SINTEF IKT

Postadresse: 7465 Trondheim
Besøksadresse: O S Bragstads plass 2C
7034 Trondheim
Telefon: 73 59 30 00
Telefaks: 73 59 10 39

Foretaksregisteret: NO 948 007 029 MVA

SINTEF RAPPORT

TITTEL

VOCALS : Beskrivelse av demonstratoren

FORFATTER(E)

Trym Holter

OPPDRAGSGIVER(E)

Norges Forskningsråd / NTNU

RAPPORTNR.	GRADERING	OPPDRAGSGIVERS REF.	
SINTEF A5538	Åpen	Torbjørn Svendsen, IET/NTNU	
GRADER. DENNE SIDE	ISBN	PROSJEKTNR.	ANTALL SIDER OG BILAG
Åpen	978-82-14-04388-4	403356	13
ELEKTRONISK ARKIVKODE		PROSJEKTLEDER (NAVN, SIGN.)	VERIFISERT AV (NAVN, SIGN.)
VOCALS - Beskrivelse av demonstrator.doc		Erik Harborg	Erik Harborg
ARKIVKODE	DATO	GODKJENT AV (NAVN, STILLING, SIGN.)	
	2008-02-26	Truls Gjestland, forskningssjef	

SAMMENDRAG

Rapporten beskriver en demonstrator som er utviklet som et enkelt eksempel på et multimodalt brukergrensesnitt, hvor fokus i stor grad ligger på bruk av tale for interaksjon. Demonstratoren lar brukeren kombinere penn og tale som input-modaliteter.

Demonstratoren er utviklet basert på en distribuert klient/tjener-arkitektur. Klienten kjører på en mobil terminal under Pocket PC 2003, mens tjeneren kjører på en PC med Linux. Applikasjonen som er valgt for formålet er et enkelt system for turistinformasjon i Trondheim. Rapporten beskriver de to sentrale funksjonalitetene som behøves for å realisere applikasjonen, henholdsvis en løsning for håndtering av kart på en mobil terminal og en talegjenkjenner egnet for distribuert prosessering. Programvaren og metodene som er brukt beskrives, sammen med en detaljert veiledning for oppsett og bruk av demonstratoren.

Programvare og dokumentasjonen fra prosjektet ligger i prosjektfolderen på DVDen som følger rapporten.

STIKKORD	NORSK	ENGELSK
GRUPPE 1	Akustikk	Acoustics
GRUPPE 2	Taleteknologi	Speech technology
EGENVALGTE	Multimodal	Multi modal
	Brukergrensesnitt	User interface
	Distribuert talegjenkjenning	Distributed speech recognition

INNHOLDSFORTEGNELSE

1 Innledning

Denne rapporten beskriver demonstratoren som er utviklet i prosjektet VOCALS (Voice Centric User Interfaces for Location Based Services) [1]. Dette er et forskningsprosjekt innen Norges forskningsråds program ”IKT 2010” og er et samarbeid mellom Institutt for elektronikk og telekommunikasjon (IET) ved NTNU og Avdeling for akustikk ved SINTEF IKT.

Demonstratoren er et enkelt eksempel på et multimodalt brukergrensesnitt, hvor fokus i stor grad ligger på bruk av tale for interaksjon (i tråd med prosjektets tittel). Demonstratoren lar brukeren kombinere penn og tale som input-modaliteter. Modellen for dette er enkel, og likner på filosofien i Microsofts Mipad ("Multimodal Interactive Pad") [2], ofte kalt "tap and talk" eller på norsk "peik og preik". Ved å plassere pennen på et punkt i det grafiske brukergrensesnittet vil brukeren bestemme hvilken tilstand applikasjonen skal befinne seg i, samtidig som talegjenkjenneren blir aktivisert.

Demonstratoren er utviklet basert på en distribuert klient/tjener-arkitektur. Den mobile terminalen (klienten) kjører på en HP iPaq 5550 med Pocket PC 2003, mens tjeneren er en PC som kjører Linux. Applikasjonen som er valgt for demonstratoren er et enkelt system for turistinformasjon i Trondheim. Funksjonene som inngår i denne er beskrevet i kapittel 2. De to sentrale funksjonalitetene som behøves for å realisere applikasjonen er henholdsvis en løsning for håndtering av kart på en mobil terminal og en talegjenkjenner egnet for distribuert prosessering. Programvaren og metodene som er brukt i demonstratoren for dette er beskrevet henholdsvis i kapittel 3 og 4. En detaljert veiledning for oppsett og bruk av demonstratoren er gitt i kapittel 5, før oppsummering og konklusjoner følger i kapittel 6.

Utviklingsarbeidet er utført med Trym Holter som ansvarlig. Espen Helle og Terje Mugaas (begge ved Avdeling for anvendt kybernetikk, SINTEF IKT) har vært ansvarlige for implementeringen av talegjenkjenningsklienten og Geir Anker (tidligere ved Avdeling for anvendt matematikk, SINTEF IKT) har vært ansvarlig for implementering av kartløsningen. Ole Hartvigsen (tidligere ved Institutt for datateknikk, NTNU) har utført hovedtyngden av systemintegrasjonen.

