ינואר 2010-2011

|  |
| --- |
|  |



**מגישים:** אביעד היטנר, יעקב הוזארסקי, צוריאל סלברמן, יוסף דוידסון

**מנחה:** הרב שמואל עוזיאל.

**ישיבת אמית נחשון המגמה הטכנולוגית**

|  |  |
| --- | --- |
| אמי"ת נחשון בית שמש | שדה הקרב העתידי |

תוכן עניינים

[1. תיאור מצב קיים 2](#_Toc324247292)

[2.היבט חברתי - צבאי 3](#_Toc324247293)

[רובוט מהו? 3](#_Toc324247294)

[רובוטיים צבאיים: 4](#_Toc324247295)

[3. הבעיה המועלת לדיון 6](#_Toc324247296)

[4. הפתרון המוצע 6](#_Toc324247297)

[5. תיאור המערכת 7](#_Toc324247298)

[6. קהל היעד 8](#_Toc324247299)

[7. היבט טכנולוגי 10](#_Toc324247300)

[מיקרו מעבד 10](#_Toc324247301)

[פעולת ומבנה המעבד 10](#_Toc324247302)

[תקשורת טורית ומקבילית 11](#_Toc324247303)

[מצלמה 12](#_Toc324247304)

[ייצוג צבע במחשב: 12](#_Toc324247305)

[תקשורת למחשב 14](#_Toc324247306)

[התוכנה Visual Basic: 15](#_Toc324247307)

[מאפיינים של Visual Basic: 15](#_Toc324247308)

[User Control: 18](#_Toc324247309)

[8. היבט מדעי - STS 28](#_Toc324247310)

[אלגוריתם חיפוש הדרך הכי קצרה 28](#_Toc324247311)

[אלגוריתם :A\* 31](#_Toc324247312)

[מבוא: שטח החיפוש 32](#_Toc324247313)

[החל חיפוש 33](#_Toc324247314)

[נתיב ניקוד 34](#_Toc324247315)

[המשך חיפוש 36](#_Toc324247316)

[סיכום שיטת \*A: 39](#_Toc324247317)

[9. מחקר ופיתוח : 41](#_Toc324247318)

[תיאור מבנה הרובוטים: 41](#_Toc324247319)

[תיאור מבנה הרובוט הרודף: 42](#_Toc324247320)

[תיאור מבנה הרובוט הנרדף: 43](#_Toc324247321)

[10. הסבר התוכנית + האלגוריתם חיפוש 44](#_Toc324247322)

[11. ביבליוגרפיה 76](#_Toc324247323)

# 1**. תיאור מצב קיים**

כיום יש שימוש רב בשליטה מרחוק במערכות שנות בתחומים שונים:

**בתחום הצבאי –**רובוטים צבאיים הם רובוטים חצי אוטונומיים או נשלטים מרחוק שעוצבו למטרות צבאיות. הם משמשים להרבה פעילויות שונות לרבות: חילוץ והצלה, איתור, תקיפה מטרות מרחוק ובדיקת בתים ללא סיכון חיי אדם.

**בתחום הביטחוני:**

רובוטים משמשים לכל מיני פעולות גם בתחום הביטחוני כמו:

-מערכות ניטור לצורך אבטחת מבנים וגבולות.

-פירוק פצצות. במקרה של זיהוי חפץ חשוד חבלני המשטרה, משתמשים ברובוט מונחה מרחוק על מנת לפרק את הפצצה.

**בתחום הרפואי:**

בשנים האחרונות התפתח תחום הרובוטיקה גם לכיוון הרפואי. השילוב בין התחומים היה יכולת הדיוק והנגישות הגבוהה של הרובוט. הרובוטים נמצאים כבר בשימוש ובשלבי פיתוח מתקדמים במספר תחומים ברפואה ובהם: ניתוחים בפולשנות מזערית, ניתוחי גב, ניתוחי מוח (ביופסיות), ניתוחים אורולוגייים וניתוחי לב. יישומים נוספים נבחנים כל העת. תחום הרובוטיקה הרפואית כמו תחום הרובוטיקה בכלל הוא רב תחומי ומשלב בתוכו בקרה, הנדסת מכונות, הנדסת תוכנה, אלקטרוניקה, חיישנים ואינטליגנציה מלאכותית.

- שימוש ברובוטים עוזר גם לצורך סיוע למוגבלים בביצוע פעולות יום- יומיות.

# 2.היבט חברתי - צבאי

## רובוט מהו?

רובוט הוא מכונה אוטומטית הניתנת לתכנות, בעלת יכולת תנועה, הנשלטת על ידי בקר ממוחשב ומסוגלת לבצע פעולות מורכבות יחסית, הניתן לתכנות.

הרובוט עצמו הוא שילוב כל הרכיבים ומהערכות עליהם אנו עובדים. הרובוט בעצם יעזור לנו להדגים את הרעיון של הפרויקט שבו הרובוט יבצע את הפקודות הנאמרות מהמשתמש.

איך יראה שדה הקרב העתידי? באיזה טקטיקות לחימה ישתמשו? כיצד יוכרעו מלחמות?

היסטורית הרובוטים שהשתתפו בשדות הקרב הולכות זמן רב אחורה, החל ממלחמת העולם השנייה והמלחמה הקרה. "גלית" היה רכב מימי מלחמת העולם השנייה שהיה מתפוצץ ומשמיד טנקים ומערכות חי"ר שפותח ע"י הורמאכט (צבא גרמניה) ונשלט ע"י שלט רחוק.



הרובוט גוליית בשימוש של חיילי הורמאכט

הטלטנק

הטלטנק הסובייטי היה טנק שנשא נשק והיה בלתי מאוייש (נשלט ע"י שלט רחוק). היו עוד הרבה רובוטים שנועדו לריגול, ציטוט והעברת מידע.

השימוש ברובוטים בעת מלחמה היה נחשב פעם לתחום המדע בדיוני אולם כיום הוא מפותח ונחקר בצורה גבוהה מאוד. יש המאמינים שבעתיד ילחמו במערכות נשק אוטומטיות, והצבאות מתחרות ביניהם על פיתוחים והמצאות חדשות. ישראל היא אחת המדינות המובילות בעולם בתחום מערכות הנשק האוטומטיות בפרט ובתחום הרובוטים בכלל.

כמו שלמדנו, רובוטים צבאיים הם תחום מאוד מתפתח כיום ולהלן כמה דוגמאות:

### רובוטיים צבאיים:

#### http://upload.wikimedia.org/wikipedia/commons/thumb/b/bc/ANDROS-sap1.jpg/250px-ANDROS-sap1.jpgאנדרוס MarkV-A1

רובוט סילוק פצצות (ס"פ) של יחידת יהל"ם, צה"ל מדגם אנדרוס MarkV-A1

אנדרוס MarkV-A1 הוא רובוט ס"פ תוצרת ארצות הברית שתוכנן על ידי נורת'רופ גראמן. הוא שייך לסדרת רובוטי רמוטק אנדרוס (Remotec ANDROS) הכוללת עוד רובוטי סילוק פצצות.

גודלו הוא כשל מכסחת דשא גדולה והוא שוקל 363 ק"ג (800 פאונד).

הרובוט מצויד בזרוע תלת-מפרקית ארוכה שבקצהה צבת, מצלמת וידאו ורובה shotgun לטיפול במטעני חבלה. המזקו"ם **(מ**ערכת **ז**חלים, **ק**פיצים **ומ**רכובים או בשפה הלא מקצועית "שרשרת") של הרובוט נרשם כפטנט והוא מתוכנן לעבירות בתנאי שטח קשים. בסך הכל לרובוט יש 4 מצלמות וידאו צבעוניות, כולל מצלמה המורכבת בראש הרובוט. הרובוט כולל גם מספר מיקרופונים, פנסים וחיישנים אחרים.

אנדרוס MarkV-A1

#### http://upload.wikimedia.org/wikipedia/commons/thumb/9/97/D9N-RaamHaShachar02.jpg/250px-D9N-RaamHaShachar02.jpgרעם השחר – דחפורי D9N

הD9N מופעל על ידי שלט רחוק. ומשמש את פלוגת "הבזק" ששייכת ליחידת יהל"ם.

"רעם השחר" הוא רובוט חדשני ורוב הפרטים עליו הם חסויים. הוא פותח על ידי הטכניון הישראלי בשיתוף פעולה עם צה"ל שהחלה לפני כ4 שנים. בעקבות ההצלחה של הדגם, נמצאים עוד שני D9 בפיתוח נוסף בטכניון שיכנסו בקרוב למערך הלחימה של "הבזק".

הכלי עצמו מצויד בשלל מצלמות כדי לתת תמונה מלאה של מקום העבודה של "רעם השחר". הכלי עצמו השתתף בהרבה פעילויות , החל מעזה וגם בלחימה בלבנון השנייה. הכלי עצמו נשלט בידי מערכת שליטה אלחוטית מיוחדת. כאשר מפעיל הכלי יושב ברכב מיוחד, ומפעיל את הכלי ממנו.במקרה של בעיות תקשורת וכדו'. הכלי נהפך לכלי רגיל. ואפשר להפעילו מתא העבודה הרגיל.

#### כיפת ברזל

כיפת ברזל היא מערכת מתנייעת ליירוט רקטות קצרות טווח, המפותחת על ידי חברת רפאל.

חברת רפאל מציגה את המערכת כפתרון יעיל נגד רקטות קצרות טווח ופגזי 155 מ"מ, הפועל ביום ובלילה, בכל תנאי מזג אוויר, ומסוגל להתמודד עם מספר רב של איומים בו זמנית, וכן כמערכת נ"מ המסוגלת ליירט מטוסים הטסים עד לגובה של 10 ק"מ.‏ סוללה אחת אמורה להגן על שטח של עד 150 קמ"ר.

המערכת מורכבת ממכ"ם חיפוש והנחיה, מרכז בקרה וטילי יירוט.

מכ"ם המערכת מזהה שיגור של רקטה ומעביר מידע על מסלול מעופה למערכת השליטה והבקרה (שו"ב), המחשבת לפי מידע זה את מקום הפגיעה הצפוי. אם מיקום זה מצדיק יירוט, משוגר טיל יירוט כנגד הרקטה (טיל היירוט משוגר כמעט-אנכית). הראש הקרבי של טיל היירוט מתפוצץ בסמוך לרקטה, באמצעות מרעום קרבה, קודם שהגיעה אל המטרה, במקום שבו לא צפוי נזק לצד המיירט.

# 3. הבעיה המועלת לדיון

רוב המערכות הקיימות כיום לא פועלות באופן אוטומטי לגמרי.

הם אמנם מבצעים פעולות שיועדו לכך אוטומטית , אך הם רק חלק ממערך גדול יותר שמבצע סדרת פעולות לצורך השגת מצב מסוים.

מערכים אלו כוללים בד"כ מרכיב אנושי שמקבל את ההחלטות על פעולות למען השגת מטרה מסיומת. כפי ששמתם לב כל המערכות הקיימות בתחומים שצוינו למעלה זקוקות לאדם לצורך הפעלתם. אמנם הם לא מאוישות אך ללא אדם הם לא יוכלו להשיג את מטרתם.

# 4. הפתרון המוצע

פיתוח מערכות אוטונומיות שמסוגלות לבצע את מטרתן המוגדרת מראש ללא אדם המתפעל את המערכת, ע"י כתיבת אלגוריתם לכל מטרה.

בפרויקט נפתח מערכת לדוגמה, שתפעל באופן עצמאי לביצוע מטרתה, תוך כדי זיהוי תנאי השטח וכל האובייקטים המצויים ב"זירת העבודה, ניתוח כל הגורמים ומציאת הדרך המתאימה והיעילה לביצוע הפעולה שהוגדרה כמטרת המערכת.

לצורך הדגמת הפתרון וחקירתו בחרנו בפרוייקט שלנו להתמקד בתחום הצבאי-בטחוני, שבו הצורך לבצע פעולות המסכנות חיי אדם ע"י רובוטים, הוא גדול מאוד.

אנו מתכוונים לפתח מערכת המדמה מערכת המפעילה רובוט שמסוגל להשיג ולתקוף מטרת אויב בשטח ללא צורך הפעלת אדם.

# 5. תיאור המערכת

הפקודות נשלחות ע"י משדר אל בקרת הרובוט בשטח

מצלמה מצלמת בזמן אמת את "הזירה"

הנעת הרובוט

תוכנת מחשב (1) מעבדת את התמונות ומזהה את תוואי "הזירה","האויב" ואובייקטים הנעים בו

תוכנת מחשב (2) מחשבת ע"י אלגוריתם את הדרך הקצרה

ליצירת מגע עם "האויב"

המערכת תכלול רובוט שמטרתו לנוע ולתקוף אויב שנע בשטח. המערכת תזהה את תוואי השטח ואת האובייקטים המצויים בו ע"י צילום ה"זירה" וניתוחה ע"י תוכנת מחשב. המערכת תזהה את השטח בו הרובוט יכול לנוע, את המכשולים המצויים בדרכו אל המטרה, לאחר מכן תוכנת מחשב תחשב עפ"י אלגוריתם מסוים את הדרך המהירה ביותר להשגת האויב שנע , ותחתור למגע.

# 6. קהל היעד

המערכת שאנו נפתח לצורך הדגמת מערכות אוטונומיות, מיועדת בעיקר לשימוש צבאי-בטחוני, אך מערכת זו רק תשמש כדוגמה לפיתוח מערכות אוטונומיות שיכלו להתפתח לתחומים אחרים ע"י כתיבת אלגוריתמים למטרות שונות, ותפנה גם ליעדים בתחומים אזרחיים וכו'

# 7**. היבט טכנולוגי**

## מיקרו מעבד

מיקרו מעבד הוא רכיב דיגיטאלי אשר ניתן לתכנות. כמו כן, הוא כולל בתוכו את כל הפעולות של יחידת עיבוד מרכזית. פקודות אלה הן פקודות אלמנטאריות ביותר, כאשר כל פקודה בנויה מרצף קצר של ביטים, שמשמעותו מובנת על-ידי המעבד. רצף זה קרוי שפת מכונה. המיקרו מעבד מחובר לכל הרכיבים והוא בעצם הרכיב השולט על פיקוד הרובוט ע"י שליחת אותות לכל הרכיבים.

כמו כן הוא מבצע את התקשורת לשליחת וקבלת נתונים מהרשת העצבית לרובוט.

### פעולת ומבנה המעבד

כל מעבד מורכב ממספר יחידות המבצעות פעולות שונות. במעבדים מודרניים המבוססים על שיטת [צינור עיבוד נתונים](http://he.wikipedia.org/wiki/%D7%A6%D7%99%D7%A0%D7%95%D7%A8_%D7%A2%D7%99%D7%91%D7%95%D7%93_%D7%A0%D7%AA%D7%95%D7%A0%D7%99%D7%9D) (Pipeline), כל אחת מיחידות אלו פועלת בנפרד.

לצורך ביצוע [פקודה](http://he.wikipedia.org/wiki/%D7%A4%D7%A7%D7%95%D7%93%D7%94_(%D7%9E%D7%97%D7%A9%D7%91)) בודדת המעבדים משתמשים באחד מהסטים של הפעולות הבאות

* - Fetchחישוב כתובת הפקודה הבאה לביצוע, וקריאת הפקודה הבאה לביצוע מהזיכרון.
* Decode - פיענוח הפקודה, והבאת ערכי ה[רגיסטרים](http://he.wikipedia.org/wiki/%D7%90%D7%95%D7%92%D7%A8_(%D7%9E%D7%97%D7%A9%D7%91%D7%99%D7%9D)) בהם הפקודה משתמשת.
* Execute - ביצוע הפעולה האריתמטית/לוגית שהפקודה דורשת באמצעות ה-ALU. במקרה שהפקודה היא פקודת חישוב, כאן מתבצע החישוב. במקרה והפקודה היא פעולת קריאה/כתיבה בזיכרון, כאן מחושבת הכתובת של הנתון בזיכרון. במקרה של פקודת הסתעפות (Branch/Jump), כאן מחושב קיום התנאי להסתעפות, ומתבצעת ההחלטה האם להסתעף, או לא.
* Memory - קריאה/כתיבה של נתון בזיכרון, במקרה של פקודת קריאה/כתיבה בזיכרון.
* Write Back - כתיבת תוצאת החישוב משלב הExecute או תוצאת הקריאה משלב הMemory אל רגיסטר היעד.

## תקשורת טורית ומקבילית

תקשורת טורית היא הדרך שבה מקשרים בין סימולטור הרשת העצבית לתכנת המיקרו מעבד. התקשורת אחראית על הקישור בין הסימולאטור רשת עצבית לתוכנה של המיקרו מעבד.

כידוע, בתקשורת טורית המידע עובר ביט אחר ביט, בעוד שבתקשורת מקבילית, כשמה כן היא, עוברים מספר ביטים במקביל בכל פרק זמן, מה שאומר לפי ההיגיון שתקשורת מקבילית היא הרבה יותר מהירה. כיום, כל חיבורי USB וה-MP3 שלנו למחשב עובדת בתקשורת טורית, למה זאת?

לחיבור מקבילי (פרללי Parallel) יש מס' ערוצי מידע מול החיבור הטורי (סריאלי Serial) שעובד על ערוץ אחד (או 2 הנפוץ יותר היום - פלוס ומינוס). החיבור הטורי עדיף מכיוון שאין לו הרבה בעיות שקיימות בחיבור הפרללי כמו רעש (Noise) שנפתר בחיבור סריאלי ע"י סיכוך שאי אפשר לשים עד קווים של החיבור הפרללי (שיותר דקים וקיים בהם רעש מוגבר). ז"א - יחס רעש לאות בחיבור סריאלי נמוך משמעותית מחיבור פרללי. עוד נקודה חשובה היא שבחיבור פרללי קיים ה ISI - Intersymbol interference שבעצם גורם לאות תתעוות ובכך בעצם מוריד את קצב, הבעיה הזאת לא קיימת בחיבור סריאלי. וכמובן, תקשורת פאללית מוגבלת למקסימום מרחק קצר מאוד ביחס לתקשורת סריאלית. ככה שלמרות שלתקשורת מקבילית יש יותר ערוצים בשביל להעביר מידע, הם מגיעים עם הרבה בעיות שמגבילות את הקצב העברה המקסימלי ושימוש נוח.

## מצלמה

מצלמה היא מכשיר היוצר תמונה ומאפשר לקבע אותה לאורך זמן. מבחינים בין מצלמה (באנגלית Stills Camera או Photo Camera) ובין מצלמת וידאו המצלמת רצף של תמונות, בדרך כלל באופן אלקטרוני (מצלמה המצלמת סרטים על גבי סרט צילום נקראת מסרטה). מקור השם העברי במילה "צלם" שמשמעותה דמות. מקור השם הלועזי "קאמרה" נובע מהמונח קאמרה אובסקורה, שפירושו בלטינית "חדר אפל". זה היה המנגנון המוקדם להטלת תמונות: חדר שלם שימש באופן דומה לחלק הפנימי של מצלמה מודרנית. באותה התקופה לא הייתה עדיין דרך לאחסן את התמונה (פרט לציורה), והיא אבדה ברגע שהסצנה המצולמת השתנתה.

המצלמה שאנחנו השתמשנו בה היא: Logitech QuickCam Pro 9000.

### ייצוג צבע במחשב:

יצוג צבע במחשב הוא הדרך בה מוגדר צבעו של כל פיקסל במסך כדי לשלוט בתצוגת המחשב.  
קיימים מספר קידודים לייצוג צבע באופן כמותי. במכשירי מדידה משתמשים בקואורדינטות צבע מדויקות ביותר, לדוגמה אהובי או XYZ. בדפוס משתמשים בדרך כלל בשיטה הקרויה CMYK בה יש דיו בארבעה צבעים - ציאן, מג'נטה, צהוב ושחור. תחושת הצבע הנוצרת בעת צפייה בדף המודפס נגרמת מאיסוף האור שמוחזר מהדף. על כל נקודה בדף עשויה להיות מונחת דיו בכמות שונה מכל אחד מארבעת הצבעים, וחלק האור שמוחזר מהדף הוא זה שאינו נבלע על ידי הדיו.  
באופן דומה, תחושת הצבע הנוצרת בעת צפייה בצג מחשב נגרמת מאיסוף האור הנפלט משלושה סוגי פיקסלים (אדום, ירוק וכחול) על ידי העין. לכן מכונה השיטה RGB - ראשי תיבות של אדום, ירוק, כחול. הצבע על הצג נקבע על ידי שלוש העוצמות האלה. נהוג לסמן, אם כן, את הצבעים לפי הכמות של אדום, ירוק וכחול שיש לשלב יחד על מנת ליצור את הגוון הדרוש באופן הבא: (R, G, B) כאשר R מייצג את כמות הצבע האדום הדרושה, B את כמות הצבע הכחול וכן הלאה .  
ישנן מספר שיטות לסימול העוצמות השונות. בשפת HTML (לעיצוב ממשקים של אתרי אינטרנט) ובמגוון יישומים נוספים מסמנים את הצבע על ידי רצף של שלושה מספרים הקסדצימליים דו ספרתיים, המייצגים את העוצמה היחסית של האור האדום, הירוק והכחול בצבע. בחלק משפות המחשב מוסיפים לפני קוד הצבע את הסימון סולמית (#). בשפות עיליות מסוימות ניתן לשלוט בצבע בקלות יתרה תוך קביעת שם הצבע במקום הגדרה מספרית. למשל, כדי לכתוב טקסט בצבע כתום בשפה שבה נכתבים הערכים בוויקיפדיה, משתמשים בפקודה <span style="color:orange;"> תפוזים </ span>.  
קביעת העוצמות של מרכיבי הצבע השונים נעשית על ידי השמת המספרים השלמים בין 0 ל 255 לכל אחד מן המשתנים R, G ו-B, כאשר 0 פירושו לא להאיר בכלל את הפיקסל, ו -255 מסמל הארה במלוא העוצמה. (לעתים מסמנים עוצמה יחסית על ידי חלוקה ב -255, ואז הערכים הם בין 0 ל -1). לשם הנוחות נהוג לתרגם את מספרים אילו לבסיס הקסדצימלי (בסיס 16) בו יש צורך רק ב -2 תווים כדי לייצג מספרים בטווח הנ"ל. באופן זה מקצים לכל אחד משלושת צבעי היסוד (RGB) שני תווים משמאל לימין. בכל צבע הטווח נע מ -00 ל-FF (מ -0 ל -255 בבסיס דצימלי) שמבטאים רוויה של אותו הצבע. מכיוון שבכל מרכיב ניתן ליצור 256 גוונים שונים, מספר הצבעים השונים שניתן לייצג באופן זה הוא 256 בחזקת שלוש שהם 16,777,216 צבעים שונים.

**דוגמאות לצבעים קלאסים:**

* פיקסל אדום ייוצג כך: R=255, G=0, B=0 או כך: R=FF, G=00, B=00 או בכתיבה מקוצרת: .FF0000
* פיקסל מג'נטה ייוצג כך: R=255, G=0, B=255 או כך: R=FF, G=00, B=FF או בכתיבה מקוצרת. FF00FF:
* פיקסל שחור ייוצג כך R=0, G=0, B=0 או כך (בכתיבה מקוצרת): 000000.
* פיקסל לבן יוצג כך R=255, G=255, B=255 או כך (בכתיבה מקוצרת): FFFFFF.

את הכתיבה המקוצרת לעתים מכנים RRGGBB שכן שתי הספרות השמאליות מייצגות את כמות האדום, שתי האמצעיות את כמות הירוק והימניות את הכחול..

טבלה שמדגימה את צבעי היסוד:

|  |  |  |
| --- | --- | --- |
| מיוצג בששה אפסים | 000000 | שחור |
| שני תווים ראשונים | **FF**0000 | אדום |
| שני תווים אמצעיים | 00**FF**00 | ירוק |
| שני תווים אחרונים | 0000**FF** | כחול |
| מיוצג בששה F | FFFFFF | לבן |

## תקשורת למחשב

Universal Serial Bus או בקיצור USB הוא תקן תקשורת בין מחשבים וציוד היקפי. הוא מיושם לרוב בדרך של יציאה טורית הנפוצה בעיקר במחשבי פנטיום 2 ומעלה ובמחשבי מקינטוש מדגם G3 ומעלה. במערכות ההפעלה של מיקרוסופט, היציאה נתמכה בצורה חלקית החל מחלונות 95 OSR2 ובצורה מלאה בחלונות 98 ומעלה.

ממשק ה-USB בא לספק יציאה סטנדרטית מהירה ואמינה לחיבור מגוון רחב של התקנים. ממשק ה-USB בא להחליף את היציאות הטוריות והמקביליות הישנות (COM, LPT) המוגבלות במספר ההתקנים שניתן לחבר אליהן, והינן איטיות וקשות להתקנה. לעומת יציאות אלה, מאפשר ממשק ה-USB חיבור מספר רב של התקנים למחשב (עד 127), כאשר ניהול המשאבים הוא אוטומטי. הממשק תומך בהעברת נתונים מהירה מאוד של עד 12 מגהביט לשנייה בגרסת 1.1 USB, ועד 480 מגהביט לשנייה בגרסת USB 2.0. כמו כן, הממשק מאפשר לחבר ולנתק התקנים ללא צורך באתחול המחשב.

כבל USB מכיל 4 גידים: שניים מהם משמשים לצורך העברת נתונים ושניים מהם מספקים מתח חשמלי נמוך של 5 וולט. מתח חשמלי זה משמש לשם הפעלת התקנים בעלי צריכת חשמל נמוכה, כגון: מקלדת, עכבר, ומצלמת אינטרנט.

התקני USB יכולים לצרוך זרם עד לסך של 500 מיליאמפר מממשק ה-USB אליו הם מחוברים. אם סך הזרם הנצרך הוא פחות מ-50 מיליאמפר, ממשק ה-USB לא יופעל.

## התוכנה Visual Basic:

Visual Basicבקיצור (VB) היא שפת תכנות מונחית אירועים שפותחה על ידי חברת מיקרוסופט. השפה מתאפיינת בקלות ובנוחות פיתוח ובהתאמה לממשקי משתמש ומערכות עיבוד נתונים. השפה הושקה לראשונה כ-Visual Basic 1.0 ב -1991, וגרסתה האחרונה יצאה בשנת 2009 . השפה מיועדת לסביבת מערכת ההפעלה Windows בלבד.

רקע:

יסודותיה בשפת Basic שהייתה שפת תכנות פופולרית בשנות השמונים, אף שהייתה חסרה מבני בקרה בסיסיים ולכן הקוד שנכתב בה נחשב "קוד ספגטי".בעיות אלה נפתרו ב-Visual Basic.

הגרסה הראשונה פותחה והוצגה לראשונה ב - Windows בכנס העולם באטלנטה ב -20 במאי 1991, יחד עם חלונות 3.1. באותה תקופה לא היו כמעט כלי פיתוח לכתיבת ממשקים למערכת ההפעלה Windows, שהייתה עוד בחיתוליה. השימוש שייעדו לשפה במיקרוסופט היה יצירת אבות טיפוס של תוכנות, שיכתבו לאחר מכן בשפות אחרות. שפה זו שאבה רבות מהרעיונות והתחביר של שפת פסקל, כמעט בכל היבט אפשרי: צורת הגדרת בלוקים, מבנה הפונקציה, משפטי בקרה, אופרטורים ועוד, אך לא הייתה יעילה וגמישה תכנותית כפסקל.

בשלביה הראשוניים השפה הייתה חלשה למדי ובעלת ביצועים נחותים ותחביר פרימיטיבי וחסר גמישות, ולכן לא התאימה למערכות מסחריות, אך במהלך השנים, במיוחד החל מגרסאות 5 ו -6 שבהן הוספו פקדים שיתופיים, יכולת של הידור של הפרויקט לקובצי, DLL EXE ו-ActiveX וכן פקדי משתמש ומסכי MDI, היא נעשתה חזקה למדי והתאימה להיבטים תכנותיים מסוימים.

פיתוח השפה הופסק אחרי גרסה 6, ובמקומה פותחה-Visual Basic.NET, כחלק מ -. NET ולשפה זו מומשו כל עקרונות התכנות מונחה עצמים, השפה הוגמשה וחוזקה, וחוברה אל פלטפורמת הדוט נט.

