Dokumentace k projektu pro předměty IZP a IUS

Iteračné výpočty

projekt č. 2

5. prosince 2014

Autor: Andrej Barna, xbarna01@stud.fit.vutbr.cz
Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

Obsah

1 Úvod

Meranie vzdialenosti je v súčasnosti využívané pre mnohé účely, napríklad pri zememeračstve, či 3D modelovaní. Bežné prístroje sa musia spoliehať na to, že odmerajú uhol a z toho si následne vypočítajú tangens, pomocou ktorého si následne dokážu vypočítať vzdialenosť, respektíve aj výšku objektu. V tejto dokumentácii sa zaoberám problematikou výpočtu tangens a možnými riešeniami pre výpočet vzdialenosti podľa zadania.

Táto dokumentácia popisuje môj návrh aplikácie, ktorá dokáže porovnať výsledky výpočtov funkcie tangens pomocou rôznych metód a tiež dokáže vypočítať vzdialenosť objektu zo zadaného hĺbkového uhlu, respektíve aj výšku, pokiaľ je zadaný výškový uhol. Taktiež je popísaný problém zisťovania presnosti pre určitý počet iterácií pri využívaní zloženého zlomku.

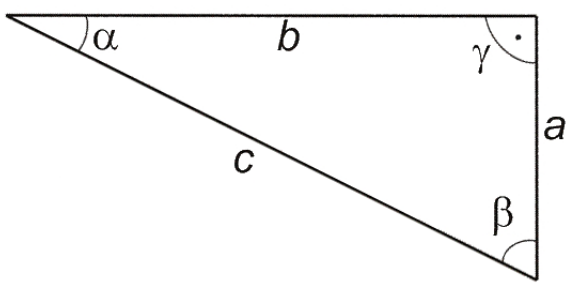
2 Analýza problému a princíp jeho riešenia

Hlavná problematika zadania spočíva vo výpočte funkcie tangens pri výpočte vzdialenosti, respektíve aj výšky objektov od miesta pozorovania. Výpočet funkcie tangens je predovšetkým známy tak, ako sa vyučuje na stredných školách. Teda buď ako podiel protiľahlej a priľahlej strany voči danému uhlu, alebo tiež ako podiel sínusu a kosínusu pre daný uhol. V zadaní úlohy je však jasne dané, že sa má tangens vypočítať z uhlu daného argumentom, čiže akékoľvek ďalšie zamýšľanie sa nad výpočtom tangens pomocou dĺžky strán je zbytočné.

V tomto momente by sa dala pri programovaní využiť jednak funkcia `tan` z matematickej knižnice jazyka C, alebo funkcie `sin` a `cos` a následne ich podielom získať tangens uhla. Avšak v zadaní úlohy je tiež dané, že sa z matematickej knižnice nesmie volať žiadna funkcia s výnimkou `isnan`, `isinf` a `tan`, ktorý sa smie využiť jedine pre zrovnávanie presnosti výpočtu tangens pomocou metódy Taylorovho radu a zreťazeného zlomku. Je teda potrebné, aby bola zavedená funkcia, ktorá dokáže vypočítať tangens, a to podľa zadania dokonca dve. Jedna má počítať tangens podľa Taylorovho radu a druhá metódou zreťazených zlomkov. Taktiež pri meraní vzdialenosti je vyžadovaná presnosť na 10 desatinných miest pre tangens, pričom je potrebné zistiť počet iterácií pre funkciu vypočítavajúcu tangens.

2.1 Goniometrické funkcie

V tejto sekcii bližšie popisujem výpočet goniometrických funkcií sínus, kosínus a tangens pre ujasnenie niektorých vzťahov medzi nimi.



Obrázek 1: Názorný pravouhlý trojuholník.

V tomto obrázku je pravouhlý trojuholník ABC, v ktorom sú uhly α, β a pravý uhol γ. Z toho vieme určiť, že strana protiľahlá pravému uhlu je prepona, teda sa jedná o stranu c. Teda

strany a, b sú odvesny. Konkrétne strana a je protiľahlá odvesna a strana b je priľahlá odvesna. Základné goniometrické výpočty sú:

$$sin(\alpha) = \frac{protiľahlá\ odvesna}{prepona} = \frac{a}{c} \tag{1}$$

$$cos(\alpha) = \frac{priľahlá\ odvesna}{prepona} = \frac{b}{c} \tag{2}$$

$$tan(\alpha) = \frac{sin(\alpha)}{cos(\alpha)} = \frac{protiľahlá\ odvesna}{priľahlá\ odvesna} = \frac{a}{b} \tag{3}$$

Tieto výpočty sú využiteľné až pri meraní vzdialenosti a výšky objektu, keďže ako som už napísal, máme daný uhol a nie strany.