2 Funksjonell beskrivelse

Sentralt i applikasjonen er en kartløsning som danner basis for resten av funksjonaliteten. Assosiert med posisjoner i kartet vil det være en samling av såkalte POI ("points of interest"). I dette dokumentet brukes en vid definisjon av POI, idet alle punkter som kan bli markert i kartet betraktes som et POI. Hvert POI tilhører en kategori, og Tabell 1 viser en oversikt over hvilke kategorier som er definert i den nåværende implementeringen av demonstratoren. Som beskrevet i kapittel 3 vil det være relativt enkelt å utvide dette med flere kategorier. I tillegg viser høyre kolonne i denne tabellen oppbyggingen av en enkel grammatikk (i ”Extended Backus-Naur Format” (EBNF)) innen hver kategori, som brukes av talegjenkjenneren. Alle grammatikkene som er i bruk i demonstratoren er inntil videre svært enkle. Det vil imidlertid være mulig å utvide disse slik at færre begrensninger legges på brukerne.

Tabell 1: POI – "point-of-interest".

Nr.	Kategori	Grammatikk (EBNF)
1	Restaurant	$restaurant = (restaurant | restauranter);
2	Hotell	$hotell = (hotell | hoteller);
3	Severdighet	$severdighet = (severdighet | severdigheter);

Ved å plassere pennen på et punkt i kartet bestemmes en posisjon. Hvordan denne posisjonen skal tolkes avhenger av hva brukeren sier. Tabell 2 og Tabell 3 viser en oversikt over de mulige spørsmål og kommandoer brukeren kan komme med. I denne beskrivelsen brukes betegnelsen "aktive POI" om POI som har blitt identifisert som en respons på forespørsler fra brukeren.

Tabell 2: Funksjonalitet - zoom og panorering.

Nr.	Penn	Tale	Respons
1	På punkt i kartet	"Zoom inn"	• Zoom inn ett nivå, sentrert rundt posisjonen • Vis aktive POI som faller innen utsnittet
2	På punkt i kartet	"Zoom mye inn" "Zoom inn mye"	• Som i punkt 1, men med større endring i målestokk
3	På punkt i kartet	"Zoom helt inn" "Zoom inn helt"	• Zoom inn til det mest detaljerte nivået, sentrert rundt posisjonen • Vis aktive POI som faller innen utsnittet
4	På punkt i kartet	"Zoom ut"	• Zoom ut ett nivå, sentrert rundt posisjonen • Vis aktive POI som faller innen utsnittet
5	På punkt i kartet	"Zoom mye ut" "Zoom ut mye"	• Som i punkt 4, men med større endring i målestokk
6	På punkt i kartet	"Zoom helt ut" "Zoom ut helt"	• Zoom ut til det minst detaljerte nivået (posisjonen neglisjeres) • Vis aktive POI som faller innen utsnittet
7	På punkt i kartet	"Flytt hit"	• Sentrer kartet rundt posisjonen (behold målestokk) • Vis aktive POI som faller innen utsnittet

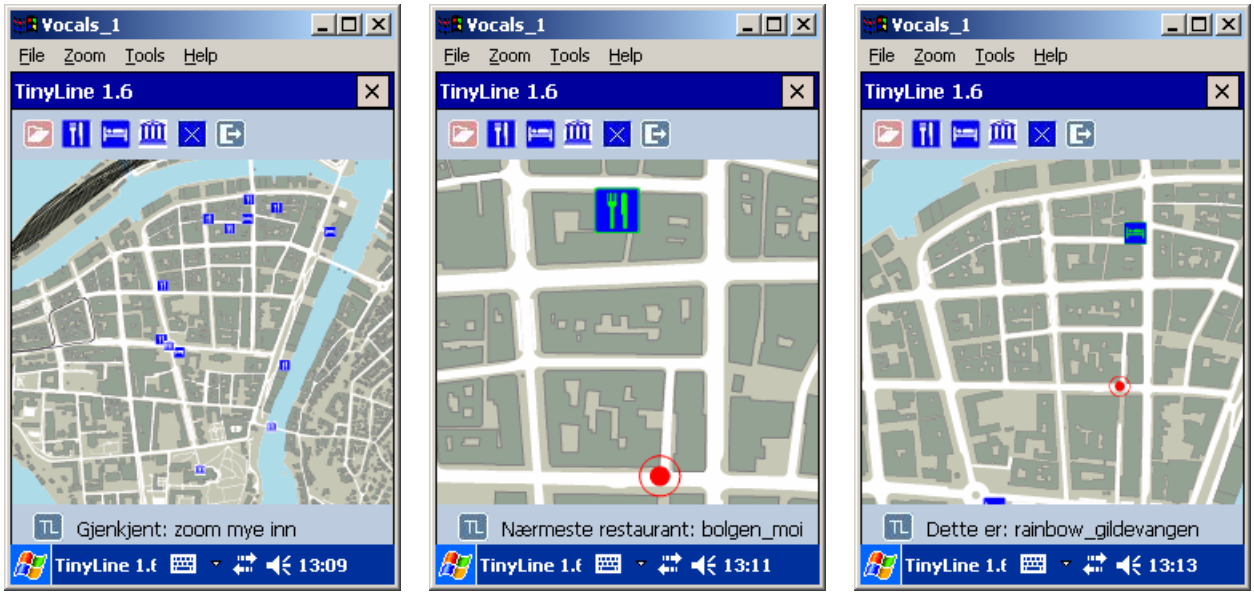
Tabell 3: Funksjonalitet - spørringer.

