Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach

Žilina 2.–5. júla 2009

VTK – VIZUALIZAČNÝ NÁSTROJ S OTVORENÝM KÓDOM

ŠRÁMEK, Miloš, (SK)

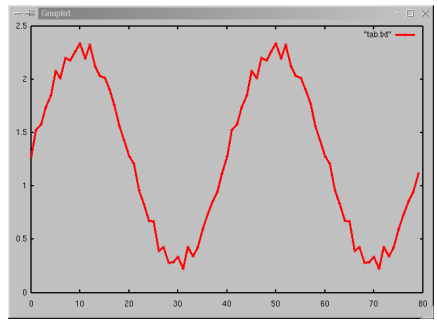
1 Úvod

Jedným z najvýraznejších trendov posledných rokov je enormný nárast objemu údajov, ktoré sú získavané simuláciou, meraním, alebo len jednoduchým zaznamenávaním udalostí. Príkladom môžu byť dáta, ktoré sú simulované na predpovedanie počasia a ktorých objem dosahuje až 1 PB (P=peta, teda 10^{15} bytov) ročne. Obdobný rozsah dát produkujú senzory snímačov v satelitnej astronómii alebo v časticovej fyzike.

Veľké objemy dát však nie sú doménou len veľkých projektov. Výsledkom dnes už bežne dostupného tomografického vyšetrenia bývajú desiatky až stovky miliónov číselných údajov. Samozrejme, že analýza takýchto objemov dát v ich pôvodnej numerickej forme nie je mysliteľná. Preto, v súlade so starou pravdou, že obrázok je hoden tisícky slov, sa často obraciame na možnosť prezentácie takýchto dát vo forme obrázkov. Tie na jednej strane skrývajú presnosť každého konkrétneho vstupného čísla, ktorú nahradzujú farbou, jasom či vhodným geometrickým tvarom. Na druhej strane však takáto abstraktná reprezentácia dokáže lepšie vypovedať o dátach ako celku, o ich charakteristických trendoch a črtách, čo používateľovi môže poskytnúť viac ako v prípade presných numerických údajov. Príkladom môže byť Obr. 1. Na jeho ľavej strane, v tabuľke, sú uvedené hodnoty namerané v 80 bodoch. Z tabuľky dokážeme, napríklad, presne zistiť, že v 15. meraní sme získali hodnotu 2.01. O dátach ako celku však toho veľa z tabuľky zistiť nedokážeme. Tu nám skôr pomôže vizuálna reprezentácia vo forme grafu na pravej strane obrázka. Ihneď vidíme, že ide o zašumený sínusový priebeh, pričom ľahko určíme aj jeho amplitúdu, periódu a úroveň šumu – teda údaje, ktoré nás obvykle zaujímajú najviac.

Vizuálnou prezentáciou číselných, ale aj nečíselných údajov sa zaoberá *vizualizácia*. Na jednej strane, z uhla pohľadu výskumníka, je to vedný odbor, ktorého primárnym cieľom je

0	1.28	20	1.28	40	1.28	60	1.28
1	1.52	21	1.21	41	1.52	61	1.21
2	1.58	22	0.96	42	1.58	62	0.96
3	1.74	23	0.84	43	1.74	63	0.84
4	1.85	24	0.68	44	1.85	64	0.68
5	2.08	25	0.67	45	2.08	65	0.67
6	2.01	26	0.39	46	2.01	66	0.39
7	2.21	27	0.43	47	2.21	67	0.43
8	2.18	28	0.28	48	2.18	68	0.28
9	2.26	29	0.29	49	2.26	69	0.29
10	2.34	30	0.34	50	2.34	70	0.34
11	2.2	31	0.22	51	2.2	71	0.22
12	2.33	32	0.43	52	2.33	72	0.43
13	2.12	33	0.34	53	2.12	73	0.34
14	2.04	34	0.42	54	2.04	74	0.42
15	2.01	35	0.6	55	2.01	75	0.6
16	1.91	36	0.73	56	1.91	76	0.73
17	1.77	37	0.86	57	1.77	77	0.86
18	1.56	38	0.94	58	1.56	78	0.94
19	1.43	39	1.12	59	1.43	79	1.12



Obr. 1: Číselná (vľavo) a grafická (vpravo) reprezentácia dát

vývoj a skúmanie metód na transformáciu vstupných dát do formy, ktorú dokážeme vnímať. Na druhej strane, z uhla pohľadu používateľa, vizualizácia predstavuje celkom konkrétne postupy, ktoré treba aplikovať, aby sme naše dáta zobrazili a tak získali možnosť na ich analýzu, pochopenie a prezentáciu. Aj keď vizualizácia dát je možná aj bez použitia počítača (papier, ceruzka, pravítko), bez počítačov by vizualizácia nikdy nebola tým, čím je dnes.

