפלט של המסנן עבור ה-frame הנוכחי.
- P_k: מטריצת השונות המשותפת של השגיאה **המעודכנת**.
- I: מטריצת היחידה.

נציין כי python מביאים ספרייה אשר מבצעת את כל החישובים לבדה, יש רק צורך לתת הגדרה ראשונית לחלק מן המטריצות.

deploy_model.py

ליבת המערכת ממומשת בסקריפט deploy_model.py, המשמש כנקודת הכניסה הראשית להפעלת המערכת כולה. סקריפט זה אחראי על תזמור כלל הרכיבים: אתחול התקשורת עם החומרה, הפעלת מודל הראייה הממוחשבת, ניהול לוגיקת המעקב וההתרעה, והקלטת פלט הווידאו. ארכיטקטורת התוכנה היא מודולרית ומבוססת על מספר רכיבים מרכזיים, בעיקר מתוך המודול vehicle_tracker.py, אשר פועלים יחד ליצירת מערכת התרעה בזמן אמת.

ממשק התקשרות עם מצלמת OAK-D Lite

התקשורת עם מצלמת ה-OAK-D Lite מנוהלת באמצעות ספריית DepthAI. התקשורת אינה ישירה, אלא מתבצעת דרך הגדרת "צינור עיבוד נתונים" (Pipeline) בפונקציה create_camera_pipeline. צינור זה מגדיר את זרימת המידע על גבי חומרת המצלמה עצמה ומאפשר העברת עומס חישובי (Off-loading) מהבקר הראשי אל המצלמה.

צינור העיבוד מורכב מהצמתים (Nodes) והקישורים (Links) הבאים:

1. ColorCamera: צומת המייצג את חיישן המצלמה. הוא מוגדר לצלם וידאו ברזולוציה הרצויה (1080p) ולהפיק שתי תתי-תמונות (streams): preview ו-video.
2. YoloDetectionNetwork: צומת זה מייצג את מודל ה-YOLO לזיהוי אובייקטים. הוא מקבל את זרם ה-preview מהמצלמה כקלט, מריץ עליו את המודל (שנטען מקובץ ה-blob.), ומפיק את תוצאות הזיהוי. כל החישוב מתבצע על גבי המעבד הייעוד (VPU) של המצלמה.
3. XLinkOut: צמתים אלו משמשים ליצירת "תורים" (Queues) שדרכם המידע המעובד נשלח מהמצלמה אל הבקר (ה- Raspberry PI). בפרויקט הוגדרו שני תורים:

- q_nn: מעביר את תוצאות הזיהוי של רשת הנוירונים.
- q_video: מעביר את פריימ-י הווידאו הגולמיים.

סנכרון זה מאפשר לבקר לקבל בכל רגע נתון גם את תמונת הווידאו וגם את רשימת הרכבים שזוהו בה, מבלי שהבקר עצמו יצטרך לבצע את החישוב הכבד של זיהוי האובייקטים.

רכיבי המעקב וההתרעה

הלוגיקה המרכזית של המערכת, האחראית על מעקב אחר הרכבים והפקת ההתרעות, מרוכזת במודול vehicle_tracker.py ומבוססת על שלוש מחלקות עיקריות:

1. TrackedVehicle – ניהול רכב במעקב

מחלקה זו מהווה ייצוג של רכב **בודד** הנמצא במעקב. כל אובייקט ממחלקה זו מכיל את כל המידע הרלוונטי על אותו רכב ספציפי, כולל:

- מזהה ייחודי (id).
- המיקום הנוכחי של הרכב בתמונה (bbox).
- היסטוריית גדלים (width_history) המשמשת לחישוב מדדים.
- אובייקט מסנן קלמן (kalman) אישי, האחראי על חיזוי מיקומו הבא ועל החלקת תנועתו.
- מדדים מחושבים כמו מהירות יחסית (speed) וזמן עד להתנגשות (ttc).
- מצב ההתרעה (warning) ודגלים נוספים כמו ignore_warning לסינון על בסיס ROI.

