VRI>

מסמך עיצוב

<יותם לויט>

<גרסה 1>

<27/05/2019>

מורים: אלי גולדשטיין וניר דוויק

היסטוריית גרסאות המסמך

|  |  |  |
| --- | --- | --- |
| תאריך | גרסה | תקציר השינויים |
| 15/5/2019 | 0.1 | מסמך ראשוני – כתיבת חלק 4 במסמך - GUI |
| 18/5/2019 | 0.2 | הוספת הקדמה, גרפים והתחלת טבלאות בחלק 2 |
| 23/5/2019 | 0.3 | סיום חלק 2 והתחלת חלק 3 |
| 29/05/2019 | 1 | גרסה סופית |



**1. הקדמה**

1.1 מטרה

*מטרתו של המסמך המוצג להלן היא להציג את הפרויקט בצורה מפושטת על כל חלקיו. המסמך מיועד לכל המעוניין להבין בפשטות את מהותו של הפרויקט, ככל שניתן, ובלי להיכנס לקוד.*

* 1. המוצר

*תכולתו של תת סעיף זה: (ניתן לתמצת מתוך מסמך האפיון)*

* *סימולטור VRI – Virtual robot interactions.*

VRI הוא "סימולטור בדיקה". מטרתו להקנות סביבה וירטואלית המאפשרת בחינת הביצועים של רובוטים, ללא חשש מפגיעה פיזית בהם או בסביבתם. יתרונו המובהק של VRI על פני האלטרנטיבה של בחינת רובוטים בעולם הפיזי-מוחשי הוא באפשרות הממשית של הימנעות מתאונה מסוכנת עקב בחינת הרובוט, אפשרות הנעדרת, באופן טבעי, מבדיקות המבוצעות בעולם הפיזי-ממשי.

* 1. קישור למסמכים קודמים

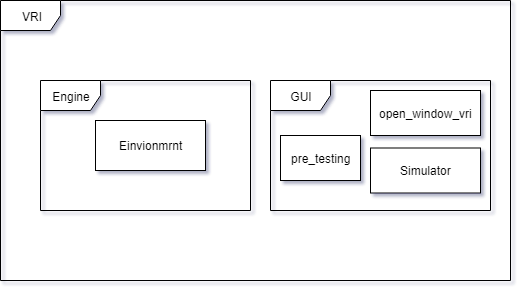
מצורף מסמך אפיון

* 1. הגדרות

GUI - Graphical user interface ממשק משתמש

OOP – תכנות מונחה עצמים. תכנות בצורה של חילוק חכם למחלקות.

1. ארכיטקטורת המערכת

חלק זה כולל את תיאור מבנה המערכת ופירוט המודולים השונים בה

2.1 מבט על הארכיטקטורה

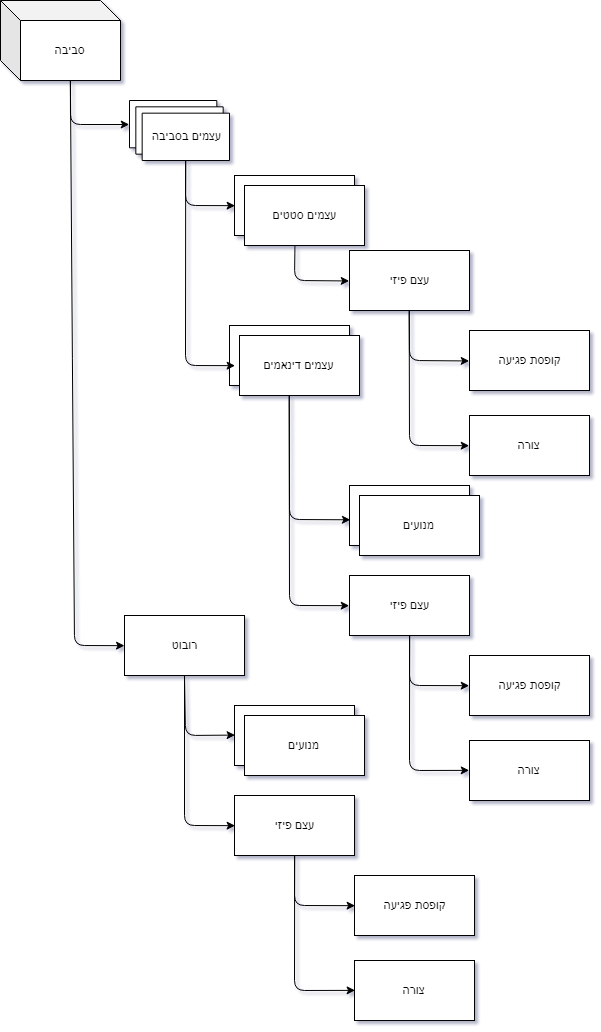
ישנם 19 קבצי-קוד המרכיבים ומפעילים את הפרויקט

