T A R T U Ü L I K O O L

MATEMAATIKA-INFORMAATIKATEADUSKOND

Arvutiteaduste instituut
Tarkvarasüsteemide õppetool
Infotehnoloogia eriala

Lavrenti Tšudakov

Arvutiprogramm laboratoorse töö „Juhuvea uurimine“ automatiseerimiseks füüsikas

Bakalaureusetöö

Juhendaja: O. Krikmann

Autor: …… “…..“ mai 2011
Juhendaja: ……. “…..“ mai 2011
Lubas kaitsmiseks: …… “…..“ …….. 2011

TARTU 2011

Sisukord

Sissejuhatus

Füüsikaõppe oluliseks osaks TÜ-s on ka laboritööde tegemine. Seal omandatakse peamised oskused laboratoorsete tööde tegemiseks. Laboritööde sooritamise lahutamatuks osaks on mõõtemääramatuste hindamisoskuste omandamine. Eraldi tööna on kasutusel laboritöö *"Juhuvea uurimine"* [6], mille käiguse ajal mõõdetakse ühe keha pikkust 50 korda nihikuga. Uuritav keha on paigutatud katseklaasi sisse selliselt, et ei oleks võimalik teostada otsemõõtmist. Selliselt mõõdetud andmete analüüsil saab õpilane konkreetse võimaluse hinnata juhuvea suurust arvutuste teel ning illustreerida andmete hajuvust keskväärtuse ümber histogrammi joonistamise teel.

Selline mõõtmine on tihti ebamugavalt tüütu üliõpilase jaoks ja võtab palju aega. Ka praktikumi juhendajal on suhteliselt ebamugav kontrollida arvutustulemusi ja nende põhjal joonistatud histogrammi õigsust. Hinnang antakse arvutatud tulemuste proportsioonide võrdlemisel mõõtmistulemustega ning histogrammi õigsuse kohta puht visuaalselt kontrollides valitud mõõtkava õigsust ning aritmeetilise keskmise asukohta (toetudes antud töö juhendaja 15 aastasele kogemusele).

Juhuvea uurimise laboritöö on tavaliselt üks esimesi töid, mis tuleb üliõpilasel sooritada ja seda enne mehaanika praktikumide sooritamist. Seda laboritööd võib lugeda suhteliselt raskeks tööks, kui arvestame, et üliõpilane peab arvutama üksikmõõtmise ja aritmeetiliste keskmiste ruuthälbe. Antud laboritöö puudutab ka mõisteid tõenäosuse tiheduse funktsioon, normaaljaotus ja usaldusnivoo, mis on enamasti gümnaasiumis jäetud käsitlemata.

Juhuvea uurimise valdkonnas [1] antakse tihti ülevaatlik kirjeldus ilma sügavamalt probleemile lähenemata. See, kuidas määramatus on seotud mõõtmiste protsessiga, eksperimendi ja mõõteriistadega, on käsitletud pealiskaudselt. Pigem seletatakse rohkem tulemust. *Statistika teooria põhikursuses* [2] tunnuse väärtuse mõõtmise osas kirjutatakse olulistedest mõistetest nagu etalon, mõõteväärtus ja mõõteskaalad, kuid protsess, kuidas need väärtused kätte saadakse, täpsuse sisu ja võimaliku vigade olulisuse kohta, ei ole seletatud midagi. Kui vaatame eksperimentide *tulemuste vigade analüüsimist* [3,4], siis seal pööratakse tähelepanu rohkem teooriale.

Rakenduste poolt eksisteerib veebipõhine programm *Vigade analüüs* [5] mille abil kasutaja saab ise proovida katseid teha, vaadata tulemusi ja lugeda rohkem probleemist. Kuid katsete arv on piiratud ainult kümnega, mõõdetakse ainult reageerimisaeg, mis võib olla seotud ka riistvara reageerimis ajaga ja andmete põhjal ei koosteta histogrammi, siis tulemused ei ole kõige kirjeldavaimad.