Pre používateľa je v prvom rade zaujímavá otázka, či je dostupný program, pomocou ktorého dokáže zobraziť svoje dáta. Niet pochýb, možností je veľa. Odhliadnuc od jednoduchých programov na kreslenie dvoj- či trojrozmerných grafov (Gnuplot, Kpl, ...), používatelia majú k dispozícii komerčné (Iris Explorer, AVS), ale aj voľne dostupné vizualizačné systémy ako Data Explorer či Mevislab a najmä veľmi populárny systém VTK, ktorým sa budeme v príspevku venovať. Okrem iného aj preto, lebo je to zaujímavý open-source projekt.

2 VTK – The Visualization Toolkit

VTK je open-source systém na počítačovú grafiku, spracovanie obrazu a vizualizáciu. Pozostáva v prvom rade z knižnice C++ tried, ktoré poskytujú základné nástroje na spracovanie rôznych typov dát, doplnené rozhraním pre interpretované jazyky Python, Java alebo TCL/Tk. Tie používateľovi umožňujú pohodlnejšie zostavovanie vlastných programov určených na riešenie daného problému. VTK dokáže spracovávať skalárne, vektorové a tenzorové údaje, ktoré sú definované nad rôznymi typmi dvoj-, troj a viacrozmerných mriežok, a to od neštruktúrovaných (údaje sú snímané s ľubovoľnou polohou) až po pravouhlé. Podporuje paralelizmus, pri ktorom sa spracovanie dát rozloží na viacero procesorov (prípadne spolupracujúcich počítačov) a tzv. prúdové spracovanie, pri ktorom sú dáta rozdelené

na menšie bloky, ktoré sa spracovávajú postupne na jednom alebo na viacerých procesoroch. Druhá možnosť zabezpečí, že na danom počítači možno spracovať úlohu ľubovoľnej veľkosti – samozrejme, že v primerane dlhšom čase.

VTK je softvér krytý veľmi voľnou licenciu[1], ktorá povoľuje šírenie programu v zdrojovej alebo binárnej forme, za podmienky (odhliadnuc od ďalších, menej podstatných), že bude distribuovaný spoločne so samotnými licenčnými podmienkami. Táto licencia bola prevzatá z operačného systému BSD [2]. BSD licencia je jednou z najznámejších open-source licencií, ktoré povoľujú ďalšie šírenie softvéru bez podmienky zverejňovania zdrojového kódu modifikácií a odvodeného softvéru. Takto sa VTK môže používať nielen v open-source produktoch, ale aj v komerčných riešeniach, pri ktorých sa zdrojový kód nezverejňuje. Táto črta odlišuje open-source (otvorený) softvér od slobodného softvéru, pri ktorom sa požaduje, aby odvodený softvér, poskytovaný používateľovi, bol krytý rovnakou licenciou ako pôvodný softvér (tu obvykle ide o niektorú z verzií licencie GNU General Public License).

Systém VTK bol pôvodne vytvorený ako sprievodný softvér ku knihe o počítačovej grafike a vizualizácii trojice autorov Schroeder, Martin a Lorensen, ktorí vtedy boli zamestnancami spoločnosti General Electric [5]. Autori však knihu aj softvér napísali vo svojom voľnom čase (so súhlasom spoločnosti), takže autorské práva zostali u nich. Vďaka svojej licencii si tento softvér rýchlo získal priaznivcov, ktorí ho nezačali len používať, ale aj sami prispievali k vývoju. VTK neskôr začala podporovať aj spoločnosť GE a ďalšie organizácie. GE začala dokonca aj predávať softvérové produkty založené na VTK – samotné VTK však (samozrejme) nie. Časť autorov neskôr GE opustila a založila spoločnosť Kitware, ktorá sa zaoberá poskytovaním služieb, súvisiacich s VTK a neskôr s ďalšími open-source produktmi a vývojom komerčného softvéru na báze svojich otvorených projektov.

Autori VTK pri vývoji svojho softvéru využili viaceré postupy a technológie [2]. V prvom rade, VTK je open-source projekt. Open-source môžeme vnímať ako technológiu vývoja softvéru, kde jedným zo základných pravidiel je zásada „zverejňuj skoro, zverejňuj často“. Používatelia sú pritom začlenení do „virtuálneho vývojárskeho tímu“ tým, že prispievajú ďalšími návrhmi, upozorňujú na chyby a prípadne ich aj odstraňujú alebo dokonca aj prispievajú svojim vlastným kódom. V tomto open-source technológia v mnohom pripomína technológiu tzv. agilných vývojových metód[3] (agile methodologies), v ktorých sa smerovanie vývoja určuje priamo počas samotného vývoja, a nie vopred vo fáze definovania cieľa, ktorá je obvykle prvým krokom v tradičnom postupe vývoja softvéru. Motivácia pre tento postup práce vyplýva najmä z toho, že v prípade vývoja špeciálneho softvéru v neprebádanej oblasti je ťažko možné vopred presne stanoviť ciele, ktoré má vývoj dosiahnuť. Tento prístup umožňuje modifikovať ciele za behu a priebežne ich prispôsobovať požiadavkám zákazníka – pričom aj samotné jeho požiadavky sa vyvíjajú v procese vývoja.

