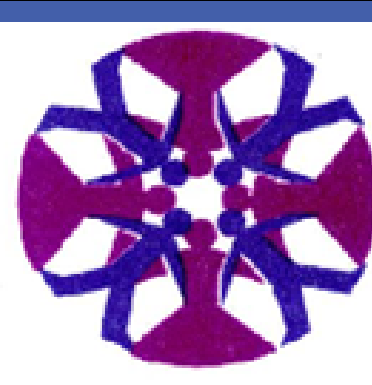
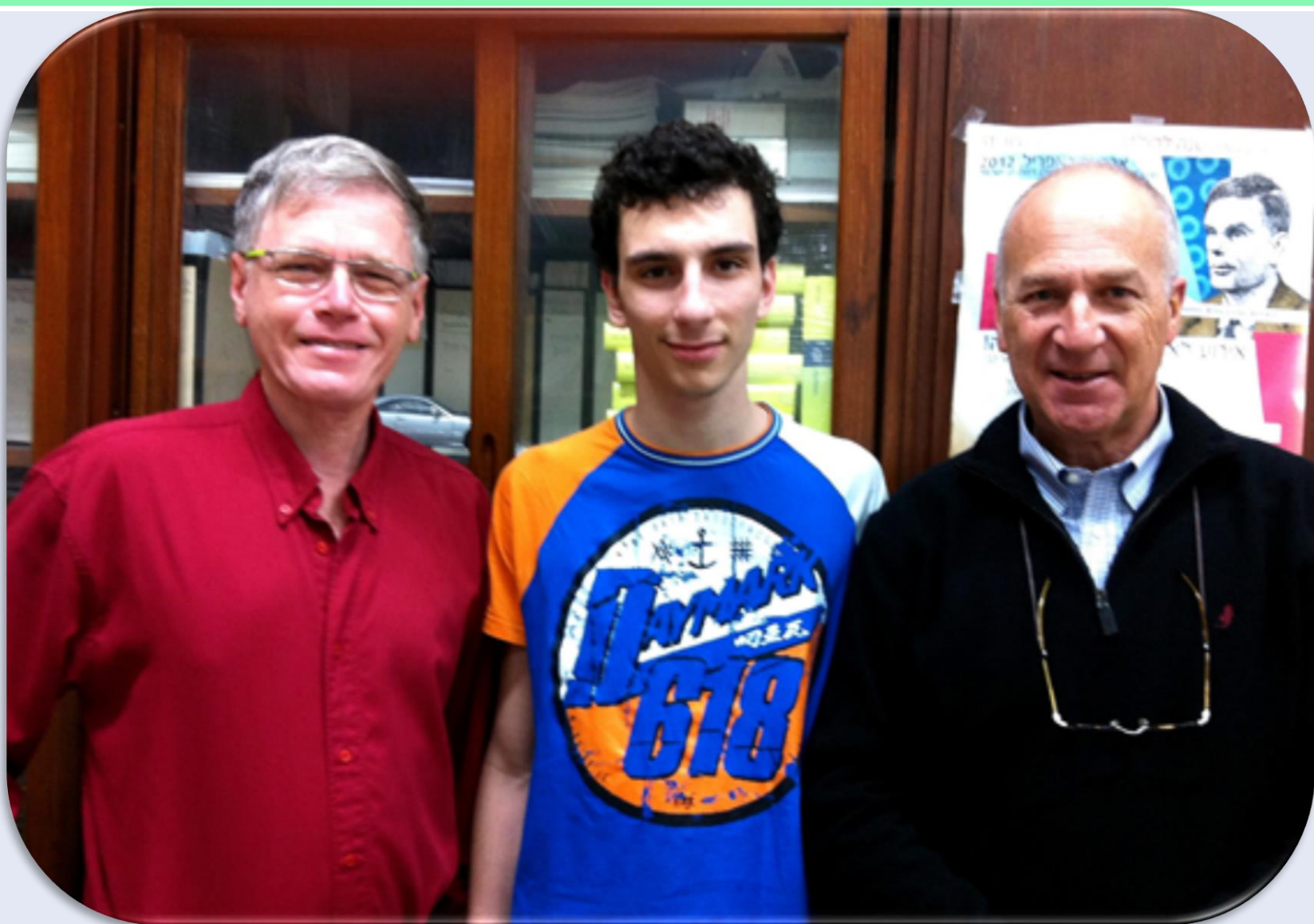




אילו בעיות עולות בעת עיצוב תוכנית התנהגותית וכיצד ניתן לפתור אותן?

