Dokumentace k projektu pro předměty IZP a IUS

Iterační výpočty

Projekt č. 2

1. prosince 2014

Autor: Jan Ondruch, xondru14@fit.stud.vutbr.cz

Fakulta Informačních Technologií

Vysoké učení technické v Brně

Obsah

1 Úvod

Funkci tangens si je možno vyjádřit vícero způsoby, přičemž u některých z nich se jedná o zajímavé algoritmické problémy. Jsou nimi například výpočet pomocí Taylorova polynomu nebo pomocí metody zřetězených zlomků. Oba tyto postupy jsou z implementačního hlediska podobné a byly v tomto projektu využity k výpočtu vzdálenosti a výšky objektu měřeného pomocí měřícího zařízení.

Tento dokument popisuje postup a výpočet 2 implementací funkce tangens a jejich následné využití k měření vzdálenosti a výšky předmětu. Navržená aplikace se spouští z příkazového řádku, kdy na standardním vstupu přečte a vyhodnotí argumenty zadané uživatelem a následně vrátí na standardní výstup vypočítané hodnoty, které odpovídají příkazům uživatele. Může se jednat buď o porovnání vypočítaných úhlů pomocí odlišných funkcí tangens, nebo o vzdálenosti, které výpočty goniometrické funkce využily.

V dokumentu se vyskytuje více částí, které se např. zabývají analýzou problémů, výpočtem jednotlivých implementací funkcí tangens a řešením problému samotného.

2 Analýza problému

2.1 Zadání problému

Cílem tohoto projektu bylo implementovat výpočet pro měření vzdálenosti a výšky měřeného objektu pomocí údajů ze senzorů natočení měřicího přístroje. Tyto údaje uživatel zadává jako argumenty do příkazové řádky. Pro úspěšný výpočet vzdálenosti objektu musí být zadán úhel α, pro výpočet výšky objektu i úhel β. Navíc je implicitně zadána výška měřicího přístroje, kterou je možno modifikovat.

2.2 Implementační detaily

Projekt bylo nutno vyřešit jen pomocí základních matematických operací *+, -, *, /*. Funkce z matematické knihovny nebyly povoleny vyjímaje funkce *tan*, kterou bylo možno použít jen pro srovnání výpočtů, funkcí *isnan* a *isinf* a konstanty *NAN* a *INF*.

2.3 Výpočty úhlů

Při implementace výpočtu funkce tangens jak pomocí Taylorova polynomu, tak i pomocí zřetězených zlomků využijeme toho, že obě metody lze poměrně jednoduše zapsat rekurentním vztahem. Z toho však také vyplývá fakt, že musí výpočty po určitém počtu kroků skončit. Jelikož víme, že obě naše funkce konvergují – výsledek se zpřesňuje, zbývá nám jen určit, jak moc chceme býti přesní. Ze zadání projektu se jedná o 10 desetinných míst. Znamená to tedy, že výpočty nebudou, ani nemohou být zcela přesné a jedná se tedy pouze o aproximaci. Důvodem nepřesnosti je konečný počet iterací a limity počítače, kdy s přibývajícím počtem iterací narůstá chybovost exaktnosti typu double.

Odchylky, které při výpočtech vznikají, změříme porovnáním námi vypočítaných hodnot a hodnot funkce tangens z matematické knihovny math.h.

3 Návrh řešení problému

3.1 Implementace tangens

3.1.1 Taylorův polynom

$$\tan(x) = x + \frac{x^3}{3} + \frac{2x^5}{15} + \frac{17x^7}{315} + \frac{62x^9}{2835} + \cdots$$

x – úhel α nebo β zadaný uživatelem v radiánech

Ze vzorce je patrný vztah dvou sousedících členů (za člen považujeme jeden zlomek). Liší se tím, že je čitatel násoben x^2 a dále koeficientem, stejně jako jmenovatel.

První člen posloupnosti: x

Rekurentní vztah pro následující člen: $a_n = \frac{čitatel\ (n)}{jmenovatel\ (n)} * x^{n+2}$

Výpočet těchto koeficientů patří mezi složitější algoritmy, kterými se zde zabývat nebudeme, a tak byly tyto hodnoty zadány explicitně jako konstanty.

Zde jsou hodnoty prvních 13 členů těchto posloupností, které bylo možno v programu použít.

