Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Demonstrace vybraných geometrických pojmů v OpenGL

2018

Zdeněk Bartal

Vedoucí práce: doc. RNDr. Michal Krupka, Ph.D.

Studijní obor: Informatika, prezenční forma

Bibliografické údaje

Autor:	Zdeněk Bartal
Název práce:	Demonstrace vybraných geometrických pojmů v OpenGL
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2018
Studijní obor:	Informatika, prezenční forma
Vedoucí práce:	doc. RNDr. Michal Krupka, Ph.D.
Počet stran:	36
Přílohy:	1 DVD
Jazyk práce:	český

Bibliograhic info

Author:	Zdeněk Bartal
Title:	Demonstration of selected geometrical concepts using OpenGL
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2018
Study field:	Computer Science, full-time form
Supervisor:	doc. RNDr. Michal Krupka, Ph.D.
Page count:	36
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Ve své bakalářské práci představují implementaci programu ve vývojovém prostředí LispWorks v jazyce Common Lisp, který je určen pro názorné předvedení vybraných geometrických pojmů za použití systému OpenGL.

Synopsis

Program introduced in this bachelor thesis developed using the LispWorks IDE and Common Lisp programming language is meant for visual demonstration of selected geometric concepts using the OpenGL system.

Klíčová slova: demonstrace počítačové grafiky; Common Lisp; OpenGL; pojmy z geometrie

Keywords: demonstration of computer graphics; Common Lisp; OpenGL; geometry concepts

Rád bych tímto poděkoval vedoucímu své bakalářské práce doc. RNDr. Michalu Krupkovi, Ph.D., za jeho vedení při tvorbě této práce. Dále bych také chtěl poděkovat své rodině a přátelům.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1 Motivace

Studium geometrie vyžaduje schopnost představit si probírané pojmy ve vícerozměrném prostoru a být schopen si uvědomit, jak dané rovnice a postupy fungují a jak dané parametry ovlivňují výsledný objekt. Neinteraktivní učební pomůcky jako například zobrazení dané scény nejsou schopny reagovat na uživatele a je pro něj tedy náročnější uvědomit si, jak daný geometrický příklad ve skutečnosti funguje. Může se i přímo stát, že si student danou situaci vyloží zcela mylně.

Nabízí se tedy vytvořit interaktivní program, který by byl schopen jednoduchým a názorným způsobem představit tyto geometrické pojmy. Hlavní výhodou oproti neinteraktivním pomůckám je možnost uživatele zadávat různé parametry pro konkrétní lekci a přímo sledovat jak tyto ovlivňují vykreslovanou scénu. Není ale možno vynechat ani výhody, které interaktivní program může potencionálně mít na prezentaci těchto problémů během výuky.

2 Geometrické pojmy

Zde jsou ve stručnosti uvedeny a představeny geometrické pojmy obsažené v programu. V této sekci vycházím zejména z učebního textu [1] a knihy [2].

2.1 Vektorové prostory

Uvažujeme množinu U. Na U existuje struktura vektorového prostoru, pokud je na ní dána binární operace $+$, která spolu s U tvoří komutativní grupu, a zobrazení $Mul_v : R \times U \to U$, kde pro $\forall r, s \in R, \forall u, v \in U$ platí:

1. $Mul_v(1, u) = u$
2. $Mul_v(r, Mul_v(s, u)) = Mul_v(rs, u)$
3. $Mul_v(r + s, u) = Mul_v(r, u) + Mul_v(s, u)$
4. $Mul_v(r, u + v) = Mul_v(r, u) + Mul_v(s, u)$

Takto daná struktura je vektorový prostor, $u \in U$ jsou vektory a $r \in R$ jsou skaláry.

Lineární kombinace vektorů $u_1, \dots u_k \in U$ a koeficientů $r^1, \dots, r^k \in R$ je vektor $Mul_v(r^1, u_1) + \dots + Mul_v(r^k, u_k)$.

Vektory $u_1, \dots u_k$ jsou lineárně závislé, pokud pro $r^1, \dots, r^k \in R$ platí, že $\exists r^n \neq 0, 1 <= n <= k$

Lineární obal $K \subseteq U$ je množina všech lineárních kombinací všech konečných n-tic vektorů z K pro všechna přirozená čísla n a značíme jej $\langle K \rangle$. K generuje U, pokud $\langle K \rangle = U$

Bází vektorového prostoru U je uspořádaná n-tice $\alpha = (u_1, \dots, u_n)$, pokud vektory $u_1, \dots, u_n$ nejsou lineárně závislé a množina $\{u_1, \dots, u_n\}$ generuje vektorový prostor U

2.2 Afinní prostory

Máme neprázdnou množinu A. Na A je dána struktura afinního prostoru, máme-li n-rozměrný vektorový prostor V a zobrazení $Add_v : A \times V \to A$, pro které platí:

1. $\forall a \in A, u, v \in V : Add_v(Add_v(a, u), v) = Add_v(a, u + v)$
2. $\forall a, b \in A, \exists! u \in V : Add_v(a, u) = b$

Množina A se strukturou afinního prostoru je n-rozměrný afinní prostor a její prvky jsou body.

Afinní kombinace bodů $a_1, \dots, a_m \in A$ a koeficientů $c^1, \dots, c^m$ je $b + Mul_v(c^1, a_1 - b) + \dots + Mul_v(c^m, a_m - b)$, kde $b \in A$. Pokud platí, že $\sum_{k=1}^{m} c^k = 1$ na bodu b nezáleží a můžeme jej vynechat.

Pro $a \in A$ a bázi $\alpha = (u_1, \ldots, u_n)$ vektorového prostoru V je dvojice $\beta = (\alpha, a)$ afinní bází A. Bod a je počátkem báze β.