Nr.	Penn	Tale	Respons
1	På punkt i kartet	"Hvor er nærmeste $POI[1]"	• Deaktiver alle POI • Tilordne eget POI til posisjonen i kartet[2] og sett aktivt • Sett nærmeste relevante POI aktivt • Zoom til begge aktive POI faller innen utsnittet • Vis aktive POI som faller innen utsnittet • Gi respons vha. tekstfeltet
2	På punkt i kartet	"Vis $POI "	• Deaktiver alle POI • Sett alle relevante POI aktive • Vis aktive POI som faller innen utsnittet
3	På POI i kartet	”Hva er dette”	• Marker det aktuelle POI med avvikende farger • Gi respons vha. tekstfeltet
4	På POI-felt utenfor kartet	Ingen tale	• Deaktiver alle POI • Sett alle relevante POI aktive • Vis aktive POI som faller innen utsnittet

I Figur 1 vises eksempler av skjermbilder fra demonstratoren. Figur 1a viser tilstanden etter at brukeren har bedt om en rask reduksjon av målestokk. I slike tilfeller der applikasjonen ikke er bedt om spesifikke opplysninger om et POI, viser tekstfeltet under kartet isteden responsen fra talegjenkjenneren. Denne funksjonen er først og fremst ment som en hjelp i forbindelse med utviklingen av systemet. I Figur 1b ser vi et tilfelle der brukeren har markert en posisjon (representert ved det røde symbolet) og spurt om nærmeste restaurant. Denne er så markert ved det relevante symbolet med avvikende fargesetting samtidig som tekstfeltet oppgir navnet på stedet. I Figur 1c ser vi igjen at brukeren har oppgitt en posisjon (i et tidligere steg av

[1] $POI = $restaurant | $hotell | $severdighet (i EBNF-notasjon).
[2] Egen posisjon betraktes her som et POI (for enkel notasjon).

interaksjonen) og har bedt om nærmere opplysninger om et POI ved å peke på det og spurt hva POI'et er. På samme måte som i det forrige tilfellet fargemarkeres POI'et mens tekstfeltet gir responsen.



(a) (b) (c)

Figur 1: Skjermbilder fra VOCALS-demonstratoren.

Ikonene som vises fra venstre mot høyre øverst i skjermbildene i Figur 1 brukes henholdsvis til å åpne et kart (se kapittel 5) samt vise restauranter, hoteller og severdigheter (se punkt 4 i Tabell 3). Det femte symbolet i rekken vil vekselvis bidra til å fjerne alle aktive POI'er og å vise alle mulige POI'er innenfor kartutsnittet. Symbolet til høyre i rekken brukes til å avslutte applikasjonen.

3 Programvare for håndtering av kartinformasjon

3.1 Verktøy

SINTEF har god erfaring med å bruke formatet Scalable Vector Graphics (SVG) [3] i distribuerte kartsystemer. SVG ble derfor valg som utvekslingsformat for kartdata i dette prosjektet. For å vise SVG kart på håndholde terminaler ble TinyLine API'et valgt [4].

SVG er en World Wide Web Consortium (W3C) standard for å beskrive todimensjonal grafikk i XML (både vektor og raster). SVG kan gjøres dynamisk og interaktivt ved å manipulere SVG dokumentets DOM (Document Object Model) ved hjelp av scripting eller et programmerings-språk (C++, Java). SVG er også godt egnet til å overføre kartgrafikk til håndholdte klienter (SVG Basic og SVG Tiny [5]). SVG-standarden er utviklet sammen med de tunge aktører innenfor grafisk publisering, mobilmarkedet og IT-industrien.

TinyLine er et SVG Tiny utviklingsverktøy for Java Micro Edition (J2ME) [6] klienter. TinyLine består av en viewer og en SDK som gir mulighet for å lage egne applikasjoner. I dette prosjektet er det tatt utgangspunkt i vieweren som er utvidet med nødvendig funksjonalitet. Det ble også gjort tester for å finne en Java Virtual Machine (JVM) som fungerte godt med TinyLine, og valget falt i denne omgang på Jeode JVM.

3.2 Kartdata

Kartdataene er FKB (felles kart base) data levert av Trondheim kommune, plan- og bygningsenheten (PBE). Datasettet har separate shapefiler for Vegsituasjon, Jernbane, Bygninger, Elv og Kyst. Datasettet ble klippet slik at utsnittet dekker Trondheim sentrum. Shapefilene ble konvertert til SVG ved hjelp av SINTEF sin egen programvare.

De aktuelle POI'ene ble manuelt lagt inn i SVG-fila, som utdraget i Figur 2 viser. Flere punkter kan legges inn etter det samme mønsteret.

```
<g id="pois">
    <g id="restaurant">
        <g id="internasjonal">
            <g>
                <desc>Bølgen &amp;amp; Moi</desc>
                <rect id="bolgen_moi" fill="#ffffff" x="73" y="-777" width="34" height="34" rx="1"/>
                <use x="73" y="-777" width="34" height="34" xlink:href="#rest"/>
            </g>
        </g>
        <g id="pizza"/>
        <g id="gresk"/>
    </g>
    <g id="hotell"/>
    <g id="severdighet"/>
</g>
```

Figur 2: Utdrag av SVG-fila med innlagte POI.

3.3 Kort beskrivelse av kildekode

Det er tatt utgangspunkt i kildekoden til TinyLine viewer for å implementere den nødvendige funksjonaliteten. De to klassene som er forandret og utvidet i forholdet til dette er MapCanvas og MouseHandler. I tillegg kommer et fåtall egenutviklede klasser som er nødvendige for integreringen med talegjenkjenneren. De viktigste klassene er beskrevet kort under:

- MapCanvas tar utgangspunkt i den opprinnelige klassen PPSVGCanvas. Denne er en utvidelse (extension) av AWT komponenten Canvas og utgjør kartdisplayet i applikasjonen. Den inneholder metodene for manipulering av kartutsnittet, så som zooming og panorering.
- MouseHandler er klassen for håndtering av mus- eller pennbevegelser. I tillegg til metoder for å tolke disse bevegelsene, inneholder klassen kallet til talegjenkjenneren.
- SpeechGetter utgjør grenseflaten mellom Java-applikasjonen og talegjenkjenneren, med metoden getSpeech.
- GrammarInterpreter står for en enkel tolkning av responsen fra talegjenkjenneren ved å sammenlikne den returnerte tekststrengen med de mulighetene som er definert som gyldige alternativer.

3.4 Oversikt over filer og kompilering

Programvaren for kartapplikasjonen ligger i tre kataloger under 'Arkiv\Demonstrator endelig versjon' i prosjektfolderen:

- src – inneholder kildekoden for applikasjonen
- svg – inneholder SVG-kart for Trondheim (og Oslo)
- TinyLine – inneholder TinyLine API'et samt den resulterende eksekverbare 'tinyline.jar'

Kompilering av koden bør skje med JDK 1.1.8 for å sikre god kompatibilitet med Jeode JVM. Katalogen src inneholder en batch-fil (compile.bat) for kompilering, som virker forutsatt at man har valgt den gitte plasseringen av JDK 1.1.8 under installering.

4 Programvare for distribuert talegjenkjenning

4.1 Bakgrunn og virkemåte

Programvaren ANSR for distribuert talegjenkjenning ble opprinnelig kjøpt fra Sprex, Inc [9]. Det viste seg dessverre relativt raskt at produktet var mindre modent enn hva som var forutsatt, og tett kommunikasjon over lengre tid var ikke tilstrekkelig til at det nådde en kvalitet som gjorde det godt egnet for videreutvikling til dette prosjektets formål. Problemene var i første rekke knyttet til klientprogramvaren. Sprex er nå tilsynelatende ikke lenger aktivt som selskap.

I den foreliggende demonstratoren brukes tjeneren fra ANSR uforandret fra leveransene som ble mottatt. Denne er beskrevet i mer detalj i kapittel 4.3. En egenutviklet klient sørger for streaming av tale (16 bits lineær oppløsning og 16 kHz punktprøvingsfrekvens) fra en iPaq til denne tjeneren og mottar resultatet fra talegjenkjenningen som en tekststreng. Klienten er beskrevet i kapittel 4.2.

4.2 Talegjenkjenneren: klienten

4.2.1 Verktøy

Klienten er skrevet i C++ og er forsøkt implementert på en måte som skal tillate flytting mellom ulike plattformer på en enkel måte. Utviklingsmiljøet som er benyttet er Visual Studio (VS) 2005 (versjon 8.0) [7]. I tillegg er Windows Mobile SDK 5.0 for Pocket PC [8] installert.

4.2.2 Kort beskrivelse av kildekode

De viktigste filene i implementeringen er ’WaveThread.cpp’, ’T_Socket.cpp’ og ’streaming lyd.cpp’. De to første inneholder metodene for henholdsvis håndtering av lydbuffer og kommunikasjon over sockets i en TCP/IP-forbindelse. Grenseflaten til resten av demonstratoren utgjøres av metodene i ’streaming lyd.cpp’, som gjør kall til de mer lavnivå metodene i ’WaveThread.cpp’ og ’T_Socket.cpp’.

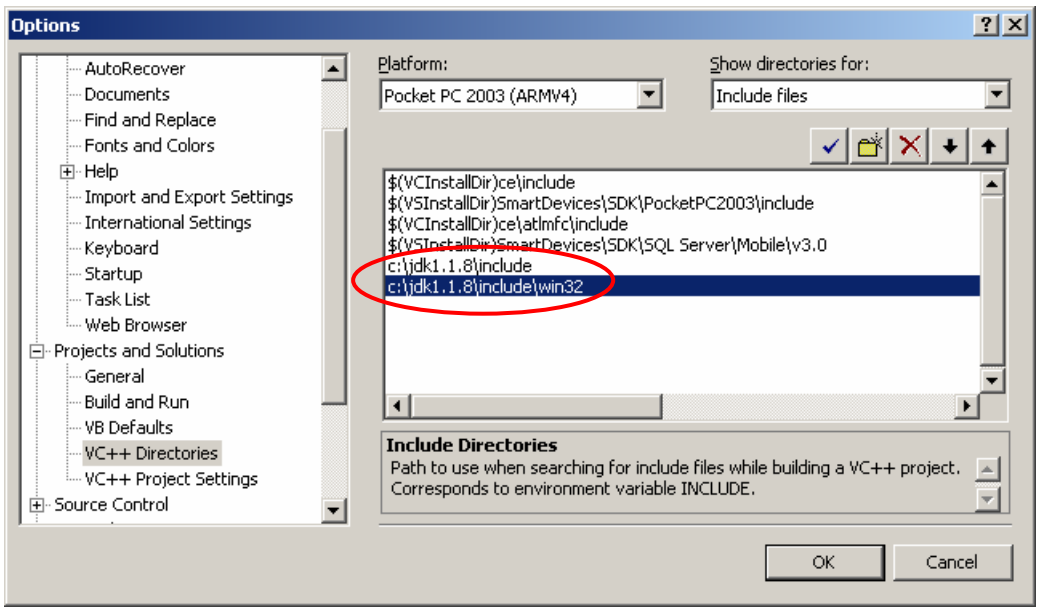
De viktigste metodene i ’streaming lyd.cpp’ er kort beskrevet under:

- Java_com_tinyline_app_SpeechGetter_getspeech er nativemetoden som blir kalt fra Javaprogrammet, ved bruk av Java Native Interface (JNI). Denne metoden inneholder parametrene for tjenerens IP-adresse (hardkodet til 192.168.14.3), hvilken port kommunikasjonen skal foregå på (hardkodet til port 4130), og lengden av audiosignalet som skal streames (hardkodet til 3 sekunder).
- initializeASR setter tjenerens IP-adresse til den ønskede verdien.
- runASR er metoden som starter fangsten av audiosignalet, streamer det til tjeneren og mottar resultatet i form av en tekststreng.
- waveInEventCB er metoden hvor audiobufferet sendes over socketforbindelsen. Som nevnt over fanges det i denne versjonen et audiosignal av fast lengde som sendes til tjeneren. Dersom en ønsker å implementere en taledeteksjonsalgoritme som et alternativ til denne prosedyren, ville dette være det naturlige stedet å inkorporere dette.

4.2.3 Oversikt over filer og kompilering

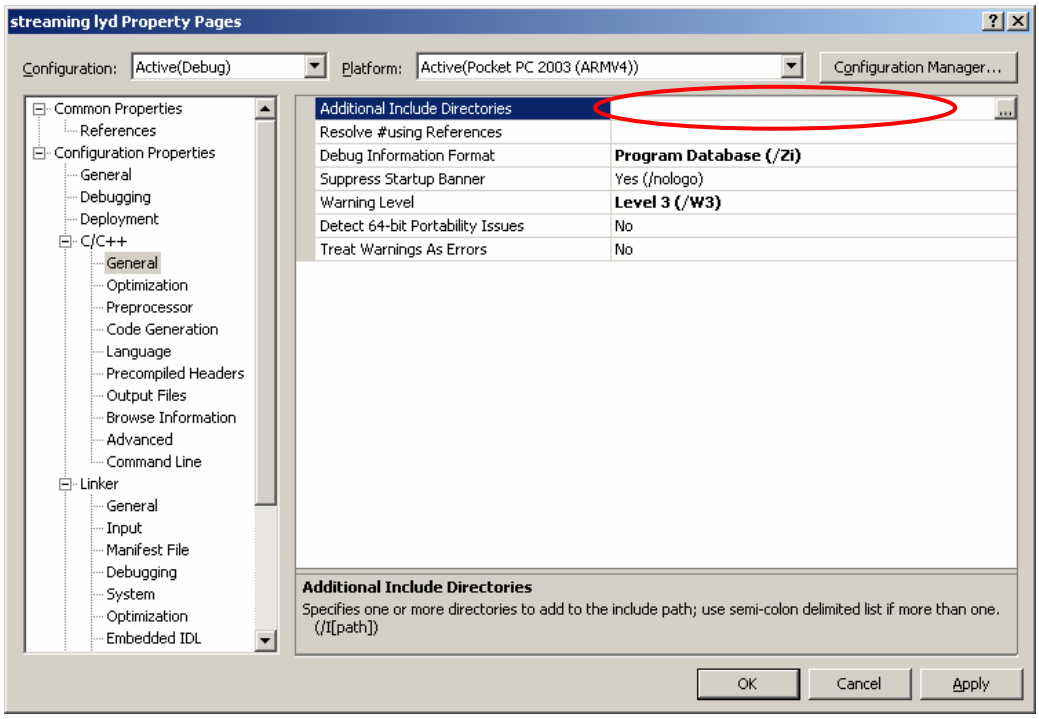
Programvaren for klienten ligger i katalogen ’Arkiv\Demonstrator endelig versjon\talegjenkjenner dll’ under prosjektfolderen. Som navnet antyder så kompileres programvaren til en ”dynamic link library” (dll) slik at det kan kalles fra Java vha. JNI. I Visual Studio åpnes prosjektet ved å åpne fila ’streaming lyd.sln’. For å unngå feil ved lenkingen må prosjektet inkludere stien til ’jni.h’ fra JDK 1.1.8. Dette kan gjøres ved å sette stien som en egenskap ved utviklingsmiljøet (’Tools’ –

'Options' – 'Projects and Solutions' – 'VC++ Directories' og velge 'Show directories for: Include files' og 'Platform: Pocket PC 2003 (ARMV4)') som indikert i Figur 3.



Figur 3: Oppsett som lar VS finne jni.h.

Alternativt kan man sette dette som en egenskap ved prosjektet, ved å høyreklikke toppnivået 'streaming lyd' i VS, velge 'Properties – C/C++ - General' og legge stiene inn under 'Additional Include Directories'. Dette er indikert av Figur 4, dog uten at stiene er lagt inn siden alternativet beskrevet over er det foretrukne i denne sammenhengen.



Figur 4: Alternativt oppsett for å la VS finne jni.h.

For å bygge dll'en på nytt kan man høyreklikke toppnivået 'streaming lyd' i VS og velge 'Rebuild'. Resultatet av dette blir en ny 'streaming lyd.dll' i folderen 'talegjenkjenner dll\streaming lyd\streaming lyd\Pocket PC 2003 (ARMV4)\Debug'.

4.3 Talegjenkjenneren: tjeneren

4.3.1 Bakgrunn

ANSRs tjenerprogramvare for talegjenkjenning heter ’sprecd’. Programvaren er bygd rundt HAPI [10] og aksepterer derfor akustiske modeller og språkmodeller som miljøet er godt kjent med fra arbeid med HAPI og HTK [11]. Programvaren foreligger som eksekverbare kode, og ligger som ’sprecd-1.4’ i ’Arkiv\Demonstrator endelig versjon\Linuxtjeneren\resources’ under prosjektfolderen. Merk at den siste versjonen vi har mottatt av programvaren er versjon 1.8. Denne inneholder imidlertid flere alvorlige feil som gjør at den ikke lar seg kjøre med den utviklede klienten. Det kan også bemerkes at Sprex selv kjører med versjon 1.4 i demonstratorene de har tilgjengelige på sine websider.

