**מדריך למתכנת**

התקנה:

יש לבצע התקנה של הספריות הנלוות כמתואר במדריך למשתמש.

לאחר שהן מחוברות ועובדות כמו שצריך, ניתן לחבר את קבצי הcpp והh ולהשתמש בהם ישירות.

\*במקרה של שימוש ישיר בקבצי הcpp יש צורך להתקין גם את linalg ולהוסיף את linalg.h אל הפרוייקט. http://github.com/sgorsten/linalg/

אפשר גם להריץ ישירות את קובץ הexe, וזה אפילו מומלץ לנסות אותו לפני המשחק עם הפונקציות בעצמכם, על מנת לוודא שהספריות הנלוות באמת הותקנו כמו שצריך.

שימוש מומלץ:

קובץ הexe מאגד בתוכו ביצוע קליברציה של הרובוט ביחס ללוח, שמירה של אותה קליברציה, ,טעינה של קליברציות קודמות וציור על הלוח מקובץ.

קובץ הexe מיועד לשימוש במקרה שאתם החלטתם שאתם רוצים ליצור ציורים יותר מורכבים מאשר אלו שתוכנה שלנו מאפשרת, או שהקליברציה דרך התוכנה הכללית ספציפית מידי למקרה שלכם.

לעומת זאת באמת עדיף לעבוד עם קבצי הcpp ישירות אם אתם מנסים ליצור קליברציה שונה משלכם, רוצים לשנות את דרך הציור, או לבצע קריאה של פורמט חדש של קבצים על מנת לצייר על סמך הקליברציה אבל ציורים של נתמכים על ידי פורמט הקבצים הקיים.

בנוסף, אפשר להשתמש בקבציה הcpp ישירות גם כאשר הפרוייקט אינו בהכרח קשור ישירות אל הקליברציה הזאת והציורים בפרט, שהרי הם גם מספקים מעטפת אל הפורמטים המשמשים את הרובוט, שאינם נוחים במיוחד לעומת פורמט הfloat3 שהוא בשימוש נרחב הרבה יותר וביצוע מניפולציה מתמטית עליו נוחה בהרבה.

יכולות ותכונות קבצי הcpp,h:

קיימים לנו 4 קבצים:  
lineMath.h

lineMath.cpp

Roboard.h

Roboard.cpp

**lineMath:**

זוג הקבצים הללו מאגד בתוכו את האספקט המתמטי של חישוב המישור של הלוח, ותרגום מיקומים בדו מימד על הלוח (הנמצאים בקבצי הציור שלנו) אל מיקומים בתלת מימד על הלוח האמיתי.

יש אפשרות לבצע שינוי וקריאה של הנורמל שמתעדכן בupdateBasis וחשוב על מנת להחליט על כיוון ההתקדמות שלנו במהלך קליברציה.

בסוף קליברציה, נקראת הפונקציה getNewBasis שמחשבת מתוך 3 נקודות הנמצאות על הלוח את המישור של הלוח, ושומרת אותו אצלה. (וגם מחזירה אותו) על מנת שכאשר נקרא בהמשך לפונקציות תרגום מדו מימד אל תלת מימד נוכל ללבצע את החישוב ביחס לבסיס האחרון שנגענו בו.

יש אפשרות להשתמש בupdateBasis כדי לקבוע את הבסיס ביחס אליו יתבצעו החישובים בעצמכם. הפונקציה הזו נקראת באופן רגיל כאשר אנחנו מבצעים טעינה של בסיס מהזיכרון.

נותרו הפונקציות getLine, getCircArc, getQuadBezierCurve, getCubicBezierCurve כאשר כל הפונקציות הללו מתרגמות מיקומים דו מימדיים אל מיקומים תלת מימדיים ביחס לבסיס של הלוח.

כל אחת מהן מחזירה את סדרת הנקודות בה הרובוט צריך לעבור על מנת לצייר על הלוח האמיתי את הציור הרצוי.

בפונקציה getLine יש את הפרמטר drawing שמיועד לכך שאם תרצו לזוז בין שתי נקודות מבלי לצייר על הלוח, תוכלו לשלוח false והרובוט יזוז על קו שהוא פשוט קצת רחוק יותר מהלוח, כך שהוא לא יצייר עליו במהלך ההליכה על הקו הזה.

**Roboard:**

כאן יש לנו כמה קטגוריות של פונקציות.

יש לנו את פונקציות התרגום של float3 והפורמט שבו הרובוט משתמש. רק חשוב לזכור שאם לא תקראו עם ערכים אל הזווית סיבוב של הזרוע והחוזק של האצבעות זה יתמלא בערכים דיפולטיים.

יש את זוג הפונקציות initRobotAPI, disconnectFromRobot שמתחברות ומתנתקות מהרובוט. הפונקציה של ההתחברות לרובוט גם מאתחלת את כל הפונקציות מהספריות שאנחנו נרצה להשתמש בהן.

יש פונקציות שמחזירות מידע על הרובוט או שולחות מידע לרובוט כבר בצורה של float3. (GetMyForce, MoveToPoint, GetMyCommand, GetMyPosition, GetMyAngularForce)

יש את הפונקציות ששומרות וטוענות את הבסיס של הלוח, שאותנו משמשות לקליברציה, אבל בעצם הן פשוט שומרות וטוענות ווקטור של ארבע float3. (saveBasis, loadBasis )

mainLoopForDrawLine זוהי הפונקציה המרכזית של הציור. היא מקבלת סדרה של נקודות (לא בהכרח קו ישר) ועוברת ביניהן בקצב הנקבע על יד הפרמטרים שלה.

ואחרונה drawFile זוהי הפונקציה שקוראת קובץ שכתוב בפורמט שיתואר בהמשך ומתרגמת זאת לפקודות על הרובוט. כדי שהיא תעבוד כמו שצריך, יש צורך קודם לקבוע בסיס לlineMath כדי שכל החישובים מדו מימד אל תלת מימד יתבצעו כמו שצריך.

פורמט קבצי הציור:

כל שורה מתארת "משיכת מכחול" של הרובוט.

כל פרמטר של שורה מופרד בנקודותיים.

הראשון בשורה מתאר את סוג השורה.

L – קו, C – מעגל (או קשת על מעגל), B – עקומת בזייה קובית, Q – עקומת בזייה ריבועית.

קו:

קו מקבל x התחלה, y התחלה, x סיום, y סיום והאם לצייר את הקו הזה או לא. מה שנותן לנו את האופציה לצייר קווים בלתי נראים פשוט על מנת להזיז את הזרוע ממקום למקום מבלי לצייר בדרך.

לדוגמא:

L:-0.05:0.23:-0.05:0.23:F:

מעגל:

מעגל מקבל x מרכז, y מרכז, רדיוס, ברדיאנים נקודת התחלה על המעגל וכמות רדיאנים לצייר.

לדוגמא:

C:-0.04:0.23:0.01:0:3.1416:

עקומת בזייה ריבועית:

העקומת בזייה הריבועית מקבלת 3 נקודות בעזרתן היא מחשבת את המסלול עליה לצייר.

עקומת בזייה קובית:

העקומה בזייה הקובית מקבלת 4 נקודות בעזרתן היא מחשבת את המסלול עליה לצייר.

לדוגמא:

B:-0.05:0.26:-0.05:0.36:0:0.33:-0.05:0.36: