Marc Ilsø Poulsen, Peter Bartram Hansen og Bjarne Antonsen 9/12-2010
dLego: Projekt Session 1 Rapport
Varighed: 4½ time

Formål

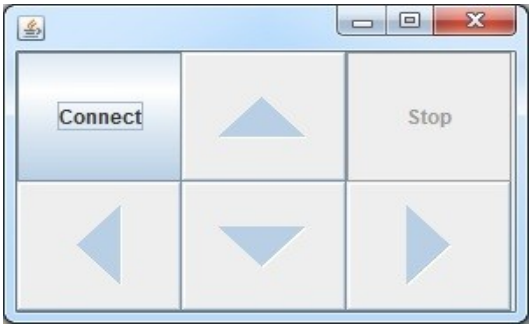
Til denne første lab session i forbindelse med vores projekt ville vi undersøge mulighederne for brugen af Bluetooth til at fjernstyre en enkelt NXT enhed, herunder udarbejdelse af et program, som på enkel vis skulle være i stand til at kunne fjernstyre NXT'en.

Plan

Allerførst ville vi bygge en så simpel robot som mulig, så vi ikke kom til at bruge for lang tid på denne del af projektet på dette tidlige stadie i processen. Vi ville først udforme en simpel brugerflade i form af et GUI, som skulle bruges til at fjernstyre NXT-enheden fra en PC. Dernæst var planen at udvide GUI'et, sådan at programmet til fjernstyring kom til at anvende Bluetooth og derved ville kunne kommunikere med NXT-enheden.
Som udgangspunkt skulle applikationen til fjernstyring køre på en PC, men senere kunne denne eventuelt porteres til en smartphone, sådan at GUI'et blev Android-baseret. Vi vurderede, at en mobil enhed ville være et godt valg til at fjernstyre robotten med.

Resultater

Da formålet med denne første session blot var at give robotten simple fjernstyringskommandoer var udformningen af denne ikke så vigtig på nuværende tidspunkt, så vi besluttede os for at anvende vores tidligere gængse robot, som var LEGO Mindstorms 9797 robotten.
Vi besluttede os for at implementere GUI'et i Java Swing frameworket, da dette var mest oplagt, når vi i forvejen anvendte Lejos API'en. Vores GUI i denne første iteration kom til at se således ud:



Idéen med denne første GUI var at kunne oprette en Bluetooth forbindelse til robotten, for derefter at gøre det muligt at navigere NXT-enheden frem, tilbage, at lade den dreje og at stoppe den.
Ved at undersøge bl.a. Lejos API'en fandt vi frem til, at vi kunne anvende klassen lejos.pc.comm.NXTConnector til at oprette en Bluetooth-forbindelse til en given NXT-enhed. Ved hjælp af metoderne connectTo() og getDataOut() var vi i stand til først at forbinde til NXT-enheden fra vores GUI og dernæst at få returneret en DataOutputStream, som der kunne anvendes til at sende kommandoer til NXT-enheden.
Vores GUI indeholdte følgende kode, som håndterede MouseEvents i vores GUI:

(LegoBluetoothGUI.java)

```
	public void mouseClicked(MouseEvent e) {

		try {
			if(dir.equals("stop")) {
				dos.writeInt(0);
				dos.flush();
			}
			if(dir.equals("forward")) {
				dos.writeInt(1);
				dos.flush();
			}
			if(dir.equals("backward")) {
				dos.writeInt(2);
				dos.flush();
			}
			if(dir.equals("connect")) {
				conn.connectTo(deviceName);
				dos = conn.getDataOut();
				enableButtons();
			}
		catch(IOException ex) {
			ex.printStackTrace();
		}

	public void mousePressed(MouseEvent arg0) {

		try {
			if(dir.equals("left")) {
				dos.writeInt(3);
				dos.flush();
			}
			if(dir.equals("right")) {
				dos.writeInt(4);
				dos.flush();
			}
			}
		catch(IOException ex){}
	}
```

Ovenstående kode forklarer i høj grad sig selv og sørger blot for at håndtere MouseEvents og sende kommandoer til NXT-enheden ved at skrive til en DataOutputStream.

På NXT-siden implementerede vi en klasse RCRobot, hvis ansvar blot var at læse de indadgående kommandoer fra PC'en og ved hjælp af TachoPilot klassen udføre navigation. Følgende kode viser de essentielle dele af dette program:

(RCRobot.java)

```
	private final static int EOS = -1;
	private final static int STOP = 0;
	private final static int FORWARD = 1;
	private final static int BACKWARD = 2;
	private final static int LEFT = 3;
	private final static int RIGHT = 4;

	public void run() {

		...

		LCD.drawString("Waiting",0,0);
		NXTConnection conn = Bluetooth.waitForConnection();
		LCD.drawString("Connected",0,0);
		DataInputStream is = conn.openDataInputStream();

		LCD.drawString("Data: ", 0, 2);

		while(true) {

			try {

				int command = is.readInt();

				LCD.drawInt(command, 6, 2);

				if (command == EOS) System.exit(1);

				switch (command) {
				case STOP:
					pilot.stop();
					break;
				case QUIT:
					System.exit(0);
					break;
				case FORWARD:
					pilot.forward();
					break;
				case BACKWARD:
					pilot.backward();
					break;
				case LEFT:
```

```
                        pilot.rotate(90, true);
                        break;
                case RIGHT:
                        pilot.rotate(-90, true);
                        break;
                }
        ...
        }
```

Programmet venter blot på, at der bliver oprettet en forbindelse fra en klient, hvorefter et control-loop sørger for at læse og udføre de navigeringskommandoer, som der bliver modtaget fra klienten. Vi valgte blot at hardcode, at robotten skulle dreje 90 grader, når der blev sendt en integer af værdi 3 eller 4 (LEFT eller RIGHT). Dette var ikke optimalt med hensyn til navigation af robotten, da det jo ville give problemer i forhold til at bevæge robotten derhen, hvor brugeren ønskede det, men vi løb tør for tid til denne første session og ville blot konstatere, at vores implementation kunne anvendes til det erklærede formål.

Konklusion

Efter denne første lab session i forbindelse med vores projekt har vi et bedre overblik over mulighederne ved at anvende Bluetooth til fjernstyring af en NXT-enhed. Vi har klart nogle mangler i forhold til at dreje robotten, da det kun er muligt at dreje 90 grader på nuværende tidspunkt. Det er vores plan, at vi skal have ændret på dette til næste session, så det bliver muligt at dreje et vilkårligt antal grader. Vi nåede ikke særlig langt med den Android-baserede applikation til denne lab session, men hvis vi får tid, så vil vi prøve at kigge nærmere på dette til en senere lab session.