Softvér, ak sa má použiť v náročných aplikáciách, musí byť kvalitný. V tejto oblasti vývojári VTK využili niektoré postupy tzv. extrémneho programovania, ktoré zdôrazňuje význam

[1] `http://www.vtk.org/VTK/project/license.html`
[2] `http://en.wikipedia.org/wiki/BSD_licenses`
[3] `http://agilemanifesto.org/`

testovania. Konkrétne, využili tzv. testom riadený vývoj, kde testy správnosti sa vyvíjajú súčasne s programom, ba niekedy aj skôr. Napísaním testu sa vlastne stanoví špecifikácia, ktorú potom samotný program implementuje. Existencia testov umožňuje pravidelné vykonávanie regresných testov, ktoré sa v prípade VTK vykonávajú každý deň (či skôr, každú noc). Výsledok testov je dostupný na druhý deň ráno, takže autor má možnosť si overiť, či jeho zmeny mali požadovaný účinok. Testy VTK sa vykonávajú až v 50 rôznych verziách na rôznych platformách[4]. Tento postup zabezpečuje rýchle odstraňovanie chýb (obvykle do 24 hodín) a stálu funkčnosť vývojárskej verzie softvéru.

3 Použitie VTK

3.1 Inštalácia

VTK nebýva súčasťou základnej inštalácie operačného systému, treba si ho doinštalovať. Je dostupný pre všetky bežné operačné systémy. Zdrojový kód poslednej stabilnej verzie možno získať zo stránok `http://www.vtk.org/VTK/resources/software.html`. Pre Ubuntu (a Debian) Linux je dostupná aj binárna verzia VTK, ktorú spolu s príkladmi, dátami a dokumentáciou nainštalujeme príkazom

```
sudo apt-get install vtklib5 vtk-python vtk-examples vtkdata vtk-doc
```

Balíky sú v úložisku `universe` a sú dosť rozsiahle. VTK je na príslušnej stránke dostupný v binárnej verzii aj pre Windows. Ak pre iné prostredia nie je binárna verzia k dispozícii, alebo ak chceme použiť najnovšiu verziu, systém je potrebné preložiť.

3.2 Jednoduchý príklad

Na zobrazenie (vizualizáciu) vstupných dát treba, bez ohľadu na ich typ, vykonať niekoľko základných operácií a definovať niekoľko objektov. Prvým krokom je vytvorenie objektu na reprezentáciu zdroja, ktorým môžu byť buď dáta zo súboru, alebo ktorý môže byť definovaný priamo vo VTK. Na obrázku 2 je vstupným objektom kužeľ, definovaný interne prostredníctvom metódy `vtkConeSource`. Takýto objekt zatiaľ zobraziť nevieme, existuje vo svojej pôvodnej abstraktnej forme. Na ďalšie zobrazenie ho musíme skonvertovať na množinu zobraziteľných primitív, akými sú trojuholníky, úsečky, body a iné (v tomto prípade bude kužeľ reprezentovaný ihlanom). Táto operácia sa nazýva mapovanie a je súčasťou každého vizualizačného programu. Samotné mapovanie môže byť v závislosti na vstupe zložitou operáciou, pri ktorej sa, napríklad, môže povrch objektu zosnímaného tomografom aproximovať sieťou tisícok trojuholníkov.

[4] Výsledky nočných testov VTK zverejnené na stránke `http://www.cdash.org/CDash/index.php?project=VTK`

Riadok

```
coneMapper.SetInput(cone.GetOutput())
```

demonštruje základnú koncepciu spracovania dát vo VTK. Vidíme, že objekt `cone` má akýsi výstup, ktorý sa pripája na vstup objektu `coneMapper`. VTK je vizualizačný systém založený na toku dát – tie „pretekajú“ *filtrami*, kde sa transformujú. VTK filtre sú v mnohom podobné filtrom, ako ich poznáme z unixových operačných systémov. V našom prípade ide o transformáciou ihlanu, ktorý je definovaný svojimi rozmermi a počtom stien, na množinu polygónov.

Takto vytvorený objekt by sme už mohli zobrazovať, potrebujeme však okno (*vtkRenderWindow*), v ktorom sa to bude diať a nástroj, ktorý to dokáže (*vtkRenderer*). S objektom budeme chcieť manipulovať, a tak pridáme manipulátor (*vtkReaderWindowInteractor*). Zatiaľ nám však chýbajú zobrazovacie parametre, akými napríklad sú uhol pootočenia, zväčšenie, farba a podobne. Tie spravuje „herec“ (vtkActor), ktorého pridáme a priradíme mu náš objekt, konvertovaný na polygóny. Posledným krokom je pridanie herca k rendereru a spustenie renderovania a interakcie.