2.2 Výpočet goniometrických funkcií využitím Taylorových radov

Taylorove rady sú zvláštne mocninné rady, ktoré sú nazvané po matematikovi Brookovi Taylorovi, ktorý ich publikoval, avšak objavené boli už predtým Jamesom Gregorym. Tieto rady aproximujú približnú hodnotu rôznych matematických funkcií, avšak teraz sa zaoberám len goniometrickými funkciami sínus, kosínus a tangens. Keďže sú tieto rady nekonečné, tak sa bežne využíva len niekoľko prvých hodnôt, zvaných tiež Taylorov polynóm. Uhol sa udáva v radiánoch.

$$sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots \tag{4}$$

$$cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots \tag{5}$$

$$tan(x) = \sum_{n=1}^{\infty} \frac{B_{2n}(-4)^n(1-4^n)}{(2n)!} x^{2n-1} = x + \frac{x^3}{3} + \frac{2x^5}{15} + \dots \quad pre\ |x| < \frac{\pi}{2} \tag{6}$$

Z týchto vzorcov sa vždy dá jednoznačne určiť o akú hodnotu sa upraví výsledok pri každej iterácii. Pri výpočte sínusu a kosínusu vypočítaná hodnota osciluje pri reálnej hodnote sínusu, vzhľadom na to, že sa v rade strieda pričítanie a odčítanie nasledujúcej iterácie. Tangens však neosciluje, hodnota vypočítaného tangens sa asymptoticky blíži k reálnej hodnote zdola, keďže sa v rade pre tangens všetky prvky sčítajú. V tomto prípade by bolo veľmi jednoduché zistiť, či sa dosiahla požadovaná presnosť na 10 desatinných miest, keďže stačí vypočítať hodnotu ďalšieho prvku radu a ak je menší ako požadovaná presnosť tak sa dá povedať, že bol nájdený požadovaný počet iterácii.

Hlavný problém Tayloravho radu tangens je potreba Bernoulliho čísel. Vzhľadom na komplikovaný princíp výpočtu Bernoulliho čísel by teda táto metóda bola pomerne náročná na výpočet tangens. Výpočet by sa dal zjednodušiť, ak by sa použila pevne daná, vopred vypočítaná postupnosť Bernoulliho čísel, ale v tom prípade by sa už dalo uvažovať priamo o tom, že by sa pri výpočte tangens použila vopred vypočítaná postupnosť čitateľov a menovateľov pre určitý počet prvých prvkov radu.

Táto metóda je využitá len pri porovnaní presnosti výpočtu tangens rôznymi spôsobmi, pričom sa v nej využíva predom daná postupnosť čitateľov a menovateľov, ako to je explicitne požadované v zadaní. Konkrétne sa jedná o prvých 13 členov.

2.3 Výpočet tangens využitím zreťazených zlomkov

Výpočet tangens využitím zreťazených zlomkov je prehľadnejšie zapísaný, než Taylorov rad a teda je ľahšie mu porozumieť, ale tiež je jednoduchšie ho naprogramovať. Vzhľadom na to, že nie je potrebné zavádzať ďalšie funkcie pre výpočet Bernoulliho čísla sa dá táto metóda využiť aj pre väčší počet iterácií, čo by v prípade výpočtu využitím Taylorovho radu bolo buď časovo, alebo aj priestorovo náročné, alebo by to v prípade zavedenia polynómu dokonca nebolo možné, ak by sme chceli počítať s väčším počtom iterácii, ktorý by už nebol obsiahnutý v polynóme.

Sú dva varianty vzorcu pre výpočet tangens pomocou zreťazených zlomkov:

$$tan(x) = \cfrac{x}{1 - \cfrac{x^2}{3 - \cfrac{x^2}{5 - \cfrac{x^2}{7 - \ddots}}}} \qquad tan(x) = \cfrac{1}{\cfrac{1}{x} - \cfrac{1}{\cfrac{3}{x} - \cfrac{1}{\cfrac{5}{x} - \cfrac{1}{\cfrac{7}{x} - \ddots}}}} \tag{7}$$

Výpočet metódou zreťazených zlomkov má však jednu základnú nevýhodu, ktorá sa pri výpočte Taylorovým radom nevyskytuje. Konkrétne sa jedná o zistenie po koľkých iteráciách sa dosiahne požadovaná presnosť. Táto komplikácia vzniká v dôsledku toho, že sa zreťazený zlomok počíta od najvnorenejšieho zlomku.