בכל פעם שרכב חדש מזוהה, נוצר עבורו אובייקט TrackedVehicle חדש, אשר מלווה אותו כל עוד הוא נמצא בשדה הראייה.

2. VehicleTracker – מנהל המעקב הראשי

זוהי המחלקה המרכזית המנהלת את **כלל הרכבים** הנמצאים במעקב. היא מחזיקה רשימה של כל אובייקטי ה-TrackedVehicle הפעילים. בכל פריים, הפונקציה הראשית שלה, update, מבצעת את הפעולות הבאות:

- חיזוי (Prediction): מורה לכל אובייקט TrackedVehicle לחזות את מיקומו הבא באמצעות מסנן קלמן.
- שיוך (Data Association): משווה בין רשימת הזיהויים החדשה שהתקבלה מהמצלמה להן רשימת הרכבים הקיימים. השיוך מתבצע על בסיס מדד "חיתוך על פני איחוד" (IoU).
- עדכון ומחיקה: אם נמצאה התאמה, הרכב הקיים מתעדכן במיקום החדש. אם זוהה רכב חדש ללא התאמה, נוצר עבורו אובייקט TrackedVehicle חדש. מהצד השני, רכבים שלא נצפו במשך מספר פריימים מוגדר מראש (REMOVE_TIME_FRAME) נמחקים מהמעקב.
- חישוב מדדים והתרעות: מפעילה את חישוב ה-TTC והמהירות עבור על רכב, ומעדכנת את מצב ההתרעה שלו.

3. WarningStateManager – ניהול מערך ההתרעות

כדי למנוע מצב שבו השמעת קובץ קולי תעצור או תעכב את לולאת עיבוד הווידאו הראשית (שחייבת לרוץ בזמן אמת), פותח רכיב זה. תפקידו הוא לנהל את השמעת ההתרעות ב**תהליכון (Thread) נפרד**.

- הלולאה הראשית מעדכנת את ה-WarningStateManager האם קיים רכב כלשהו במצב התרעה.
- התהליכון הנפרד בודק באופן רציף את המצב הזה. אם נדרשת התרעה, הוא מנגן את קובץ השמע באופן אסינכרוני, מבלי להפריע לזרימה של לולאת עיבוד התמונה.

ארכיטקטורה זו מבטיחה שהמערכת תישאר רספונסיבית ותעמוד בדרישות של עיבוד בזמן אמת, תוך הפרדה ברורה בין לוגיקת הראייה, לוגיקת

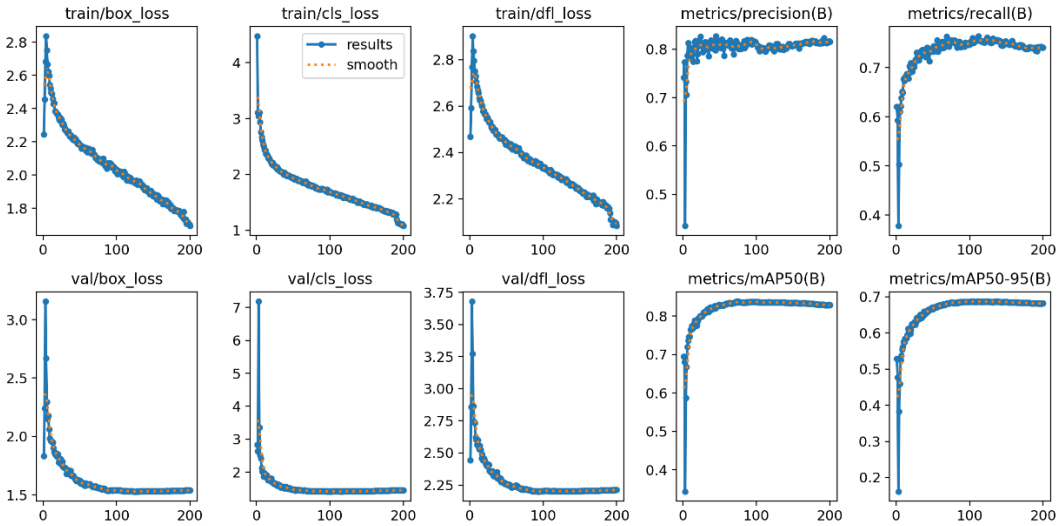
המעקב ולוגיקת ההתרעה.