Vidíme, že použitie VTK spočíva v špecifikácii a správnom zoradení modulov, ktoré zabezpečia potrebné a požadované operácie. Horeuvedená základná štruktúra vizualizačného programu pritom zostáva rovnaká, menia sa len jednotlivé moduly – triedy, pomocou ktorých dosiahneme požadovaný účinok. Napríklad (Obr. 3), zmenou dvoch riadkov, ktoré na Obr. 2 definujú kužeľ ako vstupný objekt, za načítanie tomografických dát, špecifikáciu povrchu objektu prostredníctvom prahovej hodnoty a vytvorenie trojuholníkového modelu povrchu dostaneme obrázok objektu zosnímaného tomografom – v tomto prípade ľudskej hlavy. Ostatné časti programu pritom zostanú rovnaké.

4 Softvér na báze VTK

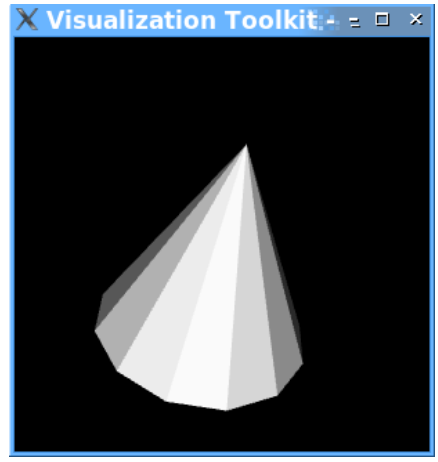
V predchádzajúcej časti sme videli, ako možno VTK použiť na vizualizáciu priamo využitím základných nástrojov, ktoré VTK poskytuje prostredníctvom knižnice tried. Táto možnosť je však dostupná len pre skúsenejších používateľov, ktorí navyše zvládajú programovanie v niektorom z jazykov, ktoré systém podporuje (Python, C++, Java a Tk/Tcl). Triedy VTK sú však priamo navrhnuté tak, aby umožňovali vytváranie nadstavieb, ktoré zložitosť programovania skryjú za používateľsky prívetivejšie grafické rozhranie. Keďže VTK je open-source systém, takýchto nadstavieb vzniklo viac, a to rovnako komerčných ako aj voľne dostupných s otvoreným kódom[5].

Slicer Program Slicer[6] poskytuje nástroje na segmentáciu, registráciu a trojrozmernú vizualizáciu multimodálnych dát so zameraním na vizualizáciu dát pochádzajúcich zo štúdii

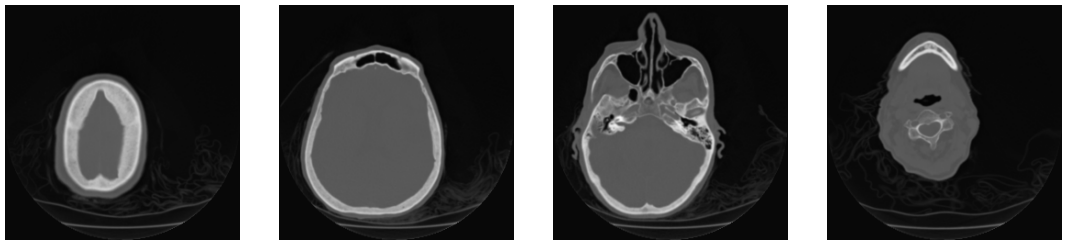
[5] na stránke `http://www.vtk.org/Wiki/VTK_Tools` je zoznam 27 aplikácií využívajúcich VTK

[6] `http://www.slicer.org`

```
#! /usr/bin/python

# nacitanie potrebnych rozsireni VTK
from vtk import *

# Vytvorenie abstraktneho objektu, ihlanu
cone = vtkConeSource()
cone.SetResolution(12)

# Priprava objektu na renderovanie jeho
# transformaciou na renderovatelne polygony
coneMapper = vtkPolyDataMapper()
coneMapper.SetInput(cone.GetOutput())

# Vytvorenie renderovacieho okna
renWin = vtkRenderWindow()
renWin.SetSize(300,300)

# Vytvorenie renderera
ren = vtkRenderer()
renWin.AddRenderer(ren)

# Povolenie zakladnej interakcie
iren = vtkRenderWindowInteractor()
iren.SetRenderWindow(renWin)

# Spojenie renderovatelneho objektu so scenou
# prostrednictvom herca (actor)
coneActor = vtkActor()
coneActor.SetMapper(coneMapper)

# Pridanie objektu (herca) k~rendereru
ren.AddActor(coneActor)

# spustenie renderovania a~interakcie
iren.Initialize()
iren.Start()
```