Pro libovolné $b \in A$ je afinními souřadnicemi vzhledem k bázi β (n + 1)-tice:

$$b_\beta = \begin{bmatrix} (b-a)^1_\alpha \\ \vdots \\ (b-a)^n_\alpha \\ 1 \end{bmatrix}$$

3 OpenGL

V této sekci vycházím zejména z [3] a [4].

OpenGL(Open Graphics Library) je multiplatformní grafická knihovna nezávislá na konkrétním programovacím jazyce.
OpenGL bylo vytvořeno v roce 1992 firmou Silicon Graphics Inc. jako nástupce grafické knihovny IRIS GL.
Poslední verzí OpenGL je 4.2, ale v této bakalářské práci je využita pouze verze 2.1. Toto představovalo při tvorbě práce jisté problémy, nicméně verze 2.1 je nejnovější verze kterou implementace OpenGL v prostředí LispWorks podporuje.

3.1 Konvence názvů

OpenGL zavádí konvenci podle níž jsou pojmenované funkce a datové typy. Pro datové typy je tato konvence GL<název datového typu> Funkce jsou pojmenované následovně:

- Předpona závisející na knihovně v níž je funkce definována viz. tabulka 1
- Vlastní název funkce
- Počet parametrů, pokud není pro různé verze funkce stejný
- Typ parametrů viz. sloupec přípona v tabulce 2
- Pokud jsou parametry ukazatele na pole příslušného typu: písmeno v

Tabulka 1: OpenGL knihovny

Knihovna	Předpona
OpenGL	gl
OpenGL Utility Library	glu
OpenGL Utility Toolkit	glut

Tedy například máme funkci color, která způsobí renderování vykreslovaných objektů v příslušné barvě. Tato funkce má několik variant, například:

- glColor3d
  tři parametry typu GLdouble
- glColor4f
  čtyři parametry typu GLfloat
- glColor4ui
  čtyři parametry typu GLuint

3.2 Datové typy

Z důvodu co největší nezávislosti na platformě zavádí OpenGL vlastní datové typy uvedené v tabulce 2.

Tabulka 2: Datové typy OpenGL

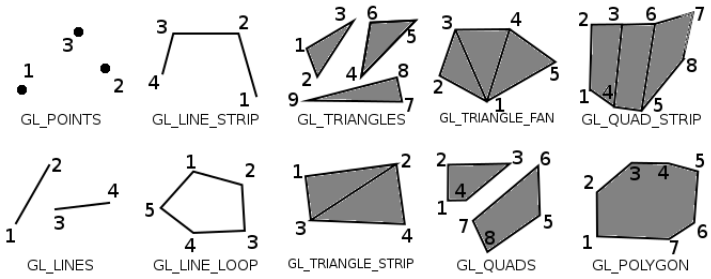
Datový typ	Interní reprezentace	přípona
GLbyte	8-bit integer	b
GLshort	16-bit integer	s
GLint, GLsizei	32-bit integer	i
GLfloat, GLclampf	32-bit floating point	f
GLdouble, GLclampd	64-bit floating point	d
GLubyte, GLboolean	8-bit unsigned integer	ub
GLushort	16-bit unsigned integer	us
GLuint, GLenum, GLbitfield	32-bit unsigned integer	ui

3.3 Grafická primitiva

Základem vykreslování v OpenGL jsou předdefinovaná grafická primitiva určující, jak budou zadané vertexy interpretovány.

Verze OpenGL 2.1, kterou tato práce využívá, definuje deset prvků zobrazených na obrázku 1.

Obrázek 1: OpenGL primitiva



Všechny příkazy pro vykreslování těchto objektů musí být umístěny mezi příkazy glBegin(primitivum) a glEnd(). Základním příkazem je zde glVertex určující vertex k vykreslení. Pokud není uveden správný počet vertexů pro dané primitivum, jsou nadbytečné vertexy ignorovány. Dále mohou být uvedeny příkazy, které modifikují stav vykreslovaného vertexu. Nejčastěji využívanými příkazy jsou:

- glColor
  udává barvu, která má být použita

- glTexCoord
  mapující dané souřadnice textury na daný vertex
- glNormal
  udávající normálu

4 LispWorks

LispWorks je multiplatformní vývojové prostředí pro jazyk Common Lisp dostupné pro následující platformy:

- Windows
- macOS
- Linux
- FreeBSD
- Solaris

4.1 Instalace

Bezplatná verze LispWorks Personal Edition je k dispozici pro stažení na webových stránkách výrobce http://www.lispworks.com/ v sekci downloads. Stažení této verze je podmíněno vyplněním webového formuláře. Stažený soubor o velikosti ~50MB je standardní instalátor pro zvolený operační systém. Při stahování souboru je také možno samostatně stáhnout dokumentaci v PDF formátu.

4.2 CAPI

Common Application Programming Interface je multiplatformní balíček pro vytváření uživatelských rozhraní zahrnutý v LispWorks a hlavní důvod pro zvolení LispWorks pro tvorbu této bakalářské práce i přes některá omezení LispWorks i CAPI. Pro použití CAPI v LispWorks stačí před příkaz uvést capi: či použít příkaz (use-package 'capi)

4.3 FLI

Foreign Language Interface je balíček pro interakci s kódem napsaným v jiném jazyce. V této bakalářské práci není přímo využíván a je zde uveden pouze pro referenci.