Selleks, et aidata üliõpilastel mõista juhuvea tekkimist lihtsa arvutiprogrammi abil ning mõõtmistulemuste hajumist kiirelt illustereerida histogrammil, seab käesoleva töö autor järgmised eesmärgid:

- uurida laboritööd „Juhuvea uurimine“ ja otsustada milliseid osasid on võimalik asendada prograamse simulatsiooniga;
- anda ülevaade programmi loomiseks tööriistadest ja võimalustest;
- luua programm, mis on suuteline andmeid sisse lugema, salvestama, töötlema ja esitama neid graafiliselt;
- testida programmi mõne kasutaja peal, et kõrvaldada jämedad ebakorrektsused

Töö koosneb neljast peatükist. Esimeses peatükkis antakse ülevaade füüsika laboritööst *„Juhuvea uurimine“* [6], kuidas toimub kogu mõõtmisprotsess ja millised osad oleks võimalik asendada arvuti programmiga.

Teises peatükis kirjeldatakse miks on kasutatud *Adobe Flash* [11] tehnoloogiat ja antakse kasutatud programeerimiskeele *Actionscript 3* [12] ülevaade. Esitatakse, millised oleksid alternatiivid, tarkvara arenduse pakettide ülevaade ja nende maksumus.

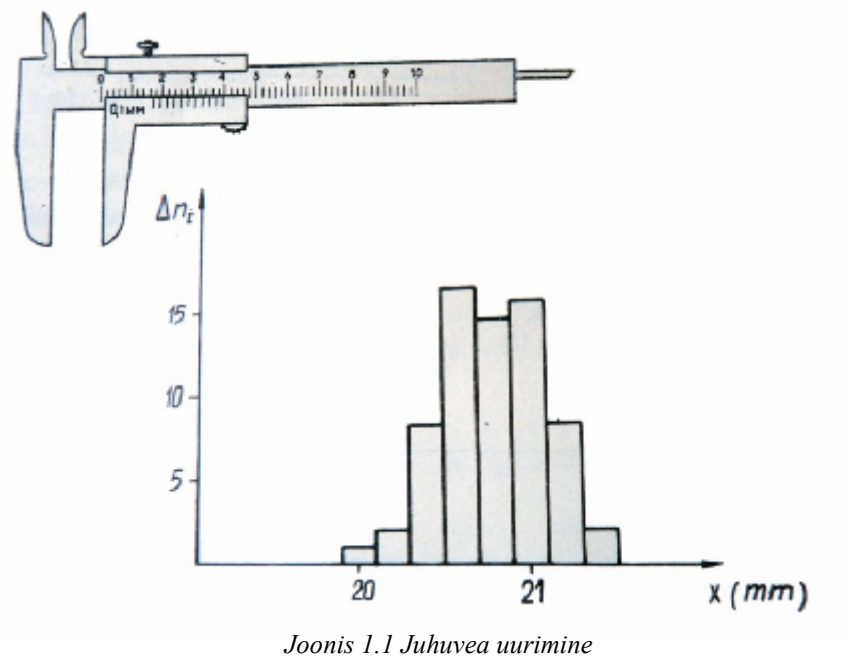
Kolmas peatükk kirjeldab järgmisi aspekte: programmi ehitus, graafiline liides, olulised funktsioonid ja parameetrid.

Neljandas peatükis räägitakse loodud programmi testimisest ja ilmnenud komplikatsioonidest. Peatutakse edasiarengu võimalustel.

Lisana on esitatud CD plaadil loodud programmi lähtekood ja kompileeritud failid niisugusel viisil, et neid saaks kohe kasutada.

1. Laboritöö „Juhuvea uurimine“

Füüsika õppekavas on ettenähtud laboritöö *„Juhuvea uurimine“ [6]*. Seda konkreetset laboritöid peavad tegema ka need üliõpilased, kes soovivad saada füüsika õpetajaks gümnaasiumi osas.



Joonis 1.1 Juhuvea uurimine

Üldiselt kogemused näitavad, et mingi suuruse mõõtmiste tulemused võivad varieeruda. Põhjusi on palju: mõõteriista skaala jaotised ei ole täpselt võrdsed, skaala kriipsud ei ole ühesuguse laiusega, osuti ise on lõpliku paksusega, kuid ka objekti omadused võivad sõltuda mõõtmistingimustest (temperatuurist, õhuvooludest, valgustingimustest jne.). Vea suurus sõltub ka mõõtja vilumustest. Kuigi ühtki füüsikalist suurust ei saa määrata absoluutselt täpselt, on võimalik hinnata vahemikku, millesse määrava suuruse tõeline väärtus teatava tõenäosusega satub. Laboratoorne töö on väga mahukas ja vajab palju täpsusi joonistamisel, mõõtmise ja arvutuste tegemisel.

