VTApi v. 1.0

Technická zpráva - FIT - VG20102015006 - 2011 – 01

Petr Chmelař, Vojtěch Fröml
Tomáš Volf, Jaroslav Zendulka

Fakulta informačních technologií, Vysoké učení technické v Brně

1. prosince 2011

Abstrakt

VTApi je API (rozhraní pro programování aplikací) založené na databázi PostgreSQL a knihovně počítačového vidění OpenCV. Je orientováno na zpracování záznamů video a obrazové informace – kategorizace, vyhledávání a porovnávání.

Abstract

VTApi is a PostgreSQL database and OpenCV API (Application Programming Interface) for the VT project. It is oriented towards processing of records containing image and video information – categorization, searching and comparison.

Obsah

1. Úvod

Hlavním cílem projektu VideoTeror je definovat, zkoumat a vytvořit funkční vzorek systému pro zpracování záznamů obrazového a video charakteru za účelem boje proti terorismu. V rámci projektu jsme předpokládali vytvoření systému, který bude založen na jednom počítači nebo počítačovém clusteru.

Tento systém bude vybaven "datovým skladem" do kterého se budou umisťovat záznamy obrazů a videosekvencí a databází, v níž budou uloženy výsledky analýzy dat na různých úrovních. Dotazy na výsledky analýzy tak budou prováděny ve formě databázových dotazů. Algoritmy analýzy budou získávat údaje z původních dat v kombinaci s dotazováním výsledků v databázi. Výsledky budou ukládány zpět do databáze. Tímto postupem se vytvoří analytické zařízení umožňující kombinaci řady přístupů a stane se flexibilním zařízením pro pokročilou analýzu dat. Předpokládá se replikace takového zařízení v případě přání zadavatele. Funkce, které by mohly být v zařízení implementovány, jsou například:

- Skladování anotovaných videozáznamů, například typu "anotovaná videosekvence" s identifikací typů scéna a výskytem objektů ve scéně. Samotné pořizování takových dat není předmětem projektu a spíše se předpokládá získání dat od zadavatele.
- Extrakce příznaků z obrazů a videosekvencí, například založených na detekci klíčových bodů a obdobných (SIFT, SURF, MSER apod,), integrálních a lokálních vlastností obrazů (histogramy jasů, barev, gradientů), lokální příznaky (LBP, LRF, Haarovy apod.)
- Strojové učení například pro parametrizované trénování metodou SVM pro analytické úlohy včetně implementace některých vybraných analytických úloh, jako jsou například detekce typu scény na základě anotovaných příkladů, sumarizace videosekvencí, detekce objektů (obličeje osob, dopravní značky apod.)
- Aplikace dalších metod získávání znalostí z dat, například pro detekci nežádoucích událostí, perzistentní sledování podezřelých osob v prostoru pokrytém mnoha kamerami apod. Vybrané biometrické metody pro identifikaci osob a objektů (například pro zkoumání vztahů mezi mírami objektů, metriky dynamiky pohybu, případně identifikaci osob).

Na základě tohoto cíle bylo vytvořeno aplikační rozhraní VTApi, což je soubor tříd (knihovna) pro zpracování záznamů videa a obrazové informace – kategorizace, vyhledávání a porovnávání, cílem je sjednotit a urychlit vývoj aplikací v oblasti počítačového vidění (v rámci projektu VideoTeror na FIT VUT v Brně). Aplikační rozhraní využívá nástroje pro zpracování obrazové informace (OpenCV, http://opencv.willowgarage.com/wiki/) a post-relační databázi PostgreSQL (http://www.postgresql.org/) pro uchování metadat k obrazovým datům.