Pri výpočte metódou zložených zlomkov sa ďalšími iteráciami znižuje hodnota v menovateli najvnorenejšieho zlomku, keďže sa od menovateľa odčíta ďalšia iterácia. Keďže tento zlomok je v menovateli ďalšieho zlomku, tiež ako menšenec, tak zníži hodnotu menovateľa zlomku, do ktorého patrí. Takto sa postupne pri každej ďalšej iterácii prenáša výsledok do ďalších - vyšších - zlomkov až kým sa daná hodnota neprenesie k hlavnému zlomku, kde už ovplyvní pôvodného menovateľa, ktorý sa tým zníži.

Z matematiky je známe, že zlomok je vlastne podiel, pričom čitateľ je delenec a menovateľ je deliteľ. Znížením deliteľa sa teda zvýši výsledná hodnota zlomku. Takto sa dá konštatovať, že každou iteráciou sa výsledok funkcie `cfrac_tan` približuje k reálnej hodnote tangens, ku ktorej sa však len asymptoticky blíži, tak to to bolo aj u Taylorovho radu.

Tiež sa z toho dá konštatovať, že každou iteráciou sa znižuje hodnota, o ktorú je zmenšená odchýlka od reálneho tangensu, vzhľadom na to, že čím viac je zlomok vnorený, tým menší je jeho vplyv na konečný výsledok.

3 Návrh riešenia problému

3.1 Analýza vstupných dát

Formát, v akom majú byť vstupné dáta určené argumentami programu, je presne daný v zadaní. Pre prvý argument je potrebné porovnávanie reťazcov znakov s možnými operáciami, pre číselné argumenty je potrebné skontrolovať ich platnosť pri načítaní. Konkrétne treba skontrolovať, či formát vstupných údajov je zhodný s očakávaným dátovým typom, či uhly patria intervalu (0;1.4>, či výška meracieho prístroja patrí intervalu (0;100> a či rozsah iterácií, pre

ktoré sa má presnosť výpočtu tangens porovnávať, nie je menší než 1 alebo väčší než 13. Taktiež je potrebné dbať na voliteľné argumenty pri meraní vzdialenosti. Na všetky tieto operácie sú v štandardnej knižnici jazyka C zabudované funkcie `strtod`, `strtol`, `strcmp`, čiže pre spracovávanie argumentov nie je potrebné tvoriť nové funkcie.

3.2 Výpočet tangens

3.2.1 Metóda Taylorovej rady, taylor_tan

Táto metóda je využitá len pri porovnaní presnosti výpočtu tangens rôznymi spôsobmi, pričom sa v nej využíva predom daná postupnosť čitateľov a menovateľov, ako to je explicitne požadované v zadaní. Konkrétne sa jedná o prvých 13 členov. Teda pri každej iterácii i sa len pričíta k i-1:

$$f(i) = f(i-1) + \frac{čitateľ_{i-1} * x^{i*2-1}}{menovateľ_{i-1}} \tag{8}$$

3.2.2 Metóda zreťazených zlomkov, cfrac_tan

Pre výpočet funkcie tangens som zvolil prvý variant zreťazeného zlomku, ktorý sa nachádza v kapitole 2.3. Ako som už v tej kapitole spomínal, zložené zlomky je potrebné vypočítavať od najvnorenejšieho prvku. Pre jednoduchosť spracovávania iterácií sa dá v iteráciách spravovávať len menovateľ, pričom sa následne už len vydelí x menovateľom získaným týmto spôsobom. V praxi by sa výpočet menovateľa pre iteráciu i dal zapísať ako:

$$f(i) = i * 2 - 1 - \frac{x^2}{f(i+1)} \tag{9}$$

Výsledný menovateľ získame, ak vypočítame iterácie od 1 do požadovaného počtu iterácií. Pokiaľ je i rovné požadovanému počtu iterácií, tak sa ako f(i+1) vloží do vzorca dvojnásobok požadovaného počtu iterácií znížený o 1. Následne už len odčítame od seba získané výsledky.