סיכום ומסקנות

ניתוח תוצאות

אימון מודל

מודל ה-YOLO אומן על סט אימון המכיל 10000 תמונות וסט ולידציה של 2500 תמונות אשר הורדו ממאגר המידע של Open Images ומכילות רכבים. אלו תוצאות האימון:



איור 11 - תוצאות האימון

- הפסדים באימון:
  - box_loss – מודד את שגיאות החיזוי של ה-bounding box במהלך האימון. ירידה בערך זה מצביעה על שיפור ביכולת המודל להתאים את התיבה למיקום ולגודל האובייקט.
  - cls_loss – מודד את שגיאת הסיווג של האובייקטים. ירידה בערך זה מראה שהמודל משתפר בזיהוי נכון של רכבים.
  - dfl_loss – מודד את דיוק מיקום הגבולות של התיבה ברמת הפיקסל. ירידה בערך זה מצביעה על כך שהמודל מצליח למקם את קצוות התיבה באופן מדויק יותר סביב האובייקט.
- הפסדים בוולידציה:
  - box_loss – מודד את שגיאות החיזוי של ה-bounding box על סט הוולידציה. ירידה בערך זה מעידה שהמודל יודע להכליל את מה שלמד גם על דוגמאות חדשות.
  - cls_loss – מודד את שגיאת הסיווג בסט הוולידציה. ירידה בערך זה מראה שהמודל מצליח לזהות נכון רכבים גם על נתונים שלא ראה קודם.
  - dfl_loss – מודד את דיוק הגבולות ברמת הפיקסל בסט הוולידציה. ירידה בערך זה מצביעה על כך שהמודל מצליח לשמר דיוק גאומטרי גם מחוץ למידע עליו אומן.
- Overfitting או Underfitting:
  - Overfitting - מתרחש כאשר המודל "משנן" את נתוני האימון במקום ללמוד מהם עקרונות כלליים. כתוצאה מכך, הוא מציג ביצועים מעולים על הנתונים שהוא כבר ראה, אך נכשל כשהוא נתקל בנתונים חדשים. ניתן להבחין בכך כאשר:

- גרף שגיאת האימון (train loss) ממשיך לרדת, בעוד שגרף שגיאת האימות (val loss) מתחיל לעלות או נשאר שטוח ברמה גבוהה.
- יש פער גדול וגדל בין ביצועי המודל על סט האימון לבין ביצועיו על סט האימות.
- המודל מגיע לרמת דיוק (accuracy) גבוהה מאוד על נתוני האימון, אך הדיוק על נתוני האימות נמוך משמעותית ואינו משתפר.

- Underfitting - מתרחש כאשר המודל פשוט מדי ולא הצליח ללמוד אפילו את הדפוסים הבסיסיים בנתוני האימון. כתוצאה מכך, הוא מציג ביצועים גרועים באופן כללי, גם על הנתונים שהוא כבר ראה וגם על נתונים חדשים. ניתן להבחין בכך כאשר:
  - גרף שגיאת האימון (train loss) וגרף שגיאת האימות (val loss) נשארים גבוהים ואינם יורדים לרמה סבירה.
  - המודל מציג ביצועים נמוכים גם על סט האימון וגם על סט האימות.
  - הגרפים "משתטחים" בשלב מוקדם מאוד של האימון, מה שמעיד על כך שהמודל מיצה את יכולת הלמידה המוגבלת שלו ואינו משתפר יותר.

מניתוח הגרפים נובע כי קיימת ירידה ברורה ועקבית הן בגרפי האימון והן בגרפי האימות. מבחינת box_loss הדבר מעיד על כך שהמודל למד בהצלחה למקם את המלבנים סביב כלי הרכב בצורה מדויקת יותר ויותר ככל שהאימון התקדם. מבחינת cls_loss הדבר מעיד על כך שהמודל לומד לזהות רכבים בביטחון גבוה. מבחינת dfl_loss הדבר מעיד על כך שהמודל משפר את דיוק המיקום של התיבות התוחמות ברמה גבוהה.

