PV251 Vizualizace

Jaro 2017

Výukový materiál

---

8. přednáška: Koncepty interakce a interakční techniky

Interakcí v kontextu vizualizace informací je mechanismus, pomocí kterého dokážeme ovlivnit, co a jak uživatel vidí.

Třídy interakčních technik

Existuje několik tříd interakčních technik:

- **Navigace** – uživatel mění pozici kamery a škálování pohledu (co je mapováno na obrazovku). Příkladem je rotace či zoomování.
- **Výběr** – ovládací prvky pro identifikování určitého objektu, sady objektů či oblasti zájmu, na které pak aplikujeme určité operace, jako například zvýrazňování, mazání či modifikace.
- **Filtrování** – ovládací prvky pro redukci velikosti dat, která mapujeme na obrazovku – odstraňováním záznamů, dimenzí nebo obojího.
- **Rekonfigurace** – uživatel může měnit způsob mapování dat na grafické entity nebo atributy. Příkladem je reorganizace dat nebo jejich rozložení. Takto poskytujeme uživateli různé pohledy na zobrazovanou datovou množinu.
- **Kódování** – uživatel mění grafické atributy, jako například velikost bodů či barvu čáry. Cílem je odhalit různé vlastnosti dat.
- **Spojování** – uživatel může využít nástroje pro spojování různých pohledů nebo objektů za účelem zobrazení vzájemně souvisejících položek.
- **Abstrahování/konkretizace** – ovládací prvky pro modifikaci level-of-detail.
- **Hybridní techniky** – ovládací prvky kombinující některé z výše uvedených technik. Například zvětšování prostoru obrazovky, který je využíván pro zobrazení detailu určitých dat, a zároveň zmenšování prostoru věnovaného „nezajímavým“ datům, která jsou zobrazena proto, aby byl zachován kontext.

Do dnešního dne byla vyvinuta řada technik a nástrojů pro interakci s daty a obecně vizualizací dané informace. I když se některé z nich jeví jako zcela nesouvisející s ostatními, sdílejí základní principy a mají společný cíl. Seznámíme se s identifikací tříd interaktivních operací, které popíšeme jako **operátory** a dále definujeme tzv. **operand** (prostor, na který je operátor aplikován).

Nyní si detailněji popíšeme řadu interakčních operací, které jsou běžně používány při vizualizaci dat a informace. Seznam nebude samozřejmě vyčerpávající, ale měl by pokrývat typické interakční techniky. Více detailní a komplexnější popis dalších technik můžete nalézt v Keimově klasifikaci (http://nm.merz-akademie.de/~jasmin.sipahi/drittes/images/Keim2002.pdf) či taxonomie Eda H. Chi (http://www-users.cs.umn.edu/~echi/papers/infovis00/Chi-TaxonomyVisualization.pdf).

Je nutné zmínit, že jednotlivé interakční operátory mohou být součástí několika navrhovaných tříd interakce a že téměř všechny operátory mohou být automatické v dané vizualizaci, a to i v její neinteraktivní části. Příkladem může být zoomování, které je dostupně téměř ve všech vizualizacích. Na zoomování se totiž můžeme dívat jako na generování nové vizualizace, zvláště pokud musíme zobrazovat různá data.

Operátory navigace

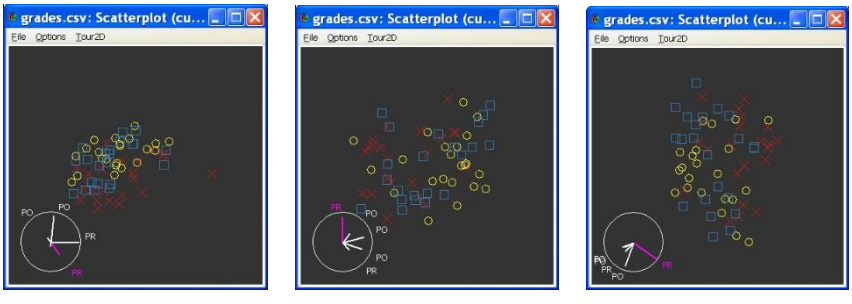
Navigace (někdy označována termínem exploration - prozkoumávání) je využívána pro vyhledání podmnožiny vstupních dat, která mají být prozkoumána, orientace pohledu na tato data a level-of-detail (LOD). Dotazovaná podmnožina může být určena pomocí určitého vizuálního vzoru nebo to může být část dat, která podléhá dalšímu detailnějšímu prozkoumání. V 3D prostoru je navigace typicky určena pozicí kamery, směrem pohledu, tvarem a velikostí objemu pohledu (viewing frustum) a stupněm LOD.

Ve vizualizacích podporujících několik rozlišení najednou odpovídá LOD sestupování nebo postupu nahoru v hierarchii dat.

Operátory navigace mohou pracovat s absolutními nebo relativními souřadnicemi v daném prostoru.

Navigace může být automatická nebo řízená uživatelem.

Příkladem automatického prozkoumání je průlet podél cesty nad multidimenzionálními daty, která pokrývá většinu nebo dokonce všechny možné orientace datového prostoru při jejich promítnutí do 2D prostoru (viz obrázek). Uživatel může ovlivňovat velikost kroku mezi jednotlivými pohledy.



Operátory výběru

Při výběru uživatel izoluje podmnožinu komponent pro zobrazení, které dále podléhají dalším operacím, jako je zvýraznění, mazání, maskování či přesun do středu oblasti zájmu. Dosud byly vyvinuty různé varianty výběru a vhodnou variantu zvolíme většinou tak, že musíme rozhodnout, jaký výsledek očekáváme. Například, měla by nová selekce nahradit tu stávající nebo by ji měla spíše doplnit/obohatit?

Granularita výběru je rovněž jedním z hlavních témat. Kliknutím na danou entitu na obrazovce můžeme vybrat pouze danou nejmenší adresovatelnou komponentu (např. vrchol na hraně) nebo můžeme vybírat širší region kolem vybrané lokace (např. celý objekt, oblast na obrazovce, povrch, …).

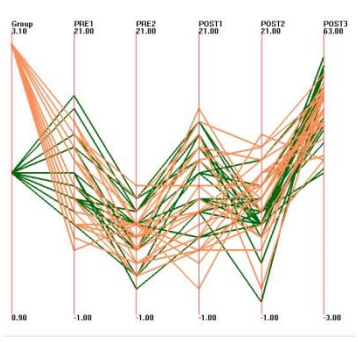
Výběr může být určen mnoha různými způsoby. Uživatel může klikat na jednotlivé entity, „kreslit“ přes výběr entit (např. držením tlačítka myši při pohybu nad objekty zájmu) či jiným způsobem izolovat entity (například výběrem do obdélníku, lasa, …).

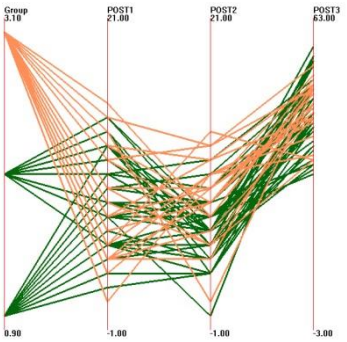
Výběry mohou být vytvářeny rovněž nepřímo, kdy systém vybere prvky, které vyhovují sadě omezení zadaných uživatelem. Příkladem je výběr uzlů v grafu, které mají určitou maximální vzdálenost od vybraného uzlu.

Operátory filtrace

Filtrace, jak už název napovídá, redukuje počet dat, která mají být zobrazena, a to pomocí nastavení různých omezení, která specifikují, která data budou zachována a která odstraněna. Příkladem takovéhoto filtru je tzv. *dynamic query specification*, která byla popsána Shneidermanem a spol. (http://www.cs.umd.edu/~ben/papers/Shneiderman1994Dynamic.pdf).