2. Specifikace

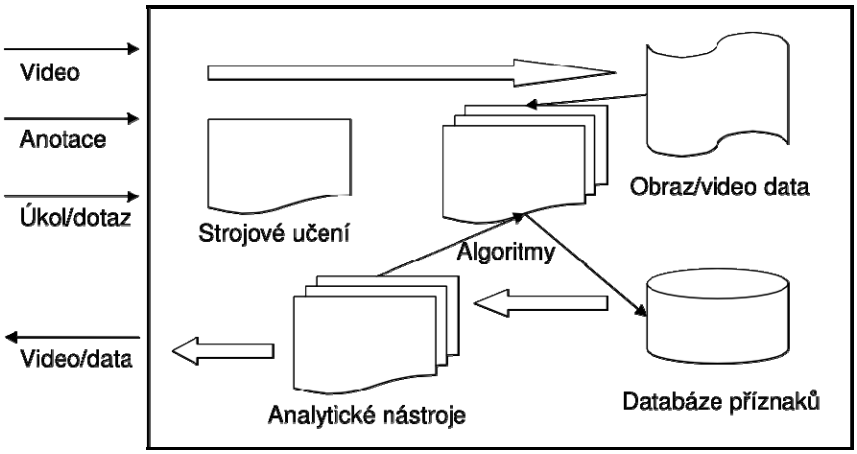
V rámci projektu VideoTeror, jsme se shodli na následujících termínech, které reprezentují třídy objektů, které je nutné zpracovávat a uchovávat:

- **Dataset** je množina (multimediálních) dat spolu s metadaty (popisem dat). Datasety jsou logicky vzájemně disjunktní, ale některý může být založen na několika dalších.
- Sekvence (**Sequence**) je množina snímků (videa) nebo obrázků, základní jednotka datasetu. V sekvenci je předpoklad časové uspořádanosti snímků.
- **Interval** je podmnožina sekvence (množina snímků) zvolená tak, aby bylo možné jim přiřadit shodné informace (metadata či příznaky). Příkladem je sledovaný objekt ve videu nebo jeho scéna. Metadata mohou být obecně libovolná, ale jsou vždy vytvářena procesem.
- Proces (**Process**, úkol, operace) je konkrétní instance metody (**Method**, algoritmus, nástroj), která definuje strukturu dat. Proces tedy definuje (vkládá) data a reprezentuje většinu aktivit navrhovaného zařízení

Následují doplňující pojmy:

- Obrázek (**Image**) a rámec videa (**Frame**), případně jejich kolekce (**Images** a **Video**), v jejich běžné interpretaci, reprezentují instance intervalů, respektive sekvencí.
- **Tag** je indexační termín. Tyto jsou (v hierarchii) přiřazené multimediálním datům pro jejich popis (anotaci).
- Selekce (**Selection**) je podmnožina logicky souvisejících metadat vhodně zvolená tak, aby operace (dotazy) nad nimi byly efektivní a umožňovaly přirozené řetězení procesů (vstupem jednoho je výstup druhého nebo obrazová data). Speciální příklady selekce jsou intervaly nebo tagy.
- Klíč-hodnota (**Key-Value**) je základní mechanismus uspořádání metadat ve VTApi, datová struktura umožňující uchování dat ve dvojici <klíč, hodnota> tak, aby při změně definice dat nebylo nutné měnit kód API.

Z návrhu projektu bylo převzato základní blokové schéma, uvedené na obrázku 1, které je sice pouze ideovou ilustrací sestavení celého zařízení, ale vhodně jej ilustruje (terminologie byla v průběhu projektu sjednocena, viz výše).



Obrázek 1: Blokové schéma a ilustrace funkcionality VTApi.

2.1. Příklady použití

Na základě potřeby použití navrhovaného zařízení jsme definovali dvě modelové situace.

Situace 1: Sledování objektů více kamerami

V objektu je několik kamer, každá snímá z jiného místa/úhlu. Probíhá záznam videa z kamer (MD). Obecné dotazy na databázi poté jsou:

- Zobraz video sekvence v určitých časových rozpětích z určitých kamer (jeden dataset?) (7:00 - 11:30; 12:00 – 18:00).
- Zobraz zrychlené/zpomalené (násobek) záznamy v daných časových okamžicích z výběru kamer (z datasetu) (7:00 – 11:30; 12:00 – 18:00).
- Možnosti synchronizace videa v rámci datasetu (shift videa do minulosti/budoucnosti), příkladem můžou být stereokamery, rozladěné hodiny na kamerách apod.

