Project 1 Report

Joong Hyuk An

Mario Farren

Victor Sandberg

1. Description:  
This project was about learning to trace geometrical figures with the robot and create elegant and neat code.  
  
2. Difficulties:  
The biggest difficulties were found in Milestone 1 where the square had to be precisely tailored in the code. The angles for turning the wheels in exactly 90 degrees using the method Motor.Rotate () had to be found by trial and error.   
Introducing the diameter into the project was really difficult at first too but eventually got the hang of it and made the robot make the exact polygons we wanted.

3. Most Interesting:  
The synchronization of the two engines when turning is something very interesting and amazing to watch. It is very interesting also how the computer mounted on the robot can calculate when to begin breaking in order to stop at the precise degrees that had been programmed previously in Eclipse.

4. Approximate Error errors :   
The robot is very precise when travelling in a straight line. It travels the 36 inches when doing the square and no error can be perceived by the naked eye. When tracing the square without the differential pilot, the robot began to deviate from the square path with every turn. Once the differential pilot was introduced, no errors could be perceived in the turns.   
  
5. Sources of Error:  
The performance of the engines is different which contributes to crooked starts or uneven accelerations. Other errors are introduced with imprecise measurements of the robot’s dimensions.

6. The source code:

**public** **class** Legos {

**public** **static** **void** main(String[] args) {

DifferentialPilot pilot = **new** DifferentialPilot(5.6, 11.9, Motor.*C*, Motor.*A*, **false**); // parameters in inches

pilot.setRotateSpeed(150);

pilot.setTravelSpeed(25);

pilot.setAcceleration(1000);

*go*(pilot);

}

/\*\* Makes the robot do two polygons and two circles \*\*/

**public** **static** **void** go(DifferentialPilot pilot) {

*polygon*(pilot, 61, 3, **true**);

*polygon*(pilot, 61, 5, **false**);

*circle*(pilot, 180, 91.48, **true**);

*circle*(pilot, 90, 30.48, **false**);

}

/\*\* Makes the robot trace a polygon

**@param** Diffentail pilot

**@param** the length of the sides of the polygon

**@param** the number of sides in the polygon

**@param** the direction of the robot, if true the robot turns left, if false the robot turns right \*\*/

**public** **static** **void** polygon(DifferentialPilot pilot, **int** sidesLength, **int** numberOfSides, **boolean** direction) {

Button.*waitForAnyPress*();

**int** i = 1;

**if** (direction) {

i = -1;

}

**for** (**int** k = 0; k < numberOfSides; k++) {

pilot.travel(sidesLength);

pilot.rotate(i\*(180-((180\*(numberOfSides-2))/numberOfSides)));

}

}

/\*\* Makes the robot trace a circle

**@param** Diffentail pilot

**@param** the degree of witch the NXT robot will turn

**@param** the radius that the NXT robot moves along

**@param** the direction of the robot, if true the robot turns left, if false the robot turns right \*\*/

**public** **static** **void** circle(DifferentialPilot pilot, **double** degree, **double** radius, **boolean** direction) {

Button.*waitForAnyPress*();

**int** j = 1;

**if** (direction) {

j = -1;

}

pilot.arc(j\*radius, j\*degree);

}

**}**