בעקבות או במקביל להצלחה של VB יצאו תוכנות דומות כמו דלפי של בורלנד ומחולל היישומים גישה של מיקרוסופט.

### מאפיינים של Visual Basic:

• תכנות ויזואלי של הממשק: התכנות הוויזואלי, שבו חלק ממלאכת התכנות מומר בגרירת פקדים (שהם אובייקטים תכנותיים שלמים) וציורם על - גבי טופס. עדכון המאפיינים של הפקדים נעשה בבחירת עכבר מתוך תיבות רשימה קיימות, והפעלתם באמצעות קריאה לשיטות הקיימות בהן. התפיסה הזו הורחבה לכלים בונה המחלקה שמאפשר בתכנות ויזואלי ליצור מחלקות, ולבונה השאילתה המאפשר ליצור שאילתות SQL בצורה גרפית. כמו כן, הפרדיגמה התכנותית ב-Visual Basic היא שתחילה יוצרים את הממשק הגרפי בציור הפקדים והגדרתם ולבסוף מגיעים אל הפונקציונאליות של הקוד שמאחורי האובייקטים של הממשק, כך שהולכים מן הכלל אל הפרט.

• תכנות מונחה אירועים: לכל אחד מהפקדים המוצגים בממשק המשתמש מוגדרת בתוכנית סדרה של שגרות, המתבצעות כאשר מתרחש האירוע המתאים, כגון כניסה לפקד, יציאה ממנו, פעולת עכבר או מקלדת עליו, שינוי תוכנו ועוד. הביצוע הסדרתי המסורתי של תוכנת מחשב, באופן לינארי פקודה אחר פקודה, הוחלף בשפה זו בביצוע מונחה אירועים: כל אירוע מפעיל את השגרה המתאימה לו.

• פשטות וידידותיות: התכנות בשפה זו פשוט וקל יחסית בניגוד לשפות שיש בהן מורכבות וסיבוכיות רבה (טיפוס מחרוזת אחד לעומת מספר רב של טיפוסי מחרוזת בשפת C++). בשפה זו קיימת אוריינטציה אנושית, כך שלפקודות והמילים השמורות והפונקציות של השפה יש משמעות באנגלית פשוטה שאינה טכנית. בשפה זו מוטמעות פונקציות רבות כחלק ממנה, כך שקל ופשוט למשל לבצע פעולות על מחרוזות. גם התכנות הוויזואלי מפשט את הכתיבה בשפה זו. אבל מאידך המחיר בתשלום על הפשטות הוא בהעדר גמישות תכנותית, ומוגבלות רבה בניסיון לבצע דברים מורכבים.

• אוטומציה פנימית: השפה לוקחת על עצמה חלק מעבודת המתכנת היכן שהדבר אפשרי: כך למשל בשפה ניתן להשתמש במשתנים מבלי להגדירם כלל. בפעולות השמה מתבצעת המרה (קסטינג) אוטומטית בין סוגי משתנים כאשר הדבר אפשרי. ניתן להגדיר משתנה מסוג משתנה (variant) שיכול לשנות את הטיפוס שלו לכל סוג במהלך הריצה, לפי סוג הנתונים שמשימים אליו. שימוש נפוץ בו הוא השמה של פונקציה שמחזירה מערך בגודל שאיננו ידוע מראש. אפשר להגדיר מערכים דינמיים שניתן לשנות את גודלם תוך כדי ריצה. בנוסף, קיימות בשפה בדיקות פנימיות שונות, ללכידת באגים נפוצים כמו למשל לכידת חריגה מאיברי מערך.

• מודולריות: שפת VB הינה שפה "פתוחת קצוות" (Open-Ended). לשפה זו היו אפשרויות הרחבה, באמצעות פקדי VBX, שנכתבו בשפת. יורשיהם של פקדים היו פקדי OCX (שכיום מכונים פקדי ,(ActiveX וניתן לכותבם בשפות שונות, באמצעות טכנולוגיית .COM הפקדים הינם למעשה מיני תוכנות, המשמשים כאבני הבניין של התוכנית, ויוצרים בה מבנה מודולרי. בהמשך נוצרו פקדים נוספים לפי נושאים ותחומים, באופן שכיסו חלק נרחב מנושאי התכנות הקיימים, והביאו נושאי תכנות מסובכים לידי פשטות תכנותית, החל מפקד טורי שמטפל ביציאות המחשב ופקד היוצר תקשורת לאינטרנט וכלה בפקד שנותן פונקציונליות של לוח שנה. הפקדים ופקדי המשתמש של VB הינם למעשה יישום של תכנות מונחה עצמים.

•שפת תכנות סינתטית: שפה שהיא שילוב של שפה מפורשת ושפה מהדרת. השפה במקורה הינה שפה מפורשת, בה כל פקודה מורצת ברגע שמגיעים אליה, כך שאין כל צורך להדר את הקוד כולו, לפני כל ריצת ניסוי, ואפשר גם לשנות קוד תוך כדי ריצה. ומצד שני התוכנית המוגמרת, זו המוגשת למשתמש, עוברת הידור לשפת מכונה, כך שמתקבלת לבסוף תוכנת יעילה יותר (הגם שהיא עדיין איטית משמעותית משפות תכנות כמו .(C

הטופס:

הטופס הוא לב ליבה של שפה זו. התפיסה היא שמתחילים בשלב הוויזואלי.יוצרים את כל הטפסים של הפרויקט, נותנים תפקודיות ראשונית לכל טופס, באמצעות ציור של פקדים מובנים ושל פקדי Active X על גבי הטופס (שווה להכרזה של משתנה). קינפוג מאפייני הפקדים שבאמצעות חלון התכונות, וכתיבת קוד VB היכן שצריך (גישה נוחה לקוד הרלוונטי בהקלקה כפולה על הטופס או על הפקד). בחלון הקוד אפשר לקבל את רשימת כל אירועים הטופס ופקדיו, ובבחירת אירוע נוצרת לה אוטומטית שגרת אירוע עם כל הפרמטרים הרלוונטיים. בשגרה יש לכתוב קוד או לקרוא לשגרה / פונקציה שנמצאת במקום אחר. במקרה שעדיין הפונקציונאליות הדרושה חסרה, אפשר ליצור הפניה לפונקציות בקובצי DLL או לקובצי COM, או להכריז על פונקציות שנמצאת ב-APIs של המערכת, או לכתוב את הקוד הנצרך במודול או במחלקה.

בגישה הזו הטופס משמש כמיכל (מיכל) שמחזיק בתוכו את הפקדים השונים, כך שכאשר מעתיקים את הטופס כל הפקדים שבו מועתקים אף הם. בנוסף הטופס משמש גם כאוסף לכל הפקדים (גביה) וכך אפשר באמצעות הטופס לעבור על כל הפקדים באופן סדרתי באמצעות לולאה. ישנם חלק מהפקדים שיש להם מאפיין של מיכל, כך שהם יכולים לשמש בטופס כתת - טופס, ולהכיל בתוכם פקדים שונים (יעיל לקבץ פקדים סביב נושא מסוים). ויש חלק מהפקדים שמכילים בתוכם אוסף של פריטים כמו למשל פקדי תיבת רשימה ותיבה משולבת שמכילים אוסף של רשימות.

### User Control:

זהו טיפוס פקד מיוחד שנוצר בידי המתכנת, שהינו מעין מיני טופס, ובעצם הוא מעין ActiveX פנימי של VB. כשם שאפשר ליצור בשפות תכנות טיפוסי משתנים, המבוססים על משתנים קיימים, כך אפשר ב-VB ליצור טיפוסי פקדים חדשים, המבוססים על פקדים קיימים או על שיטות גרפיות של השפה. יוצרים אותו כשם שיוצרים טופס, וברגע שהוא קיים במערכת, נוסף אייקון שלו בסרגל הכלים ואפשר לצייר אותו בכל טופס בפרויקט כשם שמציירים פקד רגיל. זהו למעשה יישום של התפיסה המונחת עצמים והתפיסה המודולרית. לרכז פונקציונאליות במחלקה סגורה, שמתקשרת עם החוץ באמצעות תכונות ומתודות חיצוניות. יחודו על סתם מחלקה שהוא בנוסף לקוד יכול להיות בעל מופע חזותי.שימוש פשוט יכול להעשות בו על מנת להכליל טופס אחד בטופס אחר, אך כמובן שאפשר ליצור באמצעות טיפוס זה פקדים מורכבים. הפקד מתאפיין בקלות שבה ניתן ליצור אותו, ויתרון נוסף בו שניתן לראות את האפקט התכנותי שבו בטופס אחר, גם שלא בזמן ריצה. VB הורישה את הטיפוס הזה לסביבת הדוט נט.

# 8. היבט מדעי - STS

### אלגוריתם חיפוש הדרך הכי קצרה

לא מספיק שיש את הרובוט הרודף עם הרבה אמצאים טכנולוגים מפותחים אם לוקח לרובוט המון זמן וקצת מזל בכדי למצוא את יריבו ולחסל אותו. אנו רצינו שבתוך זירה אקראית הרובוט הרודף יוכל למצוא את הדרך הכי קצרה לרובוט הנרדף ולחסל אותו בזמן הכי קצר בלי שיסתובב באקראית בתוך הזירה עד שימצא אותו. הבעיה שעמדה מולנו הייתה איך למצוא את הדרך הכי קצרה בתוך זירה שהתנאים ומיקום של כל דבר בה משתנה בכל פעם ששולחים את הרובוטים חד נגד השני, במיוחד כשמיקום הרובוטים ההתחלתי גם כן משתנה מפעם לפעם. ישנם הרבה אלגוריתמים הבנויים בשביל שפות התכנות המאפשרות מציאת הדרך הכי קצרה מהמיקום ההתחלתי ליעד הרצוי. לדוגמא יש את האלגוריתמים:

* אלגוריתם החיפוש A\*
* גיזום אלפא-ביתא
* אלגוריתם תקציד ((Branch and Bound
* עץ מינימקס
* אלגוריתם דייקסטרה

האלגוריתמים הנ"ל שייכים לסוג אלגוריתמים בשם "חיפוש קומבינטורית".

אלגוריתמי "חיפוש קומבינטורית" משיגים יעילות על ידי הקטנת גודל יעיל של איזור החיפוש או היוריסטיות המעסיקים כאשר משתמשים בכמה אלגוריתמים, אחד אחרי השני, מובטח למצוא את הפתרון האופטימאלי, בעוד אחרים יכולים רק להחזיר את הפתרון הטוב ביותר בשטח שאותו אנו חוקרים.

תקציר של כל האלגוריתמים כולל פירוט על האלגוריתם שהשתמשנו בו:

**עץ מינימקס:**

בתורת המשחקים, **עץ מינימקס** הוא עץ (סוג של מבנה נתונים) הפורס את האפשרויות למשחק של שחקן א', את התגובות של שחקן ב' לכל פעולה של שחקן א', את תגובותיו של א' לתגובותיו של ב' וכך הלאה. מעשית מוגבל עומקו של העץ על ידי הזמן וזיכרון המחשב). העומדים לרשותנו (לדוגמה, מחשבי שחמט בונים עצים של כ-9 מהלכים קדימה

העלים בעץ שנוצר הם מצבים סטטיים שנגיע אליהם לאחר רצף של מהלכים (הנתיבים בעץ הם למעשה תרחישים אפשריים). ניתן ציון לכל מצב סטטי שכזה (מצב סטטי בלוח שחמט לדוגמה), שישקף כמה המצב טוב מבחינתנו.

מובן שאם נבחר במהלך (ענף) המוביל לעלה עם הציון הגבוה ביותר, לא מובטח לנו שאכן נגיע לאותו עלה. העלה הטוב ביותר הוא התסריט האופטימאלי מבחינתנו, וניתן להניח שיריב לא יוביל אותנו בהמשך הנתיב אלא יסיט אותו לנתיב הטוב ביותר מבחינתו. משפט המינימקס מאפשר לנו לדעת מה המהלך הטוב ביותר שנוכל לעשות, על בסיס המידע הנמצא בעץ.

נסרוק את העץ החל בעלים, דרך הקודקודים, (קודקוד הוא צומת פנימי בעץ) שמעליהם עד השורש (למטה למעלה) הציון שניתן לכל קודקוד הוא הערך הגבוה ביותר של העלים/הקודקודים שתחתיו, אם אותו קודקוד מסמל מהלך שלי, והערך הנמוך ביותר של העלים/קודקודים שתחתיו אם אותו קודקוד מסמל מהלך של היריב (כי ברור שהוא יבחר באפשרות הטובה ביותר בשבילו - הכי גרועה בשבילי).

כשנגיע לשורש, נבחר בקודקוד שמתחת לשורש עם הציון הטוב ביותר.

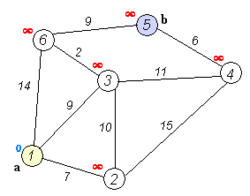
בעיה קשה בבניית עץ מינימקס היא הזיכרון הרב שהוא צורך. מספר הקודקודים שיש לפתח עולה בטור הנדסי ככל שנעמיק את החיפוש. קיימת שיטת "גיזום" המבטלת בנייה של ענפים שברור לנו עוד בשלב מוקדם כי הם לא מועילים לחיפוש שלנו. ניתן כך להקטין מספר הצמתים בעץ לשורש המספר שהיה מתקבל ללא הגיזום (בממוצע). שיטה זו נקראת גיזום אלפא-ביתא.

**גיזום אלפא-ביתא:**

**גיזום אלפא-ביתא** היא שיטת אופטימיזציה עבור עצי-חיפוש מסוג מינמקס. מטרת השיטה לצמצם את מספר תתי העצים עליהם יש להלך בעת הערכת מהלך אפשרי בעץ מינמקס. שמה של השיטה ניתן לה כאנלוגיה לגיזום עצים הגיזום הוא אלגוריתם. אופטימיזציה קלאסי, במובן שאינו משנה את התוצאה שהיה מחזיר האלגוריתם המקורי (חיפוש מינמקס), אלא רק מחזיר אותה תוצאה בזמן קצר יותר. במקרה הגרוע ביותר, האלגוריתם עובד בסיבוביות זהה לזו של חיפוש מינמקס רגיל. עם זאת, באופן פרקטי ותוך שימוש בשיטות משלימות, מספק האלגוריתם את התוצאה המקורית של מינמקס, בזמן קצר בהרבה. אופן הגיזום הוא פשוט: במהלך החיפוש לעומק ניתן לזנוח פתרונות חלקיים ברגע שברור שהם גרועים מפתרונות שכבר ראינו.