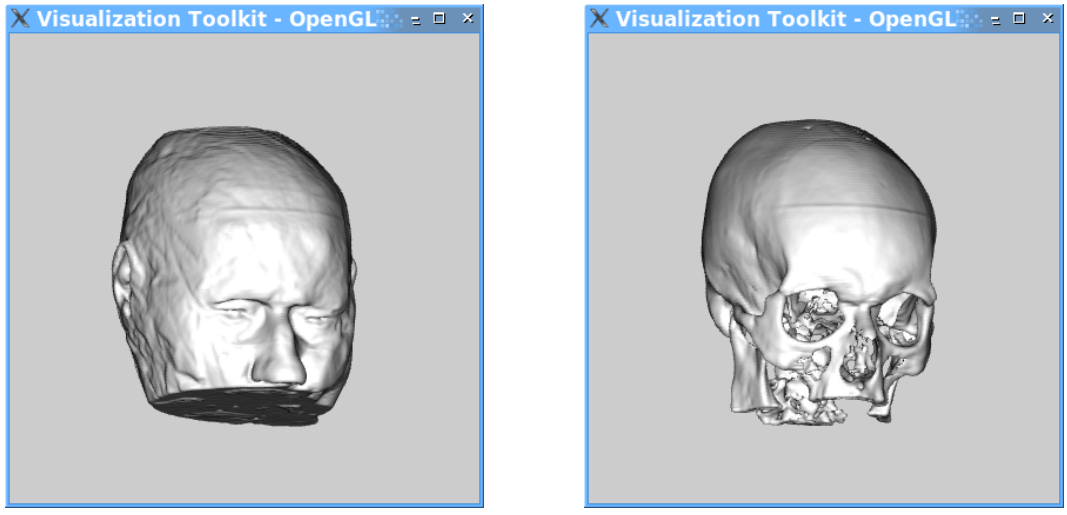
Obr. 2: Ukážka jednoduchého VTK kódu v jazyku Python



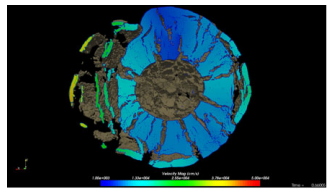
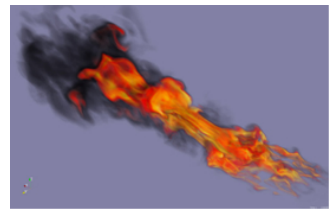
...

```
# Nacitanie vstupneho suboru s~rozmermi 170x190x184 bodov
# a~nastavanie parametrov
reader = vtkImageReader()
reader.SetFileDimensionality(3)
reader.SetFileName("tot2.raw")
reader.SetDataScalarTypeToUnsignedChar()
reader.SetDataExtent(0,169,0,189,0,183)

# specifikacia objektu prostrednictvom jasovej urovne 'prahova_hodnota'
skinExtractor = vtk.vtkContourFilter()
skinExtractor.SetInput(reader.GetOutput())
skinExtractor.SetValue(0, prahova_hodnota)
```

...



Obr. 3: Jednoduchou zmenou definície objektu z Obr. 2 môžeme zobraziť objekt snímaný tomografom. Hore: štyri zo 184 rezových obrázkov, z ktorých pozostávajú vstupné dáta. Stred: Načítanie dát do VTK, Dolu: Zobrazený trojuholníkový model vytvorený zo vstupných dát. Model bol vytvorený tak, že jasová hladina na úrovni `prahova_hodnota` bola "vydláždená" trojuholníkmi. Vľavo: povrch pokožky, prahova_hodnota=40, vpravo: povrch kosti, prahova_hodnota=100

Obr. 4: Vľavo: vizualizačný klaster RedRose s 264 uzlami v Sandia NL, stred: šírenie trhlín pri výbuch v centre asteroidu, vpravo: simulácia horenia (`http://www.psc.edu/general/software/packages/paraview/`)