4.3.2 Akustiske modeller

De akustiske modellene brukt i prosjektet er basert på skjulte markovmodeller (HMM – hidden Markov models) opprinnelig utviklet i et tidligere prosjekt utført for NRK [12]. Forprosesseringen som er benyttet er den samme i begge disse prosjektene. De ordinterne trifonmodellene fra NRK-prosjektet er trent basert på taleopptak med studiokvalitet, og for å tilpasse disse til bruk med håndholdte terminaler, ble det samlet noe nytt talemateriale for bruk i adapsjon. Totalt ble det samlet talemateriale fra 11 talere, hvorav 5 kvinner og 6 menn. Hver taler har i utgangspunktet ytret 5 repetisjoner av hvert av 42 ord definert i en tidlig kravspesifikasjon for demonstratoren. I tillegg har hver av talerne lest ca. 50 setninger, hvor teksten er hentet fra NRK-materialet. Mer detaljer om spesifikasjonen av og prosedyrene for opptakene er beskrevet i [13] og [14].

Modellene som er benyttet i demonstratoren er en transformert versjon av NRK-modellene. Adapsjonen benytter ”Maximum Likelihood Linear Regression” (MLLR) basert på én global transformasjon. Adapsjonsmaterialet som er benyttet er de 50 setningene med tekst fra NRK-materialet som er lest av hver av de 11 talerne. Adapsjonsmaterialet utgjør altså omlag 550 setninger. Mens MLLR vanligvis benyttes for å tilpasse et sett akustiske modeller til én spesifikk taler, er ideen i denne sammenhengen altså å tilpasse modellene slik at de passer til den talekvaliteten man kan forvente med en mobil terminal.

4.3.3 Grammatikk

Som nevnt tidligere er grammatikkene som benyttes i demonstratoren enkle ordnettverk. Hele grammatikken er gjengitt i Figur 5 i EBNF-notasjon.

```
$inout = inn | ut;
$speed = helt | mye;
$restaurant = restaurant | restauranter;
$hotell = hotell | hoteller;
$severdighet = severdighet | severdigheter;

$poi = $restaurant | $hotell | $severdighet;
$manipulate_view = zoom $inout |
                   zoom $speed $inout |
                   zoom $inout $speed |
                   flytt hit;
$manipulate_poi  = hva er dette |
                   hvor er naermeste $poi |
                   vis $poi;
$cmd             = $manipulate_view | $manipulate_poi;

([sil] $cmd [sil])
```

Figur 5: Grammatikken brukt i demonstratoren.

Denne grammatikken legger naturlig nok relativt strenge begrensninger på brukeren. En naturlig utvidelse av konseptet ville være å inkludere såkalt ”garbage”-modeller for å fange opp fraser som ikke følger denne strenge strukturen. En mer ambisiøs utvidelse ville være å gå i retning av naturlig språk. I tillegg til den økte kompleksiteten i talegjenkjenneren ville dette også stille større krav til enheten som skal tolke resultatene fra gjenkjenneren.

5 Veiledning for oppsett og bruk av demonstratoren

5.1 Maskinvare brukt til demonstratoren

Demonstratoren er utviklet og testet for en iPaq 5550 som klient og en Linux-PC som tjener. iPaq 5550 har en Intel PXA255 prosessor og har 128MB RAM. Linux-PC’en benyttet i dette arbeidet er en Compaq Evo N600c med 256 MB RAM. Prosessoren i denne er en Intel Pentium III-M som klokkes ved 1,2 GHz. Maskinen kjører Red Hat Linux release 9 med kjerne 2.4.20-8. Et PCMCIA innstikkskort benyttes for å sette opp WLAN.

5.2 Forberedelser og installasjon av programvare

Programvaren til tjeneren forutsetter at konfigureringsfila som skal lastes inn finnes i en fastsatt katalog, /usr/share/sprex/ansr/resources. Det er her antatt at resten av de nødvendige ressursene også finnes her, enten som filer eller som symbolske lenker. Alternativet til dette er å endre konfigureringsfila og evt. scriptet som starter tjeneren, slik at de reflekterer de faktiske plasseringene av ressursene. Tabell 4 viser en oversikt over de nødvendige filene, der plasseringen er gitt med utgangspunkt i /usr/share/sprex/ansr/resources. Disse finnes i prosjektforderen under katalogen Linuxtjener/resources/.

Tabell 4: Filer som behøves for talegjenkjenningstjeneren.

Filnavn	Kommentar
sprecd-1.4[3]	Tjenerprogrammet
settoppWLAN.csh	Script for å starte WLAN (antar at interface eth1 benyttes for WLAN).
settoppANSR.csh	Script for å starte sprecd
em.cfg -> VOCALS/vocals.cfg[4]	Lenke til konfigureringsfil (navn og plassering kan ikke endres)
VOCALS/vocals.cfg	Konfigureringsfil
VOCALS/vocals.dct	Uttaleleksikon
VOCALS/vocals.slf	Grammatikk i ”standard lattice format”
VOCALS/vocals.lis	Modelliste
VOCALS/HMM/MMF_extended.vad.global	HMM’er i HTKs MMF-format

For å installere programvaren for klienten må det først opprettes en forbindelse mellom iPaq’en og en Windows-PC vha. ActiveSync ([15], se også kapittel 5.6.1). Filer kan da flyttes fram og tilbake vha. Windows Explorer. De nødvendige stegene er beskrevet under:

1. Installer Jeode JVM hvis det ikke allerede er gjort
   (denne installeres fra ‘Arkiv\HP iPaq Pocket PC companion CD’, som også ligger under prosjektfolderen: setup.exe -> Enhance Your Experience -> h5550 Software -> Pocket PC Applications).
2. Legg TinyLine-katalogen under roten \ på iPaq’en. Folderen framstår gjerne som ’My Windows Mobile-Based Device’ i Windows Explorer.
3. Legg svg-katalogen under ’My Documents’ på iPaq’en.
4. Endre navn på streaming lyd dll.dll til speech.dll. Fila finnes i folderen ’ Arkiv\Demonstrator endelig versjon \\talegjenkjenner dll\streaming lyd\streaming lyd\Pocket PC 2003 (ARMV4)\Debug’.
5. Legg speech.dll under Windows på iPaq’en.