Töö käik on järgmine:

1. Mõõdetakse antud eset 50 korda, selle ajal hoitakse nihikut nii, et skaala poleks nähtav, sellega välditakse soovi saada eelmise tulemusega samasugust tulemust. Peale näidu üleskirjutamist nihutatakse liikuvat haara. Mõõtetulemused kantakse tabelisse

ridade kaupa (vasakult paremale), see vähendab grupikeskmiste süstemaatilise erinevuse ohtu. See osa programmis võib olla asendatud mingi graafilise objektiga, mille mingit parameetrit kasutaja peab määrama või mõõtma hiirega.

2. Leitakse 50 tulemuse aritmeetilise keskmist $\bar{l}$ ja järgmise valemi

$$s_x = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}}$$

abil ruuthälve s_I. Pärast seda leitakse gruppide aritmeetilised keskmised I_j, gruppide aritmeetilise keskmise $\bar{\bar{l}}$ ja valemi

$$s_{\bar{x}} = \sqrt{\frac{\sum_{j=1}^{n}(X_j - \bar{X})^2}{N-1}}$$

abil gruppide aritmeetiliste keskmiste ruuthälve $s_{\bar{I}}$. Veendutakse, et $\bar{l} = \bar{\bar{l}}$ ja kontrollitakse, kas kehtib seos $s_{\bar{I}} \approx \frac{s_I}{\sqrt{n}}$ kus n on mõõtmiste arv ühes grupis (antud juhul $n = 5$). Programmis kõik arvutused peavad olema teostatud automaatselt ilma kasutajapoolse osaluseta.

3. Koostatakse üksikmõõtmiste jaotuse histogrammi. Selliseks jagatakse suurima ja vähima mõõtetulemuse vahe 10 võrdseteks osaks pikkusega Δl_i . Mastaapi võib valida millimeeterpaberil vabalt. Loetakse igasse vahemikku langevate mõõtetulemuste arvu Δn_i ja joonistatakse iga vahemiku kohale ristküliku kõrgusega $f_i = \frac{\Delta n_i}{50 * \Delta l_i}$ kus Δn_i on vahemikku Δl_i sattuvate mõõtetulemuste arv. Kui mõni tulemus langeb

vahemikke eraldavale rajale, loetakse see ülespoole kuuluvaks. Histogrammil märgitakse ka ära $\bar{l}$. Programmis tehtud arvutuste abil peab olema kujundatud sarnane histogramm, et kohe visualiseerida kogu töö tulemusi. Samuti tulemust peab olema salvestatud, et seda vajadusel saaks kohe sisselugeda ja kasutada.

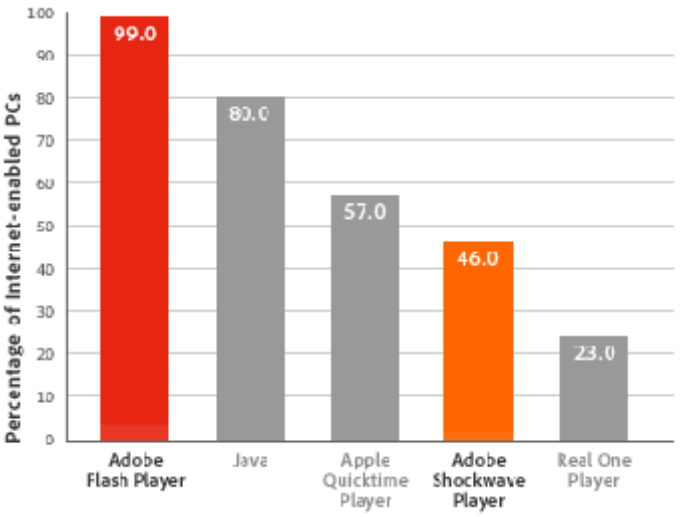
Mõõtmisprotsess on aeglane, sest kordusmõõtmiste tegemine ja kirjapanek võtab aega kui mõõtmise keha on paigaldatud katseklaasi sisse kogu protsess on üsna aeglane. Iga kord andmeid kantakse tabelisse käsitsi, selle abil tehakse arvutusi ja joonistatakse histogrammi millimeetripaberile selleks, et juhuvea uurimise tulemust visualiseerida. Samuti histogrammi joonistamine vajab palju täpsusi ja ettevaatusi. Kui töö käigus tekkib mingi häire või tehakse katkestust, siis tekib suur tõenäosus, et lõpptulemus on ebatäpne või eksklik.