ובאופן כללי, הגרפים מראים כי באימון לא התקיים Overfitting או Underfitting.

- מדדי ביצועים:
  - precision – מודד את אחוז הזיהויים של המודל שהיו נכונים בפועל (כלומר, כמה מהזיהויים ה אמיתיים ולא שגויים). ערך גבוה מעיד על שיעור נמוך של False Positive.
  - recall – מודד את אחוז האובייקטים האמיתיים שבתמונה שהמודל הצליח לזהות. ערך גבוה מעיד על שיעור נמוך של False Negative.
  - mAP50 – דיוק ממוצע משוקלל (mAP) ברמת חפיפה של $IoU \geq 0.5$. מודד עד כמה המודל מצליח לזהות אובייקטים בצורה נכונה כאשר נדרשת חפיפה של לפחות 50% בין התיבה החזויה לבין התיבה האמיתית.
  - mAP50-95 – מדד מחמיר יותר, המחשב את ה-mAP על פני מספר ספי חפיפה שונים (IoU מ-0.5 עד 0.95). ערך זה משקף את הדיוק והיציבות של המודל בזיהוי תיבות במיקומים מדויקים יותר.

הרצה בצורה סינטטית

בנוסף לבחינות המערכת בזמן אמת, בוצע שלב הערכה מקיף בסביבה סינתטית מבוקרת, באמצעות הרצת המערכת על גבי סרטוני וידאו שהוקלטו מראש. לגישה זו מספר יתרונות מרכזיים: היא מאפשרת חזרתיות מלאה על ניסויים להשוואה מדויקת בין גרסאות, ומספקת סביבה נוחה לכוונון פרמטרים עדינים של המערכת.

- <u>כוונון פרמטרים של מערכת המעקב</u>

הסביבה המבוקרת שימשה לכיול ומיטוב של פרמטרים אלגוריתמיים במודול המעקב (vehicle_tracker.py), במטרה להגיע לאיזון אופטימלי בין רגישות ליציבות. הפרמטרים המרכזיים שכוונו הם:

- סף הזמן למחיקת אובייקט (REMOVE_TIME_FRAME): נקבע הערך המתאים לאיזון בין מחיקה מוקדמת מדי של רכב שנעלם לרגע מהזיהוי, לבין השארת "עקבות רפאים" של רכבים שכבר יצאו מהסצנה.
- מרווח הדגימה לחישוב TCC (METRIC_HISTORY_GAP): כוונן כדי להבטיח חישוב TTC יציב מספיק, אך רגיש

לשינויים אמיתיים במהירות ההתקרבות.

- תצורת מסנן קלמן: בוצעו התאמות עדינות למטריצות הרעש (Q ו-R) כדי להשיג מעקב חלק וצפי מהימן של תנועת הרכבים.
- גבולות אזור העניין (ROI_MIN, ROI_MAX): נקבעו הגבולות האופטימליים לסינון יעיל של תנועה בנתיבים צדדיים מבלי לפספס איומים רלוונטיים.
- רמת ביטחון (CONFIDENCE_THRESHOLD): כוונן סף רמת הביטחון של הזיהויים שהמודל מוציא.

- <u>הערכת ביצועי המודל ובחינת חלופות</u>

הבחינה הסינתטית היוותה כלי מרכזי להערכה כמותית ואיכותית של מודלי הראייה הממוחשבת השונים שפותחו. כל מודל נבחן מול אותם סרטונים, והושווה על בסיס הקריטריונים הבאים:

- זיהויים שגויים (False Positives): בחינת הנטייה של המודל לזהות בטעות אובייקטים שאינם רכבים.
- פספוסים (False Negatives): בחינת יכולת המודל לאתר את כל הרכבים הרלוונטיים בסצנה, בדגש על תנאים מאתגרים.
- דיוק התיבה התוחמת: הערכת איכות המיקום של התיבה התוחמת סביב הרכב, והיציבות שלה בין פריימים עוקבים.