4.4 Implementace OpenGL v LispWorks

Implementace systému OpenGL v LispWorks je umístěna v adresáři `<LWinstalace>/lib/<verze>/examples/opengl/`. Pro využití této implementace je potřeba načíst a zkompilovat soubor `load.lisp`, např použitím příkazu `(load <cesta k load.lisp>)`. Poté stačí před příkaz uvést opengl: či použít příkaz (use-package 'opengl) Hlavní část implementace je tvořena FLI definicemi pro OpenGL funkcionalitu, kódem pro jednoduchou integraci s CAPI a dále pomocnými definicemi pro jednodušší práci s C funkcemi z prostředí jazyka Common Lisp, například práce s vektory ve stylu jazyka C.

4.4.1 Konvence názvů

Názvy OpenGLfunkcí a konstant jsou implementací transformované do konvence využívané jazykem Common Lisp.

Konstanty: název `OPENGL_CONSTANT` je transformován do podoby `*OPENGL-CONSTANT*`. Tedy například konstanta `GL_TEXTURE_2D` je nazvána `*GL-TEXTURE-2D*`

Funkce: zde je situace podobná, funkce `openglFunction` je implementací definována jako `opengl-function`. Tedy například `glBegin` je pojmenována `gl-begin`.

4.4.2 Struktura implementace

- examples
  Tento adresář obsahuje několik příkladů využití OpenGL v CAPI aplikaci. Plně funkční jsou tyto ukázky pouze na systému Windows.
- doc.txt
  Soubor obsahuje popis systému a instrukce pro jeho instalaci a použití.
- compile.lisp, defsys.lisp, host.lisp, load.lisp, loader.lisp a pkg.lisp
  Slouží k definici a načtení implementace
- constants.lisp
  Definice konstant systému OpenGL.
- types.lisp
  Definice datových typů systému OpenGL.
- vectors.lisp
  Obsahuje funkcionalitu pro vytváření a manipulaci s poli ve stylu jazyka C.
- fns.lisp
  Tento soubor obsahuje FLI definice pro všechny funkce základní OpenGL knihovny.
- ufns.lisp
  Zde jsou uvedeny FLI definice pro vybrané funkce z GLU(Graphics Library Utility).
- capi.lisp
  Obsahuje definici třídy `opengl:opengl-pane`, což je potomek třídy `capi:output-pane` a slouží pro snadnou integraci implementace OpenGL s CAPI.
- xfns.lisp, win32.lisp, gtk-lib.lisp, xm-lib.lisp, msw-lib.lisp a cocoa.lisp
  Jsou soubory specifické pro vybrané operační systémy, či prvky konkrétní instalace LispWorks.

5 Uživatelská dokumentace

5.1 Technické požadavky

Program je určen pro operační systémy Windows a macOS a grafické karty s podporou OpenGL verze 2.1. Dále je vyžadována funkční instalace LispWorks či LispWorks Personal Edition verze alespoň 6.1.1.

Je dále nutné poznamenat, že vývoj programu a převážná část testování probíhala na systému Windows a při spuštění v macOS trpí program drobnými chybami způsobenými drobnými změnami chování rozhraní CAPI na různých platformách.

Na dalších platformách podporovaných CAPI nebyl program testován.

5.2 Instalace

Program je distribuován jako sada zdrojových souborů v jazyce Common Lisp pro vývojové prostředí LispWorks a není jej tedy potřeba, či možné, instalovat. Spustitelná verze programu není zahrnuta, neboť LispWorks Personal Edition neumožňuje vytváření spustitelných souborů.

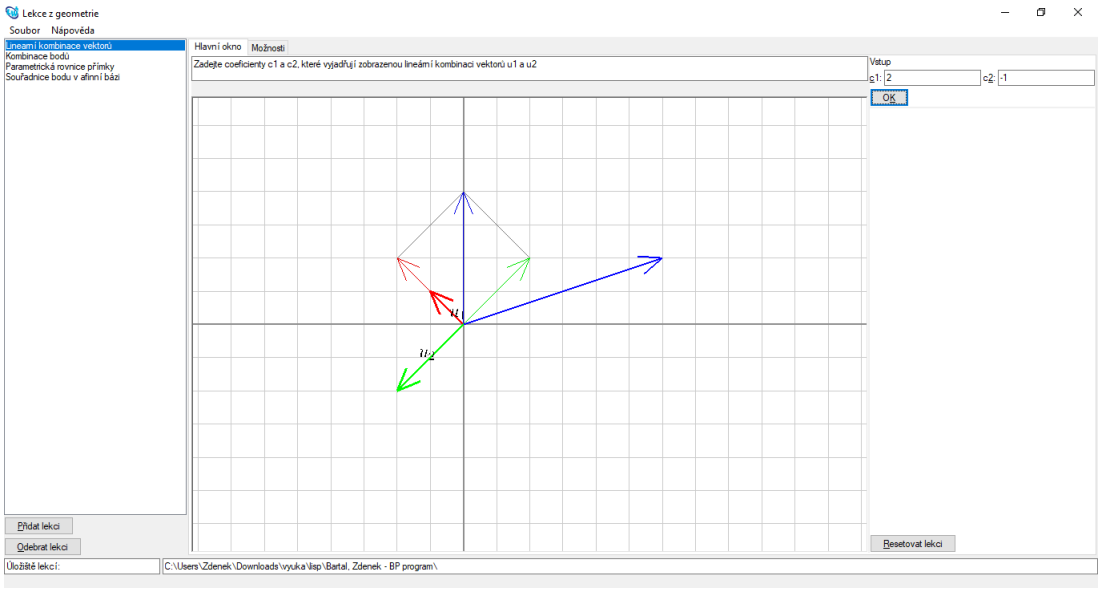
5.3 Spuštění

Pro spuštění programu je potřeba zkompilovat soubor `load.lisp`, což je možno učinit volbou File->Load v prostředí LispWorks. Dále je nutné zadat příkaz `(display-lesson-interface)`, což je možné udělat například v Listeneru.