Lõputöö eesmärk on kirjutada programmi nii, et juhuvea tekkimise protsess toimuks arvuti abil. Valminud versioonis õpilane peab hiirega otsustada joonistatud ruudu keskpaiga, pärast seda positsiooni juhuslikult muudetakse ja tehakse järnev mõte. Kui vajalik mõõtmete arv on täis, tehakse kõik arvutused, joonistatakse histogrammi ja antakse võimalus tulemust salvestada.

2. Tööriist

Liikvel on palju erinevaid programme, millega sissejuhatuses püstitatud eesmärkide saavutada. Antud töö puhul langes valik platvorm *Adobe Flash.* Kuna programm eeldab palju tegemisi graafikaga ja peab olema lihtne kasutamiseks ja paigaldamiseks, otsustati kasutada *Adobe Flash* [11] tehnoloogiat ja programmeerimiskeelt *Actionscript* versiooniga *3* [12]. Sellega programmi on võimalik kasutada laialt enamustel arvutitel, käivitades ainult interneti brauserit, kui ka *Adobe Flash* [11] tagab kiirust graafikaga töötlemise protsessis. Kuna valitud programmeerimiskeel on võimsas objektorienteeritud keel, siis on kindlustatud, et tulevikus vajaduse korral on võimalik lihtsalt funktsionaalsust laiendada.

2.1. Adobe Flash populaarsus, mugavus, võimalused.



Joonis 2.1 Adobe Flash plugin leviala

Tänapäevane *Adobe Flash* on väga levinud ja populaarne platvorm veebirakenduste loomiseks, statistika näitab, et *Adobe Flash* plugin toetus on 99% arvutitel, millised on veebipõhised [7]. Oma tugeva objektorienteeritud keelega *Actionscript 3.0* tagab võimalust luua programme, millised on keerulised ja võimsad rakendused, kus tegeletakse dünaamiliselt muudetava graafikaga. Platvormi töö on tagatud kõikides kaasaegsetes Interneti brauserites ja operatsioonsüsteemides nagu *Windows*, *Macintosh* ja *Unix*.

Mugavuse poolt *Flash* kasutamine ei vaja iga rakenduse korral lisafaile paigaldamist. Loodud rakendus laaditakse kiiresti ja tagab intellegentsemat andmete salvestamist vähemällu nii, et iga järgneva kasutamise korral ei pea kogu programmi arvutile laadida. Lihtne failide töödeldamine, nagu kirjutamine ja lugemine selle töö raames loodetava tarkvara juhul on

tagatud täiesti. Programmeerijal on vabadus valida mitu hulgast tarkvara arenduse keskkonnadest, tuntumaid nendest on:

- Adobe poolne *Flash Professiona,l* mille viimane stabiilne versioon on *CS 5* (seisuga 07.09.2010), mis praegu pakutakse osana programmide paketis *Adobe Creative Suite*. Originaalselt programm oli loodud *Macromeida* poolt 1996. aastal ja pakkus ainul baasi võimalusi animatsiooni ja lihtsa interaktiivse sisalduse loomiseks. Oma pikka elutsükliga see arendamise tarkvara oli ehitatud niisugusel viisil, et pakkuda kõiki võimalike instrumente ja komponente tarkvara loomise protsessile. Kõige olulisemad nendest on: töörist programmikoodi loomiseks, testimiseks sisseehitatud kompilaatoriga ja debuggeriga, riistad primitiivsete graafikaliste elementide loomiseks, tekstitöötlus, vektor ja rastri graafika toetus, graafikale võivad olla rakendatud erinevaid dünaamiliselt muutuvaid filtreid, video ja helitoetus, võimsad komponentide kogud, kust arendaja saab lihtsalt valida programmi osasid, nagu nuppe, kontroll kastid ja nii edasi. Viimase versiooni *CS 5.5* hind on 699.00$ või 1 799$ kui seda osta *Creative Suite 5.5 Web Premium* paketti osana.

.

- *Powerflasher* poolne *Flash Developer Tool (FDT)* - see keskkond on väga sarnane arendamise tarkvaraga *Eclipse IDE*. Enamasti see on programmikoodi loomise vahend, mis muudab kogu protsessi intuitiivsemaks ja tulemuslikuks, kuid omab laiendatud funktsionaalsust võrreldes *Flash Professional* vahendiga. Kõige tähtsamad erinevused ja laiendused on võimsas süntaktiline vigade ja lähtekoodi lisavalgustus, automaatne täitmine, šabloonid, formateerija, “refactoring”, deklaratsioonile hüppe ja palju muud. *FDT 4 Pure* versiooni hind on 153$, võimsaim versioon *FDT 4 Max* maksab 831$.