Po vypočítaní menovateľa dosiahneme výslednú hodnotu tangens pre daný počet iterácií, ak vydelíme x menovateľom.

3.3 Zistenie potrebného počtu iterácií

Presný počet iterácií, ktorý je potrebný pre výpočet tangens je obtiažne jednoznačne určiť. Pre tangens uhla, ktorý patrí do intervalu (0;1.4> určeného zadaním platí, že čím väčší je uhol, tým je väčšia hodnota tangens a teda pri rovnakom počte iterácií dosiahne väčší uhol menšiu presnosť, než menší uhol. Teda čím väčší je uhol, tým viac iterácií je potrebných pre dosiahnutie požadovanej presnosti. Z toho sa dá dedukovať, že musia existovať nejaké pomyselné hranice, od ktorých je potrebný väčší počet iterácií než pre predchádzajúcu hranicu, aby sme dosiahli požadovanú presnosť.

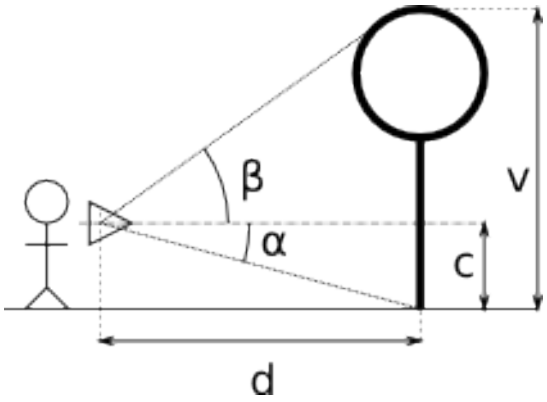
Problém tohto konštatovania spočíva vo výpočte týchto hraníc. Vzhľadom na tento problém som považoval za potrebné vytvoriť funkciu, ktorá bude pre každú hodnotu do určitého počtu desatinných miest skúmať kedy je pre aktuálne testovaný uhol potrebný väčší počet iterácií, než pre ten predchádzajúci. Každopádne, ak by sme chceli pomocou takejto funkcie určiť tieto hranice na väčší počet desatinných miest, tak by to bolo mimoriadne časovo náročné, čo sa dá obísť dvoma spôsobmi:

- raz určiť, pre ktoré intervaly je aký počet iterácií potrebný pre požadovanú presnosť
- využívať funkciu s presnosťou na menší počet desatinných miest

Druhá možnosť je neoptimálna, keďže v konečnom dôsledku funkcia musí prebiehať pri každom spustení programu a teda zväčšuje časovú náročnosť a tiež pri zníženej presnosti by bol možno postačujúci menší počet iterácií. Teda ja som zvolil prvú alternatívu a tieto hraničné hodnoty som si uložil do poľa. Teda keď sa volá funkcia na výpočet tangens, tak sa zavolá funkcia, ktorá porovná daný uhol s hranicami v poli a podľa toho určí koľko iterácií postačí pre výpočet s presnosťou na 10 desatinných miest.

3.4 Výpočet vzdialenosti

Výpočet samotnej vzdialenosti sa vykonáva pomocou goniometrických funkcií, ktoré boli popísané v kapitole 2.1.



Obrázek 2: Upresnenie uhlov a strán použitých vo výpočtoch.

Tento obrázok, ktorý som prevzal zo zadania, som považoval za výbornú názornú ukážku pre popis výpočtov. Uhly α a β sú uhly, z ktorých sa vypočítava tangens. Úsečka c predstavuje výšku meracieho zariadenia a úsečka v predstavuje výšku objektu. Úsečka d je vzdialenosť objektu od meracieho zariadenia.

Podľa zadania uhol α musí byť určený a voliteľne sa môže určiť výška meracieho zariadenia c a uhol β, avšak bez uhla β sa nedá určiť výška objektu. Keďže výška meracieho zariadenia je 1.5 metra, ak nie je explicitne inak určená, tak pre výpočet vzdialenosti stačí poznať tangens uhla α, keďže tangens je pomer protiľahlej strany (v tomto prípade je dĺžka protiľahlej strany rovnaká ako je výška meracieho prístroja) a priľahlej strany (čo je v tomto prípade vzdialenosť objektu od meracieho zariadenia). Teda vzdialenosť d sa vypočíta ako