Dotazy/situace týkající se trackingu:

- Časově závislé (umístění, velikost) ROI trackovaného objektu; anotace uložená v DB z jednoho videa; Dotaz: ukaž mi ROI sledovaného objektu v ostatních videích v datasetu (dotaz tvořící nové anotace pomocí algoritmu).
- Situace: kamera + termokamera na jednom místě.
- Situace s termokamerou. Dotaz: zobraz úseky videí (abstrakt frames) z datasetu, kde pohybující se objekty měly vyšší teplotu než 30°C (detekce obličejů).
- Otázka: Nebylo by vhodné ukládat i mapu sledovaného místa (datasetu) včetně polohy kamer u jednotlivých videí? Může pomoci synchronizaci kamer/trackovaných objektů, výpočtu vzdálenosti objektu (obecně algoritmům).
- Určení vzdálenosti sledovaného objektu – stereokamery.

Situace 2: Klasifikace segmentů videa

Video záznamy uložené v databázi jsou rozděleny na segmenty, které lze reprezentovat jedním nebo více klíčovými snímky (TRECVid SIN). Druhou alternativou je, že se pracuje pouze s klíčovými snímky, tedy obrázky, ale může se jednat o libovolné obrázky v databázi (ITS). Cílem je extrahovat informace z obrázků tak, aby je bylo možné klasifikovat do předem definovaných kategorií (v naší terminologii: přidělit jim tagy):

- Možnost ukládat obrázky/klíčové snímky do databáze (úložiště)
- Možnost načtení obrázků z úložiště a jejich vyhledávání podle tagů přiřazených (anotace) nebo automaticky zjištěných (klasifikace). Uchovávat a dotazovat tyto tagy.
- Podobnostní dotaz na obrázky
- Možnost přidat proces (2 typy metod – anotace a klasifikace).
- Přidat sekvenci (soubor obrázků).
- Přidat intervaly (obrázky).
- Možnost vložit výsledky procesu.
- Update výsledků procesu, na dotaz: ke kterým obrázkům není spočítaný proces.

2.2. Definice funkcí systému

Na základě modelových situací, provedené podrobnější analýzy a rešeršní činnosti v oblastech zpracování záznamů video a obrazové informace, byla zpřesněna vnější i vnitřní funkcionalita zařízení. Je rozdělena do sedmi kategorií:

1. *Elementární, obecněji a opakovaně použitelné algoritmy*
   Do této kategorie patři vnitřní funkcionalita a algoritmy, které budou využívány řadou úloh, jejichž řešení bude systém podporovat. Půjde především o algoritmy pro extrakci příznaků, jako jsou barevné histogramy, SIFT, SURF, založených na waveletech a frekvenčních transformacích, další specifikované například ve standardu ISO/IEC 15938 (MPEG-7) a jiných metadat, jejich efektivní ukládání a zpřístupnění. Předpokládáme použití, resp. vyvinutí řady specializovaných extraktorů. Dále sem patří podpora technik strojového učení a dolování v datech, zejména pro účely klasifikace, jmenovitě algoritmy AdaBoost, WaldBoost, SVM (Support Vector Machine) a GMM (Gaussian Mixture Models).
2. *Techniky a algoritmy detekce objektů a tvarů v obraze/videu*
   Předpokládáme vývoj technik a algoritmů pro detekci a identifikaci předdefinovaných objektů – specializované algoritmy pro detekci konkrétních předdefinovaných objektů, například pro obličeje, dopravní značky, psy, jízdní kola atd., tj i metody pro jejich klasifikaci. Dále rozpoznávání typu scény – z jednotlivých snímků, případně vylepšení na videosekvenci statistickým zpracováním řady snímků (předpokládají se typy jako „zprávy“, „vesnice“, fotbalový zápas“

apod.). Do této kategorie také patří základní funkčnost VTApi – vyhledávání podle popisu geometrie – pomocí Houghovy a obdobných transformací, extrahovaných příznaků, grafových gramatik, klasifikace objektů apod.