5.4 Ovládání programu

Příkaz `(display-lesson-interface)` způsobí vytvoření okna programu, které je zobrazeno na obrázku 2.

Obrázek 2: Hlavní okno programu



V dolní části je zobrazena cesta k současnému úložišti lekcí.

5.4.1 Menu

Menu zahrnuje dvě položky.

- Soubor,
  Obsahuje položky pro smazání, aktualizování a načtení seznamu lekcí na disku. Změnu složky, která je využita pro toto úložiště. A tradičně ukončení programu.

- Nápověda
  Zde je umístěna možnost O programu, která zobrazí okno se stručnými informacemi o programu Lekce z geometrie.

5.4.2 Seznam lekcí

Tato část obsahuje seznam dostupných lekcí. Lekci je možno vybrat kliknutím na příslušnou položku či klávesami šipka nahoru/dolů, když je seznam aktivní. Pokud je při startu programu možno načíst seznam lekcí, je automaticky vybrána první lekce v seznamu.

V dolní části panelu jsou dále umístěna tlačítka pro přidání lekce z disku a pro odebrání lekce ze seznamu. Lekce je na disku reprezentována jako textový soubor a příponou lisp. Obsahem souboru je s-výraz, který se musí vyhodnotit na instanci třídy `lesson`, typicky funkce `make-instance` s příslušnými argumenty. Při přidávání souborů do programu je doporučeno tyto soubory nejdříve

otevřít v libovolném textovém editoru, a ujistit se, že neobsahují žádný nežádoucí kód. Dokumentace třídy `lesson` je dostupná v sekci 6.4.

5.4.3 Lekce

V tomto prostoru jsou dvě záložky:

- Hlavní okno
- Možnosti

Hlavní okno slouží pro zobrazení lekce a příjem uživatelského vstupu. Horní textové pole obsahuje stručné instrukce pro uživatele. Funkčnost pravého panelu závisí na vybraném režimu lekce. V textovém režimu slouží pro příjem uživatelského vstupu, zatímco v grafickém režimu zobrazuje vstupní argumenty lekce a argumenty generované na základě uživatelského vstupu. Nezávisle na režimu lekce také obsahuje tlačítko které resetuje lekci. Hlavní část okna zobrazuje samotnou lekci a v grafickém režimu také přijímá uživatelský vstup. Zobrazenou část je možno manipulovat podle následující tabulky 3.

Tabulka 3: Seznam příkazů

Příkaz	Efekt
$\leftarrow$	Posunutí scény doleva
$\rightarrow$	Posunutí scény doprava
$\uparrow$	Posunutí scény nahoru
$\downarrow$	Posunutí scény dolů
+ nebo scroll	Přiblížení kamery
- nebo scroll	Oddálení kamery

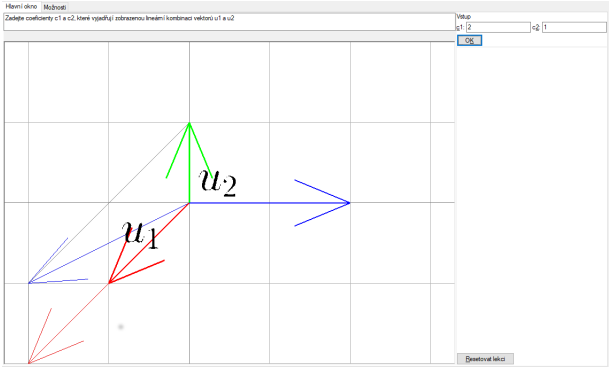
Záložka Možnosti obsahuje nastavení lekce, které závisí na samotné lekci a tlačítko pro restart lekce.

5.5 Dostupné lekce

5.5.1 Lineární kombinace vektorů

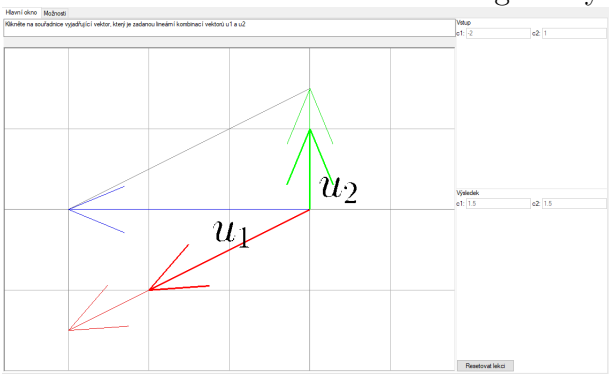
Tato lekce má dva možné režimy. V textovém (obr. 3) režimu má uživatel za úkol zadat koeficienty c_1 a c_2 takové, že zobrazený vektor je lineární kombinací $c_1 \times u_1 + c_2 \times u_2$.

Obrázek 3: Lineární kombinace vektorů - textový režim



V grafickém režimu zobrazeném na obrázku 4 je nutno kliknout na souřadnice, které představují lineární kombinaci $c_1 \times u_1 + c_2 \times u_2$.