Pro určení rozsahu zájmu v datech jsou určeny slidery, při jejichž manipulaci je vizualizace okamžitě updatována, aby reflektovala změny vyvolané uživatelem. Tento způsob dotazování pomocí nastavení rozsahů je pouze jedním způsobem aplikace filtrování. Jiná metoda vybírá položky, které chce zachovat nebo naopak schovat. Příkladem je funkce schovávání sloupců v Excelu.





Obrázek ukazuje využití filtrace pro zjednodušení pohledu na data a jejich interpretace. Dochází k filtrování řádků a sloupců za použití nástroje XmdvTool.

Rozdíl mezi filtrováním a výběrem následovaným mazáním nebo maskováním je malý, nicméně zásadní. Filtrování je nejčastěji prováděno **nepřímo** – například specifikace filtru není prováděna na samotné vizualizaci dat, ale v odděleném dialogovém okně nebo rozhraní. Filtrování je často prováděno před samotným zobrazením dat, abychom se vyhnuli zobrazení přílišného množství dat na obrazovce.

Oproti tomu je výběr prováděn nejčastěji **přímo**, kdy označujeme zobrazené objekty například klikáním myší do scény. Další operace provedené nad vybranou množinou mohou vést k výsledku, který je stejný jako za použití filtrace.

Operátory rekonfigurace

Rekonfigurace dat v dané vizualizaci je často využívána pro odhalení vlastností nebo pro vypořádání se se složitostí či měřítkem dat. Pomocí reorganizace dat, jako například odfiltrování určitých dimenzí a přeskládání těch zbývajících, můžeme uživateli poskytnout různé nové pohledy na data. Příkladem je nástroj založený na tabulkové vizualizaci, který třídí řádky a sloupce za účelem zvýraznění trendů a korelací mezi daty.

Dalším příkladem rekonfigurace může být změna dimenzí používaných pro řízení x-ové a y-ové souřadnice vykreslených značek.

Populárními metodami rekonfigurace dat jsou již dříve zmíněné PCA (principal component analysis) či MDS (multidimensional scaling), které se snaží zachovat vztahy mezi daty ve všech dimenzích při jejich projekci do nižší dimenze (často 2D).

Operátory kódování

Jakákoliv datová množina může být využita pro generování bezpočtu různých vizualizací. Vlastnosti dat, které nejsou viditelné v jednom způsobu vizualizace, mohou být zřejmé při použití jiného typu vizualizace. Například pro určitou datovou sadu může využití bodového diagramu vést k překrytí bodů, zatímco například při použití paralelních souřadnic mohou mít body svou unikátní reprezentaci.

Mnoho dnešních vizualizací podporuje zároveň několik typů vizualizací, protože jediný typ vizualizace nemůže podchytit efektivní zobrazení všech úkolů, které chce uživatel nad daty provádět. Každá z vizualizací je nejvhodnější pro jistou podmnožinu datových typů, jejich vlastností a úkolů zadaných uživatelem.

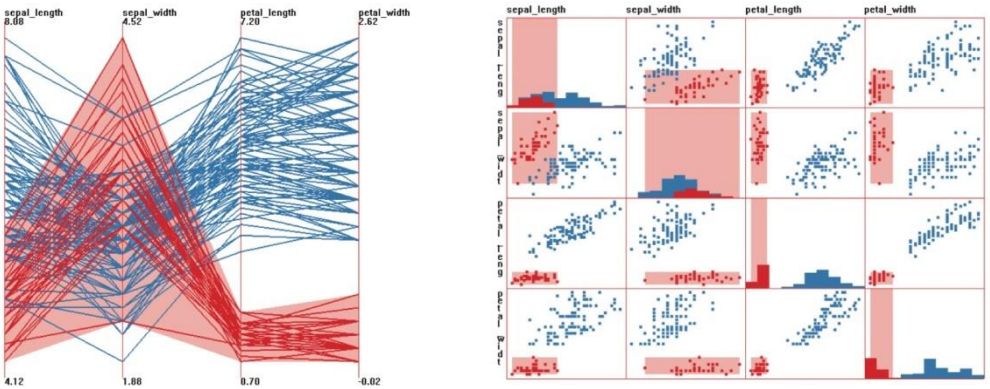
Kódování dat je prováděno různými typy mapování a pohledů na data, pomocí nichž uživatel může data řádně prozkoumat. Další formy kódovacích operací mohou například modifikovat použitou barevnou mapu, velikost grafických entit či jejich tvar. Toto může být považováno za různé variace daného typu vizualizace a pomáhá odhalit oblasti zájmu. Použitím různých variací dané techniky můžeme překonat i řadu omezení použité vizualizační techniky. Například problém překrývání bodů v bodových diagramech může být odstraněn roztřesením bodů nebo zvolením velikosti jednotlivých bodů takovým způsobem, že velikost daného bodu je odvozena od počtu bodů v té stejné pozici.

Další atributy grafických entit, které můžeme ovlivňovat, zahrnují průhlednost, texturování, styl čáry či výplně, ale také dynamické atributy, jako například úbytek intenzity či míra přeblikávání.

Je třeba zmínit, že tyto efekty mohou být často imitovány použitím transformací přímo na datech místo na jejich grafické reprezentaci.

Operátory spojování

Běžné použití operací výběru je spojení vybraných dat v jednom pohledu s odpovídajícími daty v pohledech dalších. Existuje řada způsobů spojení mezi podokny daných aplikací, nicméně zřejmě nejpoužívanější formou komunikace mezi okny v moderních vizualizačních nástrojích je tzv. *linked selection*. Popularita této formy vychází zejména ze skutečnosti, že každý z pohledů na data může odhalit zajímavé vlastnosti a že zvýraznění jedné z těchto vlastností v jednom pohledu může při jejím současném zobrazení v dalších pohledech pomoct vystavět „bohatší“ mentální model této vlastnosti (viz obrázek). V paralelních souřadnicích vybereme zkoumaný klastr, který je v matici odpovídajících bodových diagramů zobrazen pomocí obdélníka.



Pokud je povoleno interaktivně měnit výběr dat, nazývá se tento operátor *brushing*, kdy uživatel může soustavně měnit výběr v jednom pohledu, a odpovídající spojená data v ostatních pohledech jsou zvýrazněna. Další silnou stránkou techniky linked brushing je specifikace komplexních omezení pro daný výběr. Každý typ pohledu je optimalizován pro

zdůraznění jistého typu informace. Například můžeme specifikovat časové omezení při použití vizualizace obsahující časovou osu, omezení na jména polí při použití list view a geografická omezení pro mapy.

V určitých případech může uživatel chtít odpojit *(unlink)* některé vizualizace, kdy ponecháme daný pohled, ale chceme prozkoumat jinou oblast dat nebo jinou datovou množinu. Některé systémy povolují uživateli určit pro každé okno, zda se má informace přenášet do dalších oken nebo ze kterých oken toto okno přijímá vstup.

Některé typy interakce mohou být lokální vzhledem k danému oknu (např. zoom), zatímco jiné jsou sdílené mezi všemi okny (např. přeskládání dimenzí).

Operátory abstrahování/konkretizace

Při zobrazování velkého množství dat je často vhodné se soustředit pouze na určitou podmnožinu dat, o které zobrazíme detaily (konkretizace), zatímco na ostatních částech dat redukujeme stupeň detailu (LOD) (abstrahování). Jednou z nejpopulárnějších technik tohoto typu jsou **distorzní operátory** (*distortion operators*). Zatímco někteří vědci klasifikují tyto distorze jako vizualizační techniku, jedná se ve skutečnosti o transformaci, která může být aplikována na jakýkoliv typ vizualizace. Podobně jako zoomování či pannning (panorámování) je distorze vhodná pro interaktivní prozkoumávání dat. V minulosti bylo navrženo mnoho distorzních operátorů (nazývaných též *funkce*). To zahrnuje metody, které deformují celý analyzovaný prostor nebo metody aplikující deformace pouze lokálně.