$$d = \frac{c}{tan(\alpha)} \tag{10}$$

Následne sa dá pomocou uhla β vypočítať vzdialenosť od vrchu objektu po bod, ktorý sa nachádza v rovnakej výške ako meracie zariadenie, keďže trojuholník, v ktorom sa nachádza uhol β má v tom bode pravý uhol. Teda pre získanie celkovej výšky je ešte potrebné pripočítať výšku meracieho prístroja. Z toho vyplýva, že

$$v = d * tan(\beta) + c = \frac{c * tan(\beta)}{tan(\alpha)} + c \tag{11}$$

Aby sa vzdialenosť d dala vypočítať, tak tan(α) nesmie byť 0, čiže uhol α nesmie byť 0. Táto podmienka nemôže nastať, keďže tangens nadobúda 0 v hodnote k*π, avšak uhol musí byť v intervale (0;1.4> a teda sa dá z týchto vzorcov vypočítať vzdialenosť a výšku objektu podľa vstupných údajov.

3.5 Špecifikácia testov

V tejto kapitole uvádzam niekoľko príkladov, ako by sa mal program správať pri rôznych nežiadaných a aj žiadaných vstupoch. Všetky tieto údaje sú argumenty, s ktorými sa spúšťa program.

Test 1: Chybná syntax ⟶ Detekcia chyby.

```
klobasa
--tan 1.2
-m 1.4 -c
```

Test 2: Nezmyselné dáta ⟶ Detekcia chyby.

```
--tan nan 10 13
-m inf 1
```

Test 3: Dáta mimo povolený rozsah hodnôt ⟶ Detekcia chyby.

```
--tan 1.5 3 8
-c 103 -m 0 -1.9
```

Test 4: Správnosť výpočtu ⟶ Predpokládaná správna hodnota.

Vstup:

```
-m 1
```

Výstup:

```
9.6313892390e-01
```

Vstup:

```
-c 3 -m 0.7 0.4
```

Výstup:

```
3.5617254964e+00
4.5058733869e+00
```

Vstup:

```
--tan 1.1 3 3
```

Výstup:

```
3 1.964760e+00 1.758401e+00 2.063583e-01 1.959819e+00 4.940536e-03
```

Vstup:

```
--tan 0.3 8 13
```

Výstup:

```
8 3.093362e-01 3.093362e-01 7.908119e-13 3.093362e-01 5.551115e-17
9 3.093362e-01 3.093362e-01 2.886580e-14 3.093362e-01 5.551115e-17
10 3.093362e-01 3.093362e-01 1.054712e-15 3.093362e-01 5.551115e-17
11 3.093362e-01 3.093362e-01 5.551115e-17 3.093362e-01 5.551115e-17
12 3.093362e-01 3.093362e-01 5.551115e-17 3.093362e-01 5.551115e-17
13 3.093362e-01 3.093362e-01 5.551115e-17 3.093362e-01 5.551115e-17
```

Vstup:

```
-m 0.3 0.9
```

Výstup:

```
4.8490922156e+00
7.6106234032e+00
```

Vstup:

```
-c 1.7 -m 0.15 1.3
```

Výstup:

```
1.1248205560e+01
4.2217188781e+01
```

4 Popis riešení

Pri implementovaní programu som využil pre výpočet tangens pomocou zloženého zlomku vzorec 7, pričom som využíval poznatky, ktoré som uviedol v predošlých kapitolách. Taktiež som implementoval funkcie `cfrac_tan` a `tay_tan` podľa kapitoly 3.2, ktoré boli dané ako podúlohy zadania.

4.1 Ovládanie programu

Vstupné dáta pre program sú očakávané ako argumenty pri spustení. Ich syntax je daná zo zadania, pričom prvý argument určuje operáciu. Legálne operácie sú:

- `--help` - vytlačí na štandardný výstup nápovedu k programu, bez argumentov
- `--tan` - slúži pre porovnávanie presnosti výpočtu tangens
- `-c` alebo `-m` - slúžia na výpočet vzdialenosti, respektíve aj výšky

Pre `--tan` sú požadované argumenty typu `double integer integer`, pričom `double` určuje uhol, pre ktorý sa presnosť výpočtu bude porovnávať a premenné typu `integer` udávajú rozsah iterácií (vrátane tých hodnôt), nižšia hodnota má byť na prvom mieste.

