בס"ד

אמיר מזרחי

אריאל אשכנזי

פרוייקט מטלב – גולף

מבוא

הפרוייקט אותו יצרנו מדמה משחק גולף. ביכולתו של השחקן לבחור דרגת קושי (המתבטאת בגודל החור אליו יש לקלוע) וכן את מקדם חיכוך המשטח. דרגת קושי גבוהה יותר מזכה במספר נקודות רב יותר עבור קליעה. התוכנית צוברת עבור השחקן את הנקודות עד לסיום המשחק. מספר הנקודות עבור קליעה קטן בהתאם לזמן שלקח לשחקן לבצע את הקליעה.

רקע תיאורטי כללי ומשוואות הדינמיקה

המשחק מבוסס על משוואות תנועה פשוטות תחת השפעת כח חיכוך. מהירות הכדור ההתחלתית נקבעת ע"פ גודל וכיוון הוקטור היוצא ממרכז הכדור אל הנקודה בה לחץ השחקן עם העכבר על מערכת הצירים. מהירות הכדור המקסימלית נקבעת ע"פ נוסחה המופיעה בקוד ונועדה לדמות משחק ריאלי ככל הניתן.

בתנועתו פועל על הכדור כח חיכוך ששוה למכפלת מסת הכדור, מקדם החיכוך וקבוע הגרביטציה האוניברסלי (g). לכן תאוטת הכדור היא (מסת הכדור מצטמצמת):

Ax=g\*µ\*cos(teta)

Ay=g\*µ\*sin(teta)

כשאר טטא היא זווית הוקטור הנ"ל עם ציר X של מערכת הצירים.

התקדמות הכדור מתבצעת בהפרשי זמן של period=0.033 שניות. בכל נקודת זמן מחושבת מהירות הכדור הנוכחית (לכל ציר בנפרד) ע"פ הנוסחה v[t]=v[t-1]-a\*period. אם מהירות הכדור בנקודת הזמן הנוכחית קטנה מ a\*period אזי תנועת הכדור נעצרת.

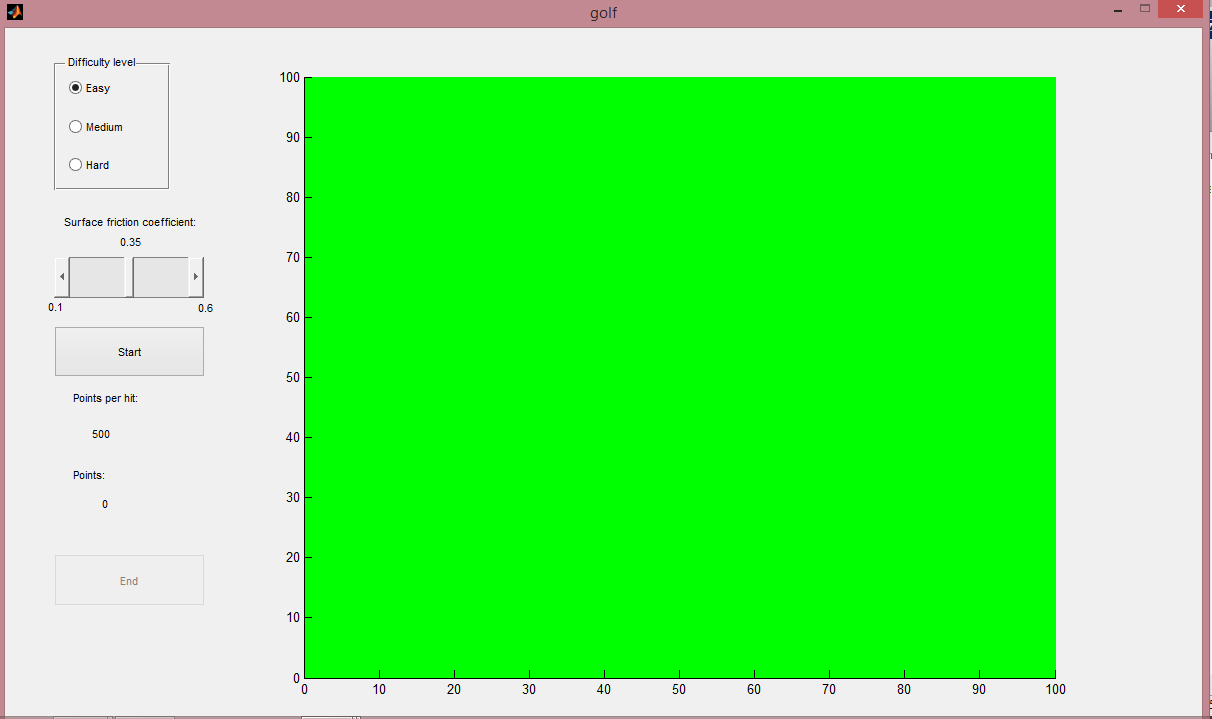
מיקום הכדור בכל נקודת זמן מחושב ע"י (לכל ציר בנפרד) x[t]=x[t-1]+v[t]\*period. אם מיקום הכדור חופף למיקום החור ומהירות הכדור קטנה דיה (הנוסחא המדויקת מופיעה בקוד והיא מותאמת כך שמהלך המשחק יהיה ריאלי ככל הניתן) אזי תנועת הכדור נעצרת והשחקן זוכה במספר הנקודות הראוי.