* שלושה קבצים אחראים על ממשק המשתמש: מסך פתיחה, מסך קבלת מידע ומסך סימולציה.

אלה שמות הקבצים:

* + open\_window\_vri
  + pre\_testing
  + Simulator
* שלושה קבצים אחראים על הלוגיקה בין הסימולטור והמסכים לבין המסכים עצמם:
  + open\_window\_vri.support
  + pre\_testing.support
  + Simulator.support
* שלושה קבצים אחראים על הגיאומטריה האנליטית בפרויקט:
  + Point,
  + Vector
  + StraightLint
* שני קבצים אחראים על התנגשות בין אובייקטים בסביבה ועל צורה של האובייקטים:
  + Hit\_Box
  + Parallelogram
* הקובץ Object\_builder מקנה אפשרות לסימולציה פיזית לעצמים בסביבה.
* Wall נותן לעצם משמעות בסביבה.
* Environment מרכז את כל העצמים והרובוט בסביבה אחת ואחראי על האינטראקציה בין כל העצמים.
* Robot מתאר עצם פיזי בעל מנועים ויכולת תזוזה
* שלושה קבצים האחראים על מנן התזוזה של אובייקטים זזים כמו Robot:
  + Linear
  + Motor
  + Motor\_motion
* קובץ Error אחראי על שמירתם של כל קודי השגיאות וניתן לקרוא להם דרכו.

2.2 פירוט רכיבי המערכת:

****תיאור מבנה המנוע

מחלקות המנוע:

|  |
| --- |
| Environment - מחלקה |
| תכונות:   * Self.objects - מילון המכיל כמפתח את המספר הסידורי של האובייקט וכערך את האובייקט עצמו * Self.robot - הרובוט שיופעל בסביבה * Self.boundaries - עצם מסוג מקבילית ישרה שמדממה את הגבולות של הסביבה |
| פעולות:   * \_\_inti\_\_ - initiation of object * Check\_robot\_in\_boundaries - **בודק אם הרובוט נמצא בגבולות הסביבה, ומחזיר "אמת" אם הרובוט בגבולות ו"שקר" אם הוא מחוץ** * Check\_onj\_robot\_crash**- בודק אם הרובוט התנגש באחד האובייקטים בסביבה. אם יש התנגשות מחזיר אמת אם אין התנגשות מחזיר שקר** * **Check\_crash- בודק אם יש התנגשות של הרובוט בכללי. מחזיר "אמת" אם יש ו"שקר" אם אין** * **Add\_obj- מוסיף עצם למילון העצמים** * **Add\_obj\_list מקבל רשימה של אובייקטים ומוסיף אותם למילון** * **Move\_robot - מקבל פעולה רצויה (ישר, אחורה, ימינה, שמאלה בשיטת הWASD) ומזיז את הרובוט לפיה** * **Get\_key - מקבל עצם שנמצא במילון ומקבל את המפתח שלו** * **Delete\_obj - מוחק אובייקט מהמילון** * **Static - environment\_from\_file ממיר את קובץ בפורמט xml לסביבה** * **Static - data\_path - מקבל שם של קובץ ומחזיר את הכתובת שלו בספרייה של המידע** * **Static - handle\_objects\_from\_file- מטפל בעצמים בקובץ בפורמט xml** * **convert\_env\_to\_file - ממיר את הסביבה לקובץ בפורמט xml** |

|  |
| --- |
| **Hit\_Box – יורש מParallelogram – המחלקה נועדה כדי לתת מימוש לצורה** |
| **אין משתנים** |
| **אין פונקציות** |