Pre `-m` alebo `-c` je to zložitejšie. Argument `-c` je voliteľný a očakáva argument typu `double`, pričom tento argument udáva výšku meracieho zariadenia. Ak je daný, tak sa očakáva ako tretí argument `-m` a za ním tradičná konštrukcia, inak sa argument `-m` očakáva na prvom mieste. Argument `-m` očakáva ako ďalší argument uhol α, udaný v radiánoch, zapísaný vo formáte `double`. Za ním môže byť voliteľný argument, ktorý udáva veľkosť uhla β, tiež vo formáte `double` - ak nie je daný, tak sa vypočíta len vzdialenosť objektu.

Ak program dostane nesprávnu operáciu, alebo ak sú ostatné argumenty v zlom formáte, tak vypíše na chybový výstup hlásenie a ukončí sa. Inak vykoná požadovanú operáciu a výsledok vypíše na štandardný výstup.

4.2 Voľba dátových typov

Pre ukladanie uhlov, výšky meracieho zariadenia, vzdialenosti aj výšky objektu som použil dátový typ `double`, keďže pri výpočtoch treba pracovať s desatinnými číslami a strata, ktorá by nastala konverziou reálneho čísla na celé číslo by bola neprípustná. Dátový typ `double` je tiež podľa zadania explicitne vyžadovaný pri výpočtoch. Pri porovnávaní presnosti výpočtu tangens používam pre uloženie rozsahu iterácií dve celočíselné premenné typu `integer`. Tieto dátové typy sú vhodné pre implementáciu vzhľadom na to, že pokrývajú rozsah hodnôt, ktoré sa v nich dá očakávať.

4.3 Vlastná implementácia

Vo funkcii `main` sa zavolá funkcia `spracujArgumenty`, ktorá porovná prvý argument, s ktorým bol spustený program s názvami možných operácií. Následne buď vypíše chybové hlásenie, ak neboli zadané argumenty, alebo ak názov operácie nebol rozpoznaný, alebo vykoná príslušnú činnosť pre vybranú operáciu.

Pre operáciu `--help` len vypíše na štandardný výstup reťazec, v ktorom je uložená nápoveda. Pre operáciu `-m` alebo `-c` zavolá funkciu `spracujMeranieVysky`, pre operáciu `--tan` zavolá funkciu `spracujTan`. Funkcie `spracujTan` a `spracujMeranieVysky` obe spočiatku spracujú požadované argumenty programu a zisťujú, či sú dáta v správnom formáte.

Po tomto kroku funkcia `spracujTan` vypíše pre požadovaný výber iterácií zrovnanie výpočtu tangens pomocou rôznych funkcií a následne sa program ukončí.

Funkcia `spracujMeranieVysky` zistí pomocou funkcie `getIterCount` potrebný počet iterácií pre presný výpočet tangens, následne pomocou funkcie `cfrac_tan` vypočíta tangens a z neho vypočíta vzdialenosť objektu, respektíve aj výšku a výsledok nakoniec vypíše. Následne sa program ukončí.

Funkcia `taylor_tan` slúži k výpočtu tangens pre účely porovnávania, pričom jej prototyp sú súčasťou zadania spolu s prototypom funkcie `cfrac_tan`.

V programe je aj obsiahnutá funkcia určená pre zistenie hraníc, od ktorých je potrebný vyšší počet iterácií. Pri overovaní správnosti nameraných hodnôt sa vyskytovali výnimky, ak funkcia prepočítavala presnosť na 7 a viac desatinných miest, avšak tieto výnimky potrebovali väčšinou o 1 iteráciu menej pre dosiahnutie požadovanej presnosti a teda podmienka presnosti nie je kvôli týmto výnimkám narušená.

5 Záver

Program dokáže vypočítať vzdialenosť objektu od miesta merania, ak je známy uhol, pod ktorým je zosnímaná spodná časť objektu. Ak je známy aj uhol, pod ktorým je zosnímaný vrch objektu, tak je možné vypočítať aj výšku objektu. Taktiež je možné porovnať viacero metód výpočtu tangens.

Tento program bol testovaný v operačných systémoch Linux/GNU a Microsoft Windows, pričom v závislosti od operačného systému môže nastať mierne odlišný výsledok po výpočte tangens, respektíve sa môže aj mierne líšiť forma výpisu dát. Odchýlka výpočtu medzi operačnými systémami môže nastať na nižších desatinných miestach.

A Metriky kódu

Počet súborov: 1 súbor

Počet riadkov zdrojového textu: 363 riadkov

Veľkosť kódu programu: 3508B

Veľkosť statických dat: 3432B