**אלגוריתם תקציד (Branch and Bound)**

תקציד (BB או B & B) הוא אלגוריתם כללי למציאת פתרונות אופטימליים שלבעיות אופטימיזציה שונות, בעיקר אופטימיזציה דיסקרטית ולא קומבינטורית. הוא מורכב ספירה שיטתית של כל הפתרונות מועמדים, בהם תת גדולות של מועמדים עקרים מבוטלים בהמוניהם, באמצעות לתחום ההערכה העליון והתחתון של כמו להיות מותאם.  
שיטת הוצע לראשונה על ידי AH לנד ו- AG דויג בשנת 1960 עבור תכנות דיסקרטית.

****אך האלגוריתם הכי שימושי שמצאנו היה האלגוריתם חיפוש המפורסם "**אלגוריתם דייקסטרה**".

**אלגוריתם דייקסטרה**, פרי יצירתו של אדסחר דייקסטרה, פותר את בעיית מציאת המסלול הקצר ביותר מנקודה בגרף ליעד. מכיוון שניתן למצוא באמצעות אלגוריתם זה, בזמן זהה, את המסלולים המהירים לכל הנקודות בגרף, בעיה זאת נקראת לעתים מציאת המסלולים הקצרים מנקודה.  
האלגוריתם עובד על גרף נתון, מכוון או לא מכוון, בעל משקולות אי - שליליות על הקשתות. המשקולות בגרף מסמלות מרחק. משמעותו של המסלול הקצר ביותר בין שתי נקודות היא המסלול בעל סכום המשקולות הנמוך ביותר בין שתי הנקודות.  
תוצאת האלגוריתם של דייקסטרה זהה לתוצאת אלגוריתם בלמן - פורד אך אלגוריתם בלמן - פורד פועל גם על גרפים הכוללים קשתות שמשקלן שלילי. לעומת זאת, זמן הריצה של אלגוריתם דייקסטרה מהיר יותר.

**פעולת האלגוריתם**  
בתמצית ניתן לסכם את פעולת האלגוריתם כך:  
עבור כל קודקוד, מסומן האם ביקרו בו או לא ומה מרחקו מקודקוד המקור.בהתחלה כל הקודקודים מסומנים כאילו לא ביקרו בהם, ומרחקם מוגדר כאינסוף. לולאת האלגוריתם:  
כל עוד נותרו קודקודים שלא ביקרנו בהם:  
מסמנים את X (הקודקוד הנוכחי. באיטרציה הראשונה זהו ים) כקודקוד שביקרו בו.  
עבור כל קודקוד Y שהוא שכן של X וגם לא ביקרנו בו:  
Y מעודכן, כך שמרחקו יהיה שווה לערך המינימלי בין שני ערכים: מרחקו הנוכחי,ומשקל הקשת המחברת בין X לבין Y בתוספת המרחק בין S ל-X.  
בוחרים קודקוד X חדש בתור הקודקוד שמרחקו בשלב הזה מצומת המקור Sהוא הקצר ביותר מבין כל הקודקודים בגרף שטרם ביקרנו בהם.  
האלגוריתם מסתיים כאשר ביקרנו בכל הקודקודים.  
סיבוכיות האלגוריתם תלויה במבנה הנתונים השומר את הקודקודים שטרם ביקרנו בהם, אם מדובר ברשימה או במערך, הסיבוכיות היא ב-**\ O(|V|^2)** , אם משתמשים בערימה בינארית סיבוכיות הריצה משתפרת ל \ O(|E|log|V|) ואם משתמשים בערימת פיבונאצ'י הסיבוכיות היא\ O(|E|+|V|log|V|) -.

**רעיון האלגוריתם**

אלגוריתם דייקסטרה מבוסס על שני רעיונות אלגוריתמיים. הראשון הוא שימוש בתת - בעיות כדי לפתור את בעיית המסלול הקצר ביותר.  
יהי P המסלול הקצר ביותר מקודקוד S לקודקוד ט ייתכן ש-P עובר בקודקודים רבים. עבור כל קודקוד V, החלק ממסלול P שמגיע מ-S אליו מהווה את הדרך הקצרה ביותר האפשרית להגיע מ-S ל-V. אם לא כן, ניתן היה לקצר את המסלול מ-S ל-V ובכך להקטין את המרחק הכולל מ-S ל-T, ואם כך P אינו המסלול הקצר ביותר, בסתירה להגדרתו.  
עקרון זה מאפשר לבסס את מציאת המסלול הקצר ביותר על מעין תכנון דינאמי.הפתרון לבעיה מצוי בפתרון לתת הבעיות שלה. המסלול הקצר ביותר מ-S ל-T יימצא על סמך המסלולים הקצרים ביותר מ-S לקודקודים שקדמו ל-T.  
עיקרון שני הוא חמדני. בכל שלב מצויים במצב שבו נקבע מרחקם הקצר ביותר מ-S של חלק מהקודקודים, אך לא נקבע מרחקם של קודקודים אחרים. הרעיון החמדני קובע כי בכל שלב יש להתבונן באלו מהקודקודים שלא נקבע מרחקם ושיש אליהם קשת מקודקודים שמרחקם כבר נקבע, וליטול מקבוצה זו את הקודקוד שהדרך אליו מ-S, שעוברת כולה בקודקודים שכבר נקבע מרחקם, היא הקצרה ביותר.  
ההגיון בבסיס בחירה זו הוא שאין קצר יותר מהקצר ביותר. לא ניתן להגיע אל קודקוד בדרך קצרה יותר מאשר על ידי בחירת הדרכים הקצרות ביותר הזמינות בכל שלב. עיקרון זה אינו נכון כאשר יש משקולות שליליות לקשתות. ייתכן שדווקא קשת ארוכה תוביל אותנו לקשת שתסמל מרחק שלילי גדול, וסכום המרחק הכולל לקודקוד יהיה נמוך יותר מאשר אם נלך בעקבות הקשת הקצרה שנבחרה מיידית.

האלגוריתם של דייקסטרה הוא ממש טוב בשבילנו אולם אנו צריכים אלגוריתם יותר מדיוק שלא ממפה את כל המפה ורק אז מוצא את הדרך הכי קצרה. רצינו למצוא אלגוריתם שיוצא מתוך דייקסטרה, שמבוסס עליו, שעושה את העבודה בפחות זמן כדי שכל פעם שהרובוטים נעים הוא יחשב את הדרך הכי קצרה החדשה. בשביל דבר כזה היינו צריכים להתמקד יותר למקום יותר קטן ולא את כל הזירה כולה. לכן החלטנו להישתמש באלגוריתם החיפוש הכי ייעודי בשבילנו המכונה בשם A\*.

## אלגוריתם :A\*

אלגוריתם A\* יכול להיות מסובך למתחילים. אמנם יש מאמרים רבים ברשת המסבירים \*A רוב כתובים לאנשים שמבינים את הבסיס כבר. מאמר זה מיועד למתחילים נכון.

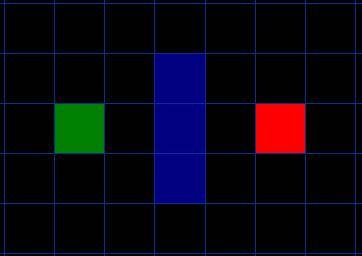
מאמר זה אינו מנסה להיות עבודה סופית בנושא. במקום זאת הוא מתאר את היסודות ומכין אותך לצאת ולקרוא את כל אותם חומרים אחרים ולהבין מה הם מדברים. קישורים כמה מן הטובים ביותר ניתנים בסוף מאמר זה, על פי קריאה נוספת.

לבסוף, מאמר זה לא תוכנית ספציפית. אתה אמור להיות מסוגל להתאים את מה כאן על כל שפה למחשב. כפי שאפשר לצפות, עם זאת, כללתי קישור לתוכנית מדגם בסוף מאמר זה. חבילת המדגם מכיל שתי גרסאות: אחת ב-C + + ואחד בליץ בסיסי. הוא מכיל גם הרצה אם אתה רק רוצה לראות \* בפעולה.

אבל אנחנו מקדים את עצמנו. בואו נתחיל מההתחלה ...

### מבוא: שטח החיפוש

נניח שיש לנו מי שרוצה להגיע מנקודה א 'לנקודה ב' נניח כי הקיר המפריד בין שתי נקודות. זו באה לידי ביטוי בהמשך, עם ירוק להיות נקודת התחלה, ואדום להיות ב 'נקודת סיום, ואת ריבועים כחולים מלאים להיות קיר באמצע.



הדבר הראשון שאתה צריך לשים לב הוא שיש לנו לחלק את שטח החיפוש שלנו לתוך רשת רבוע. לפשט את שטח החיפוש, כפי שעשינו כאן, הוא הצעד הראשון של מציאת הדרך הכי קצרה. שיטה מסוימת מפחית שטח החיפוש שלנו למערך 2 פשוט ממדי. כל פריט במערך מייצג את אחד הריבועים על רשת, ומעמדו נרשמת "ריק" או "מלא". נתיב נמצא בידי להבין איזה ריבועים שאנחנו צריכים לקחת כדי להגיע מ-A ל-B לאחר נתיב נמצא, האדם שלנו עובר במרכז הכיכר 1 למרכז הבא עד היעד הוא הגיע.

נקודות אלה מרכז נקראים "צמתים". כשאתה קורא על מציאת הדרך הכי קצרה במקום אחר, אתה בדרך כלל רואה אנשים מדברים צמתים. למה לא פשוט לקרוא להם ריבועים? כי אפשר לחלק את השטח שבו אתה מעוניין למצוא את הדרך הכי קצרה שלך למשהו אחר מאשר ריבועים. הם יכולים להיות מלבנים, משושים, משולשים או כל צורה, באמת. צמתים יכולים להיות ממוקם בכל מקום בתוך הצורות - במרכז או לאורך הקצוות, או בכל מקום אחר. אנחנו משתמשים במערכת זו, עם זאת, כי זה הכי פשוט.

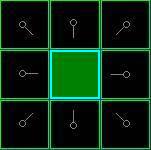
### החל חיפוש

ברגע שיש לנו פשוט יותר שטח החיפוש שלנו למספר לניהול של צמתים, כפי שעשינו עם פריסת רשת לעיל, השלב הבא הוא לערוך חיפוש כדי למצוא את הנתיב הקצר ביותר. אנו עושים זאת על ידי הפעלת בנקודה, בדיקת ריבועים סמוכים, ובאופן כללי מחפש כלפי חוץ עד שנמצא היעד שלנו.

אנחנו מתחילים את החיפוש על ידי ביצוע הפעולות הבאות:

1. מתחילים בנקודת המוצא ולהוסיף אותו "רשימה פתוחה" של ריבועים כדי להיחשב. הרשימה פתוחה היא כמו סוג של רשימת קניות. כרגע יש רק פריט אחד ברשימה, אבל יהיה לנו יותר מאוחר יותר. הוא מכיל ריבועים שעלולים ליפול בדרך שאתה רוצה לקחת, אבל אולי לא. בעיקרון, זו רשימה של ריבועים שצריך לבדוק.
2. תראו את כל הריבועים ניתן להשגה או "ריקים" הסמוכים לנקודת המוצא, תוך התעלמות ריבועים עם קירות, מים, או שטח לא חוקי אחר. להוסיף אותם לרשימת פתוח, יותר מדי. עבור כל אחד הריבועים האלה, לשמור להצביע בשם "ריבוע האם" שלו. את הדברים האלה בכיכר הורה חשוב כאשר אנחנו רוצים לאתר את דרכנו. זה יוסבר בהמשך.
3. זרוק את הכיכר החל מרשימת פתוח, ולהוסיף אותו "רשימה סגורה" של ריבועים כי אתה לא צריך להסתכל שוב עכשיו.

בשלב זה, אתה צריך משהו כמו באיור הבא. באיור זה, ריבוע ירוק כהה במרכז הכיכר הוא החל שלך. זה המתואר תכלת כדי לציין את הכיכר נוספה לרשימה הסגורה. כל הריבועים הסמוכים כעת ברשימה פתוחה של הריבועים לבדיקה, והם מתוארים בירוק בהיר. לכל מצביע האפור מצביע בחזרה לחברה האם שלה, שהיא הכיכר החל.



לאחר מכן, אנו בוחרים באחת הכיכרות הסמוכות ברשימה פתוחה, פחות או יותר לחזור על התהליך מוקדם יותר, כמפורט להלן. אך הכיכר לא נבחר? 1 עם עלות ה-F הנמוך ביותר.

### נתיב ניקוד

המפתח בקביעת ריבועים לשימוש בעת חישוב הנתיב הוא המשוואה הבאה:

F = G + H

איפה

* G = עלות התנועה לנוע מנקודת ההתחלה לנקודת ההתחלה נתון על הרשת, בעקבות שביל שנוצר כדי להגיע לשם.
* H = עלות התנועה המשוער לעבור מרובע נתון על רשת אל היעד הסופי, לנקודה ב ', זו המכונה לעתים קרובות האוריסטי, מה שיכול להיות קצת מבלבל. הסיבה מדוע היא נקראת כך משום שהיא ניחוש. אנחנו באמת לא יודע את המרחק בפועל עד שנמצא את הדרך, כי כל מיני דברים יכולים להיות בדרך (קירות, מים וכו '). ניתנת לך דרך אחת לחשב את H במדריך זה, אבל יש רבים אחרים שתוכלו למצוא במאמרים אחרים באינטרנט.

הדרך שלנו נוצר על ידי שוב ושוב עובר רשימת הפתוח שלנו ובחירת מרובע עם ציון ה-F הנמוך ביותר. תהליך זה יתואר ביתר פירוט קצת יותר במאמר. ראשית אני של et להסתכל יותר מקרוב על הדרך שבה אנו מחשבים את המשוואה.