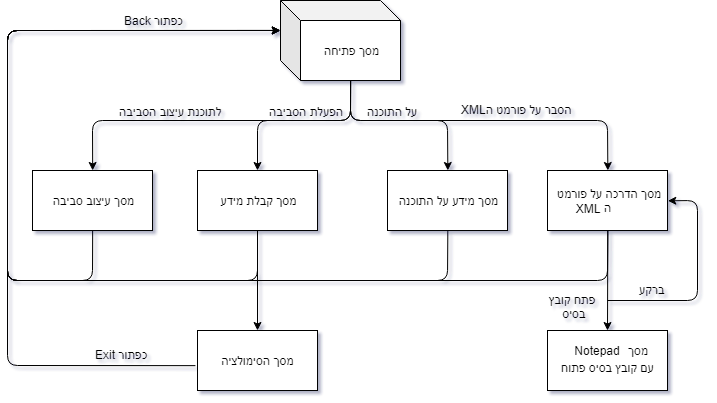
|  |
| --- |
| **ObjectBuilder - יורש מObject** |
| תכונות:   * **Self.shape - צורה שמרכיבה את האובייקט** * **Self.hit\_box - מסגרת פגיעה של האובייקט** |
| פעולות:   * **\_\_init\_\_ - initiation of object** * **Draw - מציין את האובייקט** * **Move\_bt\_units - את האובייקט בסביבה (במנוע) לפי יחידות** * **Rotate - מסובב את האובייקט בסביבה (במנוע) במעלות** * **Static – find\_edge - מוצא את שתי הנקודות הכי קיצוניות של הצורה** |

|  |
| --- |
| **Robot - יורש מObjectBuilder-** |
| תכונות:   * **wheel - רדיוס הגלגל** * **Self.motor1 - מנוע אחד** * **Self.motor2 - מנוע שני** * **- Self.center\_lineווקטור נקודתי המתאר את מרכז הכובד של הרובוט ואת כיוון הנסיעה שלו** * **- Self.motor\_weightמשקל הרובוט** * **Self.shape - צורה שמרכיבה את האובייקט** * **Self.hit\_box - מסגרת פגיעה ש לאובייקט** |
| פעולות:   * **Move - מקבל פעולה רצויה (ישר, אחורה, ימינה, שמאלה בשיטת הWASD) ומזיז את הרובוט לפיה** * **Rotate - מסובב את הרובוט בסביבה (במנוע) במעלות** * **convert\_robot\_to\_txt – ממיר את הרובוט למחזורת בפורמט xml** * **static – robot\_from\_file - ממיר את מחרוזת בפורמט xml לרובוט** |

|  |
| --- |
| **Wall – יורש מ- ObjectBuilder** |
| תכונות:   * **Self.shape - צורה שמרכיבה את האובייקט** * **Self.hit\_box - מסגרת פגיעה ש לאובייקט** |
| פעולות:   * **convert\_obj\_to\_txt – ממיר את הקיר למחרוזת בפורמט xml** * **static – wall\_from\_file – ממיר פורמט xml לאובייקט קיר** |