3. *Techniky a algoritmy sledování objektů a vyhodnocení trajektorie*
   Zde předpokládáme vývoj technik sledování objektů rozpoznatelných v obrazu (tracking), který otáčí kamerou, technik konstrukce trajektorie pohybujících se objektů, dotazování nad časoprostorovými daty trajektorií, vyhodnocení a kategorizace trajektorií, zjišťování anomálií apod.

4. *Techniky a algoritmy zpracování videosekvencí*
   Do této kategorie patří funkcionalita a algoritmy, jejichž cílem je zefektivnit práci uživatele pracujícího s videem. Předpokládáme podporu a vývoj algoritmů pro digitální stabilizaci obrazu ve videosignálu, kontrolu kvality a spojitosti digitálního videostreamu, vyhledávání ve videu, rozpoznávání scény, sumarizaci apod. Dále sumarizace a vyhledávání videosekvencí – buď video ve videu nebo opakující se videosekvence, případně i synchronizace videosekvencí (smyslem je například zjistit narušení videostreamu, kontrolovat kvalitu a spojitost digitálního videostreamu apod.). Počítáme rovněž s aplikací dalších metod získávání znalostí z dat, například pro detekci určitých událostí či stavů.

5. *Akcelerace algoritmů*
   V souladu s informacemi v přihlášce projektu bude součástí i řešení problematiky akcelerace vybraných algoritmů v DSP a/nebo GPU jednotkách. Kromě toho předpokládáme pro řešení vybraných problémů urychlení využitím paralelismu u vícejádrových systémů, např. pro paralelní trénování klasifikátoru.

6. *Podpůrné funkce a nástroje*
   Mezi podpůrné funkce bude sloužit podpora pro import videa, případně i metadat pro situace, kdy předzpracování proběhne externě, půjde o anotované video apod. Protože řada úloh pokročilého zpracování videa využívá technik strojového učení, které vyžadují pro vytvoření modelu anotovaná data, předpokládáme jako podpůrnou funkcionalitu možnost anotace.

7. *Ostatní metody*
   V této kategorii předpokládáme „nadstavbové“ metody a techniky, jejichž zařazení do řešení projektu bude závislé na požadavcích, podmínkách pro řešení, dostupnosti dat pro experimentování apod. V úvahu připadají například techniky a algoritmy zpracování videa z multikamerových systémů – metody pro zjištění topologie multikamerových systémů a sledování objektů pohybujících se mezi nimi, vyhodnocení trajektorie objektů v prostorech střežených mnoha kamerami za použití jako HMM a dalších statistických a

pravděpodobnostních, či Techniky a algoritmy zpracování termosnímků a záznamů z termokamer.

2.3. Datový model

Datový model zjednodušeně ilustrovaný na obrázku 2 do značné míry kopíruje definice uvedené v úvodní kapitole a operace, které k nim logicky náleží. Všechny třídy dědí ze třídy KeyValues, která zajišťuje základní operace potřebné pro správu dvojic klíč-hodnota (asociativní pole), na nichž je model VTApi založen.