Hodnoty čitatele

1.	1
2.	1
3.	2
4.	17
5.	62
6.	1382
7.	21844
8.	929569
9.	6404582
10.	443861162
11.	18888466084
12.	113927491862
13.	58870668456604

Hodnoty jmenovatele

1.	1
2.	3
3.	15
4.	315
5.	2835
6.	155925
7.	6081075
8.	638512875
9.	10854718875
10.	1856156927625
11.	194896477400625
12.	2900518163668125
13.	3698160658676859375

3.1.2 Zřetězené zlomky

$$\tan(\mathrm{x}) = \cfrac{x}{1 - \cfrac{x^2}{3 - \cfrac{x^2}{5 - \cfrac{x^2}{7 - \cdots}}}}$$

x – úhel α nebo β zadaný uživatelem v radiánech

Úhel tangens se u této metody počítá „od spodu nahoru“, tzn. nejprve vypočítáme nejspodnější člen a postupně zlomek zmenšujeme.

Zlomek dekomponujeme na jednotlivé elementy, které se rekurentně ve vzorci opakují. Čitatel si tedy označme *a*, jmenovatel *cf* a menšenec *b*.

$n = počet\ iterací\ (zadány\ uživatelem)$

$a = x^2$

$b = 2 * n - 1$

$cf_1 = \frac{a}{b}$ pro první člen (možný způsob inicializace výpočtu), pro ostatní $cf = \frac{a}{b - cf}$

Pro získání konečného výsledku je nutno iterovat $n - 1\ krát$ z důvodu výpočtu posledního členu (posledního ve výpočtu) odlišným způsobem: $\frac{x}{b-2-cf}$.

Pro výpočet úhlu tangens metodou zřetězených zlomků existuje i jiný vzorec, který má namísto hodnoty x^2 hodnotu 1 a na místě menšence $\frac{2*n-1}{x}$, kdy *b* odpovídá stejné hodnotě jako u mnou použité implementace. Oba vzorce jsou velice podobné a jejich implementace téměř totožná. Já jsem si pro výpočet vybral výše uvedený způsob.

3.2 Vzdálenost a výška měřeného objektu

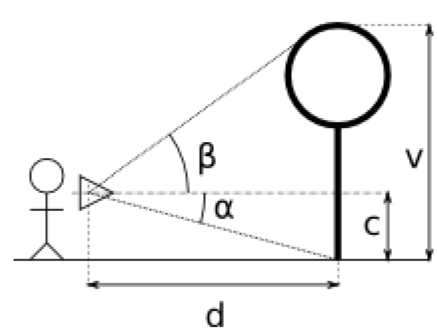
α – úhel α (argument příkazového řádku *A*)

β – úhel β (argument příkazového řádku *B*)

d – vzdálenost měřeného objektu od měřicího přístroje

c – výška měřicího přístroje

v – výška měřeného objektu



Pro výpočet vzdálenosti jsem použil úhel α. Pokud je uživatelem zadán i úhel β, vypočtena je i výška objektu. Výpočet obou úhlů je realizován použitím výpočtu zřetězených zlomků, který je přesnější – přesněji řečeno, konverguje rychleji než výpočet Taylorovým polynomem.

Množství iterací potřebných pro přesnost výsledků na 10 desetinných míst je 9. K tomuto číslu jsem dospěl po testování v celém validním intervalu pro výpočet úhlů: $(0,\ 1.4\rangle\ rad$.

3.2.1 Zjišťování dostatečného počtu iterací

Pro zjištění dostatečného počtu iterací jsem si vytvořil pomocný program, který porovnával výpočty hodnot v odlišných iteracích. V případě, že se následující iterace od té předcházející již neměnila, získal jsem počet dostatečných iterací pro určitou hodnotu.

Testované hodnoty – z intervalu $(0,\ 1.4\rangle$ (testování probíhalo po desetinných místech, v mezních hodnotách jsem testoval místa setin).

Výsledek testování: pro hodnoty blížící se spodní hranici – tedy 0 stačily 3 iterace, přímou úměrou se však jejich počet zvyšoval spolu s rostoucími hodnotami úhlu a úhel $1.4\ rad$ vyžadoval 9 iterací.