אם מיקום הכדור יוצא מגבולות המגרש (1000 יחידות לכל כיוון) אזי תנועת הכדור נעצרת ומופיעה למשתמש הודעה שהכדור יצא מגבולות המגרש.

אופן הפעלת היישום

בפונקציית הפתיחה של האובייקט הגרפי קבענו את משתני התוכנית לערכי ברירת המחדל שלהם. לדוג' רמת הקושי נקבעה להיות קלה, מקדם החיכוך נקבע להיות 0.35 וכן הלאה. בפונקציה זו קיים דגל פנימי handles.isRandom שניתן לשנותו ל-0 כך שמיקום החור יהיה קבוע ולא רנדומלי.

בשלב זה מופיעה מערכת הצירים ריקה. מסך הפתיחה נראה כך:



ביכולתו של המשתמש לבחור את רמת הקושי ע"י לחיצה על הכפתור המתאים, הניקוד עבור קליעה ישתנה בהתאם. כמו כן יכול המשתמש לשנות את מקדם החיכוך של המשטח ע"י הסליידר. טווח הערכים מתאים לטווח הערכים המציאותי.

כאשר המשתמש מעוניין להתחיל במשחק לוחץ הוא לחצן start.

לחיצה זו מאתחלת את המשחק- יוצרת אובייקט טיימר ומשייכת לו את פונקציית עדכון הנקודות (@update\_points), מכבה את לחצני בחירת הרמה, מקדם החיכוך ולחצן התחלת משחק, מאפסת את וקטור מהירות הכדור וקובעת את גודל המשטח ל-100 יחידות בכל כיוון. פונקציה זו קוראת לפונקציה buildGame.

פונקציית buildGame אחראית על יצירת האוביקטים הגרפיים על המשטח- יוצרת את הכדור ואת החור, ממקמת אותם (כאשר מיקום החור מוגרל בטווח ערכים מסויים) ויוצרת את הקו המציג את וקטור המהירות.