- *FlashDevelop* - on avatud koodiga arenduskeskkond ja toimetaja, mis toetab põhilisi võimalusi programmi koodi loomiseks. Võrreldes ülalnimetatud lahendustega pakutakse vähem funktsionaali, kuid juhul kui tegu on mitte ülemahukama projektiga, siis seda piisab eduka programmi koodi kirjutamiseks, testimiseks ja kompileerimiseks. *FlashDevelop* on tasuta

tarkvara, see muudab selle tööriista valiku kõige veetlevamaks. Tuleb arvestada, et tasuta kompilaatori kasutamine nõuab selle lisapaigaldamist.

Selle töö autoril on kahe esimese produkti kasutamise endine kogemus, iga nendest võiks rahuldada programmeerija kõiki vajadusi, kuid see tõstab lõpplahenduse hinda sobimatuks. Hariduse ja enesearendamise eesmärgina tehtud valik *FlashDevelop* poolele, sellega loodava tarkvara lõpphind võrdub ainult kulunud tundide arvuga.

Selle osa kokkuvõttena saab öelda, et *Adobe Flash platvorm* [11] annab võimalusi luua veebipõhiseid rakendusi mugavate ja mitmekesistega vahenditega. Loodud tarkvara võib olla heietatud laiale kasutajate grupile ilma takistuseta ja toetuse tarkvara lisapaigaldamiseta.

2.2. ActionScript 3.0 ülevaade

ActionScript on objektorienteeritud, prototüübi põhine, funktsionaalne, imperatiivne keel, loodud kompanii *Macromedia inc*. poolt 1998. aastal. Keel on *ECMAScript [9]* dialekt, mis tähendab *JavaScripti* süntaksi ja semaantikat omandamist. Algusena oli ehitatud nii, et kontrollida lihtsat 2D graafikat, fookusena võttes animatsiooni toetust, aga teiselt poolt pakuti suhteliselt vähe skriptimise võimalusi. Praegune stabiilne versioon on 3.0, mille abil programmeerija on suuteline luua ja kontrollida keerulisemaid veebirakendusi. Viimane versioon ilmus 2006. aastal koos *Flash Player 9*. Ise keel on avatud koodiga, tema spetsifikatsiooni pakutakse tasuta, kuid lisaks sellele on võimalus kasutada tasuta kompilaatorit, nagu *Adobe Flex* osana, ja tasuta virtuaalmasinat. *Actionsript 3.0* on juba rohkem sarnane keeltedega Java ja C#, kasutab sarnasi andmetüüpe ja paradigmaid, tema baas implementatsioon on *ECMAScript 4 [9]*. Süntaks ja grammatika on ehitatud niimoodi, et luua väljendeid, avaldusi, muutujaid, funktsioone, klasse ja objekte. Lisaks sellele suur osa keelt on loodud niisugusel viisil, et tagada graafika töötlust, muutmist ja kiire animeerimist, XML parsimist, andmete töötlemist, piiratud OpenGL ja DirectX toetust. Lähtekoodi sisaldavad failid omavad laiendust *.as*. AsctionScript abil koostatud programmid võivad olla käivitatud erinevates keskkondades:

- *Adobe Air* – käivitab *Adobe Flash* applikatsioone mitteveebipõhina, toetab HTML ja

JavaScript sisut. Selleks, et seda kasutada, kasutajal peab olema paigaldatud lisatarkvara operatsioonsüsteemi tasemel.

- *Flash Player* – käivitab veebipõhiseid programme, tüüpiliselt kasutatakse ainult brauseritega koos, aga on ka võimalus kasutada seda iseseisvalt.
- *Flash Lite* – rohkem mõeldud mobiilseadmetele, sellega funktsionaalsus on piiratud, et tagada seadmete ressursside mõistliku kasutamist.

Selle arvestades kõik ülalnimetatud *ActionScript 3* omadused on sobilikud selle töö raames loova programmile, käivitamise keskkonnaks oli valitud *Flash Player*, kui kõige levinum viis veebipõhiste rakendamiste käivitamiseks.