[3] sprecd-1.4 legges i katalogen /usr/local/bin for å kunne kjøre med med scriptet settoppANSR.csh. For andre plasseringer må settoppANSR.csh endres tilsvarende.
[4] Symbolsk lenke settes opp med kommandoen ”ln –s VOCALS/vocals.cfg em.cfg”

Dersom WLAN-forbindelsen mellom iPaq og Linux-maskinen *ikke har vært konfigurert tidligere*, er det nødvendig å sette opp denne. Dette kan gjøres slik (steg 1 på PC, steg 2 – 8 på iPaq):

1. Sett opp WLAN'et på Linuxmaskinen (som root) vha. settoppWLAN.csh (se også kapittel 5.3). Status for WLAN-koblingen kan sjekkes med kommandoene iwconfig/ifconfig
2. Slå på WLAN på iPaq'en ('Start – iPaq Wireless – WLAN').
3. Koble til nettverket 'ambidemo'.
4. Trykk på "Connections"-ikonet (til venstre for volumikonet på verktøylinja øverst).
5. Trykk 'Settings – Advanced – Network Card – Network Adapters' og velg 'My network connects to: Work'.
6. Velg 'iPAQ USB Wireless Adapter' i 'Tap an adapter to modify settings'.
7. Velg 'Use specific IP address' og bruk disse verdiene:
   - IP address: 192.168.14.5 (192.168.x.y er "private" adresser)
   - Subnet mask: 255.255.255.0
   - Default gateway: 192.168.14.1
8. Skru WLAN'et av og på (grønn diode vil blinke på iPaq når WLAN er aktivt), og forhåpentligvis så virker det som forventet. Dette sjekkes gjerne med 'ping' mot tjeneren med fast IP 192.168.14.3 (se kapittel 5.5 for omtale av PocketPing).

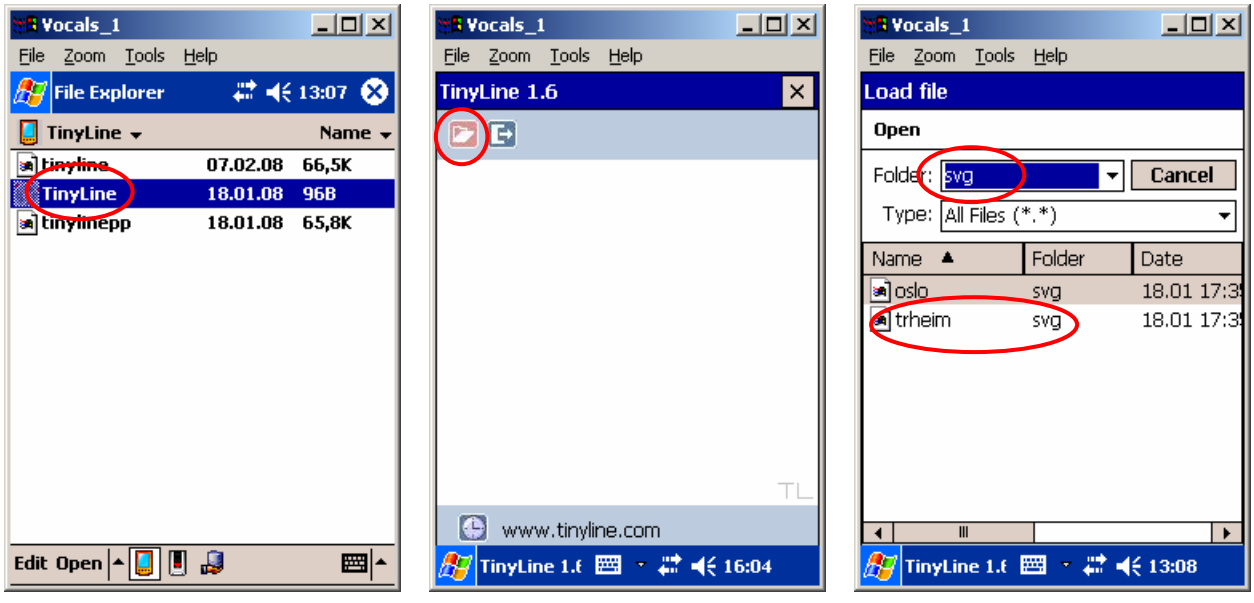
5.3 Start av tjeneren

Tjeneren startes enklest ved å logge på Linuxmaskinen, starte en ny terminal og gå til katalogen /usr/share/sprex/ansr/resources. Deretter følges denne prosedyren:

1. Sett opp WLAN'et på Linuxmaskinen med ./settoppWLAN.csh
2. Sett opp talegjenkjenneren med ./settoppANSR.csh

5.4 Start av klienten

For å starte klienten må TinyLine-applikasjonen først startes. Til dette trengs ekvivalenten til 'Windows Explorer', som under Pocket PC 2003 heter 'File Explorer'. Denne startes ved å velge 'Start – Programs – File Explorer'. Naviger deretter til 'TinyLine.lnk' under 'My Device\TinyLine', se Figur 6a. Ved å klikke på denne starter applikasjonen og lar brukeren velge hvilken svg-fil som skal lastes ved å trykke på ikonet i øvre venstre hjørne, se Figur 6b. Velg så 'Folder: svg' fra nedtrekksmenyen (se Figur 6c) og velg 'trheim.svg'. Etter en kort stund skal applikasjonen være klar til bruk.