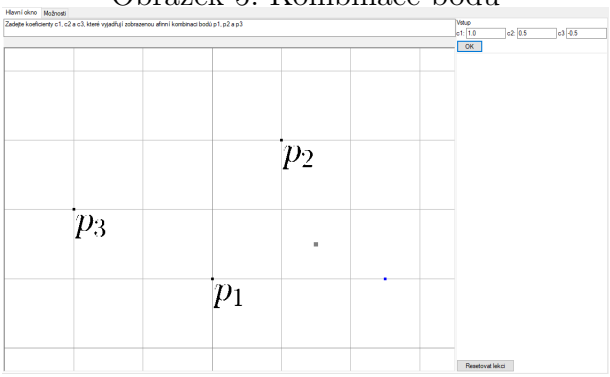
Obrázek 4: Lineární kombinace vektorů - grafický režim



5.5.2 Kombinace bodů

Tato lekce graficky znázorňuje lineární, afinní a konvexní kombinace dvou a tří bodů. Lekci zobrazuje obrázek 5.

Obrázek 5: Kombinace bodů



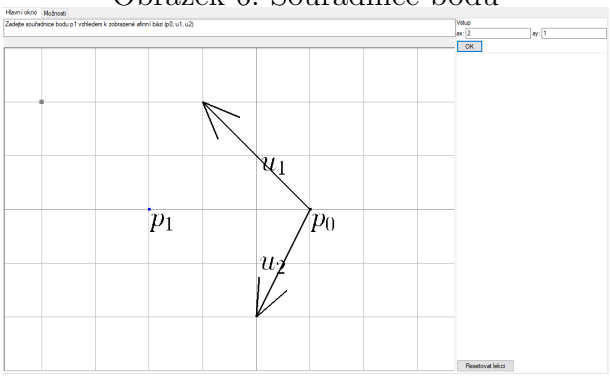
Uživatel musí zadat koeficienty c_1, c_2 a případně c_3 takové, že zobrazený bod je příslušnou kombinací $c_1 * p_1 + c_2 * p_2$ nebo $c_1 * p_1 + c_2 * p_2 + c_3 * p_3$.

Tato lekce má několik možností:

- Typ kombinace
  Kontroluje typ kombinace bodů.
- Počet bodů
  Udává, kolik bodů bude v lekci využito.
- Dodatečný krok souřadnicového systému lekce
  Určuje s jakým odstupem se mají souřadnice vykreslit.

5.5.3 Souřadnice bodu

Obrázek 6: Souřadnice bodu

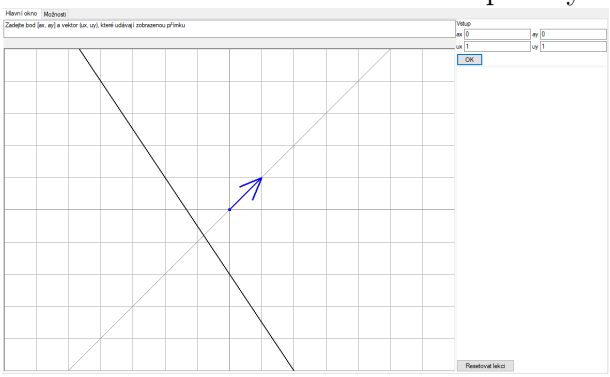


V této lekci je úkolem uživatele zadat souřadnice bodu p_1 vzhledem k zobrazené afinní bázi (p_0, u_1, u_2).

5.5.4 Parametrická rovnice přímky

Tato lekce graficky znázorňuje parametrickou rovnici přímky. Lekci zobrazuje obrázek 7. Cílem lekce je správné zadání bodu $[a_x, a_y]$ a vektoru (u_x, u_y) v afinním prostoru (R^2) tak, aby tyto vyjadřovaly zobrazenou přímku.

Obrázek 7: Parametrická rovnice přímky



6 Programátorská dokumentace

Testování převážně probíhalo na stroji s následujícími vlastnostmi:

Operační systém	Windows 10
Procesor	Intel(R) Celeron(R) 2955U @ 1.40GHz (2 CPUs)
Paměť	8GB RAM
Grafická karta	Intel(R) HD Graphics 4000

Program byl vytvořen pomocí objektového paradigma. Zdrojový kód je rozdělen do několika hlavních celků:

- Třída `manager`
  součást sloužící pro řízení a komunikaci ostatních celků programu
- CAPI definice
  definice tříd rozšiřujících funkcionalitu CAPI
- OpenGL rozhraní
  objektové zapouzdření OpenGL kódu
- Definice lekce
  základní struktura lekce
- Pomocné definice
  třída `event-passer` převzatá z učebních materiálů pro předmět Geometrie a kolekce pomocných funkcí, kódu pro načtení aplikace
- lu-solve
  balíček pro řešení lineárních funkcí, jehož kód byl převzat z http://nr-lisp.blogspot.com/

Nyní následuje popis základních celků aplikace.

6.1 Třída `manager`

Toto je prvek jehož funkcí je koordinovat komunikaci ostatních celků programu. Z převážné části jde o předávání uživatelského vstupu konkrétní lekci a naopak.

6.2 CAPI definice

Touto částí jsou definice tříd rozšiřujících funkcionalitu CAPI a pomocné funkce pro vytváření CAPI objektů.

6.2.1 Pomocné definice

Pomocné třídy `menu-item-event-passer`, `list-panel-event-passer`, `tab-layout-event-passer`, `radio-button-panel-event-passer` a `push-button-event-sender` pouze spojují funkcionalitu CAPI komponent a třídy `event-passer` pro posílání zpráv objektům.

Třída `lesson-interface` obsahuje hlavní uživatelské rozhraní.

Funkce `create-lesson-interface` a `create-opengl-scene` slouží pro vytvoření daných CAPI objektů.

6.2.2 Třída `input-field`

Tato třída reprezentuje textové vstupní pole, které ověřuje, zda-li uživatelem zadaný text splňuje podmínku vyjádřenou danou funkcí. Pokud ano, a byl-li text změněn, je vyslána zpráva `input-changed` s argumentem, kterým je zadaný text. Jinak je text při ztrátě zaměření nahrazen hodnotou posledního přijatelného textu. Je možné tuto třídu resetovat do počátečního stavu.