|  |
| --- |
| **Parallelogram – יורש מ-Object** |
| * **self.relative\_angle – זווית בין הצלעות** * **self.main\_line – צלע מנחה מרכזית בצורה כיוונה ככיוון התנועה כמו כן נקודת ההתחלה של הקו היא נקודת ההתחלה של הצורה** * **self.relative\_line –צלע מנחה קו בזווית self.relative\_angle ביחס לצלע המרכזית** * **self.main\_line\_2 – קו שמקביל ל self.main\_line** * **self.relative\_line\_2 – קו שמקביל ל self.relative\_line** |
| * **get\_lines – מחזיר את הקווים שמרכיבים את הצורה** * **change\_relative\_angle – מקבל זווית שתהווה כערך חדש לself.relative\_angle** * **move\_by\_units – מזיז את הצורה לפי יחידות** * **get\_middle\_point – מחזיר את נקודת המרכז הגיאומטרי של הצורה** * **change\_pos – מקבל נקודה שאליה נקודת ההתחלה של הצורה תזוז** * **change\_rotation – מקבל זווית ווקטור נקודתי שמייצג את מרכז הכבידה ומסתובב סביבו את המעלות** * **change\_size – מקבל שתי אורכים ומציב אותם בקווים המנחים של הצורה** * **get\_points – מחזיר את הקדקודים של הצורה כאובייקט נקודה ברשימה** * **get\_equation – מחזיר את המשוואות של הקווים שמרכיבים את הצורה** * **get\_point\_value – מחזיר את הערכים של הנקודות בתוך רשימה של tuples** * **draw – מצייר את הצורה** * **get\_edge\_points – מחזיר את שתי הנקודות הכי קיצוניות כtuple של עצם נקודה** * **convert\_shape\_to\_txt – ממיר את הצורה למחרוזת בפורמט xml** * **is\_Colliding – מקבל צורה ובודק אם ישנה התנגשות** * **parallelogram\_from\_file - ממיר את מחרוזת בפורמט xml לצורה** |

|  |
| --- |
| **Sensor - מחלקה** |
| **Self.line\_of\_sight – ישר המראה את קוו הראייה של החיישן ואת כיוון הראייה** |
| **Get\_dis\_for\_sensor – מחזיר את המרחק בין החיישן לעצם שנכנס שקוו הראייה שלו** |

 דיאגרמה לתיאור מפת הGUI:

מחלקות הGUI:

|  |
| --- |
| **open\_window\_vri – מחלקה של הGUI** |
| תכונות:   * self.test\_b **– כפתור להתחלת בדיקה שעובר לחלון השגת מידע ופונה לפונקציה** start\_test() **בקובץ תמיכה** * self.xml\_b **– כפתור לקבלת הוראות לכתיבת סביבה ב**xml **שעובר לחלון ההוראות ופונה לפונקציה** xml\_inst() **בקובץ תמיכה** * self.designer\_b **– כפתור לפתיחת המעצב לבניית סביבה אישית שעובר לחלון המעצב ופונה לפונקציה** designer() **בקובץ תמיכה** * self.about\_b **– כפתור לפתיחת מידע על התוכנה שעובר לחלון המידע ופונה לפונקציה**about\_us() **בקובץ תמיכה** |
| פעולות:   * **support - start\_test() – הורס את חלון הפתיחה הורס את חלון הפתיחה ופותח את חלון השגת המידע** * **support - xml\_inst()– הורס את חלון הפתיחה ופותח את חלון ההוראות** * **support - designer()– הורס את חלון הפתיחה ופותח את חלון המעצב** * **about\_us() - support – הורס את חלון הפתיחה ופותח את חלון המידע** |

|  |
| --- |
| pre\_testing – מחלקה של הGUI |
| תכונות:   * **self.Entry\_file\_name – תיבת תוכן שמכילה את השם של קובץ xml של הסביבה** * **self.Back – כפתור שמחזיר למסך פתיחה ופונה לפונקציה back\_pressed() בקובץ התמיכה** * **self.Continue – כפתור להתחלת הבדיקה ופונה לפונקציה continue\_pressed()** * **self.action\_file – תיבת תוכן שמכילה את שם הקובץ הפעולה** * **self.error – תווית לרשום בה שגיאות אם יש את הצורך** |
| פעולות:   * **Back\_pressed() support -– הורס את חלון השגת המידע ופותח את חלון הפתיחה** * **continue\_pressed() – בודק אם הקבצים קיימים אם כן פונה לפונקציה continue\_pressed() בקובץ התמיכה אם לא מציג הודעת שגיאה בself.error** * **support - continue\_pressed() – מקבל את השמות של קובץ הסביבה וקובץ הפעולה, הורס את חלון הפתיחה, יוצר עצם סביבה מהקובץ ופותח את חלון הסימולציה ומפעיל את הסימולטור ומעביר לו את הסביבה לבדיקה ואת קובץ הפעולות** |