במסגרת הפרויקט, נבחנו מספר גישות אימון ומודלים:

1. ניסוי 1 - אימון על חזית הרכב בלבד: מודל שאומן על סט נתונים קטן (2,000 תמונות אימון) בו סומנה רק חזית הרכב. אף שהראה פוטנציאל, הוחלט כי גישה זו טומנת בחובה סיכון ודורשת תהליך תיוג מורכב שחורג ממסגרת הזמן של הפרויקט.
2. ניסוי 2 - אימון על סט נתונים רחב מ-Open images: מודלים ממשפחת YOLO: yolov8n, yolov10n, yolo11n אומנו על סט נתונים גדול (10,000 תמונות אימון, 2,500 תמונות ולידציה) עם תיוגים מלאים של כלי רכב שנלקחו גם כן מ- Open images. מודל ה-yolov10n מאימון זה הציג את הביצועים הטובים והמאוזנים ביותר, ולכן נבחר לשמש במערכת הסופית.
3. ניסוי 3 - אימון על סט נתונים מתויג-אוטומטית: מודל שאומן על אותו סט נתונים גדול, אך עם תיוגים שנוצרו באופן אוטומטי על ידי מודל yolo11x גדול וחזק יותר. באופן מפתיע, נמצא כי דיוק התיבות התוחמות במודל זה היה נמוך יותר בהשוואה לניסוי 2, ככל הנראה עקב "רעשי תיוג" מהמודל האוטומטי, ועל כן גישה זו נפסלה.

- <u>בחינת עמידות בתנאי סביבה משתנים</u>

השימוש בסרטונים מוכנים מראש אפשר לבחון את עמידות המערכת בתרחישים מגוונים:

- תנאי תאורה: בבחינת סרטונים שצולמו בתנאי תאורה קשים (שעות חשכה, צללים חזקים), זוהה קושי מסוים באיתור כלי רכב כהים שאינם מוארים היטב, מה שמצביע על מגבלה של המודל הנוכחי.
- ריבוי רכבים ועומס תנועה: הרצת המערכת על סרטונים המתארים כבישים עמוסים ובהם רכבים רבים בו-זמנית, הראתה כי המערכת שומרת על ביצועים יציבים וקצב עיבוד גבוה, ללא פגיעה נראית לעין ביכולת הזיהוי והמעקב.
- סוגי כבישים: לא נמצא הבדל משמעותי בביצועי המערכת בין סרטונים שצולמו בכבישים עירוניים צפופים לבין כאלה שצולמו בכבישים בינעירוניים פתוחים, מה שמעיד על יכולת הכללה טובה של המערכת לסביבות שונות.

הרצה בזמן אמת

לאחר השלמת הפיתוח והאינטגרציה של רכיבי החומרה והתוכנה, הורכבה המערכת על אופניים ונבחנה בתנאי רכיבה אמיתיים. בחינה זו נועדה להעריך את ביצועי המערכת בהיבטים של מהירות עיבוד, אפקטיביות הזיהוי וההתרעה, ומגבלות תפעוליות.

- ביצועי עיבוד וקצב דגימה (FPS) – במהלך הרצה רציפה של המערכת, נמדד קצב פריימים ממוצע (FPS) של 12 פריימים לשנייה. קצב זה, השווה ערך לזמן עיבוד ממוצע של כ-83 מילישניות לפריים, עומד בדרישות של מערכת זמן אמת. הוא מאפשר דגימה רציפה של סביבת הרוכב המספקת תמונה עדכנית של הסכנות הפוטנציאליות ומאפשרת למערכת להגיב במהירות לשינויים בכביש.
- אפקטיביות מערכת המעקב וההתרעה – בניסויי שטח, המערכת הדגימה יכולת טובה של זיהוי ומעקב אחר כלי רכב מתקרבים.