Distorze může být součástí původní vizualizace nebo může být zobrazena v samostatném okně. Distorze se liší v poměru vlastností, které u vstupních dat zachovávají. Například techniky pro distorzi textu se snaží o co největší čitelnost v dané oblasti zájmu a zbytek textu je uveden hlavně pro udržení struktury dokumentu, nemusí být však čitelný.

Distorzní operátory mohou být lineární nebo nelineární, se spojitostí nultého, prvního nebo druhého řádu (nespojité operátory je rovněž možné využít). Operátory mohou být aplikovány na struktury namísto spojitých prostorů, a proto mohou být specifické pro daný typ operandu (viz dále).

Různé operátory mají různé „otisky“ (footprints), jako například tvar nebo rozsah prostoru ovlivněný transformací. Běžné tvary otisků jsou obdélníkové či kruhové, jimž ve vyšších dimenzích odpovídají hyperboxy a hyperelipsy. Rozsah ovlivněného prostoru je obvykle specifikován vzdálenostní funkcí uvnitř deformovaného prostoru a je často multidimenzionální. Tyto rozsahy mohou být fixní či proměnlivé, řízené uživatelem či sémantikou informace.

Interakční operandy a prostory

Parametry interakčních operátorů, které jsme dosud popsali, budou diskutovány ještě dále. Ještě předtím si ukážeme kategorizaci interakčních operandů, což pomůže objasnit roli těchto parametrů, kterou mají při procesu interakce, a jejich sémantiku v rámci různých prostorů.

Interakční operand *je část prostoru, na kterou je aplikován interakční operátor.*

Abychom mohli určit výsledek interaktivní operace, musíme vědět, uvnitř jakého prostoru bude interakce prováděna. Jinými slovy: pokud uživatel klikne na dané místo nebo oblast na obrazovce, které entity si vlastně přeje označit? Mohou to být pixely, datové hodnoty či záznamy mapovány na dané místo nebo třeba část vizualizační struktury (například osa).

Bylo identifikováno několik odlišných tříd interakčních prostorů. Nyní si tyto prostory popíšeme včetně uvedení příkladů existujících interakčních technik, které do jednotlivých tříd spadají.

Rozlišujeme několik základních interakčních operandů.

- Prostor obrazovky (Pixely)
- Prostor datových hodnot (Multivariate datové hodnoty)
- Prostor datových struktur (Components of Data Organization)
- Prostor atributů (Components of Graphical Entities)
- Prostor objektů (3D Surfaces)

Prostor obrazovky (Pixely)

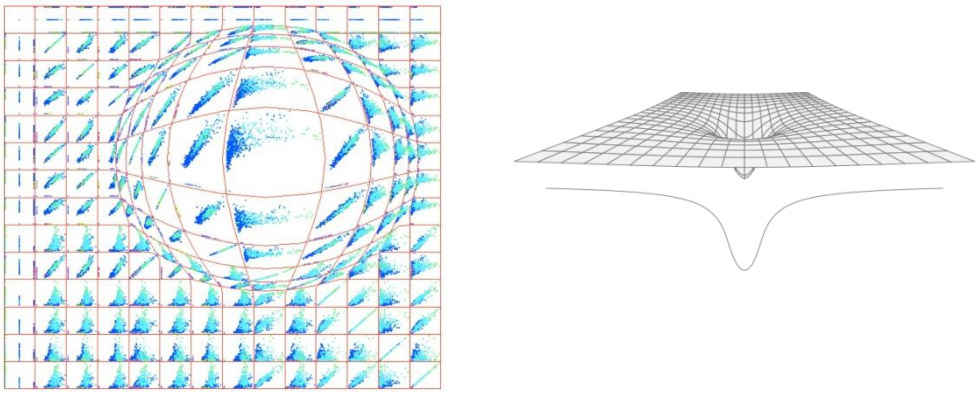
Navigace v prostoru obrazovky se typicky skládá z akcí, jako je například panorámování, zoomování či rotace. V žádném z těchto případů nevyužíváme žádná nová dodatečná data – proces se skládá z operací na úrovni pixelů, jako jsou transformace, vzorkování a replikace.

Selekce na úrovni pixelů znamená, že na konci dané operace je každý pixel klasifikován jako vybraný nebo nevybraný. Jak již bylo zmíněno dříve, výběr může být proveden nad jednotlivými pixely, obdélníkovými či kruhovými oblastmi pixelů nebo nad oblastmi libovolného tvaru, které uživatel definuje. Oblasti výběru mohou být rovněž spojité nebo nespojité.

Distorze v prostoru obrazovky zahrnuje transformaci na pixelech, například (x’, y’) = f(x, y).

Zvětšení (magnifikace) m(x, y) v daném bodě je jednoduše derivací této transformace a je užitečná pro přepínání mezi transformacemi a jejich zvětšeninami při procesu distorze. Příkladem technik v prostoru obrazovky jsou rybí oko, rubber sheet (reliéf).

Příklad tohoto typu distorze je uveden na obrázku. Vlevo je zobrazena matice grafů, vpravo pak rubber sheet.



Podíváme se blíže na jeden z typických příkladů této distorze: na rybí oko. Implementace rybího oka je poměrně přímočará. Je nutné specifikovat středový bod (c_x, c_y) transformace, poloměr čočky (lupy) r_l a velikost vychýlení (deflexe) d. Poté dochází k transformaci souřadnic obrázku do polárních souřadnic a to relativně ke středovému bodu. Efekt lupy je získán poměrně jednoduchou transformací aplikovanou v rámci poloměru r_l.

Jedna z populárních transformací tohoto typu je zadána vzorcem:

$$r_{new} = s\log(1+d(r_{old}))$$

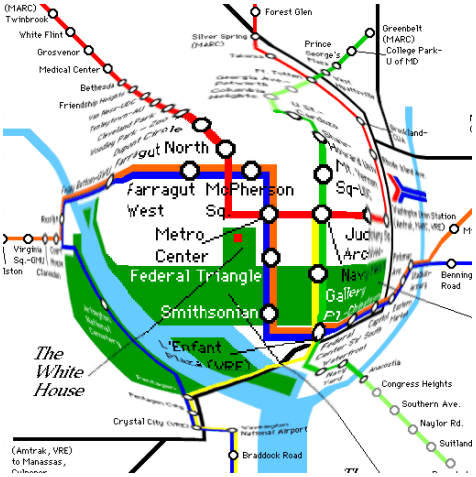
kde

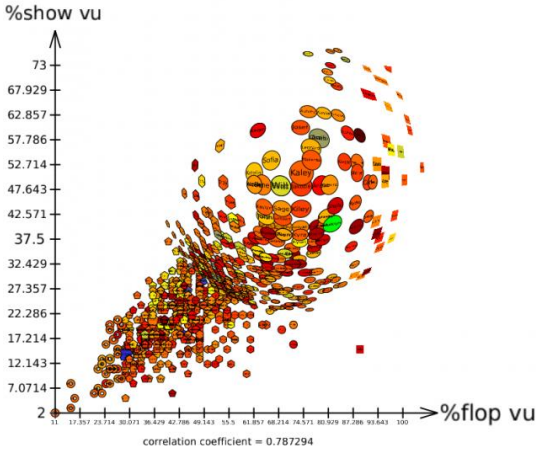
$$s = \frac{r_l}{\log(1+d*r_l)}$$

Tento vzorec zajišťuje, že poloměr bodů nacházejících se na okraji lupy je zachován na své původní hodnotě. Nyní si uvedeme pseudokód celého algoritmu:

1. Vyčistíme výstupní obrázek.
2. Pro každý pixel vstupního obrázku:
    1. Spočteme odpovídající polární souřadnice.
    2. Pokud je poloměr menší, než je poloměr lupy:
        1. Spočteme nový poloměr r_{new}.
        2. Získáme barvu tohoto místa z původního obrázku.
        3. Nastavíme tuto barvu jako barvu pixelu ve výstupním obrázku.
    3. Jinak nastavíme výsledný pixel obrázku na stejnou hodnotu, jakou má v původním obrázku.