2.3. Alternatiivid

Tänapäeval *Adobe Flash* tehnoloogial on erinevad alternatiivid, millistel on ema eeldused ja puudused, kõige tuntumad on *Java*, *Microsoft Silverlight* ja *HTML5*.

Sun Microsystems poolne *Java* abil on võimalik luua väga keerulisi visualiseerimise rakendusi, kuid sellise töö raames rakendus ei vaja suurt keerulisust. Lisaks sellele Flash plugin alustab töid tunduvalt kiiremini, kui Java plugin. Loova rakenduse eeldatavad põhiomadusteks olid mugavus ja lihtsus, milliseid *Adobe Flash* täidab täiesti ja annab võimalusi teha rakendust mõistliku aja raames.

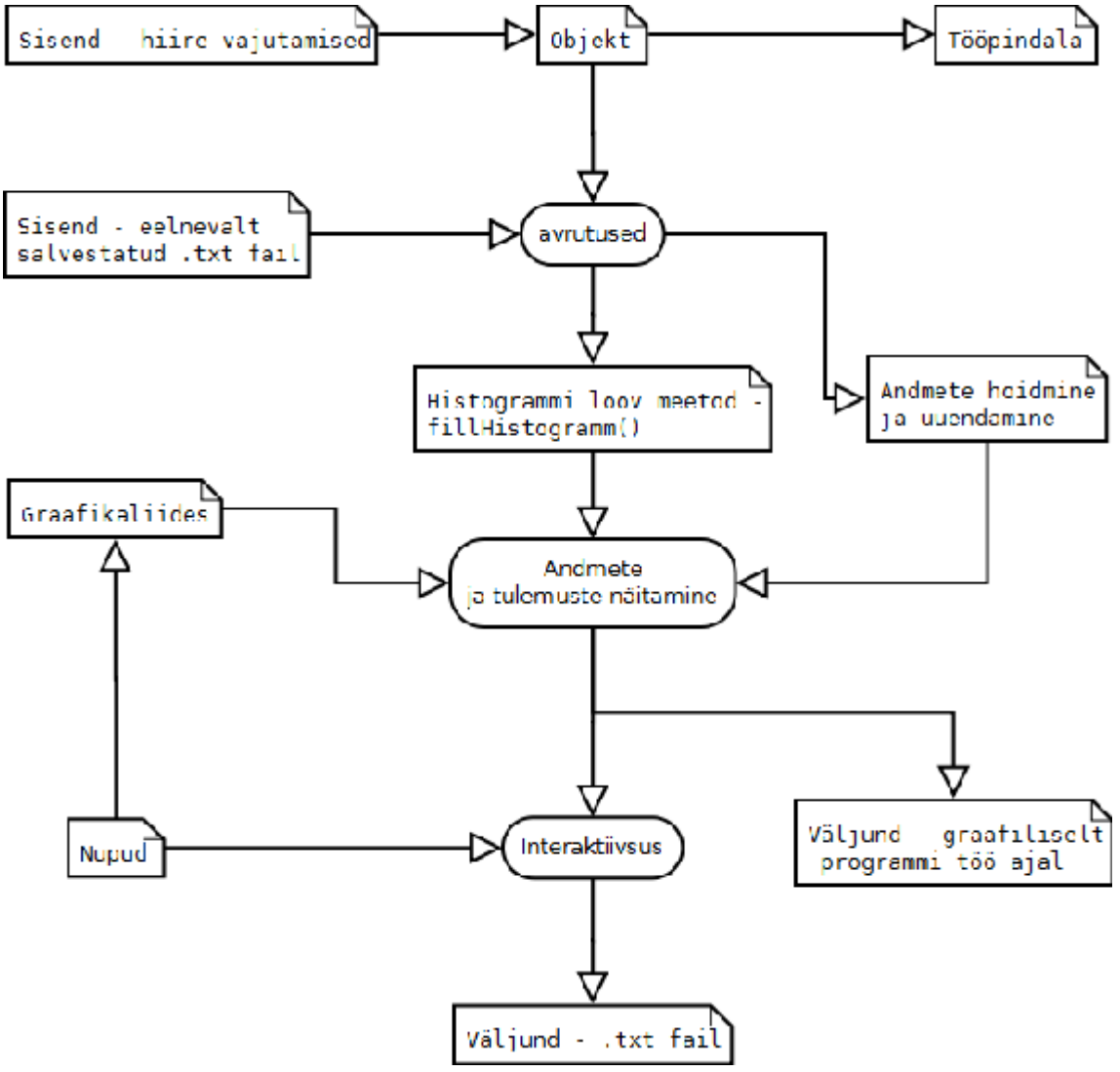
Microsoft Silverlight - võrreldes teiste tehnoloogiatega ei jõudnud veel laiendada kõikidele platvormidele, näiteks täiesti puudub operatsioonsüsteemi Unix toetus, mis tänapäeva aja raames ei ole väga mõistlik. Aga see ise on mõeldud ja realiseeritud niimoodi, et pakkuda sarnasi võimalusi nagu *Adobe Flash* pakub: video ja heli toetust, interaktiivsuse Interneti põhiste rakenduste loomist ja palju muid.