אלגוריתם המעקב, המבוסס על מסנן קלמן, סיפק זיהוי יציב והפחית "קפיצות" של התיבות התוחמות. במקביל, מסנן אזור העניין (ROI) פעל בהצלחה בסינון רכבים בנתיבים צדדיים, ובכך צמצם התרעות שוואו. מערכת ההתרעות, המבוססת על חישוב ה-TCC הוכחה כיעילה וסיפקה התרעות קוליות בזמן שאפשר לרוכב להגיב לסכנה מתקרבת.

- אתגרים ומגבלות תפעוליות – במהלך הבחינות זוהו שתי מגבלות תפעוליות מרכזיות:
  - רגישות לזעזועים מכניים: החיבור בין הבקר (Raspberry Pi) למצלמת ה-OAK-D Lite מתבצע באמצעות כבל USB 3.0. נמצא כי חיבור זה רגיש לזעזועים חזקים ופתאומיים (למשל, ירידה ממדרכה או כניסה לבור בכביש), העלולים לגרום לניתוק רגעי של התקשורת. עם זאת, במהלך רכיבה רגילה ורציפה על כביש סלול, המערכת פעלה באופן יציב וללא תקלות.
  - התחממות יתר של החומרה (Thermal Throttling): בהרצה רציפה של המערכת למשך זמן ממושך, נצפתה תופעה של התחממות יתר במצלמת ה-OAK-D Lite לאחר כ-60 דקות של פעולה, התחממות זו הובילה לקריסה של המצלמה ולהפסקת פעולת המערכת. זוהי מגבלה ידועה ברכיבי עיבוד זעירים ועתירי ביצועים, המצביעה על צורך עתידי בפתרון קירור (פסיבי או אקטיבי) לשם הבטחת פעולה רציפה לאורך זמן.

מסקנות כלליות

- היתכנות טכנולוגית – המסקנה המרכזית היא שאכן ניתן לפתח מערכת התרעה יעילה וזולה יחסית לרוכבי אופניים באמצעות רכיבי מדף. הפרויקט מדגים כיצד טכנולוגיית "בינת קצה" (Edge AI), המבצעת חישובים מורכבים על חומרה זעירה, היא המפתח לפתרונות ניידים מסוג זה.
- החשיבות של שילוב בין AI לאלגוריתמיקה קלאסית – המסקנה היא שמודל YOLO לבדו אינו מספיק. הצלחת המערכת נבעה מהשילוב בין יכולות הזיהוי של רשת הנוירונים לבין האלגוריתמים הקלאסיים (קלמן, TTC, ROI), שהפכו את המידע הגולמי להתראות חכמות ורלוונטיות.
- האתגר באיזון בין ביצועים למגבלות חומרה – הפרויקט הדגים את האתגר המרכזי במערכות משובצות מחשב (embedded systems): מציאת האיזון הנכון בין מודל מדויק ככל האפשר לבין היכולת להריץ אותו על חומרה קטנה ומוגבלת בצריכת חשמל ובקירור.

גישות נוספות לפתרון

- שימוש במאגר נתונים מותאם אישית – ביצוע תיוג ידני בהתאם לצורך הספציפי של פרויקט זה. למשל יש צורך בזיהוי רכבים אשר מתקדמים אל רוכב האופניים מהכיוון האחורי, לכן יש צורך לזהות רק את החלק הקדמי של הרכבים. אימון המודל על מאגר נתונים כזה יכול להפחית את שיעור הזיהויים השגויים (פחות False Positive) ובכך לשפר את מדד הדיוק (Precision). או עבור הצורך בתיוג יותר מדויק, בסט התיוג שמשומש הייתה שונות גדולה בין הגבולות על הרכבים ומצופה שעם תיוגים מדויקים גם יהיו זיהויים מדויקים ויציבים יותר.
- שימוש במודל חזק יותר על תשתית חיצונית – ניתן להריץ מודל כבד ומדויק יותר על שרת מרוחק, אולם גישה זו מצריכה חיבור אינטרנטי יציב.
- זיהוי לוחיות רישוי – במקום לזהות את כלל הרכב, ניתן להתמקד בזיהוי לוחית הרישוי. בדרך כלל בחזית הרכב ממוקמת לוחית רישוי, ולכן זיהוי עקבי שלה עשוי להוות אינדיקציה טובה להתקרבות רכב.