Podle typu transformace použité pro distorzi v prostoru obrazovky se musíme příslušně vypořádat s děrami nebo naopak s překrytím pixelů, ke kterému při transformaci dochází. Překrývání pixelů nezpůsobuje tak velké problémy jako díry. Přesto jsou překrývající se pixely často v konkrétních implementacích průměrovány. Díry je nutné vyřešit pomocí interpolace. Zde záleží na typu použité lupy – pro vizualizaci textu se pro interpolaci nejčastěji používá po částech lineární funkce s „rovnou“ (flat) středovou částí, aby bylo možné text v této části bez problémů číst.

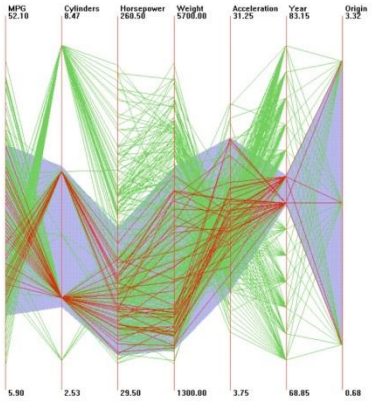




Prostor datových hodnot (Multivariate datové hodnoty)

Navigace v prostoru datových hodnot zahrnuje využití těchto datových hodnot jako mechanismu specifikace pohledu. Analogické operace panorámování (panning) a zoomování slouží ke změně zobrazovaných datových hodnot – panorámování posouvá startovní pozici rozsahu hodnot pro zobrazení, zatímco zoomování zvyšuje/snižuje velikost tohoto rozsahu.

Výběr v prostoru datových hodnot je podobný databázovým dotazům, kdy uživatel specifikuje rozsah datových hodnot pro jednu nebo více dimenzí. Toho lze dosáhnout přímou manipulací s daty, jako například brushing (interaktivní výběr) řízený daty (viz obrázek) nebo pomocí sliderů nebo jiných mechanismů pro specifikaci dotazů.

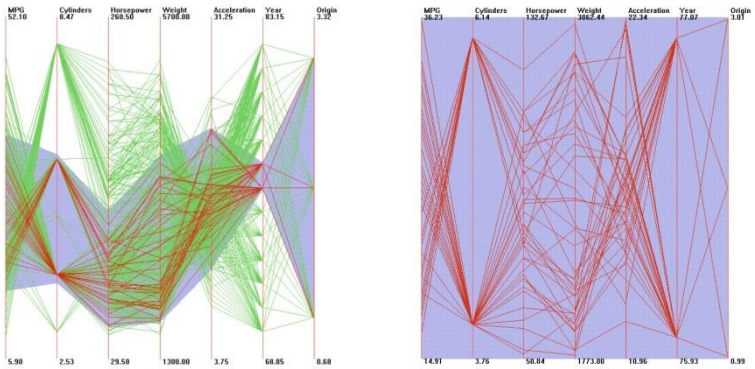


Prostor datových hodnot je pravděpodobně nejintuitivnějším prostorem, ve kterém se provádí filtrace. Při vizualizaci velkých datových množin je běžným postupem nejdříve redukovat data. Pro prostorová data je toto analogií k ořezávání dat spadajících mimo pohledový region. Pro neprostorová data tato redukce odpovídá odstranění některých záznamů, dimenzí nebo obojího.

Dimenze mohou být rovněž filtrovány, abychom mohli uživatelům prozkoumat podmnožinu dimenzí s podobnými vlastnostmi nebo vybrat reprezentanty pro jednotlivé klastry dimenzí.

Při distorzi v prostoru datových hodnot jsou datové hodnoty $(d_0, d_1, ..., d_n)$ transformovány pomocí funkce j: $(d'_0, d'_1, ..., d'_n) = j(d_0, d_1, ..., d_n)$. Tato transformace probíhá ještě před samotnou vizualizací. Ve skutečnosti může každá z dimenzí podléhat své vlastní transformační funkci $j_i : d'_i = j_i(d_i)$. V nejobecnějším případě může funkce j_i záviset na libovolném počtu dimenzí, ačkoliv v takovémto případě je možnost řízení filtrace uživatelem problematická.

Příkladem distorze v datovém prostoru je již dříve uvedený nástroj XmdvTool, kde je každá dimenze vybrané podmnožiny dat škálována takovým způsobem, aby podmnožina mohla být zobrazena v prostoru obrazovky (viz obrázek).



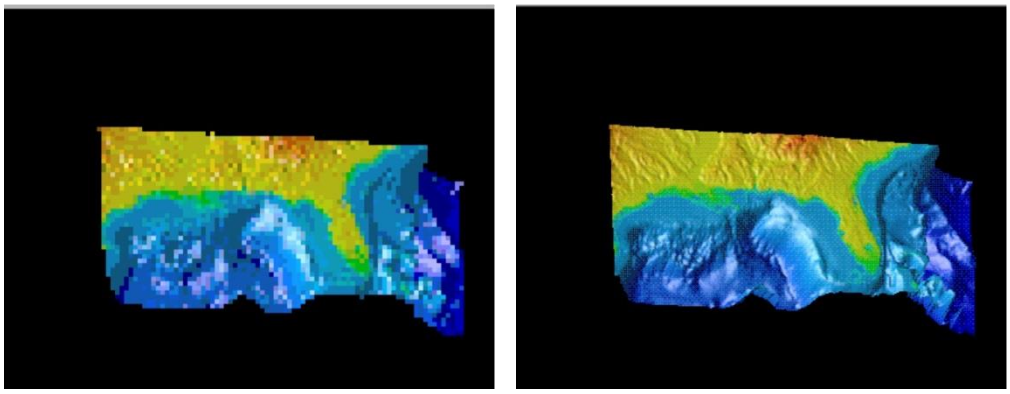
Na obrázku je vybrán N-dimenzionální hyperbox a následně je škálován přes všechny dimenze, čímž vyplníme jednotkovou hyperkostku.

Prostor datových struktur (Components of Data Organization)

Data mohou být strukturována mnoha způsoby, jako například do seznamů, tabulek, gridů, hierarchií a grafů. Pro každou z těchto struktur je možné vyvinout speciální interakční mechanismus, který určí, se kterými částmi struktury budeme manipulovat a jak se tato manipulace bude projevovat.

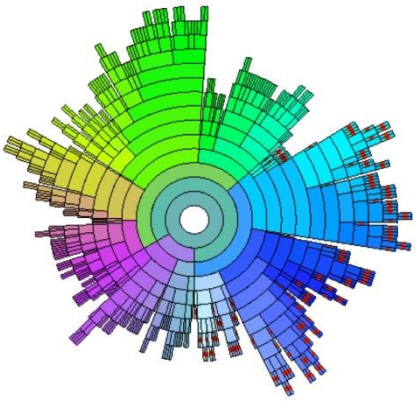
Navigace v prostoru datových struktur zahrnuje pohyb pohledové specifikace podél struktury, jako například při zobrazování sekvenčních skupin záznamů nebo při procházení hierarchickou strukturou (operace pohybu nahoru nebo dolů ve struktuře).

Jako příklad si uveďme daný obrázek, který ukazuje rozdíl mezi zoomováním v prostoru obrazovky (vlevo - zahrnující replikaci pixelů) a zoomováním v prostoru datové struktury (vpravo - zahrnující získání detailnějších dat v požadovaném rozlišení).