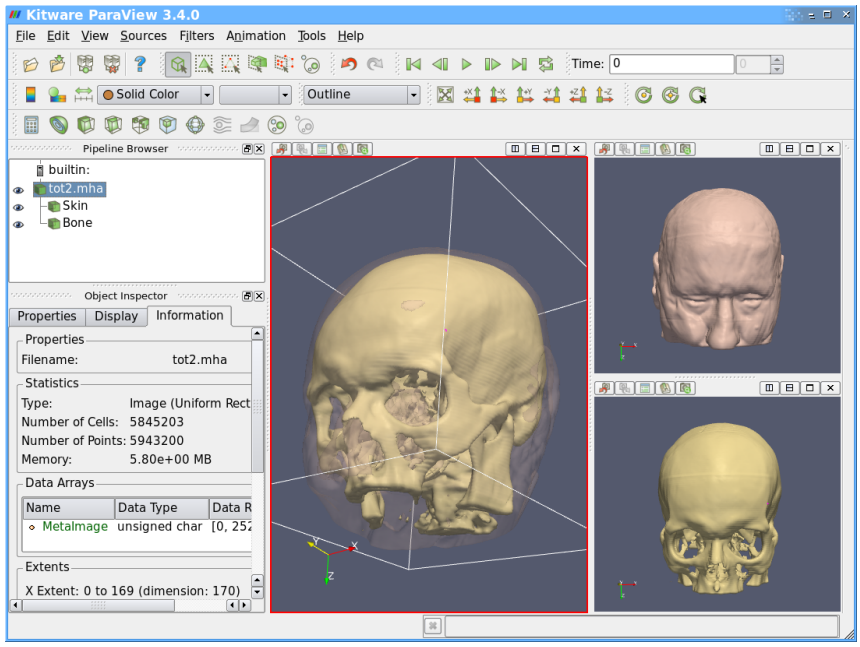
v oblasti medicínskeho zobrazovania (difúzne tenzorové zobrazovania, funkčná magnetická rezonancia) a iné biomedicínske aplikácie.

ParaView Paraview[7] je škálovateľný vizualizačný program určený pre širokú paletu platforiem od jednoprocesorových desktopov, cez viacprocesorové systémy so zdieľanou pamäťou až po počítačové klastre. Je vybavený viacerými vizualizačnými algoritmami na zobrazovanie tokov a na povrchové a objemové zobrazovanie a podporuje zobrazovanie na multiobrazovkových displejoch a stereoobrazovkách. Pri behu na paralelnom systéme Paraview dokáže pracovať s veľmi rozsiahlymi statickými aj dynamickými dátami definovanými na rôznych typoch mriežok (pravouhlé, štruktúrované, neštruktúrované).

ParaView [7] bol vytvorený autormi VTK (spoločnosť Kitware) v spolupráci s Los Alamos National Laboratory. Motiváciou od začiatku bolo vyvinúť škálovateľný vizualizačný nástroj s podporou paralelizmu a distribuovanej pamäte. Jeho vývoj neskôr podporili aj ďalšie národné laboratóriá a iné organizácie v USA. Paraview sa využíva najmä na vizualizáciu výsledkov rozsiahlych simulácií. Napríklad v Sandia National Laboratory v USA sa používa na vizualizácie výsledkov simulácií, ktoré sa robia na 10 000 uzlovom klastri, pričom sám Paraview beží na 264 uzloch s dvoma procesormi AMD a grafickým akcelerátorom NVidia Quadro (Obr. 4 vľavo). Príkladom aplikácií je vizualizácia výsledku simulácie výbuchu nálože s energiou ekvivalentnou 10 megatonám TNT v ťažisku asteroidu Golevka (mriežka s 1.1 mld. buniek) alebo vizualizácia výsledkov simulácie horenia so 150 miliónmi stupňov voľnosti.

VTK Designer 2 Program VTK Designer[8] je grafický editor na vytváranie aplikácií v prostredí VTK. Výstupom je program, ktorý sa zapisuje v ľubovoľnom z jazykov, ktoré VTK podporuje. VTK Designer je určený pre výskumníkov, ktorí potrebujú zostavovať zložité vizualizácie, ale aj pre vývojárov, ktorí chcú využiť VTK vo svojich vlastných aplikáciách. Charakteristickou črtou programu je možnosť jednoducho zostavovať programy z modulov prostredníctvom grafického rozhrania.

[7] `http://www.paraview.org`
[8] `http://www.vcreatelogic.com/oss/vtkdesigner/`



Obr. 5: Použitie Paraview na vizualizáciu tomografických dát. Zobrazené sú povrchy dvoch tkanív – pokožky a kosti (vpravo). V strede sú zobrazené v perspektívnom náhľade, pričom pokožka je priehľadná