6.2.3 Třída `input-display`

Toto je modifikace CAPI panelu sloužící pro organizaci dvourozměrné kolekce prvků `input-field`. Při odeslání daného vstupu uživatelem je daný vstup ověřen danou validační funkcí a pokud není validní, je zobrazena chybová hláška. Třídu je možno resetovat a zničit či změnit danou konfiguraci.

6.2.4 Třída `opengl-scene`

Rozhraní mezi CAPI prvky a OpenGL kódem, tato třída je schopna fungovat jako samostatný program. Podporovanou funkcionalitou je přidání/odebrání grafických objektů, převod mezi souřadnicemi okna a scény a naopak, nastavení zobrazené části a limitovaná manipulace integrovaného souřadnicového systému.

6.3 OpenGL rozhraní

Toto je zapouzdření OpenGL funkcionality do objektového systému.

6.3.1 Třída `vector4` a její variace

Toto je třída vyjadřující čtyřprvkový vektor, která je využita jak pro reprezentaci pozice, rotace a měřítka vertexu, tak i pro vyjádření barvy (třída `color`) a kvaternionu (třída `quaternion`).

6.3.2 Třída `object-transform`

Základní grafický objekt, tato třída je vyjádřením pozice vzhledem k nadřazené instanci `object-transform`, případně k prostoru OpenGL. Je možné měnit

pozici a měřítko instance. Třída také obsahuje kód pro rotaci, tento není nicméně v programu využit a není zcela ověřený.

Tato třída je předkem všech následujících tříd uvedených v této sekci.

6.3.3 Třída `textured-label`

Třída reprezentující dvourozměrnou texturu. V následující tabulce je uveden seznam přípustných hodnot, které se namapují na příslušnou texturu.

Hodnota	Textura
:u1	u_1
:u2	u_2
:p0	p_0
:p1	p_1
:p2	p_2
:p3	p_3

Pro správné vykreslování textur je nutné zavolat funkci `initialize-labels` před jejich použitím a pouze jednou během života daného OpenGL kontextu.

6.3.4 Třída `abstract-object`

Toto je společný předek pro speciální objekty a základní grafická primitiva(třída `graphic-primitive`).

6.3.5 Třída `grid-object`

Je speciální objekt definující až trojrozměrnou mřížku, která může být vykreslena v libovolném objemu OpenGL prostoru. Nicméně na hlavním testovacím stroji byla plynulost vykreslování silně ovlivněna při využití dvojrozměrné mřížky již při vykreslování více než několika desítek linek. Je možné nastavit hlavní rozestup mezi vykreslovanými linkami a rozestup sekundárních linek mezi hlavními linkami, nicméně musí platit následující rovnice $1/x = y, y \in N^+$ kde x je rozestup sekundárních linek.

6.3.6 Třída `graphic-primitive`

Jednoduchý grafický objekt, který interpretuje zadané vertexy jako dané OpenGL primitivum.

6.3.7 Třída `compound-object`

Třída pro vykreslení listu jednoduchých grafických objektů.

6.3.8 Třída `coordinate-system`

Třída vykreslující až trojrozměrnou mřížku v daném prostoru.

6.4 Třída `lesson`

Tato třída je zodpovědná za definici struktury lekce a řízení společných prvků konkrétních instancí lekce.

Vlastnost lekce `lesson-options`, která je dále zmiňována je hash-table, která udává vlastnosti lekce. Jedinou vždy povinnou vlastností je `:mode`, která může nabývat hodnot buď `:text` nebo `:graphic`. Pokud není inicializována funkcí `f-initialize-lesson-options` je nastavena na hodnotu `:text`. Udává, zda je mód lekce textový nebo grafický.

Následující vlastnosti jsou využity a měly by být nastaveny pouze pokud lekce může být v grafickém módu.

- `:num-of-points`
  Počet uživatelsky zadaných bodů, které má lekce přijímat.

- `:point-precision`
  Na jakou hodnotu mají být jednotlivé prvky uživatelsky zadaných bodů zaokrouhleny.

Speciální vlastností lekce je `:scene-grid-additional-step`, která jediná nevyvolává restart lekce pokud je změněna uživatelským vstupem. Určuje, zda-li mají být vykresleny sekundární linky v souřadnicovém systému lekce. Pro hodnotu této vlastnosti by měla platit následující rovnice $1/x = y, y \in N^+$ kde x je `:scene-grid-additional-step`. Hodnota této vlastnosti menší než 0.2 způsobila značné zpomalení vykreslování lekce při standardním měřítku na hlavním testovacím stroji.

Dále mohou být vlastnosti lekce libovolné, nicméně kromě funkce `f-initialize-lesson-options` smí být měněny pouze zprávami zasílanými prvky listu `gui-option-changer`.

Dva okomentované příklady objektu lekce jsou umístěny ve složce `../examples/`. Systém načítání/ukládání lekcí nepodporuje komentáře, proto jimi nejsou základní lekce opatřeny.

6.4.1 Inicializační argumenty

V následujícím seznamu jsou využity vlastní názvy argumentů, nicméně při vytváření objektu lekce musí být uvedeny jako `:<název argumentu>`. Tedy například `f-get-boundary` by bylo uvedeno jako `:f-get-boundary`.

`f-initialize-lesson-options`

Argumenty:

`lesson-options`

Inicializuje `lesson-options` na výchozí hodnoty. Toto je jediná uživatelská funkce, která by měla měnit hodnoty `lesson-options`.