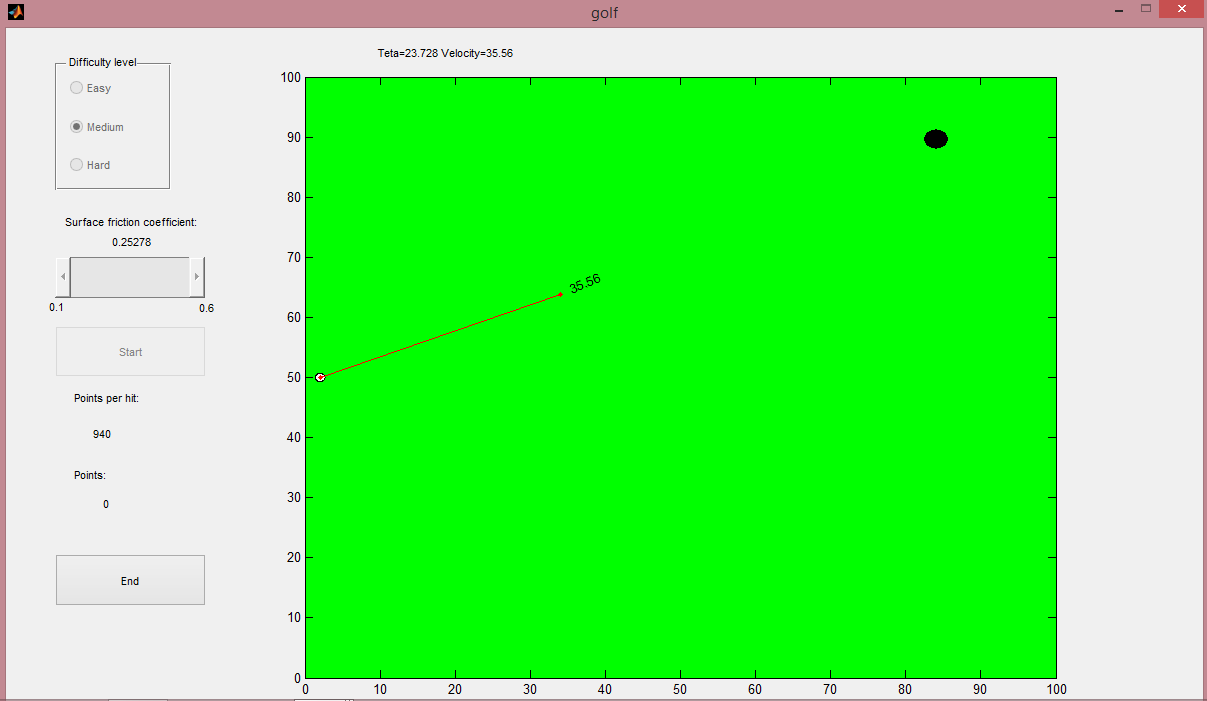
לאחר סיום פונקציית buildGame מפעילה פונקציית start את הטיימר, מגדירה את פונקציות ה-callback עבור ButtonDownFcn(@hit) וכן WindowButtonMotionFcn(@speed).

פונקציית עדכון הנקודות מורידה מסך הנקודות שניתן לקבל עבור קליעה 5 נקודות בכל חצי שניה שעוברת. מינימום נקודות לקליעה הוא 5.

הפונקציה speed אחראית להצגת קו אדום המייצג את וקטור המהירות. כמו כן הפונקציה אחראית על חישוב גודל המהירות והזווית והצגתם למשתמש.

הפונקציה hit נקראת כאשר לחץ המשתמש על מערכת הצירים והיא אחראית לקביעת המהירות ההתחלתית ולהגדרת פונקציית הטיימר להיות running. כמו כן היא מבטלת את האפשרות ללחיצה חוזרת על הצירים וכן את פונקציית תנועת העכבר.