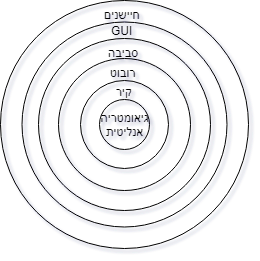
|  |
| --- |
| **Simulator – מחלקה של הGUI** |
| תכונות:   * **Self.env – הסביבה של הסימולטור** * **Self.lap – מספר הסיבוב של הסימולטור** * **self.canvas\_sim – משטח גרפי בוא יש תיאור גרפי של הסימולטור** * **Self.after – משתנה בוליאני המתאר אם להמשיך את פעולת הסימולטור או לא** * **self.menubar - שורת התפריטים בה יש ארבעה פעולות אפשריות:**    + - * **התחלה של הסימולטור – פונה לפעולה start()**       * **עצירה של הסימולטור – פונה לפעולה stop()**       * **שמירה של המצב הקיים – פונה לפעולה save()**       * **יציאה מהסימולטור למסך הרשאי – פונה לפעולה exit() בקובץ התמיכה** |
| פעולות:   * **Start() – כותב בסטטוס שהסימולטור מופעל משנה את self.after לאמת ופונה לפעולה שמפעילה את הסימולטור** * **Stop() – כותב בסטטוס שהסימולטור בעצירה ומשנה את self.after לשקר** * **Save() – שומר את המצב הקיים של הסביבה התיקייה בתוק תיקיית Data** * **Support – exit() – הורס את חלון הסימולציה ופותח את מסך הפתיחה** |

|  |
| --- |
| **Sensor - מחלקה** |
| **Self.line\_of\_sight – ישר המראה את קוו הראייה של החיישן ואת כיוון הראייה** |
| **Get\_dis\_for\_sensor – מחזיר את המרחק בין החיישן לעצם שנכנס שקוו הראייה שלו** |

2.3 דיון בנושא העיצוב הנבחר

* שפת תכנות: בחרתי ב-python לאורך כל הקוד שלי משום שלשפה זו יש ספריות רבות. דבר זה מאפשר גמישות בשיטות הפעולה ומתן אפשרויות רבות להרחבת הפרויקט.
* חלוקה למחלקות בשיטת OOP: שיטה זו יוצרת מדרגיות ושינויים בה קלים יותר. בנוסף OOP מספקת פשטות וארגון במהלך התכנות.

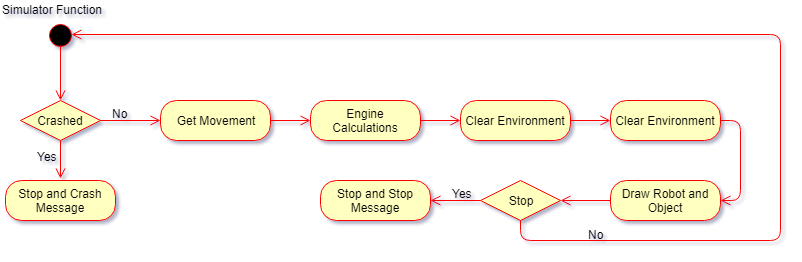
מעגלי פיתוח:



1. **חישובים מתמטיים ואלגוריתמיקה**
2. **הסימולטור**:

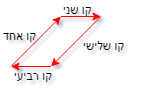
פונקציית הסימולטור:

מבט על הפעולה:



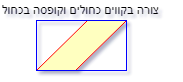
פונקציית הסימולציה היא הפונקציה שמרכזת את הGUI ומנוע הסימולציה. הפונקציה מריצה את המנוע, בודקת התנגשויות ומשנה את הGUI כך שיהיה תצורה גרפית בזמן ההרצה

הרכב של צורה:

כל עצם מורכב מקווים המתוארים בפונקציה מתמטית וכמו כן כווקטור המגביל את גודל הקו כל קו מתחיל בנקודת הסיום של הקו לפניו חוץ מהקו הראשון שהוא מתקבל ביצירת הצורה. כך נוצר פוליגון שמהווה גוף לעצם. דוגמה לפוליגון שיכול להיות:

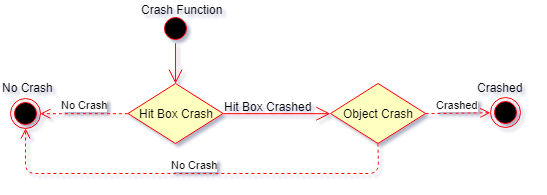
קופסת פגיעה – Hit Box:

כל עצם בסביבה יש מלבן או "קופסה" סביבו המאונך לצירי x ו-y של המסך. לקופסה קוראים "Hit Box ".