`f-generate-concise-description`

Argumenty:

`lesson-options`

Vrací textový řetězec, který poskytuje uživateli stručné instrukce pro vyřešení lekce.

`f-get-boundary`

Argumenty:

`lesson-options`
`solution-object-args`
`input-object-args`
`base-object-args`

Vrací list ve formátu `((`x_{min} x_{max}`) (`y_{min} y_{max}`))` udávající prostor, který má být vykreslen.

`f-generate-input-args-format`

Argumenty:

`lesson-options`

Vrací list udávající formát uživatelského vstupu s prvky:

- `global-validation-function`
  Funkce, jejímž argumentem je list listů formátovaných podle argumentů `format-type` a `format` a která vrací `(values successp error)`:
  - `successp`
    udává, zda-li je argument přijatelný
  - `error`
    textový řetěz udávající, proč není vstup přijatelný nebo `nil`.
- `local-validation-function`
  Funkce, jejímž argumentem je textový řetězec `input`.
  Vrací `(values successp input-object)`:
  - `successp`
    udává, zda-li je argument přijatelný
  - `input-object`
    je objekt daný `input` nebo `nil`

- `default-inputs`
  List listů formátovaných podle argumentů `format-type` a `format`, který určuje úvodní vstup.
- `format-type`
  Buď `:dimensions` nebo `:list`
- `format`
  List listů textových řetězců pokud je `format-type :list`. List `(x y)` kde x určuje počet sloupců a y počet řádků.

`f-generate-base-object-args`

Argumenty:

`lesson-options`

Vrací prvek `base-object-args` reprezentující argumenty pro vytvoření základního objektu lekce.

`f-generate-base-object`

Argumenty:

`lesson-options`
`base-object-args`

Vrací list grafických prvků, které budou zobrazeny uživateli.

`f-generate-combination-args`

Argumenty:

`lesson-options`
`base-object-args`

Toto je funkce vracející objekt `combination-args`, který bude použit pro vytvoření objektu řešení funkcí `f-generate-solution-object-args`.

`f-generate-solution-object-args`

Argumenty:

`lesson-options`
`combination-args`
Jedno z následujících:
- `combination-args`
- Uživatelský vstup zpracovaný `f-process-input`

– Návratová hodnota funkce `f-process-graphic-input`

`base-object-args`

Vrací objekt `solution-object-args` reprezentující řešení lekce.

`f-process-graphic-input`

Argumenty:

`lesson-options`

`direction`
`:in` nebo `:out`

`base-object-args`

`input-points`
list `:num-of-points` bodů jež byly zadány uživatelem a jejichž prvky jsou zaokrouhleny na `:point-precision` nebo návratová hodnota této funkce

Převádí uživatelsky předaný list bodů na formát shodný s návratovou hodnotou `f-generate-combination-args` a naopak.

`f-process-input`

Argumenty:

`lesson-options`

`direction`
`:in` nebo `:out`

`input-points`
uživatelsky předaný vstup, nebo návratová hodnota této funkce

Převádí uživatelsky předaný vstup ve formátu definovaném `f-generate-input-args-format` na formát shodný s návratovou hodnotou `f-generate-combination-args` a naopak.

`f-generate-solution-object`

Argumenty:

`lesson-options`

`solution-object-args`

Tato funkce je volána pouze pokud je vlastnost `:mode` lekce rovna `:graphic`. Vrací list grafických prvků, které budou zobrazeny uživateli.

`f-solution-correct-p`

Argumenty:

`lesson-options`

`solution-object-args`

`result`
návratová hodnota `f-generate-solution-object-args` volané se zpracovaným uživatelským vstupem a `base-object-args`

`base-object-args`

Predikátová funkce určující, je-li dané řešení správné.

`f-generate-input-object-args`

Argumenty:

`lesson-options`

`solution`
zpracovaný uživatelský vstup

`result`
návratová hodnota `f-generate-solution-object-args` s argumenty `solution` a `base-object-args`

`base-object-args`

Tato funkce vrací argumenty `input-object-args` pro vytvoření objektu udávajícího uživatelský vstup.

`f-generate-input-object`

Argumenty:

`lesson-options`

`input-object-args`

Vrací list grafických prvků, které budou zobrazeny uživateli.

`gui-option-changer`

List CAPI objektů, které umožňují uživateli měnit vlastnosti lekce. Každý z prvků by měl posílat zprávu `lesson-options-changed` s argumenty prvek a list s prvky vlastnost lekce a nová hodnota této vlastnosti.

`name`

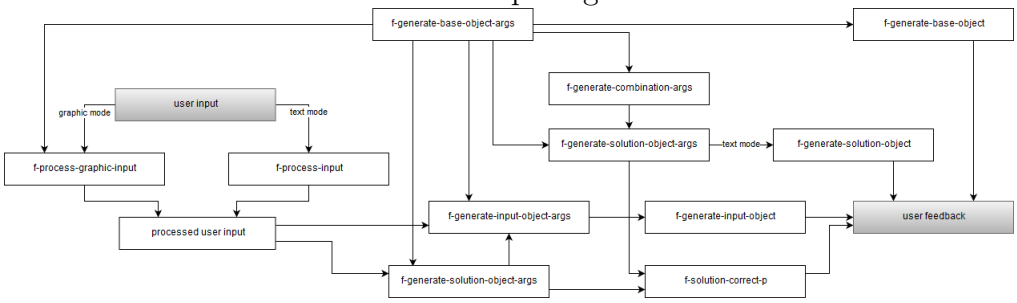
Jméno lekce

6.4.2 Popis činnosti lekce

Při vytvoření instance lekce jsou volány funkce `f-initialize-lesson-options`, `f-generate-concise-description` a `f-get-boundary`.