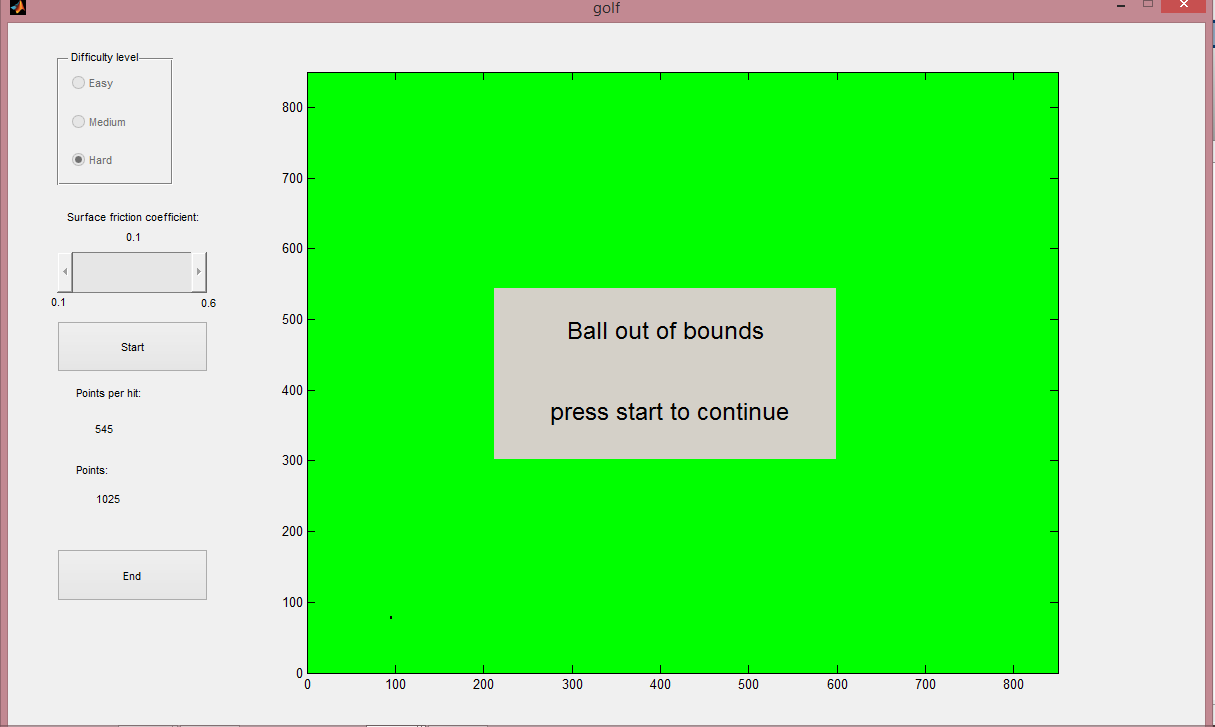
צילום מסך של תחילת משחק:



הפונקציה running אחראית על הצגת תנועתו של הכדור בעזרת משוואות התנועה שכתבנו לעיל. כמו כן בודקת הפונקציה האם הכדור נקלע אל החור. במקרה זה תודפס למסך הודעה זו:

באם לא התרחשה קליעה אזי בודקת הפונקציה האם הכדור נעצר (בהתאם לגודל המהירות כמתואר במשוואות לעיל). במקרה זה תגדיר הפונקציה את הפונקציות speed ו-hit כדלעיל על מנת לאפשר לשחקן לנסות שוב לקלוע. בנוסף הפונקציה בודקת את מיקום הכדור ומתאימה את גודל מערכת הצירים כך שיתקבל הריבוע הקטן ביותר המכיל גם את הכדור וגם את החור.

במידה והכדור יצא מגבולות מערכת הצירים תופיע למשתמש הודעה זו:



ויוכל המשתמש ללחוץ start ולהמשיך במשחק.

בכל שלב במשחק יכול המשתמש ללחוץ על לחצן end ולסיים את המשחק- לסמן באדום את כמות הנקודות שנצברה, לאפשר בחירת פרמטרים והתחלת משחק חדש. לחיצה על לחצן end מציגה למשתמש את ההודעה הבאה:



נספח א- משוואות התנועה

כאמור לעיל תנועת הכדור במשחק מבוססת על תנועה תחת השפעת כח חיכוך. כידוע, כח החיכוך הקינטי שווה לכח הנורמלי כפול מקדם חיכוך המשטח, כאשר הכח הנורמלי שווה כמובן לכח הכובד:

כאשר µ מקדם חיכוך המשטח, |**N**| גודל הכח הנורמלי, m מסת הכדור, ו-g קבוע הגרביטציה האוניברסלי השווה בערכו ל- 9.8 . הכוחות מסומנים באות מודגשת לציין שמדובר בגדלים ווקטוריים. כיוון כח החיכוך הוא בכיוון המנוגד למהירות הכדור.

בהתאם לחוק הראשון של ניוטון תאוטת הכדור בהשפעת כח חיכוך זה תהיה שווה ל:

וכיוונה ככיוון כח החיכוך.

ע"מ להפריד את רכיבי הוקטורים בבעיה להיטליהם על צירים X ו-Y נשתמש בזווית טטא שיוצר וקטור המהירות עם ציר ה-X וכך נקבל שההיטל על ציר X של וקטור כלשהו בבעיה שווה לגודל הווקטור כפול קוסינוס טטא, ואילו היטלו על ציר Y הוא גודל הוקטור כפול סינוס טטא (כאשר עבור החיכוך והתאוטה נשתמש פשוט בסימן מינוס ע"מ לקבל שכיוונם יהיה מנוגד לכיוון וקטור המהירות).