קצוות הקופסה נמצאים בערכים הכי קיצוניים של העצם שעליו הם נמצאים. מטרת הקופסה היא להקל על החישובים של התנגשות בכך שהיא מורכבת רק מארבעה ישרים וכך החישוב קצר יותר כמו כן חישובים של פונקציות שמאונכות לצירים יותר פשוטות .דוגמה לקופסה:

בדיקת התנגשות בין עצמים:

כאשר הסימולטור בודק אם יש התנגשות בין שתי עצמים הוא קודם בודק אם קורת התנגשות בין Hit Box יש עצם אחד עם Hit Box של עצם אחר. המחשה וויזואלית של הבדיקה:



איך בודקים אם שתי צורות מתנגשות:

על צורה של עצם יש קווים ממנה היא מורכבת וכך גם עצם Hit Box מורכב מצוהר שבנויה מישרים. כל ישר מתואר בעזרת פונקציה מתמטית על המישור x, y ווקטור, כדי לבדוק התנגשות צריך

* לבדוק אם לקוו מצורה של עצם אחד יש נקודה משותפת עם קוו מצורה של עצם אחר. כלומר צריך להשוות בין משוואות הישרים
* אם יוצא נקודה משותפת משמע הצורות מתנגשות. לכן:
  + בשלב ראשון בודקים את ה-Hit Box משום שמשוואות הישרים שלו מקבילים לצירים
  + אם יש התנגשות בודקים את הצורה עצמה שיכולה להיות מורכבת ממספר רב של ישרים בזוויות שונות.

1. **פורמט המידע השמור**

לפני התחלת הסימולטור המערכת מקבלת שתי קבצים מתוך תיקיית הData שנמצאת בתוך התיקייה של התוכנה, האחד קובץ שמתאר את הרכב הסביבה בפורמט xml השני קובץ Python בוא יש פעולות לרובוט.

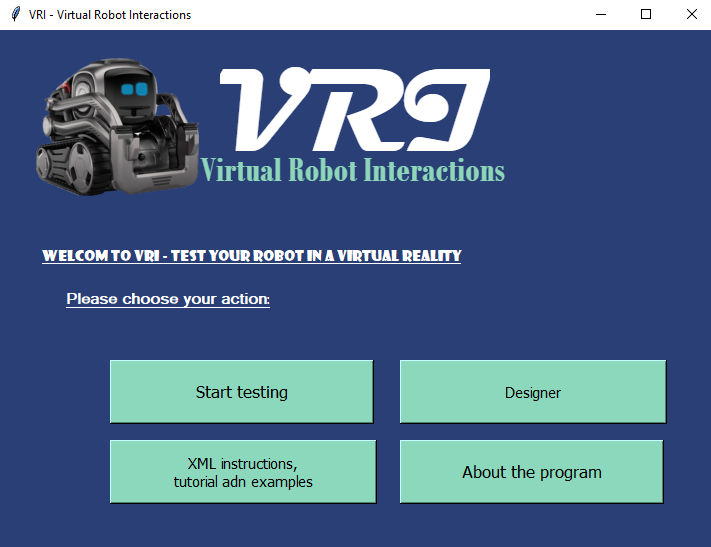
קובץ הסביבה:

קובץ הסביבה מתקבל בפורמט xml כאשר הוא נכתב במבנה היררכי כמו Tag base. כאשר רוצים לתת את המשתנים שכל עצם צריך כדי ליצורו רושמים את הערכים בין הפתיחה של הכותרת לסגירה. דוגמה לקובץ של סביבה (ראו בנספחים)