אפשרויות להמשך הפיתוח

כדי לקדם את הפרויקט ממערכת אב-טיפוס למוצר אמיתי, נדרש להעמיק את הבדיקות ולבצען בהיקף רחב ובתנאים מגוונים:

- מדגמים גדולים יותר: איסוף נתונים רחב יותר הכולל אלפי שעות רכיבה וסוגי כלי רכב מגוונים.
- בדיקות שטח ממושכות: הפעלה רציפה של המערכת לאורך פרקי זמן ארוכים, כדי לבחון את אמינותה ועמידותה.

- תנאי סביבה משתנים: בדיקות בתנאי תאורה שונים (יום, לילה, סנוור), מזג אוויר משתנה (גשם, ערפל, שמש חזקה) וסוגי כבישים שונים (עירוני, בין-עירוני, שטח).
- שיקולי עלות-תועלת: בחינת האיזון בין ביצועי המערכת לבין דרישות מעשיות כגון עלויות ייצור, צריכת אנרגיה, משקל וגודל הרכיבים, וכן רמת הנוחות והשימושיות עבור הרוכב.
- עמידות המערכת בתנאים שונים: פיתוח המערכת ומארז כך שיוכל לתפקד בתנאי שטח קיצוניים כמו חום גבוה/נמוך, רטיבות, רעידות חזקות.
- פיתוח אפליקציה לטלפון לאינטגרציה יותר נוחה למשתמש עם הכרטיס בקרה.

כחלק מהפיתוח של המוצר, יש גם ליצור תוכנית ווליידציה שתצביע על אמינות המוצר ברמה מסחרית.

נספחים

אתחול והרצה

עבור אתחול והרצה של הפרויקט ניתן לעיין בקובץ ה-README אשר מצורף ל-git repo אך נפרט גם כאן:

הורדה של ה-repository:

```
git clone https://github.com/RonBulka/bicycle_mobileye.git
cd bicycle_mobileye
```

הורדה של תלויות:

```
pip install -r requirements.txt
```

הורדה של המאגר מידע:

```
python src/downloader.py --export_labels --train_samples 6000 --val_samples 1500
```

אימון המודל:

```
python src/train_yolo.py --epochs 200 --batch 32
```

המרה של המודל ל-blob.:

```
python src/convert_model_to_blob.py
or if doesn't work use https://tools.luxonis.com/
```

הרצה של המודל על סרטון:

```
python tests/predict.py --input_name input1.mp4 --output_name output1.mp4
```

הרצה של המודל עם המצלמה:

```
python src/deploy_model.py
```

ספציפית להרצה על raspberry pi

הורדה של תלויות מערכת:

```
sudo apt update
sudo apt upgrade -y
sudo apt install -y python3 python3-pip python3-venv git
```

הורדה של תלויות Python:

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

העתקה של קובץ service:

```
sudo cp bicycle-mobileye.service /etc/systemd/system/
```

שינוי קובץ service שיכיל כתובות נכונות:

```
sudo nano /etc/systemd/system/bicycle-mobileye.service
```

אתחול והרצה של ה-service

```
sudo systemctl enable bicycle-mobileye
sudo systemctl start bicycle-mobileye
```

רשימת מקורות

[1] E. Dagan, O. Mano, G. P. Stein and A. Shashua, "Forward Collision Warning with a Single Camera", *IEEE Intelligent Vehicles Symposium*, pp. 37–42, 2004.

[2] Ultralytics, *YOLO Documentation*, https://docs.ultralytics.com

[3] Google, *Open Images Dataset*, https://storage.googleapis.com/openimages/web/index.html

[4] Raspberry Pi Foundation, *Raspberry Pi 4 Model B documentation* https://www.raspberrypi.com/documentation

[5] Luxonis, *OAK-D Lite Documentation*, https://docs.luxonis.com

[6] Velo.ai, *Copilot – AI-powered bicycle safety system*, https://www.velo.ai