K nim jsou přiřazeny třídy zjednodušující chápání jednotlivých pojmů uvedených v úvodu. Jsou to především třídy s množinou obrazových dat (Video, Images) a s jednotlivými obrázky (Frame, Image). Třída KeyValues je tedy stěžejní pro zajištění primární funkcionality API. Ostatní třídy dědí její atributy (prezentovány svými třídami):

- Commons1 zajišťuje velice základní funkce, jako je načtení konfiguračního souboru a parametrů příkazové řádky (blíže specifikované v programové dokumentaci), zajišťuje připojení k databázi (PostgreSQL), datovému úložišti (souborový systém) a dále jednotným způsobem spravuje chybové hlášení a další výpisy (log).
- Třída Select slouží ke konstrukci dotazů, které po prvém zavolání funkce next() zpřístupní požadované informace z databáze. K tomuto slouží funkce v diagramu uvedená jako getX(klíč), kde X je v reálné implementaci název datového typu požadovaného klíče, tedy zde existují funkce getString(k), getInt(k), getIntArray(k), getFloat(k), getFloatArray(k) atp.

Například, po vytvoření nového objektu image třídy Image pomocí funkce newImage z objektu třídy Sequence je objekt select inicializován na dotaz, který zpřístupní všechny obrázky dané selekce. Pomocí volání next() se atributy objektu image (zprostředkované například pomocí getString(location) nebo getName()) morfují, takže není nutné vytvářet další nové třídy, ale právě jen volat next() pro procházení všech obrázků selekce.

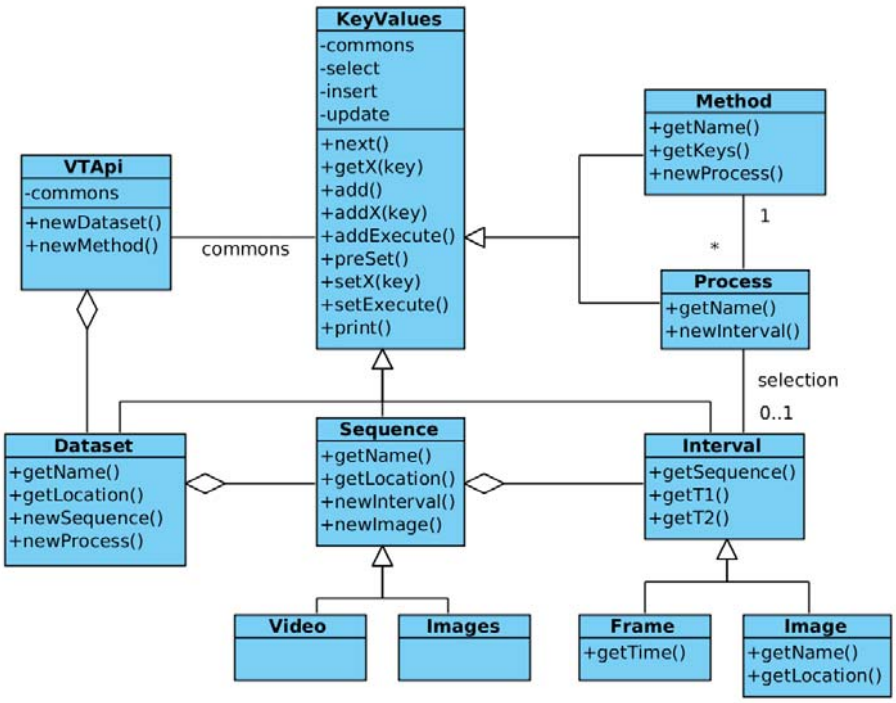
- Třída Insert zajišťuje vkládání odvozených dat, kde je to možné (Sequence, Interval, Process) nebo obecných dat prostřednictvím třídy KeyValues, jak je uvedeno v ukázce kódu v příloze 2, pomocí funkcí add() pro vkládání dat daných třídou API, resp. Primárním klíčem selekce, Obecně existují 2 způsoby vkládání – okamžitý (addExecute()) nebo dávkový (implicitně zavoláním next() nebo destruktorem).
- Třída Update obdobně umožňuje úpravu hodnot v klíč-hodnotovém systému aktuálního záznamu (zpřístupněného pomocí next()) funkcí preSet() (volitelně), setX(key, value).