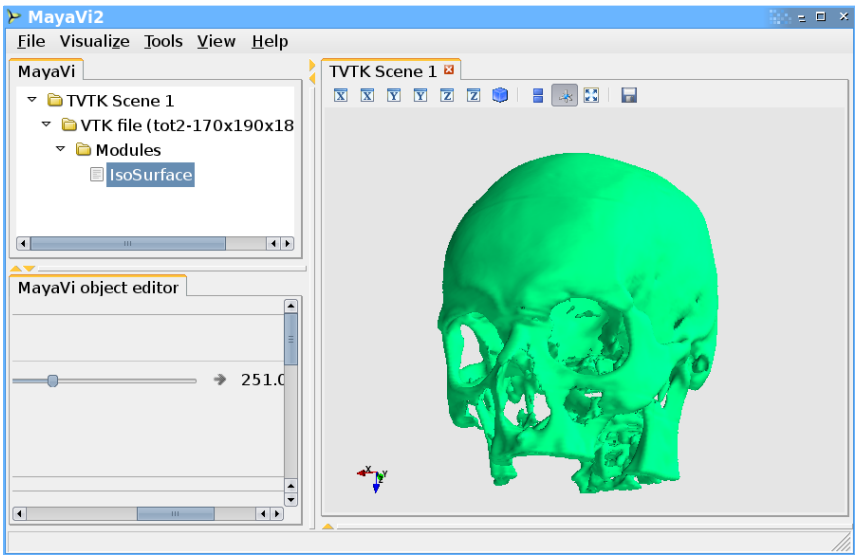
Mayavi2 Mayavi2[9] je mnohoúčelový a viacplatformový nástroj na vizualizáciu 3D dát, so zameraním najmä na vizualizáciu skalárnych, vektorových a tenzorových dát v dvoch a troch rozmeroch, ľahké skriptovanie pomocou Pythonu a ľahké rozširovanie pomocou vlastných modulov a filtrov. Súčasťou balíka je grafická aplikácia `mayavi2`, ktorá umožňuje zostavovanie vizualizácií bez potreby programovania, rovnako ako aj nástroje, ktoré umožňujú mayavi využiť ako kresliaci stroj v iných aplikáciách.

5 Ďalšie OSS programy

Pre autorov VTK a aj pre ich spoločnosť Kitware je open-source základným vývojovým modelom, v ktorom do svojej práce zapájajú komunitu vývojárov-dobrovoľníkov. Okrem základného produktu, VTK, v spolupráci s komunitou boli vytvorené aj ďalšie nástroje na spracovanie dát, ale aj na podporu samotného vývoja softvéru.

The Insight Toolkit – ITK Cieľom vizualizácie je prezentácia dát vo forme obrázkov. Častokrát však pred samotnou vizualizáciou treba dáta upraviť alebo analyzovať, čím sa získa odvodená informácia, ktorá je neskôr jadrom samotnej vizualizácie. Táto fáza tzv.

[9] `https://svn.enthought.com/enthought/wiki/MayaVi`



Obr. 6: Použitie programu Mayavi2 na vizualizáciu tomografických dát. Zobrazený je povrchy kostného tkaniva

predspracovania hrá mimoriadne dôležitú úlohu pri vizualizácii objemových dát, ktoré sú často používané v medicínskej diagnostike (tomografické metódy) a biológii (konfokálna mikroskopia). Objemové dáta sú vlastne postupnosťou dvojrozmerných obrázkov, ktoré zobrazujú priečne rezy snímaným objektom. Takýchto obrázkov možno pri jednom meraní získať až 2000, pričom jeden obrázok môže mať až 1000^2 (alebo aj viac) pixelov.

ITK [3] je nástroj na predspracovanie objemových dát so zameraním najmä na úpravu (filtrovanie, odstraňovanie šumu), segmentáciu a registráciu (priestorové zosúladenie dvoch alebo viacerých dátových objemov do spoločného súradnicového systému tak, aby ich bolo možné spolu ďalej spracovávať alebo zobrazovať). Pri *segmentácii* ide o identifikáciu objektov a tkanív v dátach. Príkladom môže byť detekcia tumoru v trojrozmernom obraze pečene. *Registrácia* je transformácia dvoch dátových objemov do spoločného súradnicového systému, ktorá je potrebná vtedy, ak je pacient snímaný viackrát, buď s cieľom zistiť vývoj choroby alebo jej liečenia, alebo ak bol pacient snímaný viacerými snímačmi, ktoré poskytujú doplnkovú informáciu.

ITK nemá prostriedky na vizualizáciu. Prepojovacie triedy však umožňujú spracovanie výstupov ITK modulov modulmi VTK a naopak. Možno teda povedať, že oba systémy sa v oblasti spracovania a vizualizácie objemových dát dopĺňajú. ITK má mnohé spoločné črty s VTK – licenciu, vývojový proces, štruktúru tried, podporované jazyky a tiež aj to, že je využívaný vo viacerých ďalších komerčných a otvorených systémoch (Mevislab, SCIRUN, [1]).

CMake VTK bol od začiatku vyvíjaný ako multiplatformový systém, s cieľom jeho použitia na všetkých hlavných operačných systémoch. Aj keď je jadro VTK napísané v jazyku C++, v rôznych systémoch sa používajú rôzne vývojové nástroje – prekladače, čo sťažuje prenos softvéru medzi jednotlivými systémami, alebo dokonca aj medzi rôznymi prekladačmi na jednom systéme. Navyše, VTK je rozsiahly systém so stovkami tried a státisícmi riadkov kódu. V čase vzniku VTK neexistoval nástroj, ktorý by umožnil písanie kódu tak, aby bol preložiteľný na všetkých cieľových platformách.