3.2.2 Výpočet vzdálenosti objektu

Pro výpočet vzdálenosti objektu je nutno znát úhel α a výšku měřicího přístroje, která je implicitně nastavena na 1.5 m. Uživatel ji však může modifikovat a nastavit v intervalu $(0,\ 100)\ m$. Vzdálenost poté vypočítáme podle definice výpočtu tangens v pravoúhlém trojúhelníku:

$$d = \frac{c}{\mathrm{tg}(\alpha)}$$

3.2.3 Výpočet výšky objektu

Pro výpočet výšky objektu je nutno znát kromě úhlu α a výšky měřicího přístroje (která je opět volitelná) také úhel β. Výšku poté vypočítáme takto:

$$v = c\ +\ \mathrm{tg}(\beta) * d$$

4 Specifikace testů

V programu se vyskytuje více oblastí, které je potřeba otestovat. Jsou jimi především výpočty úhlů tangens, hranice validních argumentů a správný syntax zadání argumentů příkazové řádky. Správnost výpočtů si můžeme ověřit např. softwarem *WolframAlpha*.

4.1 Test 1. Chybný syntax – detekce chyby

Vstup	Očekávaný výstup
./proj2 --tan 1.031 4	Wrong arguments passed
./proj2 -tan 0.756 3	Wrong arguments passed
./proj2 -c -m 0.35 1.301	Wrong arguments passed

4.2 Test 2. Chybný rozsah argumentů – detekce chyby

Vstup	Očekávaný výstup
./proj2 --tan 0.451 15	Wrong arguments passed
./proj2 -m -0.43	Wrong arguments passed
./proj2 -c 2.6 -m 0.35 1.589	Wrong arguments passed

4.3 Test 3. Správnost výpočtů – předpokládané správné hodnoty

Vstup	Očekávaný výstup
./proj2 --tan 0.942 9 11	9 1.375000e+00 1.374880e+00 1.199975e-04 1.375000e+00 0.000000e+00 10 1.375000e+00 1.374957e+00 4.315530e-05 1.375000e+00 0.000000e+00 11 1.375000e+00 1.374984e+00 1.552016e-05 1.375000e+00 0.000000e+00
./proj2 -m 1.198	5.8662549467e-01
./proj2 -m 0.349 1.005	4.1220606782e+00 7.9908848088e+00
./proj2 -c 4.5 -m 0.349 1.005	1.2366182035e+01 2.3972654427e+01

5 Implementace

5.1 Vlastní implementace

U vytváření mé implementace programu jsem vycházel z argumentů, které uživatel zadává. Funkce pro zpracování argumentů se tedy postará o standardní vstup. Podle zadaných dat se v programu inicializuje buď část funkcí zpracovávající porovnávání výsledku funkcí tangens, nebo část pro výpočty vzdálenosti a výšky objektu.

Kód začíná ve funkci main, která volá funkci starající se o argumenty, a ta inicializuje první, nebo druhou část programu.

5.2 Syntax spuštění programu

Program se spouští v příkazovém řádku a očekává některé z následující argumentů (argumenty v "[ ]" jsou volitelné). Pokud zadané argumenty nevyhovují žádné z možných variant, program vypíše chybovou hlášku.

Argumenty	Význam
./proj2 - -help	výpis nápovědy
./proj2 - -tan A N M	výpočet úhlu α zadaného argumentem *A* a následné zobrazení porovnání výpočtů z implementovaných funkcí tangens v intervalu iterací N až M. Rozsah argumentů N M: $(0 < N <= M < 14)$
./proj2 [-c X] -m A [B]	výpočet vzdálenosti, popř. výšky, pokud byl zadán argumentem *B* úhel β. Argument X nastavuje výšku měřicího přístroje. S

6 Závěr

Program měří vzdálenost a výšku měřeného objektu, nebo porovnává výsledky výpočtů tangens funkcí přesně tak, jak bylo v zadání projektu specifikováno, čímž byl můj úkol splněn. V rámci implementace jsem dbal na to, aby byly jednotlivé hranice či intervaly úhlů lehce modifikovatelné. Na druhou stanu musím ale také poznamenat, že by bylo možno program více zoptimalizovat, např. experimentálně zjistit, který ze 2 možných výpočtů pro zřetězené zlomky je časově méně náročnější.

Program byl úspěšně testován v prostředí operačního systému Linux. Kompatibilita s MS Windows není zaručena, jelikož pro něj nebyla aplikace určena.

6.1 Metriky kódu

- počet souborů: 1 soubor
- počet funkcí: 11
- počet řádků zdrojového kódu: 374
- velikost kódu programu: 11 853 byte
- velikost přeloženého souboru: 13 716 byte (systém Linux, 64 bitová architektura)