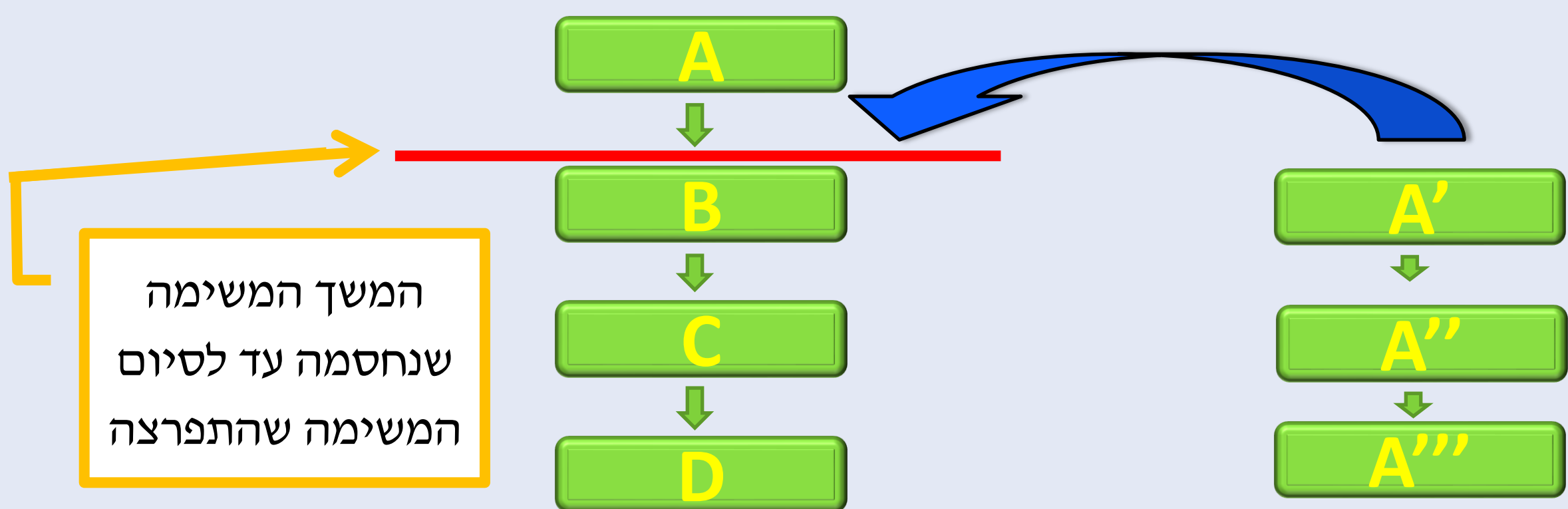


תוצאות המחקר: דוגמאות לבעיות עיצוב ופתרון

"התפרצות" של משימה למשימה אחרת והמשכה

הבעיה: על ההליקופטר היה לתקן סטייה שנוצרה עקב משב רוח ולאחר מכן להמשיך במסלול הקודם. כאשר הוכנס תיקון פשוט עבור הסטייה, שתי המשימות השתלבו ולא בוצעו אחת אחרי השנייה.

פתרון: יש לחסום בכל תהליך שמטפל במשימה שהתפרצה את הפעולות שקשורות למשימה שהופסקה. לאחר סיום הטיפול במשימה המתפרצת, תמשיך התוכנית מהנקודה בה המשימה הקודמת הופסקה.