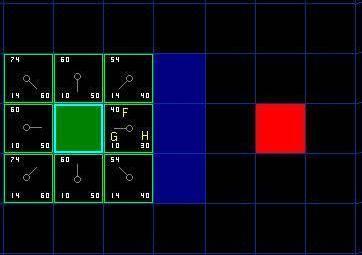
כפי שתואר לעיל, G היא העלות התנועה לנוע מנקודת ההתחלה לנקודת ההתחלה ניתן באמצעות הנתיב שנוצר כדי להגיע לשם. בדוגמה זו, נוכל להקצות העלות של 10 עד כל ריבוע אופקי או אנכי עבר, ואת העלות של 14 למהלך באלכסון. אנו משתמשים אלה מספרים כי המרחק בפועל לנוע באלכסון הוא השורש הריבועי של 2 (לא לפחד), או כ 1.414 מהעלות הכוללת של העברת אופקית או אנכית. אנו משתמשים 10 ו 14 למען הפשטות. היחס הוא צודק, ואנחנו להימנע מהצורך לחשב שורש ריבועי ואנחנו למנוע שברים עשרוניים. זה לא רק בגלל שאנחנו מטומטמים ולא אוהבים מתמטיקה. באמצעות מספרים שלמים כאלה הוא הרבה יותר מהר על המחשב גם. כפי שאתה בקרוב לגלות, מציאת הדרך הכי קצרה יכול להיות איטית מאוד אם אתה לא משתמש קיצורי דרך כאלה.

מאז אנחנו חישוב עלות G בדרך מסוימת לכיכר נתון, הדרך להבין את העלות G של הכיכר, כי היא לקחת את העלות G של האם, ולאחר מכן להוסיף 10 או 14, תלוי אם זה באלכסון או אורתוגונליים (לא באלכסון) מהכיכר כי האב. הצורך בשיטה זו תתברר עוד קצת על בדוגמה זו, ככל שאנחנו יותר מרובע מן הכיכר החל.

H ניתן להעריך במגוון דרכים, השיטה שאנו משתמשים כאן נקראת שיטת מנהטן, שבו אתה לחשב את מספר הריבועים הכולל נע אופקית ואנכית להגיע לכיכר היעד מן הכיכר הנוכחית, תוך התעלמות תנועה אלכסונית, והתעלמות מכשולים העלולים להיות בדרך. לאחר מכן להכפיל את סך של 10, העלות שלנו להעברת מרובע אופקית או אנכית. זה (כנראה) בשם שיטת מנהטן כי זה כמו לחשב את מספר רחובות ממקום אחד למקום אחר, שם לא ניתן לחצות את הרחוב באלכסון.

קריאת תיאור זה, ייתכן לנחש האוריסטי הוא רק הערכה גסה של המרחק הנותר בין הכיכר הנוכחית היעד "בקו אווירי". זה לא המקרה. אנחנו באמת מנסים לאמוד את המרחק הנותר לאורך השביל (שהיא בדרך כלל יותר). ככל הערכה שלנו היא המרחק שנותר בפועל, מהר יותר האלגוריתם יהיה. אם להעריך את המרחק הזה, עם זאת, אין הוא מתיימר לתת לנו את המסלול הקצר. במקרים כאלה, יש לנו מה שנקרא "היוריסטית קבילה".

מבחינה טכנית, בדוגמה זו, שיטת מנהטן אינו קביל כי זה קצת מגזים בהערכת המרחק הנותר. אבל נשתמש בו בכל זאת כי זה הרבה יותר קל להבין למטרות שלנו, כי זה רק אומדן קלה. בהזדמנות נדירה כאשר הנתיב וכתוצאה מכך אינו הקצר ביותר האפשרי, זה יהיה כמעט קצר. רוצים לדעת עוד? אתה יכול למצוא משוואות והערות נוספות על היוריסטיות כאן .

F מחושב על ידי הוספת G והתוצאות של השלב הראשון בחיפוש שלנו ניתן לראות באיור למטה. F, G, ועשרות ח כתובים כל ריבוע. כמו מצוין בכיכר ימינה מיידית של הכיכר מתחילים, F מודפס ימין למעלה, G מודפס שמאל למטה, ו-H מודפס הימנית התחתונה.

אז בואו נסתכל על כמה ריבועים אלה. בכיכר עם האותיות בו, G = 10. הסיבה לכך היא כי זה רק ריבוע אחד מהכיכר מתחיל בכיוון אופקי. הריבועים ממש מעל, מתחת, בצד שמאל של הכיכר החל לכולם יש ניקוד זהה של 10-G. הריבועים אלכסוניים יש עשרות גר '14.

עשרות רבות של H מחושבים על ידי הערכה של המרחק מנהטן לכיכר היעד אדום, נע רק אופקית ואנכית והתעלמות הקיר כי הוא בדרך. באמצעות שיטה זו, בכיכר ימינה מיידית של התחלה הוא 3 ריבועים של הכיכר האדומה, על ציון ה 30. ממש מעל הכיכר כיכר זה 4 ריבועים חוץ (זוכרים, רק להזיז אופקית ואנכית) על ציון ה 40. אתה יכול כנראה לראות כיצד עשרות ח מחושבים על ריבועים אחרים.

ציון F עבור כל ריבוע, שוב, מחושב פשוט על ידי הוספת G ו-H יחד.

### המשך חיפוש

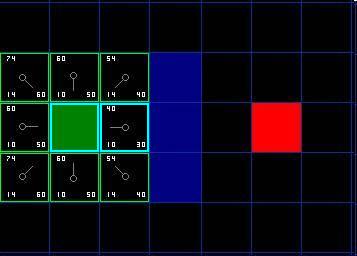
להמשך החיפוש, אנחנו פשוט בוחרים ציון הנמוך F מרובע מכל אלה נמצאים ברשימת פתוח. לאחר מכן לבצע את הפעולות הבאות עם הריבוע הנבחר:

4) זרוק אותו מהרשימה פתוח להוסיף אותו לרשימה הסגורה.

5) יש לבדוק את כל ריבועים סמוכים. התעלמות אלה הכלולים ברשימה סגורה או ריבוע "מלא" (השטח עם מים, קירות, או שטח לא חוקי אחר), להוסיף ריבועי לרשימה פתוחה אם הם לא ברשימה פתוחה. להפוך את הכיכר הנבחר "הורה" של ריבועים חדשים.

6) אם בכיכר הסמוכה כבר ברשימה פתוחה, בדוק אם זו הדרך לכיכר, כי היא טובה יותר. כלומר, לבדוק אם ציון G עבור מרובע, נמוך, אם אנו משתמשים הכיכר הנוכחית כדי להגיע לשם. אם לא, לא עושים כלום.   
מצד שני, אם עלות G של הנתיב החדש הוא נמוך יותר, לשנות את האם של הכיכר הסמוכה לכיכר הנבחר (בתרשים לעיל, לשנות את הכיוון של מצביע להצביע בכיכר הנבחר). לבסוף, לחשב מחדש גם את ה-F ו-G עשרות מרובע. אם זה נראה מבלבל, אתה תראה את זה לידי ביטוי בהמשך.

אוקיי, אז בוא נראה איך זה עובד. של 9 ריבועים הראשונים שלנו, יש לנו 8 שמאל ברשימת פתיחה לאחר הכיכר המוצא הייתה עברה ברשימה סגורה. מתוכם 1 עם עלות ה-F הנמוך ביותר הוא 1 ימינה מידית של הכיכר מתחילים, עם F ציון של 40. לכן אנו בוחרים את הכיכר כמו בכיכר הבאה שלנו. זוהי גולת הכותרת בכחול באיור הבא.



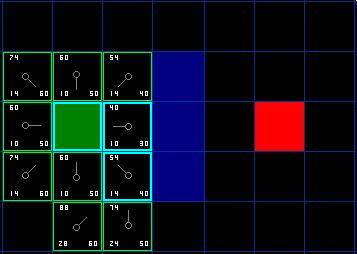
ראשית, נפיל אותו מהרשימה הפתוח שלנו ולהוסיף אותו לרשימה הסגורה שלנו (בגלל זה הוא מסומן עכשיו בכחול). לאחר מכן אנחנו בודקים את ריבועים סמוכים. ובכן, אלה ימינה מידית של ריבוע זה הם ריבועים קיר, אז אנחנו מתעלמים מהם. 1 שמאלה המיידית היא הכיכר החל. זה על רשימה סגורה, אז נתעלם גם את זה.

שאר ארבעה ריבועים כבר ברשימה פתוחה, ולכן אנחנו צריכים לבדוק אם נתיבים אלו ריבועים הם יותר טוב באמצעות ריבוע כדי להגיע לשם, באמצעות עשרות ז כנקודת ההתייחסות שלנו. בואו נסתכל על ימין בכיכר מעל הכיכר נבחרה שלנו. ציון G הנוכחי שלה הוא 14. אם אנחנו במקום עבר הכיכר הנוכחית כדי להגיע לשם, ציון G יהיה שווה ל 20 (10, שהוא ציון G להגיע לכיכר הנוכחית, בתוספת 10 יותר ללכת אנכית 1 מעל זה). ציון של 20 AG גבוה מ 14, אז זה לא דרך טובה יותר. זה בהחלט הגיוני, אם אתה מסתכל על התרשים. זה ישיר יותר כדי להגיע לכיכר, כי מהכיכר מתחיל פשוט על ידי הזזת משבצת אחת באלכסון כדי להגיע לשם, ולא נע אופקית מרובע ולאחר מכן אנכי מרובע.

כאשר אנו חוזרים על תהליך זה עבור כל 4 הריבועים הסמוכים כבר ברשימה פתוחה, אנו מוצאים כי אף אחד השבילים משתפרים על ידי עובר הכיכר הנוכחית, אז אנחנו לא משנים שום דבר. אז עכשיו הסתכלנו על כל הריבועים הסמוכים, אנחנו נעשה עם ריבוע זה, מוכן לעבור לכיכר הבאה.

אז אנחנו נעבור על הרשימה של ריבועים ברשימה הפתוח שלנו, שהוא כיום עד 7 ריבועים, ואנחנו לבחור אחד עם עלות ה-F הנמוך ביותר. מעניין לציין, כי במקרה זה, יש שני ריבועים עם ציון של 54.אז מה אנחנו בוחרים? זה לא ממש משנה. לעניין מהירות, זה יכול להיות מהיר יותר לבחור את האחרונה לרשימת פתוח. זה הטיות החיפוש לטובת ריבועים לקבל נמצאו מאוחר יותר בחיפוש, כאשר מקבלת קרוב יותר למטרה. אבל זה לא ממש משנה. (בשונה טיפול הקשרים לכן שתי גרסאות של \* עלול למצוא נתיבים שונים של אורך שווה.)

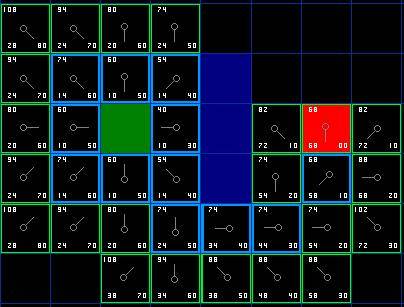
אז בואו לבחור את אחת בלבד למטה, בצד ימין של הכיכר החל, כפי שמוצג באיור הבא.



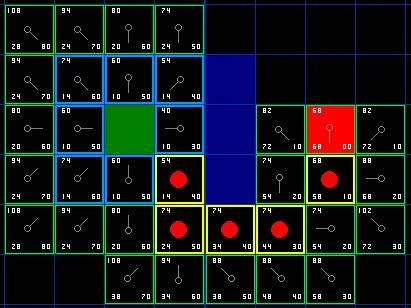
הפעם, כאשר אנחנו בודקים את ריבועים סמוכים אנו מוצאים כי 1 ימינה המיידית היא ריבוע הקיר, אז להתעלם מזה. כנ"ל לגבי אחת בלבד מעל זה. כמו כן, אנו מתעלמים בכיכר מתחת לקיר. למה? כי אתה לא יכול להגיע לכיכר באופן ישיר מן הכיכר הנוכחית ללא חיתוך פני פינת הקיר הסמוך. אתה באמת צריך לרדת הראשון ולאחר מכן לעבור אל הכיכר הזו, עוברת מעבר לפינה בתהליך. (הערה:.. כלל זה על חיתוך פינות הוא לא חובה השימוש בו תלוי כמה צמתים שלך ממוקמים)

זה משאיר חמישה ריבועים אחרים. את שני הריבועים מתחת לכיכר הנוכחי אינם כבר ברשימה פתוחה, אז נוסיף אותם בכיכר הנוכחי הופך להיות ההורה שלהם. של שלושה ריבועים נוספים, שניים מהם כבר ברשימה סגורה (כיכר ההתחלה, אחת בלבד מעל הכיכר הנוכחית, הן מודגשות בכחול בתרשים), אז אנחנו מתעלמים מהם. והכיכר האחרון, לשמאלו של הכיכר הנוכחית, נבדק כדי לראות אם הוא כל ציון G נמוך אם עוברים בכיכר הנוכחית כדי להגיע לשם. ללא הועיל. אז אנחנו עושים ומוכנים לבדוק את הכיכר הבאה ברשימה הפתוח שלנו.

אנו חוזרים על תהליך זה עד נוסיף את הכיכר היעד לרשימה סגורה, ובשלב זה נראה משהו כמו באיור שלהלן.



שים לב, את הכיכר הורה לשני ריבועים רבועים מתחת לכיכר החל השתנה באיור הקודם. לפני זה היה ציון של 28 G והצביע חזרה לכיכר מעליה ימינה. עכשיו יש לו ציון של 20 ומצביע על הכיכר ממש מעליו. זה קרה אי שם לאורך הדרך על החיפוש שלנו, כאשר ציון G נבדק והתברר להיות נמוך באמצעות דרך חדשה - כך הורה היה עבר והציונים G ו F חושבו מחדש. למרות שינוי זה לא נראה חשוב מדי בדוגמה זו, יש הרבה מצבים אפשריים שבהם זה בדיקת קבוע יעשה את כל ההבדל לקבוע את הנתיב הטוב ביותר ליעד שלכם.