כעת בכל רכיב מתבצע תנועה פשוטה תחת תאוטה קבועה ולכן ניתן לקבוע שהמהירות (בכיוון ציר i כלשהו) בזמן נתון היא:

ומיקום הגוף (בכיוון ציר i כלשהו) בזמן נתון יהיה כמובן:

בפועל תנועת הכדור במשחק איננה רציפה אלא דיסקרטית במרווחי זמן של 0.033s (כלומר תדירות של כ- 30.3 Hz , תדירות מספקת ע"מ שהתנועה תיראה כרציפה לעין המתבונן) ולכן משוואות אלו יהפכו למשוואות דיסקרטיות. כלומר, המהירות בכל אינטרוול תהא שווה למהירות באינטרוול הקודם פחות תאוטת הכדור כפול גודל האינטרוול (Period=0.033s). ומיקום הכדור בכל אינטרוול שווה למיקום הכדור הוקדם ועוד מהירות הכדור כפול גודל האינטרוול.

המשוואות שבהן השתמשנו בפועל לתנועת הכדור במשחק:

*נספח ב- פונקציית* speed

*במהלך המשחק נדרש השחקן לכוון בעזרת העכבר את המהירות ההתחלתית של הכדור. פעולה זו מתבצעת בעזרת פונקציית speed המוגדרת להיות פונקציית WindowButtonMotionFcn של האובייקט הגרפי.*

*פונקציה זו היא פונקציית callback המופעלת כאשר סמן העכבר נע על גבי האובייקט הגרפי. על מנת לקבל את מיקום העכבר ברגע קריאת הפונקציה השתמשנו בפונקציית get עבור התכונה CurrentPoint.*

*כעת הפונקציה speed מחשבת את מיקום ראש חץ המהירות ביחס למיקום הכדור ומגדירה את XData ו- YData של אובייקט הקו, שנוצר בהתחלת המשחק, בהתאם (הפונקציה לא מגדירה את הקו להגיע עד לסמן העכבר ממש וזאת על מנת לאפשר לחיצה על מערכת הצירים). כך התקבל חץ אדום היוצא מן הכדור ומצביע אל מיקום סמן העכבר.*

*כעת מחשבת הפונקציה את גודל הקו הנ"ל (חישוב גודל וקטור בעזרת משפט פיתגורס). גודל הקו הוא גודל המהירות ההתחלתית של הכדור. גודל המהירות מוצג מעל מערכת הצירים וכן בסמוך לקו בכיתוב שחור (*ראה נספח ג*). אם גודל המהירות גדול מגודל המהירות המקסימלית שהוגדרה בקוד אזי המהירות נקבעת להיות שווה לגודל המהירות המקסימלית ולא לגודל הקו. במקרה זה יוצג גודל המהירות בכיתוב אדום.*

*כמו כן מחשבת הפונקציה את הזווית טטא שיוצר הקו עם ציר ה-X. הזווית טטא שווה לatan של רכיב ה-y חלקי רכיב ה-x. עם זאת ע"מ לקבל זווית מוגדרת נכון (שהרי פונקציית טנגנס מחזורית פאי) השתמשנו במספר תנאים ע"מ לקבוע את נוסחת חישוב הזווית המתאימה. הזווית המתקבלת נעה בין 0 ל-180 מעלות ברביעים הראשון והשני, ובין מינוס 180 ל-0 מעלות ברביעים השלישי והרביעי. הזווית טטא מופיעה מעל מערכת הצירים.*

*נתוני המהירות והזווית נשמרים במקומות המתאימים ב-handles של האובייקט הגרפי.*

*נספח ג- timer*

*במהלך פעולת התוכנית נעשה שימוש באובייקט מסוג timer. האובייקט נוצר בתחילת המשחק ונמחק בסגירת האובייקט הגרפי. אובייקט הטיימר משמש לשתי מטרות שונות במהלך המשחק, כלומר קיימות שתי פונקציות callback לטיימר (הנקראות כאשר הטיימר מתקדם) והתוכנית מחליפה ביניהן בהתאם לשלבי הביצוע. כמו כן התוכנית מפעילה ומכבה את הטיימר בהתאם לנדרש.*