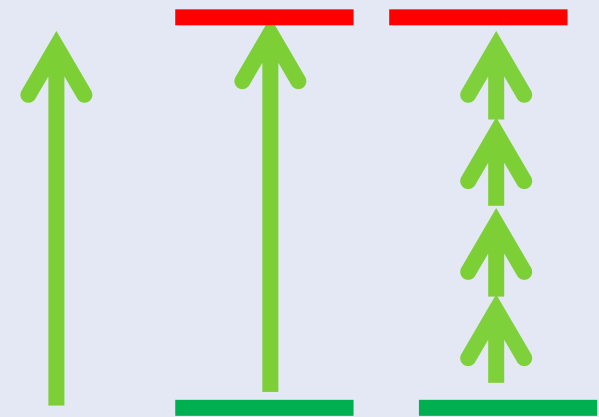


טיפול בתנועה ארוכה

הבעיה: בביצוע תנועה ארוכה בעזרת אירוע יחיד, אפשרי שהאירוע הבא יבחר לפני השלמת התנועה והפקודה הבאה תתקבל לפני השלמת הפעולה הקודמת.

פתרון: שתי שיטות לטיפול בבעיה:

- שימוש באירועים נפרדים להכרזה על תחילת הפעולה וסיומה.
- הרכבת הפעולה ממספר רב של פעולות קטנות יותר.



הפרדה בין אירועי מערכת (פנימיים) וסביבה (חיצוניים)

הבעיה: המערכת קיבלה הודעה על אירוע חיצוני לפני שסיימה לטפל באירוע הקודם.

פתרון: ניתן להפריד בין אירועים פנימיים לחיצוניים כאשר בין כל אירוע חיצוני למשנהו יגיעו כל האירועים הפנימיים.

אירועים פנימיים

- התחל צביעה אנכית
- זז ימינה
- החלף צבע

אירועים חיצוניים

- הגיע הזמן להחליף צבע
- הרוח הסיטה את הרובוט
- הגעת לקצה הקיר

מסקנות ודיון

- מצאנו פתרונות כלליים למספר סוגי בעיות שעולות בעת תכנון תוכנית התנהגותית.
- לכל בעיה, יש מספר פתרונות מתאימים.
- תבנית פתרון כלשהי, יכולה לפתור מספר סוגי בעיות.
- הפתרונות יכולים לשמש בסיס לדיון על שינויים והוספות לבסיס התכנות ההתנהגותי.