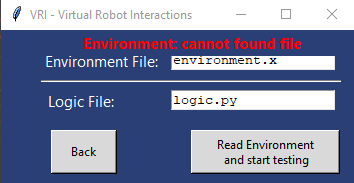
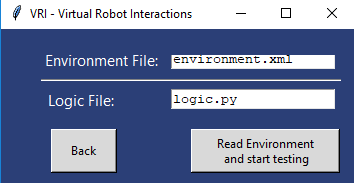
1. **ממשק משתמש**

"ממשק משתמש" של הפרויקט מורכב מאלה המסכים:

1. מסך פתיחה – החלון הראשון העולה בפתיחת התוכנה. החלון מעוצב מלוגו הפרויקט ומארבעה כפתורי בחירה: האחד, פתור להפעלת הסביבה; השני, כפתור לתוכנת עיצוב סביבה מותאמת אישית; השלישי, כפתור להסבר על פורמט ה-XML לתכנות סביבה; הרביעי, כפתור מידע על אודות התוכנה.



1. מסך קבלת המידע – החלון העולה בלחיצה על כפתור "הפעלת הסביבה". עומדת לרשות משתמש התוכנה האפשרות לשנות את ברירת מחדל של קובץ הסביבה ושל קובץ הלוגיקה של הרובוט. נוסף על כך ישנם שני כפתורים: כפתור אחד מיועד לאפשר למשתמש לחזור למסך הקודם וכפתור שני מיועד לאפשר למשתמש להמשיך בהפעלת הסביבה. במקרים של בעיה במציאת הקבצים, תעלה בחלון הודעת Error.

מסך רגיל: מסך עם הודעת Error:

1. מסך הסימולציה – חלון הסימולציה מעוצב משורת תפריט: כפתור הפעלה, כפתור עצירה, כפתור שמירה וכפתור יציאה. כפתור הפעלה מפעיל את הסימולציה; כפתור עצירה עוצר את הסימולציה; כפתור שמירה שומר על המצב הנוכחי של הסביבה; כפתור יציאה מוציא ממסך הסימולציה למסך פתיחה. נוסף על מסך הסימולציה עומדים לרשות המשתמש אלה השניים: מחוון שליטה על מהירות ההפעלה ותווית להצגת מצבם של הרובוט ושל הסביבה: מופעל, עוצר, התנגשות.



5. **נספחים**

**הוראות התקנה על המחשב:**

1. הורד את התיקייה VRI
2. הורד Pytohn 3.7
3. הורד את הספרייה xml
4. הורד את הספרייה tkinter

**הוראות הפעלה:**

1. כנס לתיקייה GUI שבתוך VRI והרץ את קובץ Open\_window\_vri

\* כדי שהקובץ סביבה ולוגיקה יפעלו יש להשים אותם בתיקיית Data

דוגמה לקובץ הסביבה הכולל סביבה, קיר ורובוט:

Environment>

<height>1000</height>

<width>1000</width>

<Robot>

<center\_of\_mass>

<Point>

<x>150.0</x>

<y>450.0</y>

</Point>

</center\_of\_mass>

<shape>

<Parallelogram>

<StraightLine>

<Point>

<x>250</x>

<y>500</y>

</Point>

<Vector>

<length>200</length>

<angle>180</angle>

</Vector>

</StraightLine>

<angle>90</angle>

<length>100</length>

</Parallelogram>

</shape>

<wheel>2</wheel>

<motor\_1>

<name>motor\_1</name>

<torque>48</torque>

<weight>210</weight>

<power>5</power>

<shaft\_diameter>6.0</shaft\_diameter>

</motor\_1>

<motor\_2>

<name>motor\_2</name>

<torque>48</torque>

<weight>210</weight>

<power>5</power>

<shaft\_diameter>6.0</shaft\_diameter>

</motor\_2>

</Robot>

<Objects>

<Wall>

<Parallelogram>

<StraightLine>

<Point>

<x>200</x>

<y>200</y>

</Point>

<Vector>

<length>100</length>

<angle>0</angle>

</Vector>

</StraightLine>

<angle>120</angle>

<length>200</length>

</Parallelogram>

</Wall>

</Objects>

</Environment>