# Online mód ve zpracování řeči

Petr Schwarz a Jan Černocký, FIT VUT Brno

April 27, 2005

Úkolem tohoto cvičení je lehce Vás uvést do on-line vstupu zvuku a on-line zpracování řeči. Upozorňujeme, že se jedná o ukázkové, téměř nepovinné cvičení a že cvičící Vám budou schopni podat základní informace, nikoliv vysvětlit podrobně synchronisaci jednotlivých vláken při nahrávání. Autorem SW pro on-line vstup je Petr Schwarz a k podrobnému pochopení se musíte "naložit" tak na půl dne do zdrojáků. Pokud ale chcete v budoucnu něco s on-line zvukem dělat, stojí to za to, protože Petr to má opravdu dobře napsané.

## 1 Základní vyzkoušení

Stáhněte si balík `energy.tgz`, rozbalte, zkompilujte pomocí `make` a spusťte exáč `energy`. V terminálu byste měli vidět obrázek odpovídající "plovoucí energii". Připojte mikrofon a něco do něj řekněte nebo zaťukejte, energie by měla vzrůst.

### Nejede ?

Mát pravděpodobně něco s mikrofonem či jeho nastavením:

- zkontrolujte, zda je zapíchnutý do správného konektoru.
- přečtěte si help k ALSA-Mixéru `man alsamixer` a pak `alsamixer` spusťte.
- ...jeho nastavení není úplně triviální ..., musíte se dostat do "capture" módu (F4), zvolit správný vstup pro zvuk (Mic) a vytáhnout "potenciometr" Capture. Na mém počítači kupodivu dostávám zvuk, i když je na nule....
- Pokud se nic nezměnilo, vraťte se do módu Playback (F3) a ověřte, zda čtete ze správného mikrofonu: MicSelect musí být v poloze Mic1.
- Pokud nepomohlo ani to, můžete v módu Playback vyzkoušet MicBoost (zesílení signálu z mikrofonu).

## 2 Kuk do zdrojáků – easy case.

Na první pohled si povšimnete, že už to není C, ale C++. Základem SW pro on-line vstup není main `energy.cpp`, ale dvě třídy (jména odpovídají názvům souborů):

- `LWFSource`, která se stará o čtení dat ze zvukovky.
- `ENE`, která provádí segmentaci na rámce a výpočet energie. Pohledem do `ene.{cpp,h}` ovšem zjistíte, že tato třída je velmi chudičká a neobsahuje prakticky žádný kód. Je totiž poděděná ze třídy `Frames`, ve které přepisuje pouze jedinou funkci (aha, je to C++, takže **metodu**): `processFrame`, která určuje, co se s přečteným rámcem udělá.
- vlastní `Frames` má důležitou funkci: produkuje jednotlivé rámce, na kterých je pak založené celé zpracování.

Nyní si otevřete main `energy.cpp` a pohleďte na na hlavní cyklus `while(1)`:

- volání `WFS.read` přečte vzorky.
- `ENE.addWaveform` je "vloží" do instance třídy pro výrobu rámců.
- v cyklu `while(ENE.getFeatures(features)` nám pak metoda `getFeatures` produkuje parametry na tolik rámců, na kolik má k disposici vzorků.
- pak se vrátíme opět na čtení nových vzorků.

Zobrazování energie je řešeno pomocí třídy `SpecGraf`, která využívá knihovnu `curses`. Abychom na obrazovce viděli nejen současnou energii, ale i její historii, je použit buffer, ve kterém se energie postupně odsouvají doleva a při výpočtu energie nového rámce je celý buffer znovu vykreslen na obrazovku: `SG.show(enebuf)`. U této "vykreslovací třídy" Vás patrně bude zajímat její inicializace:
`void SpecGraf::set(int y, int x, int dy, int dx, int nbins, float min_val, float max_val)`

- `y, x` - souřadnice rohu na obrazovce, odkud chceme vykreslovat, zde 0,0.
- `dy, dx` - šířka obrázku, který budeme vykreslovat, pokud -1,-1, použije se celé okno.
- `nbins` - počet hodnot na vykreslování.
- `minval` - minimální hodnota.
- `maxval` - maximální hodnota.

### 2.1 Na co je to dobré ?

V této laboratoři pro Vás nemáme zadání typu "udělejte toto a toto". Zkuste například popřemýšlet, jak tento SW použít k výrobě on-line rozpoznávače izolovaných slov pomocí DTW:

- napíšete si třídu zděděnou z `Frames`, která bude umět nejen energii, ale i LPCC koeficienty.
- implementujete jednoduchý VAD (Voice Activity Detector), založený třeba na tom, že energie přeleze nějaký práh, k detekci jednotlivých slov.
- pokud detekujete slovo, necháte jeho matici LPCC koeficientů srovnat se všemi referencemi a vypíšete výsledek.
- Jelikož bude VAD detekovat kdeco (šumy, jiná slova než jsou ve slovníku, atd.), uvažujte o kategorii "reject", například když bude vzdálenost u všech referencí příliš velká.
- výpočet DTW může trvat dlouho... Dejte si pozor, abyste nezačali ztrácet vzorky a zvětšete velikost bufferu pro vstup vzorků (`BUFF_LEN` v `config.h`) nebo, jste-li zkušení mazáci, zauvažujte o 3-vláknové struktuře (1. vlákno čte vzorky, druhé se stará o rámce a parametry, třetí o rozpoznávání).

## 3 Kuk do zdrojáků – difficult case.

Chcete-li zjistit, jak to přesně pracuje, pročtěte si slajdy Petra Schwarze (soubor `cv3.pdf` na webu) a naložte se do zdrojáků. Ignorujte slide 29. "Vaše práce".