(a) (b) (c)

Figur 6: Start av Vocals-klienten.

5.5 Nedkobling av tjener og klient.

Tjeneren stoppes med Ctrl-C. Klienten stoppes ved å trykke symbolet til høyre for rød ring i Figur 6b.

5.6 Installasjon og bruk av praktiske verktøy for Pocket PC

5.6.1 ActiveSync

ActiveSync er en forutsetning for å sette opp kommunikasjonen mellom en iPaq og en windows-PC, for installasjon av programvare og annet. Siste versjon av ActiveSync er tilgjengelig fra Microsofts websider [15]. Alternativt kan ActiveSync installeres fra folderen ’Arkiv\Pocket PC utilities\ActiveSync_v4.5’ under prosjektfolderen.

5.6.2 Remote Display Control

Dette programmet gjør det mulig å vise displayet til en iPaq på skjermen til en windows-PC. Det er også mulig å styre iPaq’en ved å bruke musa på PC’en. Det er et godt egnet verktøy ved demonstrasjoner til en større gruppe. Installasjonsfila ligger under ’ Arkiv\Pocket PC utilities’ under prosjektfolderen, og installasjonen utføres som følger (med iPaq tilkoblet windows-PC med ActiveSync):

1. Kjør RemoteDSP for iPaq 5550.exe på Windows-PC’en.
2. Det er mulig å ignorere advarselen om å avslutte Windowsapplikasjoner hvis ønskelig.
3. Velg folderen som foreslås, 'C:\Program Files\Microsoft ActiveSync\Remote Display Control'
4. Velg 'Install "Microsoft Remote Display Control" using the default application install directory'

For å bruke applikasjonen følges følgende framgangsmåte:

1. Start 'Active Sync Remote Display' på PC’en og velg 'Ignore' (hvis spurt).
2. Start applikasjonen ’cerdisp.exe’ ('Start – Programs') på iPaq’en
3. Velg 'Connect' og velg hostname PPP_PEER forutsatt at koblingen er satt opp med ActiveSync

5.6.3 Pocket Ping

Dette programmet gjør det mulig å kjøre ’ping’ fra iPaq’en. Dette er for eksempel gunstig for å verifisere at den trådløse forbindelsen mellom klient og tjener er operativ. Installasjonsfila ligger under ’ Arkiv\Pocket PC utilities’ under prosjektfolderen, og installasjonen utføres som følger (med iPaq tilkoblet windows-PC med ActiveSync):

1. Kjør PocketPingSetup.exe på Windows-PC’en.
2. Velg 'Install "Agang(CE) Pocket Ping 1.0" using the default application install directory'

For å starte applikasjonen velger man ’Start – Programs - Pocket Ping’ på iPaq’en.

6 Konklusjoner

Dette dokumentet har gitt en kort beskrivelse av demonstratoren som er utviklet i prosjektet VOCALS. Demonstratoren er et enkelt eksempel på et multimodalt brukergrensesnitt, hvor fokus i stor grad ligger på bruk av tale for interaksjon.

Virkemåten til demonstratoren er beskrevet, sammen med detaljer fra implementeringen og oppsettet av både kartløsningen og talegjenkjenningsteknologien som er benyttet. Beskrivelsene har tatt sikte på å være tilstrekkelig detaljerte til at personer som ikke har vært involvert i prosjektet skal kunne kompilere, installere og kjøre demonstratoren.

Referanser

[1] T. Svendsen et.al., "VOCALS - Voice Centric User Interfaces for Location Based Services", i Proceedings of the Norwegian Signal Processing Symposium (NORSIG) 2005, Stavanger, Norway, September, 2005.

[2] X. Huang et.al., "MIPAD: A Multimodal Interaction Prototype", i Proceedings of IEEE ICASSP 2001, Salt Lake City, Utah, May 2001.

[3] http://www.w3.org/Graphics/SVG, sist besøkt februar 2008.

[4] http://www.tinyline.com, sist besøkt februar 2008.

[5] http://www.w3.org/TR/SVGMobile, sist besøkt februar 2008.

[6] http://java.sun.com/javame, sist besøkt februar 2008.

[7] http://msdn2.microsoft.com/en-us/teamsystem/bb188238.aspx, sist besøkt februar 2008.

[8] http://www.microsoft.com/downloads/details.aspx?familyid=83A52AF2-F524-4EC5-9155-717CBE5D25ED&displaylang=en, sist besøkt februar 2008.

[9] http://cassandra.sprex.com/ansr/, sist besøkt februar 2008.

[10] J. Odell et.al., "The HAPI Book version 1.4", Entropic Ltd., 1999.

[11] S. Young et.al., "The HTK Book version 3.4", Cambridge University Engineering Department, 2006.

[12] T. Holter og E. Harborg, "Talegjenkjenning for teksting av direktesendte TV-programmer", SINTEF-rapport STF40F01008, 2001.

[13] T. Holter, "Spesifikasjon av opptak for VOCALS", SINTEF-notat, 2006.

[14] V. Henriksen, "Speech recordings for use in VOCALS", SINTEF-notat, 2006.

[15] http://www.microsoft.com/windowsmobile/activesync/, sist besøkt februar 2008.