Výběr v prostoru datové struktury obecně zahrnuje zobrazení této struktury a umožnění uživateli identifikovat oblasti zájmu uvnitř struktury. Příkladem je **brushing** založený na struktuře, který umožňuje řídit výběr dat uložených v hierarchii klastrů. Příkladem interakce v tomto případě je zvýraznění dat, která spadají do určité větve stromu.

Dalším příkladem je vizualizační nástroj **InterRing**, který zobrazuje hierarchii radiálně a pomocí vyplňování prostoru. Tento nástroj umožňuje poloautomatický výběr uzlů, vzhledem k jejich hierarchické struktuře.



Obrázek ukazuje příklad, kde jsou automaticky vybrány koncové uzly. Tento výběr vzniknul dotazem na koncové uzly nad jejich společným předkem.

Pro redukování množství zobrazované informace se i zde často používá filtrace. Například pro vizualizace v čase je běžnou praxí definovat určitý rozsah na časové ose, na který chceme zaměřit pozornost. Prozkoumávání okolí v grafových vizualizacích zase často obsahuje filtraci uzlů a hran, které jsou dále než je uživatelem specifikovaný počet „skoků“ (míněno jako počet hran) od daného zájmového bodu. U hierarchických metod je filtrace založena právě na stupni hierarchie.

Distorze na hierarchických strukturách je velmi běžná zejména díky hustotě informace, která je odvozena z širokých nebo hlubokých hierarchických struktur. Několik vědců se zaměřilo na techniky založené na zobrazení hierarchie radiálně – např. Andrews a Heidegger, Stasko a Zhang, Yang.

Ve všech uvedených případech jsou data uchovávána ve struktuře namísto v samotných datových hodnotách nebo v mechanismu, jak jsou vizualizována.

Formalizace této procedury je komplikovanější než v ostatních případech. Většinu distorzí však můžeme definovat pomocí mapování vektoru (D, S), kde D jsou data a S je struktura uchovávající tato data, na vektor (D’, S’), kde transformace může modifikovat data, strukturu nebo obojí.

Většina běžně používaných struktur pro ukládání dat (gridy, hierarchie, sítě, …) obsahuje logické operace, které mohou uživatelé interaktivně na těchto strukturách provádět. Při implementaci je nutné učinit rozhodnutí o stupni automatizace použité operace, a zda budou interakce specifikovány přímo ve vizualizaci nebo v samostatném dialogovém okně. Při volbě automatizované techniky je nutné zvolit mezi důkladnými, ale časově náročnými technikami a rychlými, ale nepřesnými technikami. Nyní se pokusíme zaměřit na některé z těchto návrhových rozhodnutí.

Vezměme v úvahu uspořádání dimenzí pro vizualizaci multivariate dat. Plně manuální techniky mohou v tomto případě zahrnovat manipulaci s textovými vstupy v seznamu (pomocí operace posunu – nahoru a dolů, nebo pomocí techniky drag-and-drop). V případě paralelních souřadnic či matice bodových grafů můžeme manipulovat přímo s osami. U matice bodových grafů může posun jednoho prvku vyvolat změny v dalších řádcích a sloupcích, pokud chceme například zachovat symetrii podél hlavní diagonály.

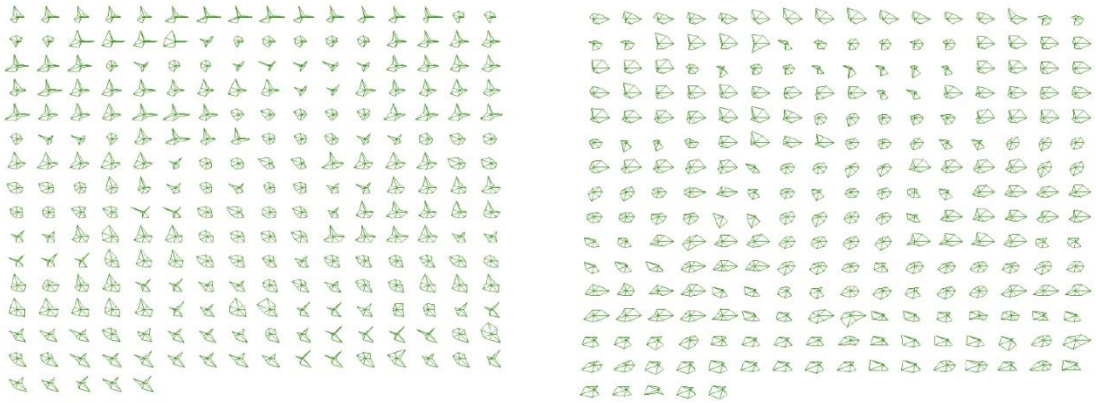
Naopak automatické přeskládání dimenzí potřebuje alespoň dvě základní rozhodnutí o návrhu: jakým způsobem měřit kvalitu uspořádání a jakou strategii zvolit pro hledání těchto kvalitních uspořádání. Pro tato rozhodnutí je možné zvolit různé metriky. Jedna z běžně používaných je součet korelačních koeficientů mezi každou dvojicí dimenzí.

Tento **korelační koeficient** mezi dvěma dimenzemi je definován následovně:

$$\rho_{X,Y} = \frac{\sum (x_i y_i - n\mu_X \mu_Y)}{(n-1)\sigma_X \sigma_Y}$$

kde *n* je počet datových bodů, X a Y jsou dvě dimenze, x_i a y_i jsou hodnoty pro i-tý datový bod, μ_X je střední hodnota v X a σ_X je standardní odchylka pro X.

Další přístup k měření kvality uspořádání může zahrnout jednoduchost interpretace. Různá uspořádání dimenzí mohou vést k zobrazení s většími či menšími vizuálními shluky nebo strukturami. Například je udáváno, že při použití glyfů reprezentujících datové body je snazší analyzovat jednoduché tvary namísto komplexních. Proto pokud jsme schopni změřit průměrnou nebo kumulativní složitost tvaru (např. počítáním prohlubní či vrcholů tvaru), můžeme tuto znalost využít pro porovnání vizuální složitosti různých uspořádání dimenzí.



Příklad je na obrázku, kdy levá část ukazuje původní uspořádání, zatímco pravá část zobrazuje výsledky po přeskládání dimenzí, kdy se snažíme redukovat konkávní oblasti glyfů a zvýšit procentuální podíl symetrických tvarů.

Pokud máme vybránu požadovanou kvalitu uspořádání, je dalším úkolem nalezení efektivní vyhledávací strategie pro nalezení těchto kvalitních uspořádání. Vyhodnocení všech možných uspořádání dimenzí je velmi náročné, pokud není počet dimenzí velmi malý (počet jednotlivých uspořádání je totiž N!). Tento počet může být dělen 2, pokud vezmeme v úvahu symetrická řešení, stále ale musíme vyhodnotit obrovský počet možností. Typickou strategií v těchto situacích je využití technik optimalizace. Problém uspořádání dimenzí je velmi podobný problému obchodního cestujícího, proto můžeme přímo použít algoritmy používané pro tento problém.

Jeden z nejjednodušších algoritmů pracuje následovně:

1. Vybereme dvě libovolné různé dimenze
2. Prohodíme jejich pozice a spočteme kvalitu uspořádání
3. Pokud je kvalita nižší než kvalita původního uspořádání, zrušíme prohození
4. Opakujeme kroky 1-3 fixně definovaným počtem iterací nebo dokud určitý počet provedených testů nevykazuje žádné zlepšení kvality

Tyto heuristické přístupy nejsou rozhodně optimální, nicméně často vedou k nalezení přijatelného řešení. Tyto přístupy se dají kombinovat i s manuálním přístupem, kdy uživatel může některá uspořádání zadat ručně na základě svých znalostí datové množiny a poté nechá systém automaticky dopočítat kvalitní uspořádání na jím modifikované množině.