אז איך אנו קובעים את הנתיב? פשוט, פשוט להתחיל בכיכר היעד אדום, ולעבוד לאחור נע מאפס לחברה האם שלה, בעקבות החצים. זה יהיה בסופו של דבר לקחת אותך חזרה לנקודת ההתחלה ההתחלה, וזה בדרך שלך. זה אמור להיראות כמו באיור הבא. עוברים את הכיכר החל A ל-B היעד מרובע הוא פשוט עניין של עוברים במרכז כל ריבוע (צומת) אל מרכז הכיכר הבאה בדרך, עד שמגיעים למטרה.

### סיכום שיטת \*A:

אוקי, עכשיו עברתם הסבר, בואו להוציא את השיטה צעד אחר צעד הכל במקום אחד

1) מוסיפים את הכיכר להתחיל (או צמתים) לרשימה פתוחה.

2) חזור על הפעולות הבאות:

א) מחפשים את הריבוע הנמוך ביותר F עלות ברשימה פתוחה. אנחנו קוראים לזה הכיכר הנוכחית.

ב) החלף אותו לרשימה הסגורה.

ג) כל אחד 8 ריבועים סמוכים לכיכר הנוכחית ...

* אם זה לא "ריק" או אם הוא נמצא ברשימה סגורה, להתעלם ממנו. אחרת לבצע את הפעולות הבאות.
* אם זה לא ברשימה פתוחה, להוסיף אותה לרשימת פתוח. להפוך את הכיכר הנוכחית האם של ריבוע זה. רשום את F, G, ועלויות ח של הכיכר.
* אם זה ברשימה פתוחה, לבדוק אם בדרך זו לכיכר כי הוא טוב יותר, באמצעות עלות G כאמצעי. עלות נמוכה יותר G אומר כי מדובר בדרך טובה יותר. אם כן, לשנות את האם של הכיכר לכיכר הנוכחית, ולחשב מחדש את ה-G ו-F עשרות רבות של הכיכר. אם אתה שומר רשימת פתוח מיון לפי ציון F, ייתכן שיהיה צורך לפנות את הרשימה כדי להסביר את השינוי.

ד) מתקבלת כאשר אתה:

* מוסיפים את הכיכר היעד לרשימה סגורה, ובמקרה זה הדרך נמצא (ראה הערה להלן), או
* לא מצליחים למצוא את הכיכר היעד, רשימה פתוחה ריקה. במקרה זה, אין שום דרך.

3) שמור את הנתיב. עבודה לאחור בכיכר היעד, עבור כל אחד מן הכיכר לכיכר האם שלה עד שתגיע לכיכר ההתחלה. זה הנתיב שלך.

הערה: באלגוריתם שפיתחנו בשפת הVB אנחנו ביססנו אותו על האלגוריתם A\* אבל החלטנו לא לעבד עם נתונים באלכסון אלא בחרנו רק להתעסק אם הכיוונים הבסיסיים – למעלה, למטה, ימינה, שמאלה.

# 9. מחקר ופיתוח :

## ערכה לטכנאי: FISHERTECHNIKS

תפקידה של תוכנית ערכה לטכנאי היא לאפשר למשתמש ולטכנאי המתחזק את המערכת לאתר תקלות ביתר קלות. לשם כך כתבנו תוכנה שנותנת תצוגה ויזואלית של הפישרטכניקס ומשתנה בהתאם לפקודות אותן אנחנו נותנים לו ובהתאם למידע שמתקבל ממנו. כך יכול המשתמש לבדוק אם התקלה היא בפישרטכניקס, ברכיבי החומרה (מערכת ההנעה, חיישן המוליכות) או בתכנות: אם התקלה היא בפישרטכניקס, כשנלחץ על האייקונים של היציאות הדיגיטליות צבעם ישתנה (והפקודה תישלח אליו, שכן היא מותנית בשינוי הצבע) אך בפישרטכניקס עצמו הנורות יישארו כבויות והרכיבים לא יעבדו. אם התקלה היא בתכנות האייקונים לא ישנו את צבעם וכל מידע שיתקבל מהכניסות הדיגיטליות והאנלוגיות לא יוצג על המסך. אם התוכנה היא ברכיבי החומרה יכולים להיות שני מקרים: אם מערכת ההנעה לא תעבוד, כשניתן אות בערוצי הפלט הדיגיטלי הכפתורים על המסך ישנו את צבעם, הנורות בפישרטכניקס ישנו את צבען אך מערכת ההנעה לא תעבוד. אם חיישן המוליכות לא יעבוד, לא יתקבל שום קלט בפישרטכניקס, ולכן המספרים בתוויות המסמלות את הקלט האנלוגי לא ישתנו.

כך נראה המסך למשתמש של הערכה לטכנאי:

. יכולת הפעלה  
כיישום של 32-bit אמיתי, LLWin 3.0 מופעל תחת Win95, 98 ו - NT.  
התוכנה מאפשרת שליטה של שניהם במקביל הישן ממשק אמנות. Nr. 30520 ו Intelligent וממשק Nr אמנות. 30402.  
  
  
2. ממשק משתמש  
תוכלו להכיר את התכונה החדשה הראשונה כל כך מהר כפי שאתה מגדיר על יצירת פרויקט חדש.  
הנה, שדה שיח מופיעה בו ניתן לבחור מבין מגוון תבניות. התבניות  
בא גדושה תת תוכניות עבור סוגים שונים מודל fischertechnik (למשל, על רובוטים ניידים ו  
תעשיית הרובוטים).  
  


 הכנסת, העברת שינוי קוביות פונקציה  
  
אתה יכול להסיר את כל אבני פונקציה זמין מארגז הכלים ולהכניס אותם לשימוש הראשון שהפעיל  
"הכנס בלוקים" הכלי.  
  
כדי להסיר חוסם פונקציה מתיבת הכלי, להשתמש בעכבר כדי לבחור את  
סמל לחסום את הפונקציה הרצויה בארגז הכלים. שמו של  
לחסום את הפונקציה יופיע. לחץ על לחצן העכבר השמאלי וגרור את  
לחסום את הפונקציה (מבלי לשחרר את לחצן העכבר) למיקום הרצוי  
על המסך. אם לשחרר את לחצן העכבר, את שדה שיח על זה  
לחסום פונקציה מסוימת תופיע. אחרי שעשית את כל הנדרש  
ערכי, לחסום את הפונקציה ממוקם על המסך. האם שני הפונקציה  
רחובות ימוקם ישירות על גבי זו, קו החיבור  
נמשכים באופן אוטומטי ביניהם. זה חיבור אוטומטי של שני  
חוסם פונקציה המכונה "התחבר אוטומטית". Auto Connect  
פונקציות כל עוד המרחב שבין גושי פונקציה לא  
יעלה על 9 נקודות המסך.  
  
אם ברצונך להוסיף תת תוכנית, לחץ על הכרטיסייה "המדד תת התכנית  
את ארגז הכלים. תת תוכנית כל זמין יוצג. הפעל את  
הרצוי תת תוכנית עם לחצן העכבר השמאלי; שם יודגשו  
בכחול. ואז להזיז את הסמן אל תת תוכנית התמונה וגרור אותו ב  
באופן המתואר לעיל למיקום הרצוי על המסך.  
  
  
אם אתה רוצה לשנות את הערכים נכנסו בבלוק הפונקציה במועד מאוחר יותר, לחץ על הפונקציה לחסום עם כפתור העכבר הימני. שדה שיח תהיה שוב להופיע.  
  
אתה יכול גם להעביר בלוקים פונקציה ידי בחירת בלוק הפונקציה עם הסמן, לחיצה על השמאלי לחצן העכבר ולאחר מכן, מבלי לשחרר את לחצן, גרירת לחסום פונקציה למיקום אחר.  
במהלך תמרון זה, קווי חיבור שנותרו על כנם. בנוסף לכך, חיבור קווי יכולים גם  
נוצר על ידי העברת בלוקים לתפקד. אם בלוק הפונקציה היא עברה כל כך הקלט שלה הוא על גבי הפלט של גוש פונקציה אחרת, קו החיבור נוצר. כנ"ל לגבי יציאות שאינן דגש על תשומות. לאחר מכן, לחסום את הפונקציה ניתן להעביר למצב סופני שלה, או יותר  
הקשרים של תשומות ותפוקות שנותרו ניתן להסיק.



מחיקת בלוקים פונקציה  
  
כדי למחוק בלוק הפונקציה, להפעיל את "גושי מחק" סמל. הסמן הופך  
מוכר ואהוב הפטיש (גירסה 2.1). אם אתה עכשיו לחץ על הבלוק פונקציה עם העכבר השמאלי כפתור, לחסום את הפונקציה וכל קווי חיבור הגובלים יימחקו. פקודה זו לא יכולה להיות הפוך.  
  
החלפת פונקציה חוסמת  
  
חוסם פונקציה מוחלפים הראשון שהפעיל "אבני החלף" סמל, ואז לגרור פונקציה לחסום מתיבת הכלי עם לחצן העכבר השמאלי והניח אותו מעל לחסום את הפונקציה להיות הוחלף. אם לחסום את הפונקציה החדשה כוללת את התשומות והתפוקות אותו כמו את הישן, את הקשר הקווים יהיה גם השתלטו באופן אוטומטי.  
  
ציור קווים  
  
לחיצה עם הכפתור השמאלי של העכבר על הסמל "" קווי צייר בסרגל הכלים הופכת את הסמן אל עפרון. הקו ניתן להשוות בין שני רחובות הפונקציה על ידי לחיצה הראשון עם כפתור העכבר השמאלי על התוצר של גוש הפונקציה הראשונה ולאחר מכן לשחרר את לחצן העכבר וכתוצאה מכך את הסמן  
הופך crosshair. זה אותות הפלט כבר כראוי "מכה". קו יכול עכשיו להיות צייר  
על ידי לחיצה על הקלט של לחסום את הפונקציה השנייה.  
  
אם אתה רוצה, אתה יכול גם לקבוע את הנתיב של קטעים בודדים לאורך הקו, או בדרך של כל הקו. ציור הוא התחיל על ידי לחיצה על כל קלט, פלט, או קו. עד אז העברת crosshair בדיוק במאונך או במאוזן לנקודה הראשונה ושוב לחיצה על הכפתור השמאלי של העכבר, אתה יהיה נמשך הקטע הראשון של הקו. קטעים נוספים נמשכים באותו אופן עד נקודת היעד הושג. נקודת התייחסות לפעולת הציור הוא תמיד נקודת הקצה קטע הקו האחרון נמשך. אם ברצונכם לקבל את שאר הקו נמשך באופן אוטומטי, פשוט לחץ על  
על נקודת היעד.  
  
קטעים בודדים ניתן להעביר לאחר הציור הושלם על ידי לחיצה על מקש CTRL  
המפתח את לחצן העכבר השמאלי בו זמנית. 

מחיקת קווים  
  
קווי ניתן למחוק על ידי הפעלת "קווי מחק" סמל בסרגל הכלים. פעולה זו הופכת את הסמן אל המחק.  
  
לחיצה עם הכפתור השמאלי של העכבר במקום כלשהו על קו החיבור מוחקת את הקו. חסר קטעים על קווי גבול אחרים הושלמו. לחיצה על כפתור העכבר הימני על קו יש ההשפעה של מחיקת הקו יחד עם כל הקווים הגובלים.  
  
עריכת קבוצות  
  
עם גירסה החדשה LLWin, כמה בלוקים פונקציה ניתן להעביר בו זמנית, להעתיק, לגזור הודבק במקום אחר.  
  
כדי להמשיך, תחילה לבחור את בלוק הפונקציה הראשונה בקבוצה עם CTRL השמאלי לחצן העכבר. גוש פונקציה הנבחר יקבל ריבוע קו שבור.  
  
המשך או על ידי בחירת בלוק הפונקציה האחרונה בקבוצה להיות  
נערך על ידי לחיצה על הלחצן השמאלי בעכבר SHIFT (וכתוצאה מכך כל  
אבני שוכב בין מודגשים גם) או על ידי בחירה נוספים  
בלוקים בודדים לתפקד ידי הקשה על CTRL השמאלי לחצן העכבר.  
  
כל רחובות הפונקציה ניתן לבחור על ידי הקשה על CTRL זה גם האופן שבו, למשל, הקבצים והתיקיות הם נבחר ב-Windows Explorer.  
  
הקבוצה הדגישה אז יכול להיות נע על ידי העסקת השמאלי של העכבר  
הכפתור באופן המתואר לעיל.  
  
עם השימוש ב-Windows הבאה פונקציות סטנדרטיות העתק גזור מחק הדבק   
הקבוצה הדגישה יכול, למשל, גם יועתקו לתוך תוכנית (העתק הדבק) או עבר (גזור והדבק) או נמחקו כליל



Private Sub cmdAction\_Click()

Dim ft As New FishFace

ft.OpenInterfaceUSB \_

ft.SetMotor ftiM2, 1

ft.Pause 1000

ft.SetMotor ftiM1, 1

ft.Pause 500

ft.SetMotor ftiM1, 0

ft.SetMotor ftiM2, 0

ft.SetMotor ftiM3, 2

ft.Pause 2000

ft.SetMotor ftiM3, 0

ft.CloseInterface

Unload Me



Private Sub Fahren()

lblStatus = "Beenden : I7 drücken"

ft.SetMotor ftiM4, ftiRight

ft.WaitForInput ftiI7

End Sub



Do

ft.SetMotor ftiM2, 0

ft.SetMotor ftiM1, 1

lblAnalog = ft.GetAnalog(ftiAX) & \_

" - " & ft.GetAnalog(ftiAY)

ft.Pause 333

ft.SetMotor ftiM1, 0

ft.SetMotor ftiM2, 1

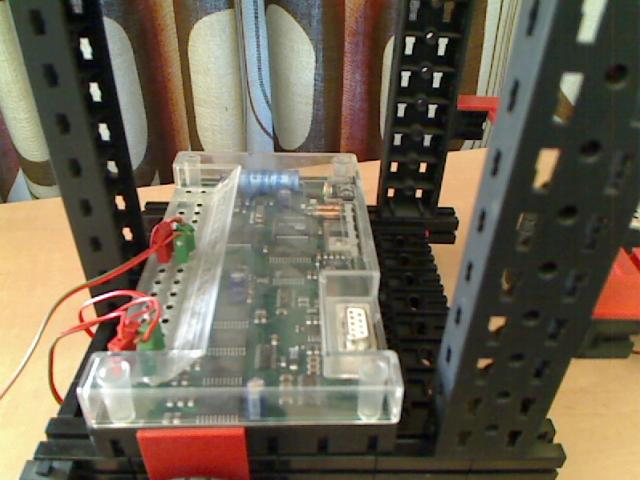
lblAnalog = ft.GetAnalog(ftiAX) & \_

" - " & ft.GetAnalog(ftiAY)

ft.Pause 333

Loop Until ft.Finish(ftiI1)

## C:\Documents and Settings\כניסה\שולחן העבודה\רובוטים.BMPתיאור מבנה הרובוטים:

הרובוטים בנויים ממן חלקי לגו שמתחברים אחד לשני. בשניהם יש לוח אם שאחראי על ניתנת הפקודות ועליו מחוברים חוטי חשמל שאחראים על העברתם. בשני הרובוטים יש שני מנועים שמחוברים לשני גלגלים שמסוגלים לנוע אחורה וקדימה.