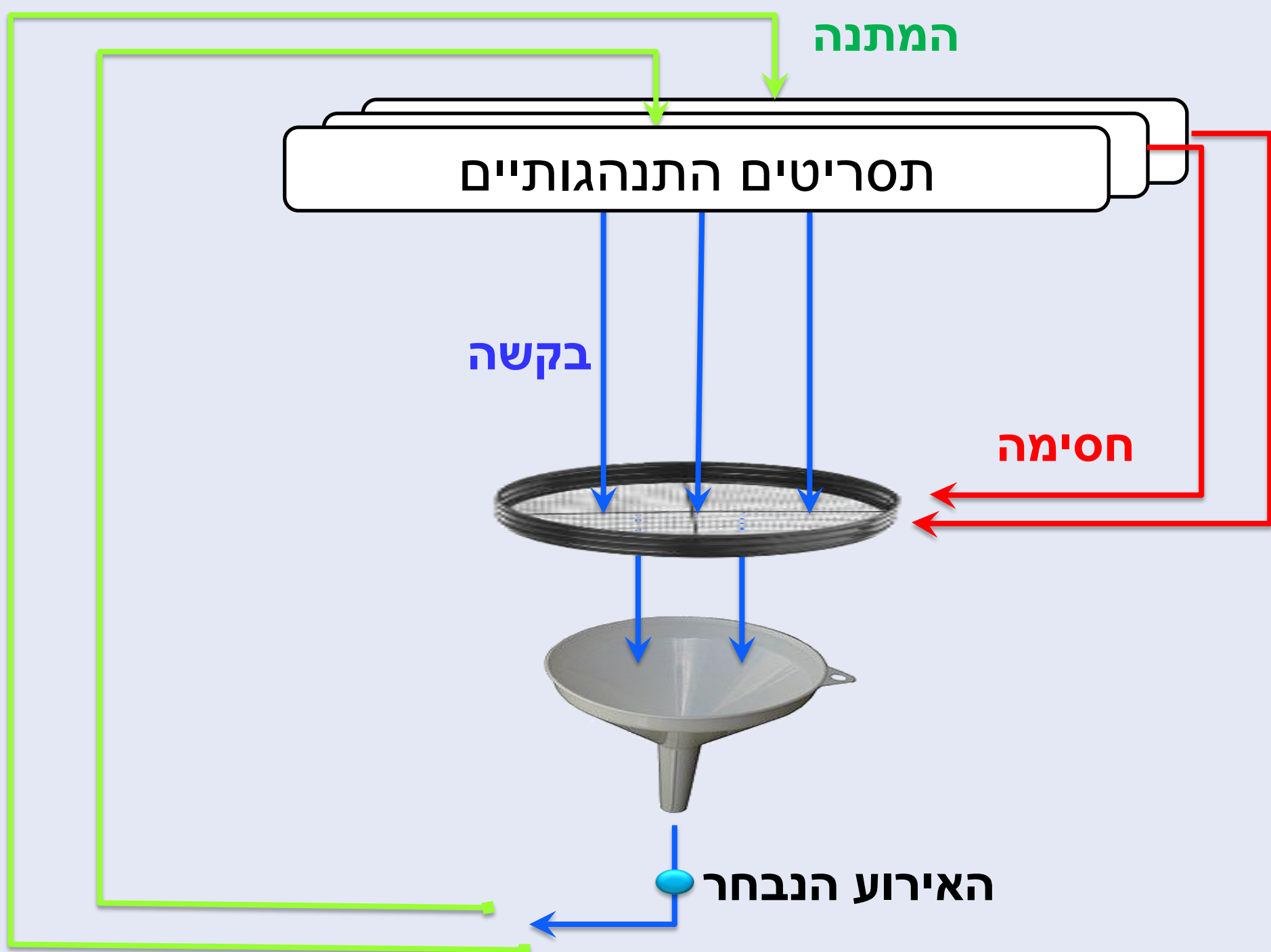
רקע תאורטי

תכנות התנהגותי היא גישה חדשה לפיתוח מערכות תגובתיות (Reactive Systems). כל רכיב של המערכת ממופה ישירות להיבט, או תסריט, מסוים של התנהגות המערכת כפי שהוא מתואר במסמך הדרישות. התכנות ההתנהגותי, הנקרא גם תכנות מונחה תסריטים, הוצג לראשונה על ידי פרופ' דוד הראל בשפה live - LSC - sequence charts . בפרויקט שלי, השתמשתי בספרייה BPL לתכנות התנהגותי בשפת JAVA .

בPPL, ישנם שני אלמנטים מרכזיים:

- Event - אירוע התנהגותי, לדוגמא: הכפתור נלחץ, ההליקופטר הגיע לנקודה X.
- b-Thread (Behavior Thread) - מבקש, חוסם ומחכה לאירועים.

כל תהליך מכריז על אירועים התנהגותיים שהוא מבקש, אירועים נוספים שהוא ממתין להם ואירועים אותם הוא חוסם, כל התהליכים מסתנכרנים. המערכת בוחרת מתוך רשימת האירועים את האירוע בעל העדיפות הגבוהה ביותר שאינו חסום ומעירה את כל התסריטים שביקשו או המתינו לאירוע זה. תהליך זה חוזר על עצמו.



שיטת החקר

בעבודה זו רצינו לזהות בעיות שעשויות לעלות בעת כתיבת תוכנית התנהגותית. לבעיות שזיהינו, פיתחנו שיטות ותבניות שונות. המחקר התבצע בשיטה מעגלית - קיבלנו משימה, תכננו ועיצבנו את התוכנית. בזמן ולאחר הכתיבה, התגלו שאלות בעיצוב התוכנה נמצאו פתרונות לבעיות והם תועדו. לאחר התיקון התווספה משימה נוספת, וחוזר חלילה. בסיום, הפתרונות הוכללו לשיטות ותבניות כלליות יותר לבעיות מאותו הסוג.