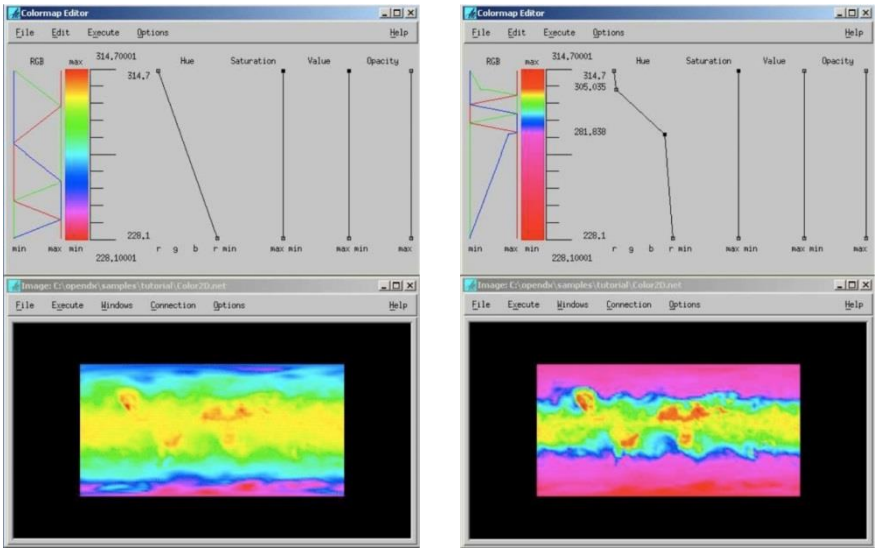
Prostor atributů (Components of Graphical Entities)

Navigace v prostoru atributů je velmi podobná navigaci v prostoru datových hodnot. Panorámování zahrnuje posun rozsahu hodnot zájmu, zatímco zoomování může být dosaženo buď škálováním atributů, nebo zvětšením rozsahu hodnot zájmu.

Stejně jako při výběru řízeném datovými hodnotami, výběr v prostoru atributů požaduje po uživateli určení podmnožiny daných atributů zájmu. Například máme-li dané znázornění barevné mapy, uživatel může vybrat jeden nebo více vstupů, které budou zvýrazněny. Podobně, pokud datové záznamy obsahují atributy jako například kvalita či neurčitost (uncertainty), pak vizuální reprezentace těchto atributů doprovázená vhodnými interakčními technikami umožňuje uživateli filtrovat nebo zvýrazňovat data na základě těchto atributů.

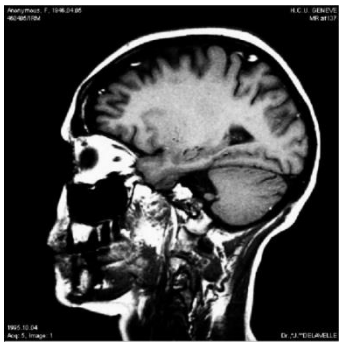
V prostoru atributů často dochází k přemapování – buď pomocí výběru různých rozsahů daného atributu, nebo výběrem různých atributů pro danou vstupní množinu. Například v systému GlyphMaker může uživatel vybrat mapování dané dimenze dat ze seznamu možných grafických atributů.

Mějme atribut A dané grafické entity. Můžeme provést distorzní transformaci aplikováním funkce k:a'=k(a). Můžeme předpokládat, že A může nabývat hodnot z rozsahu [a_0→a_1] nebo že A je specifikováno jako vektor. Například distorze barevné mapy může alokovat širší nebo naopak užší rozsah hodnot pro některé podmnožiny, čímž zvýšíme čitelnost jemných odchylek (viz obrázek). Obrázek ukazuje distorzi atributů ve formě modifikace barevné mapy, která byla vygenerována pomocí editoru barevných map v OpenDX systému. Barevná mapa je deformována takovým způsobem, aby byl větší rozsah hodnot přidělen středu rozsahu dat.



Tato forma distorze je často používána v analýze medicínských obrázků pro identifikaci oblastí zájmu.

Pravděpodobně nejpoužívanější interakce v prostoru atributů představují modifikace barevných atributů a atributů průhlednosti. Byla navržena řada technik, které se snaží o lepší využití barevného prostoru a lepší vnímání vlastností zobrazovaných dat. Například řízením kontrastu a jasu můžeme zvýraznit určitou oblast dat, čímž můžeme přilákat pozornost uživatele, který data analyzuje. Navíc mu usnadníme detekci vlastností, klasifikaci a měření dat. Příklad je na obrázku, kdy změnou kontrastu a jasu zvýrazňujeme jisté vlastnosti.



Interaktivní nástroje pro specifikování a modifikaci přenosové funkce (transfer function) se nejčastěji používají při renderování objemu (volume rendering) pro kontrolu barvy a průhlednosti, čímž dokážeme zvýraznit různé struktury uvnitř objemových dat.

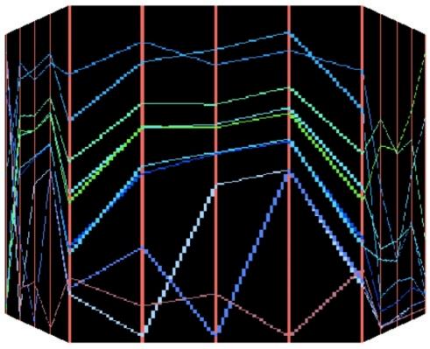
Prostor objektů (3D Surfaces)

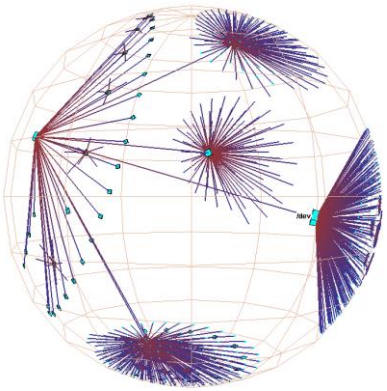
V tomto zobrazení jsou data mapována na geometrické objekty a tento objekt (nebo jeho projekce) následně podléhá interakcím a transformacím. Navigace v prostoru objektů se často skládá z pohybu kolem objektů a pozorování povrchů, na které byla data namapována. Systém podporující navigaci v prostoru objektů by měl umožňovat globální pohled na objekt a zároveň detailní pohledy. Ty mohou být určitým způsobem omezeny, aby byly uživateli rychle nabídnuty „dobré pohledy“ na objekt.

Výběr zahrnuje klikání na objekty zájmu nebo zvolení cílových objektů ze seznamu.

Typickým příkladem přemapování v prostoru objektu je změna objektu, na který jsou data mapována, jako například přepínání mapování geografických dat z roviny na kouli a naopak.

Příklady distorze v této formě interakce jsou tzv. perspektivní stěny (vlevo) a hyperbolické projekce (vpravo). Na tyto metody můžeme nahlížet jako na varianty metody založené na prostoru obrazovky, kdy objekt, na který jsou data promítnuta, zapouzdřuje distorzní funkci. Avšak po aplikaci mapování mohou povrchy podléhat dalším transformacím ve 3D, jako například rotaci, škálování a perspektivní distorzi.