1 Ve skutečné implementaci je atribut realizován dědičností.

Třídy odvozené z KeyValues obsahují jen minimum speciální funkcionality. Především se starají o konzistenci příslušejících dat (primárních klíčů) – k tomuto souží getY() metody, kde Y je v diagramu a implementaci nahrazeno názvem klíče, tedy například getName() pro název entity dané třídy nebo getLocation() pro její umístění (například datasetu nebo adresáře s obrázky). Dále obsahují některé zjednodušující funkce, aby nebylo nutné volat konstruktor, ale logicky postupovat v toku programu pomocí newC() metod, kde C je v diagramu a implementaci nahrazeno „nižší" třídou v hierarchii. Tato funkce poté zpřístupní data logicky (obsažené) související s aktuálním záznamem (pomocí next()). Tedy například metoda getSequence() po zavolání na objektu třídy Dataset vytvoří nový objekt třídy Sequence, ve ktrém budou po zavolání next() zpřístupněny všechny sekvence aktuálního datasetu, jehož název aktuálně vrací funkce getName().

Další informace o těchto třídách je možné získat z programové dokumentace na stránkách http://vidte.fit.vutbr.cz.

V neposlední řadě, třída VTApi představuje vstupní třídu API a zajišťuje připojení k databázi, jejíž model je uveden v příloze 1. Zde jsou pro jednoduchost uvedeny jen takové atributy (sloupce), které už není možné dále redukovat pro plnou funkčnost API, nicméně principem klíč-hodnota je zajištěna její plná rozšiřitelnost pomocí getX(klíč), add() a setX(klíč, hodnota), jak bylo uvedeno výše v této kapitole.



Obrázek 2: Zjednodušený model tříd VTApi.

3. Technická dokumentace

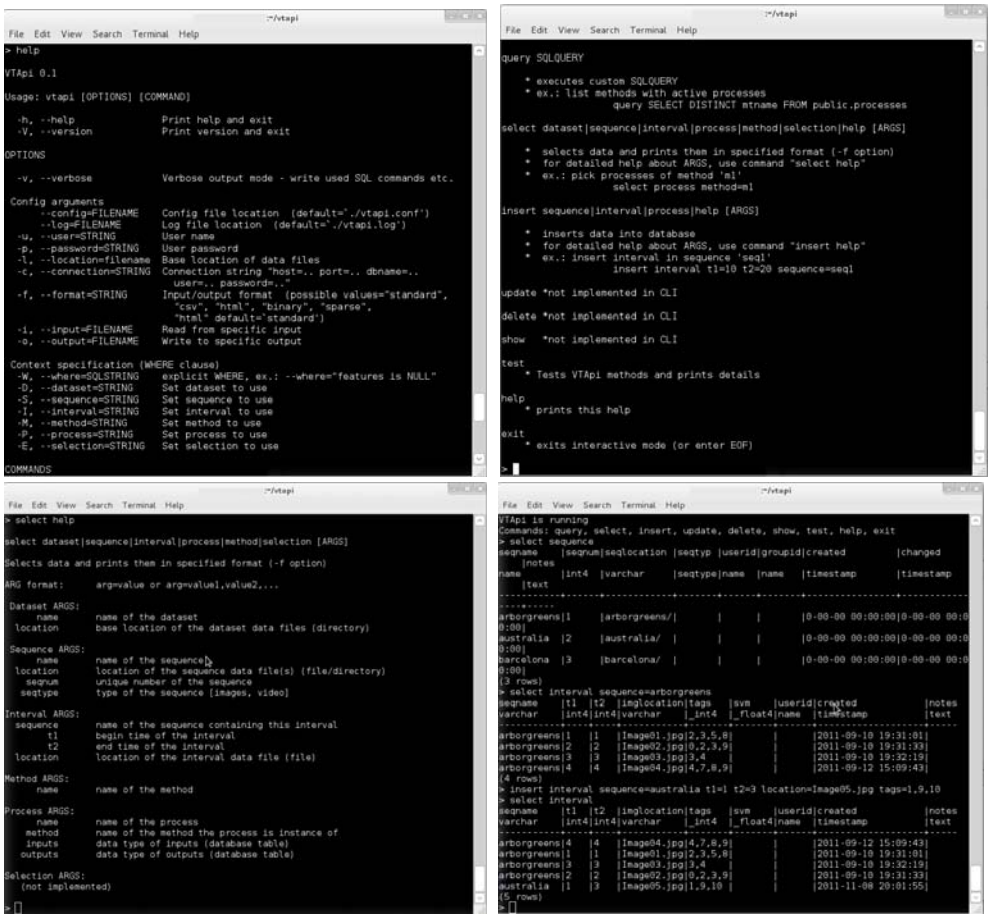
Technická dokumentace je přiložena v příloze společně s diagramem tříd v podobě generované ze zdrojových kódů (C++), které je společně s dokumentací možné získat na stránkách https://gitorious.org/vtapi. Zde se dále nachází Wiki, která popisuje základní předpoklady pro funkci (použité knihovny), dále návod na kompilaci ze zdrojových kódů (pro operační systémy POSIX/Linux a Windows) různých částí API:

- Vlastní projekt VTApi (C++, adredáře include, src).
- Rozhraní zpřístupňující API z příkazové řádky (VTCli).
- Programová dokumentace včetně plnohodnotného diagramu tříd a databáze dle případu použití „Situace 2“ (TRECVid SIN, ITS).
- Rozhraní jazyka Python za podpory knihovny Boost (VTAPyI, adresář pyi).

Technická dokumentace je dále dostupná z http://vidte.fit.vutbr.cz/vtapi.html.

3.1. VTCli

Rozhraní zpřístupňující API z příkazové řádky je tzv. samodokumentační pomocí příkazu help, jak je uvedeno na následující ilustraci. V horních dvou obrazovkách je byla vyvolána nápověda - příkaz help ($ vtapi help), vlevo dole nápověda k dotazování ($ vtapi select help) a vpravo dole je uveden příklad dotazování a vkládání.



Obrázek 3: Ilustrace nápovědy VTApi (VTCli)
a příklad funkcionality z příkazové řádky.

4. Závěr

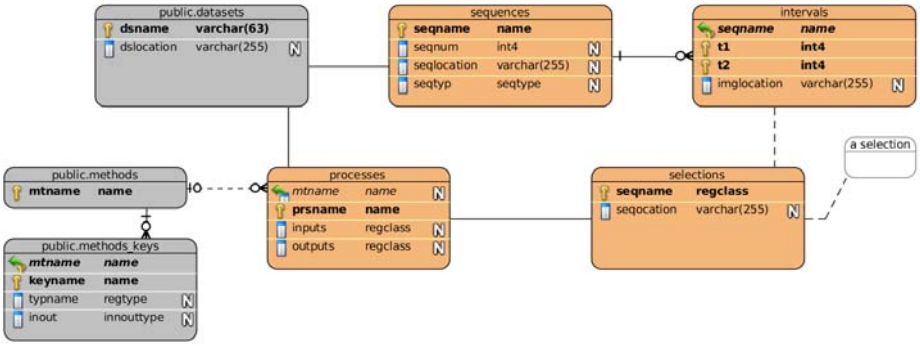
V rámci testování proběhly 2 typy testů – funkční a výkonnostní. Prvním typem funkčního testu (verifikačním) je test vestavěný a spustitelný z VTCli ($ vtapi test). Druhým typem funkčního testu (validační) je demonstrativní použití VTApi pro klasifikaci a vyhledávání obrazových dat, jak je uvedeno v případu použití „Situace 2“. V rámci tohoto použití byl proveden výkonnostní test na serveru pořízeném v rámci projektu VideoTeror (s testem na síti FIT) s následujícími výsledky:

- Množství dotazů je c.a. 1000 za sekundu (pomocí operace newInterva()).
- Množství vkládaných/upravovaných dat je c.a. 1000 za sekundu (při celkové velikosti 4MB).