*פונקציה אחת היא פונקציית update\_points. פונקציה זו משמשת לשלב בו השחקן מכוון את מהירות הכדור ותפקידה הוא להקטין את מספר הנקודות שניתן לקבל עבור קליעה ככל שהזמן עובר.*

*עבור פונקציה זו מוגדר מחזור השעון להיות 0.5 שניה. כלומר, בכל חצי שניה שעוברת יורדות 5 נקודות מכמות הנקודות שיתקבלו עבור קליעה. עם זאת, מוגדרת כמות מינימלית של 5 נקודות, כך שאם עבר די זמן וכמות הנקודות שניתן לקבל ירדה ל-5 אזי הפונקציה לא תמשיך לחסר נקודות בכל מחזור שעון. הגדרת מחזור השעון לחצי שניה נעשתה מתוך מחשבה שזהו המחזור המתאים ע"מ ליצור מטיבציה אצל השחקן לבצע את המהלך בצורה מהירה. כמובן שניתן לשנות את מחזור הזמן לכל גודל אחר.*

*הפונקציה השניה היא פונקציית running. פונקציה זו משמשת להצגת תנועתו של הכדור על מערכת הצירים.*

*עבור פונקציה זו הוגדר מחזור השעון להיות 0.033 שניות, ע"מ ליצור תדירות של כ- 30.3 Hz שתדמה תנועה רציפה של הכדור.*

*בכל מחזור שעון הפונקציה מחשבת את מהירות הכדור הנוכחית ואת מיקומו (בעזרת משוואות התנועה המפורטות בנספח א), וממקמת את הכדור על מערכת הצירים בהתאם. בנוסף פונקציה זו בודקת האם התרחשה קליעה של הכדור אל החור (אם כן- הטיימר מופסק, מוצגת הודעה למשתמש וכפתור ה-start נדלק ע"מ לאפשר למשתמש להמשיך במשחק).*

*אם לא התבצעה קליעה בודקת הפונקציה האם הכדור נעצר (החישוב נעשה כמפורט לעיל), במקרה זה הטיימר נעצר והתוכנית חוזרת לשלב בו על השחקן לכוון את מהירות הכדור. כלומר פונקציית הטיימר מוגדרת שוב להיות update\_points.*

*אחר עצירת הכדור הפונקציה בודקת כיצד יש להתאים את גבולות המסך בהתאם למיקום בו נעצר הכדור- היא מחשבת את הריבוע הקטן ביותר המכיל את הכדור ואת החור יחד ומגדירה כך את גבולות מערכת הצירים. במידה ומיקום הכדור חורג מ-1000 יחידות בכיוון כלשהו תופיע הודעה על כך למשתמש וכפתור start יופעל על מנת לאפשר למשתמש להמשיך במשחק.*

*נספח ד-הצגת המהירות בקצה הקו*

*קטע הקוד העוסק בצהגת המהירות בסמוך לקו נמצא בשורות 203-216, כחלק מפונקציית speed שתיארנו בנספח ב.*

*האלמנט האחראי על הצגת המהירות בסמוך לקו הינו אלמנט טקסט handles.speedTextLine. התכונה ‘HitTest’ של האלמנט הוגדרה להיות ‘Off’ על מנת שלא להפריע לאפשרות ללחוץ על מערכת הצירים (ובכך "לחבוט" בכדור). צבע הטקסט הוא שחור אך הוא מתחלף לאדום במידה והמהירות הנוכחית גבוהה מהמהירות המקסימלית שהוגדרה בקוד.*

*לאחר חישוב המהירות התוכנית קובעת את מיקום אלמנט הטקסט בסמוך למיקום קצה הקו (הנוסחה המדויקת המופיעה בקוד מחשבת למעשה גם היכן יש למקם את האלמנט ביחס לקו- מעליו\מתחתיו ו-מימינו\משמאלו).*

*בנוסף התוכנית מחשבת את זווית הטיית הטקסט. על מנת לדאוג על הטקסט תמיד ישר התוכנית בודקת את זווית וקטור המהירות ובהתאם קובעת את זווית הטקסט הרצויה באופן שהטקסט לא יתהפך. (החישוב המדויק מופיע בקוד)*