Proces distorze v prostoru objektů můžeme reprezentovat jako sekvenci dvou funkcí:

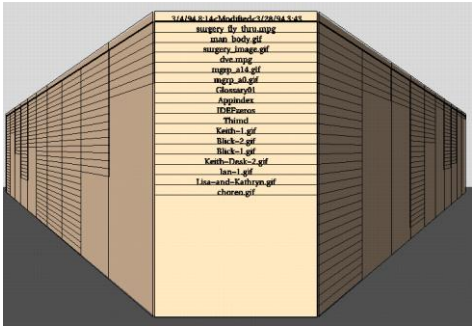
- První mapuje data (obecně parametrizována ve dvou dimenzích) na 3D strukturu:

$$(x, y, z) = g(a, b)$$

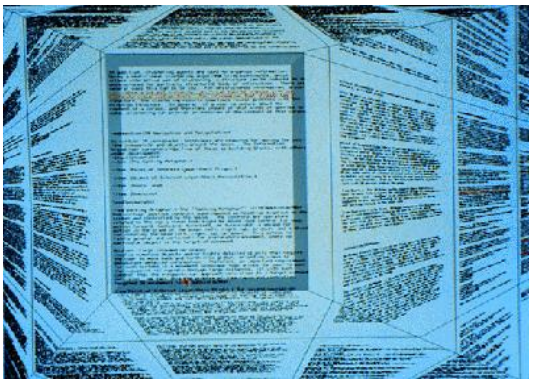
- Druhá tuto strukturu transformuje a promítá na obrazovku:

$$(i, j) = h(x, y, z)$$

Jednou z metod pro navigaci ve vizualizaci velkého počtu dokumentů a dat jsou takzvané **perspektivní stěny** (perspective walls). Tento přístup zobrazuje jeden panel pohledu na povrch objektu umístěný ortogonálně ke směru pohledu a ostatní panely jsou orientovány takovým způsobem, že mizí se vzdáleností a to způsobem definovaným pomocí perspektivní deformace.



Zjednodušená verze perspektivní stěny může být vytvořena tak, že přední stěna implementuje horizontální škálování určité části mapovaného 2D obrázku, zatímco sousední segmenty jsou podrobeny horizontálnímu a vertikálnímu škálování, které je úměrné jejich vzdálenosti ke hraně přední stěny. Navíc je na tyto segmenty aplikováno zkosení.



Tedy pokud jsou levá, střední a pravá sekce původního obrázku, který má být vykreslen na perspektivní stěnu, ohraničeny pomocí (x_0, x_{left}, x_{right}, x_1) a levý, střední a pravý panel výsledného obrázku je definován jako (X_0, X_{left}, X_{right}, X_1), pak aplikovaná transformace je následující:

– pro $x < x_{left}$:

$$x' = X_0 + (x - x_0) * \frac{(X_{left} - X_0)}{(x_{left} - x_0)}$$

$$y' = (X_{left} - x') + y\left(1 - \frac{(X_{left} - x')}{(X_{left} - X_0)}\right)$$

– pro $x_{left} \leq x < x_{right}$:

$$x' = X_{left} + (x - x_{left}) * \frac{(X_{right} - X_{left})}{(x_{right} - x_{left})}$$

$$y' = y$$

– pro $x \geq x_{right}$:

$$x' = X_{right} + (x - x_{right}) * \frac{(X_1 - X_{right})}{(x_1 - x_{right})}$$

$$y' = (x' - X_{right}) + y\left(1 - \frac{(x' - X_{right})}{(X_1 - X_{right})}\right)$$

Při použití perspektivní stěny s ní může uživatel interagovat sekvenčním procházením „stránek“ (dopředu i dozadu). Dále je možné využívat indexy pro přímý přístup (skákání) do oblasti zájmu – často je toto implementováno jako záložka vyčnívající nahoře na stránce na začátku každé sekce.

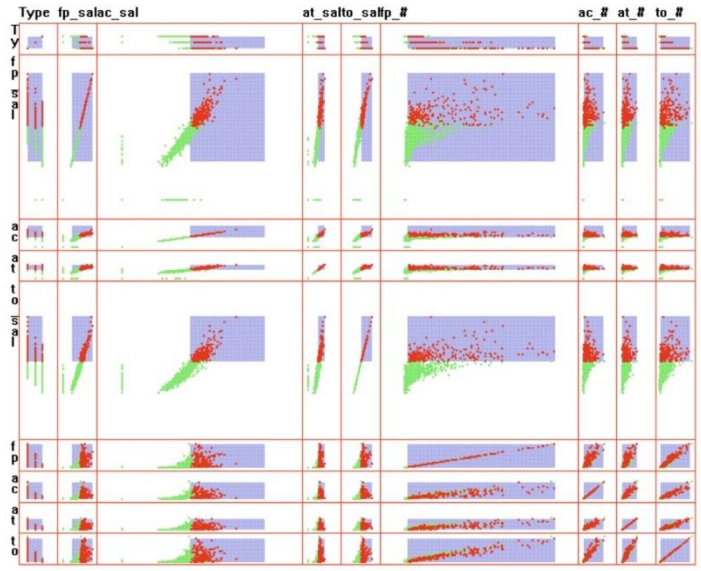
Prostor vizualizace struktur

Vizualizace se zaměřuje na strukturu, která je relativně nezávislá na hodnotách, atributech a struktuře dat. Například mřížka, do které je vykreslena matice bodových grafů, nebo osy zobrazované v různých typech vizualizací, jsou komponenty vizualizované struktury, na které se zaměřuje interakce.

Příkladem navigace při vizualizaci prostoru struktur je posun stránek v nástroji založeném na tabulkové vizualizaci nebo zoomování na jednotlivé grafy v matici bodových grafů.

Při výběru typické operace zahrnují výběr komponent, které mají být schovány, přesunuty nebo přeskupeny. Například uživatel může vybrat osu paralelních souřadnic a táhnout ji na jinou pozici, aby objevil různé vztahy mezi dimenzemi dat.

Příkladem distorze v tomto prostoru je tzv. table lens technika, která umožňuje uživateli transformovat řádky a/nebo sloupce tabulky za účelem poskytnutí násobného LOD. Tento proces aplikovaný na matici bodových grafů je znázorněn na obrázku.



Obrázek vygenerovaný nástrojem TableLens ukazuje matici bodových grafů, ve které jsou dvě buňky (a jejich odpovídající řádky a sloupce) zvětšeny na úkor ostatních buněk.

Některé z dříve popsaných technik je možné aplikovat i na prostor vizualizace struktur. Například technika rybího oka v prostoru obrazovky může být využita pro jakoukoliv vizualizaci struktury obsahující sekvence či gridy komponent. Stejná distorzní funkce může být v tomto případě použita pro nastavení rozestupů mezi osami paralelních souřadnic nebo pro nastavení velikosti buněk gridu při použití matice bodových grafů. Dalším příkladem je využití klastrování, filtrace či technik optimalizace (manuální i automatické) pro organizaci sad vizualizací na obrazovce jako jsou například objemové vizualizace obsahující stovky či tisíce různých sad s odlišným nastavením parametrů.

Klíčovým je ujištění, že uživatel je obeznámen se všemi operacemi, které může provádět nad strukturou vizualizace a využít konzistentní sadu ikon a označení, která je uživatel schopen rychle pochopit a používat. Další klíčovou funkcí je využití hladkých přechodů mezi vizualizacemi, což detailně probereme v následující sekci.

Animační transformace

V podstatě veškeré interakce v rámci vizualizačních systémů vedou ke změně zobrazeného obrázku. Některé z těchto změn jsou poměrně dramatické – například při otevření nové datové sady. Další změny mohou zachovat některé aspekty pohledu a jiné změnit. V případě, kdy si chce uživatel zachovat kontext zatímco svoji pozornost směřuje na měnící se oblast, je žádoucí poskytnout hladký přechod mezi výchozí a cílovou vizualizací. Například při rotaci s 3D objektem nebo datovou množinou je hladká změna orientace obecně mnohem lepší než skokový přechod do finální orientace. V některých případech k tomu stačí využít jednoduchou lineární interpolaci mezi počáteční a koncovou konfigurací. V jiných případech ale lineární interpolace nevede ke konstantní rychlosti změny (například při pohybu kamery po křivce). Navíc ve většině případů získáme mnohem přitažlivější výsledek za použití postupného zrychlení a zpomalení změny. V této sekci se tedy zaměříme na algoritmy, které musíme využít, pokud chceme dosáhnout této kontroly změn.