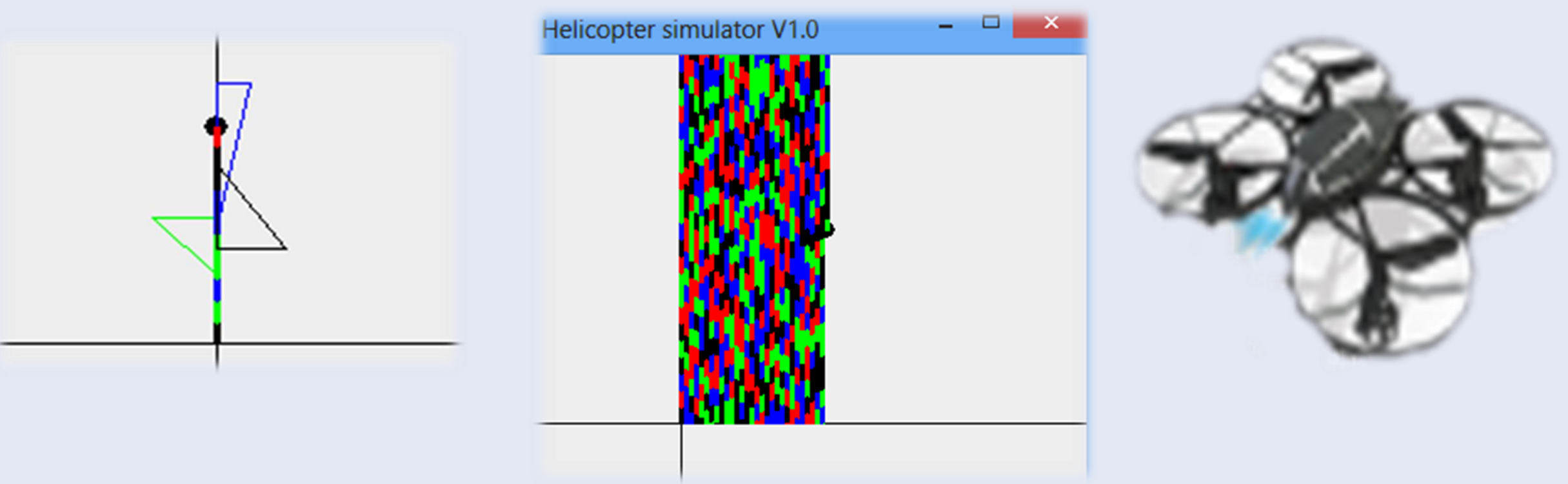
עבודת החקר גם ייצרה דוגמה נוספת במאגר הדוגמאות שממחישות את יתרונות התכנות התנהגותי בפיתוח ותחזוקת תוכנה.

המשימה

המשימה בה בחרנו לצורך הפרויקט הייתה צביעת קיר (או משטח) על-ידי ההליקופטר/רובוט. הרובוט נע במקביל לקיר ומתמודד עם תת משימות ומטלות כגון כיסוי שלם של הקיר, החלפת צבעים, תיקון סטיות עקב רוח וכדומה.

```
public class FixWind extends BThread {
    ...
    if (dx > 0) {
        bSync(eMoveLeftMaint, none, MissionMoves);
    } else if (dx < 0) {
        bSync(eMoveRightMaint, none, MissionMoves);
    }
    ...
}
```

תסריט בג'אווה: תיקון מיקום בעקבות סטייה עקב משב רוח.
בקשת אירועי תזוזה "לתחזוקה" תוך חסימת אירועי המשימה המקורית.



חזרה למסלול לאחר סטייה

צביעת קיר ב-4 צבעים

ההליקופטר - Quad Rotor