Z tohoto dôvodu autori VTK vytvorili systém `cmake` [4, 6], ktorý takýto preklad umožňoval. `cmake` je svojim účelom podobný dvojici nástrojov Automake/Autoconf, ktoré však podporujú platformovú nezávislosť len v prostredí unixových systémov a len pre kompilátory, použiteľné v programe `make`. `cmake` oproti tomu podporuje aj grafické vývojové prostredia (KDevelop, Eclipse, VisualC++ a ďalšie) tak, že z konfiguračného súboru `CMakeFile.txt` generujú buď súbor Makefile pre štandardné kompilátory používané na príkazovom riadku, alebo konfiguračné súbory jednotlivých IDE.

Úlohou programu CMake je testovanie aktuálneho systému, zistenie, aký kompilátor sa na ňom používa (v prípade prítomnosti viacerých je možný výber) a ktoré jeho nastavenia (prepínače) sú potrebné. Ďalej, zistí, kde sa nachádzajú hlavičkové súbory (v prípade prekladu programov v jazykoch C alebo C++) a potrebné knižnice. V prípade potreby dokáže generovať zdrojový kód, ktorý sa použije v preklade a napokon určí, kam sa majú uložiť výsledky prekladu.

`cmake` dnes postupne nahrádza už zastarávajúcu dvojicu Automake/Autoconf aj pri preklade ďalších systémov, akými sú, napríklad, desktopové prostredie KDE a KDE aplikácie[10].

CDash CDash je sieťový server určený na testovanie softvéru, ktorý sa vyvinul z programu Dart [6]. CDash organizuje testovanie a zbiera, analyzuje a zobrazuje jeho výsledky, ktoré môžu byť posielané klientmi umiestnenými na ľubovoľnom mieste na Internete. Výsledky sú zobrazované v podobe webovej stránky, teda vývojári majú neustály prehľad o aktuálnom stave vyvíjaného softvéru. CDash je súčasťou systému na vývoj softvéru, ktorý zahŕňa open-source programy CMake a jeho doplnkové nástroje CTest a CPack od spoločnosti Kitware ako aj ďalšie externé nástroje na návrh, správu a údržbu rozsiahlych softvérových systémov (napr. Doxygen[11] na generovanie dokumentácie z komentovaného kódu, Valgrind[12] na testovanie správnosti použitia pamäte). Príkladom použitia Servera CDash je prehľadová stránka o stave vývoja samotného CDash a o ďalších projektoch[13].

[10] `http://techbase.kde.org/Development/Tutorials/CMake`
[11] `http://www.doxygen.org`
[12] `http://www.valgrind.org`
[13] `http://www.cdash.org/CDash/`

6 Záver

V príspevku sme si ukázali softvérový systém VTK, ktorý dnes v oblasti vizualizácie dát úspešne konkuruje komerčným produktom – ak ich svojimi vlastnosťami, schopnosťami a flexibilitou skôr nezatieňuje. VTK je súčasne aj významným open-source projektom, a ako taký je príkladom úspešného využitia princípov otvorenosti vo vývoji zložitých softvérových aplikácií.

Poďakovanie

Táto práca bola podporovaná Agentúrou na podporu výskumu a vývoja na základe Zmluvy č. APVV-20-056105.

Literatúra

[1] Ingmar Bitter, Robert Van Uitert, Ivo Wolf, Luis Ibanez, and Jan-Martin Kuhnigk. Comparison of four freely available frameworks for image processing and visualization that use itk. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):483–493, 2007.

[2] Ron Goldman and Richard P. Gabriel. *Innovation Happens Elsewhere: Open Source as Business Strategy*. Morgan Kaufman Publishers, 2005.

[3] Luis Ibanez, Will Schroeder, Lydia Ng, and Josh Cates. *The ITK Software Guide: The Insight Segmentation and Registration Toolkit*. Kitware Inc., first edition, 2003.

[4] Ken Martin and Bill Hoffman. *Mastering CMake*. Kitware Inc., 2008.

[5] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit, An Object-Oriented Approach To 3D Graphics*. Prentice Hall, 1996.

[6] William J. Schroeder, Luis Ibánez, and Ken Martin. Software process: The key to developing robust, reusable and maintainable open-source software. In *ISBI 2004*, pages 648–651, 2004.

[7] Amy Squillacote. *The Paraview Guide*. Kitware Inc., 2008.

Kontaktná adresa

Miloš Šrámek (doc. Ing., PhD),
Komisia pre vedeckú vizualizáciu,
Rakúska akadémia vied, Viedeň
milos.sramek@oeaw.ac.at

Fakulta riadenia a informatiky
Žilinská univerzita

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH

Zborník príspevkov medzinárodnej konferencie OSSConf 2009

2.–5. júla 2009
Žilina, Slovensko