לוח האם

המנועים

הרובוטים. (ימני רודף, שמאלי נרדף)

### תיאור מבנה הרובוט הרודף:

Fishertechnicks intelligent interface

**מחשב PC**

פענוח תמונה

הזירה ומיקומי הרכבים

כבל USB ותוכנת קליטה

מצלמת ווידאו

תצוגת PC

מנועים

תקשורת משדר מקלט או כבל טורי

### תיאור מבנה הרובוט הנרדף:

**מחשב PC**

פענוח תמונה

**וזיהוי התקרבות האוייב**

כבל USB ותוכנת קליטה

מצלמת ווידאו

Fishertechnicks intelligent interface

(כולל flash)

מנועים

# 10. הסבר התוכנית + האלגוריתם חיפוש

WEBCAM DECLARATIONS

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnd As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam As Any) As Long

Private Declare Function capCreateCaptureWindow Lib "avicap32.dll" Alias "capCreateCaptureWindowA" (ByVal lpszWindowName As String, ByVal dwStyle As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hwndParent As Long, ByVal nID As Long) As Long

Private mCapHwnd As Long

Private Const CONNECT As Long = 1034

Private Const DISCONNECT As Long = 1035

Private Const GET\_FRAME As Long = 1084

Private Const COPY As Long = 1054

Dim m As Integer, w As Integer

Dim R As Integer, G As Integer, B As Integer

Dim Tolerance As Integer

Dim ft As New FishFace

Dim l As Integer

Dim A(4) As Boolean

Dim mat(640, 480) As Integer 'מגדירים מטריצה בגודל מסוים הממפה את כל האיזור הנתון

Dim mavoh As Boolean

Dim i, blac As Integer

Dim x0, y0 As Integer

Dim derech(134, 2) As Integer

Dim p As Integer

Dim j As Integer

Dim rob\_move(136, 6) As Integer

Dim matara As Integer

Dim position\_rodef As Integer

Dim rbp As Integer

Dim px, py As Integer

Dim green, blue As Integer

Dim green\_1, blue\_1 As Integer

Private Declare Function GetTickCount Lib "kernel32" () As Long

Dim LastTime As Long

Dim Mystep As Integer

Dim penia As Integer

Dim penia\_z As Integer

Dim kivun\_x As Integer

Dim kivun\_z As Integer

'---------------------robot-------------------------

Private Sub Command21\_Click()

ניתוח התמונה

Picture2.Picture = Picture1.Picture

מגדירים שהתמונה הנוכחית המתקבלת היא התמונה שננתח

green\_1 = 0

blue\_1 = 0

מגדירים את הגודל ההתחלתי של המשתנים

nekud = Int(Picture2.Height \* Picture2.Width / (15 \* 9 \* 3))

אנו מחלקים את השדה הנתון לחלקים קטנים 9\*16

For y0 = 0 To Picture2.Height Step Int(Picture2.Height / 9(

For x0 = 0 To Picture2.Width Step Int(Picture2.Width / 15(

אנו עוברים ריבוע ריבוע.

For m = x0 To x0 + Int(Picture2.Width / 15(

For w = y0 To y0 + Int(Picture2.Height / 9(

בודקים את הפיקסילים בתוך כל ריבוע

c = Picture2.Point(m, w)

R = c Mod 256

G = (c \ 256) Mod 256

B = (c \ 256) \ 256

מנתחים כל פיקסל לפי צבעים

If R < 85 And G < 85 And B < 85 Then

'Picture2.PSet (w, m), vbBlack

blac = blac + 1

End If

בודקים אם הפיקסל שחור ואם כן מוסיפים עוד אחד למונה

If G > 90 And (G \* 14) > ((B + R) \* 9) Then

green = green + 1

End If

בודקים אם הפיקסל שחור ואם כן מוסיפים עוד אחד למונה

If R > 70 And 2 \* B < R And 2 \* G < R Then

blue = blue + 1

End If

בודקים אם הפיקסל שחור ואם כן מוסיפים עוד אחד למונה

Next w

עוברים לפיקסל הבא בשורה

Next m

עוברים לשורה הבאה

If 4 \* green >= nekud Then

אם יותר 1/12 מהפיקסלים בריבוע צהובים

If green > green\_1 Then

ואם מספר הפיקסלים גדול יותר מאשר במשבצת הקודומת

Image2.Left = x0

Image2.Top = y0

אזי הרובוט הרודף נמצא במשבצת הזאת

green\_1 = green

והמשבצת הזאת תהיה המשבצת להשווה במשבצות הבאות

position\_rodef = (((y0 + 5) \* 9 \ 480) \* 15) + (x0 \* 15 \ 640)

ממירים את נתוני המשבצת לערכים שאפשר להכניס לאלגוריתם שמוצא את המסלול

End If

End If

If 4 \* blue >= nekud Then

If blue > blue\_1 Then

Image1.Left = x0

Image1.Top = y0

blue\_1 = blue

matara = (((y0 + 5) \* 9 \ 480) \* 15) + (x0 \* 15 \ 640)

End If

End If

אותו דבר כמו לעייל רק כדי למצוא את מיקום הרובוט הנרדף

If blac >= nekud Then

אם מספר הפיקסלים השחורים במשבצת גדול מ-שליש ממספר הפיקסלים במשבצת

Shape3(i).Left = x0

Shape3(i).Top = y0

Shape3(i).Visible = True

אז במשבצת הזאת יש מחסום

i = i + 1

במשבצת הבאה נשתמש באובייקט הבאה

mat(x0, y0) = 0

במשבצת הזאת במארך מתקבלת כחסומה

rob\_move((((y0 + 2) \* 9 \ 480) \* 15) + (x0 \* 15 \ 640), 5) = 1

באלגוריתם המשבצת מוגדרת כחסומה

Else

mat(x0, y0) = 1

אחרת המשבצת פנויה

End If

green = 0

blue = 0

blac = 0

מאפסים את המונים...

Next x0

משבצת הבאה בשורה

Next y0

שורה הבאה

Image2.Visible = True

Image1.Visible = True

התמונות המציגים את הרובוטים עכשיו נראים

End Sub

Private Sub Command3\_Click()

מאתחל את התנועה

Timer3.Enabled = True

מאתחל את הטיימר שמפעיל את תנועת הנרדף

Timer2.Enabled = True

מאתחל את הטיימר שמפעיל את תנועת הנרדף

kivun\_x = 0

kivun\_z = 0

מאפס את הכיווונים שאליהם פונים הרובוטים

ft.OpenInterfaceCOM ftiIntelligent\_IF, ftiCOM1, 12

מאפשר לשלוח פקודות לרובוטים

End Sub

Private Sub Form\_Load()

i = 0

mavoh = False

blac = 0

x0 = 0

y0 = 0

Mystep = 5

'set up the visual stuff

Picture1.Width = 640

Picture1.Height = 480

Picture2.Width = 640

Picture2.Height = 480

'inten = 5

Tolerance = 20

'Tppx = Screen.TwipsPerPixelX

'Tppy = Screen.TwipsPerPixelY

ReDim POn(640, 480)

'ReDim p(960, 1280)

STARTCAM

End Sub

Private Sub Picture1\_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)

If Button = 1 Then

STARTCAM

ElseIf Button = 2 Then

STOPCAM

End If

End Sub

Private Sub Timer1\_Timer()

'Get the picture from camera.. the main part

SendMessage mCapHwnd, GET\_FRAME, 0, 0

SendMessage mCapHwnd, COPY, 0, 0

Picture1.Picture = Clipboard.GetData

Clipboard.Clear

'state all statistics

'Label1.Caption = Int(Wo / (Ri + Wo) \* 100) & " % movement" & vbCrLf & "Real Movement: " & RealRi & vbCrLf \_

& "Completed in: " & GetTickCount - LastTime

End Sub

Sub STOPCAM()

DoEvents: SendMessage mCapHwnd, DISCONNECT, 0, 0

Timer1.Enabled = False

End Sub

Sub STARTCAM()

mCapHwnd = capCreateCaptureWindow("WebcamCapture", 0, 0, 0, 640, 480, Me.hwnd, 0)

'יצירת חלון לתמונה סרט בזיכרון

DoEvents

'הפעל מה שכתוב אח"כ גם אם אתה עסוק בדברים אחרים

SendMessage mCapHwnd, CONNECT, 0, 0

'תחבר את המצלמה שזיהית לחלון שנוצר

Timer1.Enabled = True

End Sub

'Private Sub Timer2\_Timer()

'SavePicture Picture1.Image, "C:\pics\img" & Counter & ".bmp"

'Counter = Counter + 1

'End Sub

Private Function search(rob\_move() As Integer, matara As Integer, position\_rodef As Integer) As Integer

הפונקציה מוצאת את הכיוון שבו הרודף צריך ללכת כדי להשיג את הנרדף

Dim rbp As Integer

מגדירים את הערך המשתנה שמסמל את המשבצת הנבדקת

Dim derech(134, 2) As Integer

מגדירים את המערך שבתוכו מכניסים את המסלול שהרובט צריך לעבור

Dim p As Integer

Dim j As Integer

Dim n, m As Integer

מגדירים עוד כמה ערכים

If position\_rodef = matara + 15 Then

אם הרודף נמצא במשבצת שמתחת לנרדף

search = 1

אז הרודף צריך לעלות משבצת

ElseIf position\_rodef = matara - 15 Then

אם הרודף נמצא במשבצת שמעל הנרדף

search = 3

אז הרודף צריך לרדת משבצת

ElseIf position\_rodef = matara + 1 Then

אם הרודף נמצא במשבצת שמימין לנרדף

search = 4

אז הרודף צריך לזוז משבצת שמאלה

ElseIf position\_rodef = matara - 1 Then

אם הרודף נמצא במשבצת שמשמאל לנרדף

search = 2

אז הרודף צריך לזוז משבצת ימינה

Else

אחרת, והרודף לא במשבצת שסמוכה לנרדף אז נעשה כך

For n = 0 To 134

For m = 1 To 4

rob\_move(n, m) = 0

Next m

Next n

מאפסים את כל הנתונים שבמערך

בטור הראשון של המערך מכניסים את המרחק מהמשבצת אל הרודף

בטור השני מכניסים את המרחק המשוער של כל נקודה אל הנרדף

בטור השלישי אומרים אם אפשר לבדוק את המשבצת

בטור הרביעי מכניסים את המספר של המשבצת שממנה מגיעים אם באים אליה מהרודף

בטור החמישי מכניסים אם המשבצת חסומה או לא

rob\_move(135, 1) = 100

מגדירים שמשבצת שלא קיימת רחוקה מהרודף במאה משבצות

rob\_move(position\_rodef, 1) = 0

מגדירים שהמרחק בין הרודף אל עצמו שווה 0

rob\_move(position\_rodef, 3) = 1

מגדירים שאת המשבצת של הרודף אפשר לבדוק

For j = 0 To 134

rob\_move(j, 2) = Abs((matara Mod 15) – (j Mod 15)) + Abs((matara \ 15) – (j \ 15))

Next j

בודקים את המרחק המשוער בין כל נקודה אל הנרדף

Do

rbp = 135

מגדירים שלכתחילה בודקים את משבצת 135 שאיננה קיימת

If rob\_move(position\_rodef, 3) = 1 Then

אם המשבצת של הרודף ניתנת לבדיקה

rbp = position\_rodef

היא תהיה המשבצת שנבדוק

End If

For j = 0 To 134

בודקים את כל המשבצות

If rob\_move(j, 3) = 1 Then

אם המשבצת ניתנת לבדיקה

If (rob\_move(j, 2) + rob\_move(j, 1)) < (rob\_move(rbp, 2) + rob\_move(rbp, 1)) Then

והמרחק המשוער בין הנקודה לנרדף + המרחק אל הרודף קטן מזה של המשבצת שכבר בחרנו לבדוק אז...

rbp = j

נבדוק את המשבצת הזאתי

End If

End If

Next j

בודקים את המשבצת שבחרנו

המשבצת משמאל

If rbp Mod 15 <> 0 Then

אם המשבצת לא בטור הכי שמאלי (ואין משבצת מימינו)

If rob\_move(rbp - 1, 5) <> 1 Then

והמשבצת משמאל לא חסומה

If rob\_move(rbp - 1, 3) = 0 Then

והמשבצת עוד לא נבדקה

rob\_move(rbp - 1, 3) = 1

אז המשבצת ניתנת לבחירה כ רב"פ

rob\_move(rbp - 1, 4) = rbp

משבצת האם (המשבצת שממנה מגיעים אם באים מהרובוט) הוא המשבצת רב"פ

rob\_move(rbp - 1, 1) = rob\_move(rbp, 1) + 1

המרחק בין הנקודה לרודף שווה למרחק ממשבצת האם שלה לרודף + 1

End If

End If

End If

בודקים את המשבצת מימין ועושים כדומה

If (rbp + 1) Mod 15 <> 0 Then

If rob\_move(rbp + 1, 5) <> 1 Then

If rob\_move(rbp + 1, 3) = 0 Then

rob\_move(rbp + 1, 3) = 1

rob\_move(rbp + 1, 4) = rbp

rob\_move(rbp + 1, 1) = rob\_move(rbp, 1) + 1

End If

End If

End If

בודקים את המשבצת ממעל ועושים כדומה

If rbp > 14 Then

If rob\_move(rbp - 15, 5) <> 1 Then

If rob\_move(rbp - 15, 3) = 0 Then

rob\_move(rbp - 15, 3) = 1

rob\_move(rbp - 15, 4) = rbp

rob\_move(rbp - 15, 1) = rob\_move(rbp, 1) + 1

End If

End If

End If

בודקים את המשבצת מתחת ועושים כדומה

If rbp < 120 Then

If rob\_move(rbp + 15, 5) <> 1 Then

If rob\_move(rbp + 15, 3) = 0 Then

rob\_move(rbp + 15, 3) = 1

rob\_move(rbp + 15, 4) = rbp

rob\_move(rbp + 15, 1) = rob\_move(rbp, 1) + 1

End If

End If

End If

rob\_move(rbp, 3) = 2

המשבצת לא ניתנת לבדיקה מעכשיו

Loop Until rbp = matara

עושים שוב ושוב עד שבדקנו את הנרדף

Do

derech(rob\_move(rbp, 1), 1) = rbp

במערך "דרך" מכניסים את המספר של המשבצת האחרונה שבדקנו (הנרדף)