Prvním krokem algoritmů pro řízení změny v datech je získání uniformní parametrizace proměnné nebo proměnných, které chceme během animace řídit. Pro některé proměnné, jako například pozice podél rovné čáry nebo škálování, použijeme lineární interpolaci, která poskytne konzistentní změny v následujících časových krocích. Pro další proměnné, jako pozice podél zakřivené cesty, musíme problém přeformulovat zavedením nového parametru.

Předpokládejme, že původní parametr je funkce proměnné *t*, která nabývá hodnot od 0 do 1. Například můžeme použít kubický polynom pro spočtení (x, y) pozice pro různé hodnoty t: $x(t) = At^3 + Bt^2 + Ct + D$ (stejně pro y). Nyní můžeme vytvořit seznam pozic p_i pro $0 <= i <= n$, kde *n* je počet kroků mezi počáteční a koncovou pozicí. Dělení *t* na *n* stejných podintervalů dosáhneme jednoduchým přiřazením příslušných hodnot *t* do parametrické rovnice.

Poté můžeme odhadnout délku oblouku A součtem vzdáleností mezi po sobě jdoucími body:

$$A = \sum_{i=1}^{i=n} dist(p_{i-1}, p_i)$$

Je zřejmé, že čím je menší krok mezi sousedními body, tím je přesnější odhad délky oblouku.

Pro většinu křivek je však vzdálenost mezi sousedními body různá. Proto kdybychom použili vždy výše popsaný přístup, byla by rychlost výsledné animace proměnlivá.

Při výpočtu délky oblouku je rovněž užitečné pro každý bod p_i spočítat vzdálenost d_i od počátku křivky k tomuto bodu.

Pak spočteme funkci A(i), která reprezentuje procentuální poměr vzdálenosti, kterou bod urazí v i-tém časovém kroku.

Pro zjednodušení použijeme místo proměnné i proměnnou *t* ($0.0 <= t <= 1.0$). Dále definujeme nový parametr *s* = A(t). Výsledky uložíme do tabulky, takže pro každou hodnotu *t*

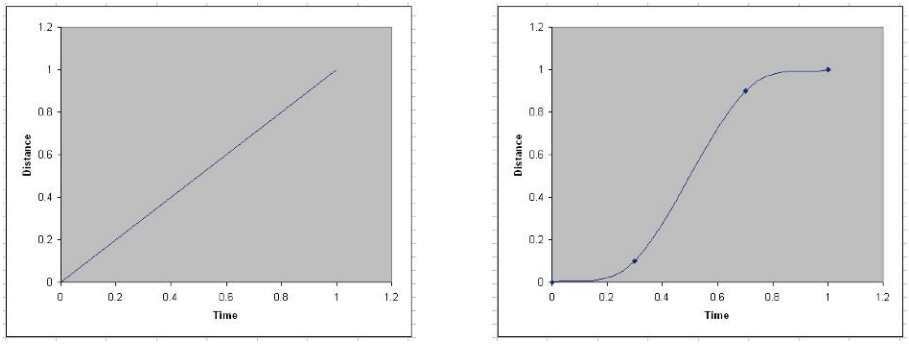
známe její odpovídající hodnotu *s* = A(t). Hodnotu *s* využijeme pro určení uniformní rychlosti (za využití lineární interpolace).

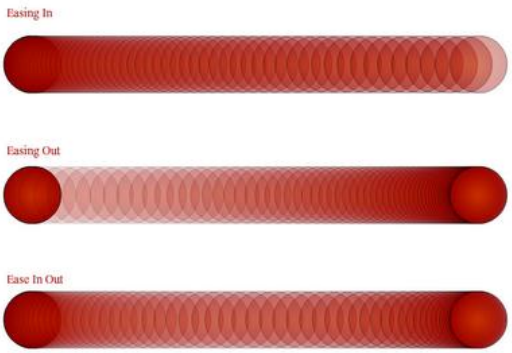
Výše uvedený postup je označován jako **reparametrizace**. Parametr *s* nyní slouží k ovládání rychlosti. Když vykreslíme parametr *s* v závislosti na čase, dostaneme rovnou čáru vedoucí z počátku (0.0, 0.0) do (1.0, 1.0). Jinými slovy, na začátku animace začínáme v původní pozici a když čas dosáhne hodnoty 1.0, jsme v koncové pozici. Rychlost jednoduše odpovídá sklonu této křivky. Co však s případy, kdy křivka není rovná, ale zakřivená? Pak části s malým sklonem představují nízkou rychlost a naopak velký sklon reprezentuje vysokou rychlost.

Protože počáteční a koncový bod jsou fixní, máme zajištěno, že skončíme, kde chceme.

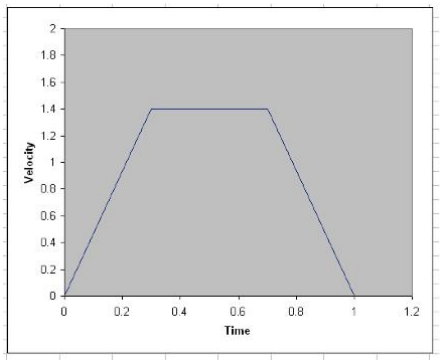
Pro určení animace mezi počátečním a koncovým bodem máme nekonečně mnoho možností pro nastavení. Můžeme dokonce na určitý časový okamžik animaci zastavit. Hlavním předpokladem je, že křivka se monotónně zvyšuje a nyní předpokládejme rovněž, že se nemůže vracet zpět.

Běžným typem křivky pro řízení animace je křivka, která reprezentuje postupné navyšování rychlosti na začátku animace z nuly na požadovanou rychlost a poté opět postupné snižování rychlosti až k nule na konci animace. Toto chování přesně reprezentuje sinová křivka. Klíčovým požadavkem je udržení hladké křivky.



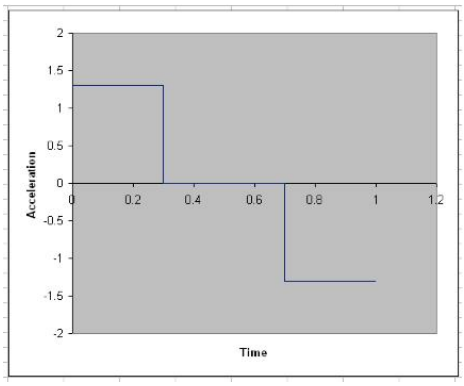


Někdy je jednodušší specifikovat pohyb pomocí **křivky rychlosti**. Rychlost je jednoduše první derivací křivky pozice. Křivka rychlosti pro případ postupného navyšování a snižování rychlosti se skládá ze segmentu, který se rovnoměrně zvyšuje od nuly, dále rovného segmentu a na konci z klesajícího segmentu končícího v nule (viz obrázek).



Prostor pod křivkou musí odpovídat hodnotě 1.0 – protože když je požadovaná rychlost příliš velká, musíme strávit více času při vzestupné a poté sestupné části.

Třetím typem křivky, která se občas používá pro kontrolu pohybu, je **akcelerační křivka**. Odpovídá druhé derivaci poziční křivky nebo analogicky první derivaci křivky rychlosti. Tvar křivky reprezentující postupně se zvyšující a poté snižující rychlost je složen ze tří horizontálních úsečkových segmentů – jeden nad osou (kladné zrychlení), jeden na ose (konstantní rychlost) a jeden pod osou (zpomalení).



Relativní pozice a délky čar nad a pod osou mohou být využity pro různé efekty a nemusí být nutně symetrické. Nicméně prostory definované vzestupnou a sestupnou fází se musí rovnat.

Poziční křivka a křivka rychlosti a akcelerace mohou být využity pro řízení jakéhokoliv atributu, který se v průběhu animace mění.