Jako možnost dalšího pokračování vidíme široké možnosti rozšíření, především v další funkčnosti a optimalizaci výkonnosti.

Výchozí bod pro získání dalších informací a programové dokumentace je na adrese http://vidte.fit.vutbr.cz/vtapi.html. Tato zpráva včetně jejích příloh zde bude pravidelně aktualizována.

Příloha č. 1: Model databáze



Obrázek 4: Minimální logický datový model (schéma databáze) VTApi.

V obrázku 4 je ilustrován minimální logický datový model databáze ve schématu public. Oranžově označené tabulky jsou specifické pro jednotlivý dataset a mohou se vyskytovat ve více schématech. Bílou barvou je označena jedna z možností vytváření selekcí (první z nich je intervals) – v tomto případě se jedná o (dočasnou) tabulku „a selection“, ale je plánováno využití úložiště založené na souborech s patřičnými manipulátory v podobě odvozené třídy KeyValues.

Příloha č. 2: Příklad použití API

Procházení datasetů v DB

```
VTApi* vtapi = new VTApi(argc, argv);
Dataset* dataset = vtapi->newDataset();
while (dataset->next()) cout << dataset->getName();
```

Procházení obrázků po sekvencích

```
Sequence* sequence = dataset->newSequence("my_seq");
if (!sequence->next) error("No my_seq");
Image* image = sequence->newImage();
while (image->next())
    cout << image->getString("imglocation");
```

Přidání sequence

```
sequence->add("my_new_seq", "location/filename");
sequence->addInt("cislo", 7);
sequence->addExecute();
```

Načtení a editace pole z intervalu (obrázku)

```
int size = 0;
float* array = image->getFloatA("test", size);


image->setFloatA(array, size);
image->setExecute();
```

Práce s libovolnou tabulkou <klíč, hodnota> je doporučeno konzultovat nejprve konzultovat s tvůrci

```
KeyValues* tags = new KeyValues(*dataset, "tags");
tags->select = new Select(*tags);
tags->select->whereString("real_name", "NULL");
while (tags->next()) …
```

Mazání je řešeno jako libovolný dotaz, opět důrazně doporučujeme nejprve konzultovat

```
tags->update = new Query(*tags,
"DELETE FROM "+ tags->getDataset() +
    ".tags WHERE tagid=7");
```

```
tags->update->execute();
```

Přidání datasetu, metody a částečně také procesu je v současné implementaci možné provést pouze prostřednictvím tvůrců API.

Příloha č. 3: Sady dat

V rámci projektu VideoTeror testujeme VTApi na následujících datových sadách (Dataset):

- TRECvid SIN (ITS) na základě dat „Internet Archive videos with Creative Commons licenses“ (50GB, 200 hodin) v MPEG-4/H.264 s trváním mezi 10s s 3,5min (IACC.1, http://www.archive.org/details/movies) spolu s anotovanými 346 koncepty (herec, letadlo, pláž, …).

Více viz http://www-nlpir.nist.gov/projects/tv2011/tv2011.html#sin.

- TRECVid MED / AVSS Challenge na základě dat „HOSDB's i-LIDS Multiple Camera Tracking Training Corpus “ (128GB, 44 hodin, 5 kamer) z letiště London Gatwick v MPEG-2 nebo AVI (i-LIDS MCTTR, http://scienceandresearch.homeoffice.gov.uk/hosdb/cctv-imaging-technology/video-based-detection-systems/i-lids) spolu s anotovanými trajektoriemi některých osob (v XML).

Více viz http://www-nlpir.nist.gov/projects/tv2011/tv2011.html#sed a http://www.itl.nist.gov/iad/mig/tests/avss/2010/.

- Volně přístupný dataset University of Washington Image Database (Washington), (300MB, 1200 obrázků) zařazených do 21 kategorí (domorodci, fotbal, yellowstone, ...).

Více viz http://www.cs.washington.edu/research/imagedatabase/.