בשורה שמספרה שווה למרחק בינו לרודף

rbp = rob\_move(rbp, 4)

ואז עושים את אותו דבר עם משבצת האם של המשבצת כך שנקבל ב-0 את הרודף

Loop Until rob\_move(rbp, 4) = position\_rodef

derech(0, 1) = position\_rodef

If position\_rodef = rbp + 15 Then

אם המשבצת שבשורה מתחת לרודף ב"דרך" (המשבצת הבאה במסלול) מעליו אז

search = 1

אז תוצאת החיפוש שווה 1

ElseIf position\_rodef = rbp - 15 Then

אם המשבצת שבשורה מתחת לרודף ב"דרך" (המשבצת הבאה במסלול) מתחתיו אז

search = 3

אז תוצאת החיפוש שווה 3

ElseIf position\_rodef = rbp + 1 Then

אם המשבצת שבשורה מתחת לרודף ב"דרך" (המשבצת הבאה במסלול) מימינו אז

search = 2

אז תוצאת החיפוש שווה 2

ElseIf position\_rodef = rbp - 1 Then

אם המשבצת שבשורה מתחת לרודף ב"דרך" (המשבצת הבאה במסלול) משמאלו אז

search = 4

אז תוצאת החיפוש שווה 4

End If

End If

End Function

End Sub

תכנות של הרובוט פישרטכניקס

Do

אתחול מספר אקראי ובחירה אקראית של מיקום (X,Y) פנוי בתמונה כדי למקם את הנרדף

Randomize

px = Int(Rnd \* 640)

Randomize

py = Int(Rnd \* 480)

מציאת מקום המוגדר כ-1 במטריצה, כלומר - פנוי

Loop Until (mat(px, py) = 1)

מיקום תמונת הנרדף

Image1.Left = px

Image1.Top = py

Image1.Visible = True

הגדרת מקום הנרדף כתפוס

mat(px, py) = 0

קביעת מספר הריבוע למטריצה עבור הנרדף

matara = (((py + 5) \* 9 \ 480) \* 15) + (px \* 15 \ 640)

Do

כנ"ל לגבי הרודף

Randomize

px = Int(Rnd \* 640)

Randomize

py = Int(Rnd \* 480)

Loop Until (mat(px, py) = 1)

Image2.Left = px

Image2.Top = py

Image2.Visible = True

position\_rodef = (((py + 5) \* 9 \ 480) \* 15) + (px \* 15 \ 640)\

Case 1 'yashar'

הפעלת תזזוזת הרובוט - ישר

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנועים

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

הפעלת רוטציית המנועים

ft.Pause 1160

הפעלת המנועים לזמן מוקצב – הזמן הדרוש לעבור מריבוע אחד בתמונה לשני

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנועים

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנועים

ft.SetMotor ftiM4, ftiRight

ft.SetMotor ftiM3, ftiRight

הפעלת המנועים

ft.Pause 1160

הפעלת המנועים לזמן מוקצב – הזמן הדרוש לעבור מריבוע אחד בתמונה לשני

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנועים

פנייה ימינה

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנוע

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 110

הפעלת הרובוט להתקדם ישר לזמן קצר, ואז לפנות ימינה, על מנת להקל על הפנייה

ft.SetMotor ftiM4, ftiLeft

ft.SetMotor ftiM3, ftiRight

ft.Pause 526

הפעלת המנוע על מנת לפנות ימינה - במקום

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 1160

תנועה ישרה בכיוון ימין

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנוע

kivun\_z = kivun\_z + 1

ערך השומר את כיוון הרובוט

פנייה שמאלה

Case 0 'smola'

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנוע

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 110

הפעלת הרובוט להתקדם ישר לזמן קצר, ואז לפנות ימינה, על מנת להקל על הפנייה

ft.SetMotor ftiM4, ftiRight

ft.SetMotor ftiM3, ftiLeft

ft.Pause 526

הפעלת המנוע על מנת לפנות שמאלה - במקום

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 1160

תנועה ישרה בכיוון שמאלה

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

ערך השומר את כיוון הרובוט

kivun\_z = kivun\_z + 3

End Select

End Sub

ערך הנשמר העוזר לתוכנית לחשב את דרכו בהמשך.

penia = (rndmove + kivun\_x) Mod 4

המשך דרך קצרה

kivun = search(rob\_move(), matara, position\_rodef)

ערך השומר את כיוון המטרה.-kivun

If kivun = 1 Then

Image2.Top = Image2.Top - Int(Picture2.Height / 9)

משנה את מיקום הרודף בתמונה (כלפי מעלה)

position\_rodef = position\_rodef – 15

משנה את ערך מיקום הרודף במערך

ElseIf kivun = 3 Then

Image2.Top = Image2.Top + Int(Picture2.Height / 9)

משנה את מיקום הרודף בתמונה (כלפי מטה)

position\_rodef = position\_rodef + 15

משנה את ערך מיקום הרודף במערך

ElseIf kivun = 2 Then

Image2.Left = Image2.Left + Int(Picture2.Width / 15)

משנה את מיקום הרודף בתמונה (כלפי ימינה)

position\_rodef = position\_rodef + 1

משנה את ערך מיקום הרודף במערך

ElseIf kivun = 4 Then

Image2.Left = Image2.Left - Int(Picture2.Width / 15)

משנה את מיקום הרודף בתמונה (כלפי שמאלה)

position\_rodef = position\_rodef – 1

משנה את ערך מיקום הרודף במערך

End If

If position\_rodef = matara

אם הרודף משיג את הנרדף אז

Then

Timer3.Enabled = False

Timer2.Enabled = False

MsgBox "gotcha"

End If

בטל את פעילות הטיימרים, הדפס הודעה: "תפסתי אותך"

Private Sub Timer3\_Timer()

If Image1.Top >= Picture2.Top + Int(Picture2.Height / 9) Then

If Image1.Top <> Picture2.Height And mat(Image1.Left, Image1.Top - Int(Picture2.Height / 9)) <> 0 Then

לולאה הבודקת אם אפשר לנוע בכיוון מסוים, ושאין מכשולים (בריבועים הסמוכים למיקום הרובוט)

A(1) = True

Else

A(1) = False

End If

Else

A(1) = False

End If

אם במשבצת אחד למעלה לנרדף אין מכשולים, אז אפשר לפי ערך בוליאני את התזוזה כלפי מעלה. אם יש מכשול, אז אסור תנועה.

If Image1.Top <= Picture2.Height - Int(Picture2.Height / 9) Then

If Picture2.Height <> Image1.Top And mat(Image1.Left, Image1.Top + Int(Picture2.Height / 9)) <> 0 Then

לולאה הבודקת אם אפשר לנוע בכיוון מסוים, ושאין מכשולים (בריבועים הסמוכים למיקום הרובוט)

A(3) = True

Else

A(3) = False

End If

Else

A(3) = False

End If

אם במשבצת אחד למטה מלרודף אין מכשולים, אז אפשר לפי ערך בוליאני את התזוזה כלפי מעלה. אם יש מכשול, אז אסור תנועה.

If Image1.Left <= Picture2.Left - Int(Picture2.Width / 15) Then

If Picture2.Width <> Image1.Left And mat(Image1.Left + Int(Picture2.Width / 15), Image1.Top) <> 0 Then

לולאה הבודקת אם אפשר לנוע בכיוון מסוים, ושאין מכשולים (בריבועים הסמוכים למיקום הרובוט)

A(2) = True

Else

A(2) = False

End If

Else

A(2) = False

End If

אם במשבצת אחד מימין לנרדף אין מכשולים, אז אפשר לפי ערך בוליאני את התזוזה כלפי מעלה. אם יש מכשול, אז אסור תנועה

.

If Image1.Left >= Int(Picture2.Width / 15) Then

If Image1.Left <> Picture2.Width And mat(Image1.Left - Int(Picture2.Width / 15), Image1.Top) <> 0 Then

לולאה הבודקת אם אפשר לנוע בכיוון מסוים, ושאין מכשולים (בריבועים הסמוכים למיקום הרובוט)

A(4) = True

Else

A(4) = False

End If

Else

A(4) = False

End If

אם במשבצת אחד משמאל לנרדף אין מכשולים, אז אפשר לפי ערך בוליאני את התזוזה כלפי מעלה. אם יש מכשול, אז אסור תנועה.

מערך הקובע את תזוזת הרודף והנרדף.

Do

Randomize

rndmove = Int(Rnd \* 4 + 1)

Loop Until A(rndmove) = True

אתחל תזוזה רנדומאלית של הנרדף, בכיוונים המורשים לפי הבוליאן.

If rndmove = 1 Then

Image1.Top = Image1.Top – Int(Picture2.Height / 9)

matara = matara – 15

תזוזה כלפי מעלה

ElseIf rndmove = 2 Then

Image1.Left = Image1.Left + Int(Picture2.Width / 15)

matara = matara + 1

תזוזה כלפי ימינה

ElseIf rndmove = 3 Then

Image1.Top = Image1.Top + Int(Picture2.Height / 9)

matara = matara + 15

תזוזה כלפי מטה

ElseIf rndmove = 4 Then

Image1.Left = Image1.Left - Int(Picture2.Width / 15)

matara = matara – 1

תזוזה כלפי שמאלה.

(לכל התזוזות גם משנים את ערך מיקום הנרדף במערך.)

End If

ערך הנשמר העוזר לתוכנית לחשב את דרכו בהמשך

penia = (rndmove + kivun\_x) Mod 4.

פנייה לרובוט

Case 1 'yashar'

הפעלת תזזוזת הרובוט - ישר

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנועים

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

הפעלת רוטציית המנועים

ft.Pause 1160

הפעלת המנועים לזמן מוקצב – הזמן הדרוש לעבור מריבוע אחד בתמונה לשני

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנועים

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנועים

ft.SetMotor ftiM4, ftiRight

ft.SetMotor ftiM3, ftiRight

הפעלת המנועים

ft.Pause 1160

הפעלת המנועים לזמן מוקצב – הזמן הדרוש לעבור מריבוע אחד בתמונה לשני

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנועים

פנייה ימינה

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנוע

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 110

הפעלת הרובוט להתקדם ישר לזמן קצר, ואז לפנות ימינה, על מנת להקל על הפנייה

ft.SetMotor ftiM4, ftiLeft

ft.SetMotor ftiM3, ftiRight

ft.Pause 526

הפעלת המנוע על מנת לפנות ימינה - במקום

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 1160

תנועה ישרה בכיוון ימין

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

כיבוי המנוע

kivun\_z = kivun\_z + 1

ערך השומר את כיוון הרובוט

פנייה שמאלה

Case 0 'smola'

ft.SetMotor ftiM3, ftiOn

ft.SetMotor ftiM4, ftiOn

הדלקת המנוע

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 110

הפעלת הרובוט להתקדם ישר לזמן קצר, ואז לפנות ימינה, על מנת להקל על הפנייה

ft.SetMotor ftiM4, ftiRight

ft.SetMotor ftiM3, ftiLeft

ft.Pause 526

הפעלת המנוע על מנת לפנות שמאלה - במקום

ft.SetMotor ftiM3, ftiLeft

ft.SetMotor ftiM4, ftiLeft

ft.Pause 1160

תנועה ישרה בכיוון שמאלה

ft.SetMotor ftiM3, ftiOff

ft.SetMotor ftiM4, ftiOff

ערך השומר את כיוון הרובוט

kivun\_z = kivun\_z + 3

End Select

End Sub

ערך הנשמר העוזר לתוכנית לחשב את דרכו בהמשך.

penia = (rndmove + kivun\_x) Mod 4

# 11. ביבליוגרפיה

1 . עמיהוד, (1998) , סדנת לימוד VISUAL BASIC , הוצאת הוד עמי, הרצליה.

2 . דר' רפי יהל , (1995) , טכנולוגיה מוכללת , האגף למדע וטכנולוגיה.

3 . אינג' וילי רוזנבלום, (1999), אלקטרוניקה תעשייתית הוצאת אורט ישראל.

אלגוריתם לעיבוד תמונה: <http://multimedia.haifa.ac.il/course/lessons/lessons.html>

במצלמות חיישן :ccd <http://www.andor.com/library/digital_cameras/?app=325>

<http://he.wikipedia.org/wiki/CCD>

איך עובדת התוכנה ?

תוכנת ויזואל בייסיק , הסבר על כך מצאנו באתר הבא:

<http://dr-vb.co.il/vb.php>

אלגוריתמים למציאת הדרך הקצרה ביותר :

<http://www.policyalmanac.org/games/aStarTutorial.html>

<http://www.ifors.ms.unimelb.edu.au/tutorial/dijkstra_new/index.html>

<http://he.wikipedia.org/wiki/RGB>

<http://en.wikipedia.org/wiki/Search_algorithm>

<http://en.wikipedia.org/wiki/Combinatorial_search>

<http://en.wikipedia.org/wiki/Minimax>

<http://en.wikipedia.org/wiki/Alpha-beta_pruning>

[http://en.wikipedia.org/wiki/A\*\_algorithm](http://en.wikipedia.org/wiki/A*_algorithm)

<http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm>

<http://en.wikipedia.org/wiki/Branch_and_bound>

<http://www.policyalmanac.org/games/aStarTutorial.htm>