Hlavní cyklus činnosti lekce vyjadřuje obrázek 8.

Obrázek 8: Princip fungování lekce



Hlavní část je provedena vždy, když uživatel vyžádá vytvoření nového příkladu. Část plynoucí od user-input je provedena vždy, když uživatel zadá nový vstup. Při změně jakékoli vlastnosti lekce kromě `:scene-grid-additional-step` uživatelem je lekce resetována.

6.5 Pomocné definice

Zde jsou uvedeny uvedeny prvky programu, které nejsou částí žádného z výše zmíněných celků.

6.5.1 Třída `event-passer`

Tato třída je převzata z učebního materiálu pro předmět Geometrie a slouží pro posílání zpráv vzhůru po hierarchii objektů.

6.5.2 Pomocné funkce

Následující vybrané pomocné funkce:

- `(~ (tolerance &rest numbers))`
  predikátová funkce určující zda-li je mezi čísly numbers nejvýše tolerance rozdílu
- `(normalize-numbers (&rest numbers))`
  vrací list normalizovaných čísel
- `(round-to (number round-to))`
  vrací číslo number zaokrouhlené na nejbližší násobek round-to

- `(try-parse-object (text type-of-object))`
  Funkce určující, zda-li textový řetězec `text` reprezentuje objekt typu `type-of-object`. Vrací `(values sucessp input-object)`.
  - `successp`
    udává, zda-li je argument přijatelný
  - `input-object`
    je objekt daný `text` nebo nil.
- `(get-rect-boundary (points &key min-offset max-offset))`
  vrací body reprezentující nejmenší obdélník obsahující body points s daným offsetem
- `(clamp-to (number min-range max-range))`
  vrací číslo number, pokud je mezi min-range a max-range, jinak min-range nebo max-range
- `(transpose (list))`
  vrací transponovaný list. List musí být list listů, které mají stejnou délku
- `(list-apply (f list))`
  vrací list výsledků aplikace funkce f na prvky transponovaného listu list
- `(generate-3nc-points (&key c-x c-y init-point))`
  vrací list tři bodů, které nejsou kolineární a leží v objemu definovaném parametry c-x, c-y
- `(other-fixnum-in-range (number constraint))`
  Vrací celé číslo, které není number a je v rozsahu constraint
- `(fixnum-in-range (constraint))`
  Vrací celé číslo které je v rozsahu constraint
- `(number-in-range (constraint step))`
  Vrací číslo které je v rozsahu constraint
- `(collinear-p (pointA pointB pointC))`
  predikátová funkce určující, tda-li jsou dané tři body kolineární
- `(line-length (pointA pointB))`
  vrací délku úsečky mezi body pointA pointB
- `(vector-length (&rest items))`
  vrací délku zadaného vektoru
- `(line-angle (x1 y1 x2 y2 &key length))`
  vrací úhel mezi zadanou úsečkou a osou x

- `(ignore-input (input))`
  vrací `(values t nil)`
- `(return-first (&rest args))`
  vrací první argument
- `(return-second (&rest args))`
  vrací druhý argument
- `(return-third (&rest args))`
  vrací třetí argument

Závěr

Program Lekce z geometrie předvádí jak základní možnosti použití systému OpenGL, tak i vybrané geometrické pojmy jednoduchým a názorným způsobem. Jednou z hlavních překážek při tvorbě této bakalářské práce byla především nízká podpora OpenGL v jazyce Common Lisp v prostředí LispWorks. I přes tyto i jiné problémy jsem se však dostal až k závěru této práce, a věřím, že se mi podařilo splnit všechny zadané cíle.

Conclusions

Program Geometry lessons demonstrates basic usage of the OpenGL system and selected geometry concepts in a simple and illustrative way. One of the main obstacles in the making of my bachelor thesis was the low amount of support for the OpenGL system in the Common Lisp programming language on the LispWorks development platform. Despite these and other obstacles I made my way to the conclusion of this thesis and I hope that I succeeded in fulfilling all of the assigned objectives.

A Obsah přiloženého DVD

V této sekci je uveden stručný popis obsahu přiloženého DVD, tj. jeho závazné adresářové struktury, důležitých souborů apod.

doc/
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

src/
: Kompletní zdrojové texty programu Lekce z geometrie se všemi potřebnými (příp. převzatými) zdrojovými texty, knihovnami a dalšími soubory potřebnými pro bezproblémové spuštění programu.

readme.txt
: Instrukce pro instalaci a spuštění programu Lekce z geometrie, včetně všech požadavků pro jeho bezproblémový provoz.

Navíc CD/DVD obsahuje:

examples/
: Dva komentované příklady souborů lekcí.

U veškerých cizích převzatých materiálů obsažených na DVD jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Pro všechny použité (a citované) materiály, u kterých toto není splněno a nejsou tak obsaženy na DVD, je uveden jejich zdroj (např. webová adresa) v bibliografii nebo textu práce nebo v souboru `readme.txt`.

Literatura

[1] KRUPKA, Michal. *Geometrie pro Informatiky*. 2018. 131 s.

[2] BICAN, Ladislav. *Lineární algebra a geometrie*. Druhá. 2009. 304 s. ISBN 978-80-200-1707-9.

[3] OPENGL ARCHITECTURE REVIEW BOARD, et al.; SHREINER, Dave. *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.4*. Fourth. 2004. 784 s. ISBN 032117383X.

[4] OPENGL ARCHITECTURE REVIEW BOARD, et al.; SHREINER, Dave; WOO, Mason; NEIDER, Jackie; DAVIS, Tom. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2*. Fifth. 2005. 896 s. ISBN 0321335732.