HTML5 - tehnoloogia annab võimalusi töötada graafikaga, kuid selliseks vajab CSS ja JavaScript kõrval kasutamist. Lisaks sellele 2011. aasta märtsi andmete alusel [10] *Internet Explorer* võtab osa 55.92% paigaldatud brauserite hulgast, kuid *HTML5* toetus on võimalik ainult *Internet Explorer 9* puhul, mis ei ole võimalik käivitada *Microsoft Windows XP* operatsioonsüsteemil.

3. Loodud tarkvara ülevaade.

Natud peatükkis esitakse programmis toimuvate protsesside ülevaadet, seletakse programmi paigaldamist ja käivitamist. Kirjeldatakse milliseid nõudeid peab rahuldma arvuti, et antud prograami tööle panna, applikatsiooni olulised osad.

3.1. Applikatsiooni töö põhimõte



Joonis 3.1 Programmi tööprotsessi skeem

Edaspidises tektis ja joonistel on lühiduse mõttes kasutatud järgmisi mõisteid:

- *objekt* – pindala, kujutatud riistküülikuna. Selle objekti tsentrit kasutaja peab sattuma silmadega ja määraha hiire kursoriga;
- *tööpindala* – programmi ala, mille sees muudab positsiooni objekt;
- *stseen* – Flash platvormi peakonteiner, kuhu lisatakse kõike graafiliste elemente;
- *kuulaja* – ActionScript 3 element *eventListener*, mis jälgib sündmusi.

Pärast programmi käivitamist kasutaja ekraanil on kujutatud järgmised põhilised programmi elemendid: nupud, tööpindala, mõõtmisepindala, katsete arv (n), histogramm. Vajadusel võib olla aktiveeritud ka mõõtetulemuste näitamine. Histogrammi joonistamise ajal, et kogu rakenduse tööpindala ei oleks kuhjanud, on tehtud suuruse vähendamine ja histogramm on kujutatud nagu väike ikoon. Ikoon ei ole lihtsalt pilt, seda uuendatakse dünaamiliselt ja see protsess on kohe nähtav. Vajaduse juhul on võimalik histogrammi suurendada, et seda teha kasutaja peab paigaldama hiire kursorit vähendatud histogrammi peale ja kohe muudab hitsogrammi mastaap. Selleks, et suurust tagasi vähendada, tuleks hiire kursorit tuua histogrammi väljapoole. Töö võib olla alustatud kohe pärast programmi käivitamist, aga soovitatakse kõik vajalikud häälestusi teha enne. Aga tuleb arvestada seda, et väärtused, millised on kasutatud vaikimisi on täiesti sobilikud programmi edukaks kasutamiseks. Mõõtmised tehakse niimoodi, et kasutaja peab otsustama oma silmadega, kus asub mõõtmisepindala täpne tsenter horisontaalse suuna suhtes ja teha hiire nuppu vajutamist just sellisele kohale. Pärast esimest vajutamist objekt alustab juhuslikult muuta oma positsiooni tööpindala raames. Positsiooni ümbermääramine kordub iga eeldefineeritud aja järgi, vaikimisi taimeri väärtus on 3 sekundit, aga see võib olla muudetud häälestuse jaotise sees. Kui katsete arv oli eeldefineeritud ja ei olnud määratud nagu „piiramata“, siis kui klikkide arv jõuab mingi arvudeni, antakse selle kohta teade, et vajalik katsete arv *n* on täis. Siis on võimalik jätkata andmete analüüsi tegemisest või salvestamist. Andmete salvestamine on võimalik teostada igal hetkel. Programmi töö ajal arvutatakse järgmisi parameetreid: praegune minimaalne ja maksimaalne mõõtmiste väärtus, iga mõõte gruppi piirid, histogrammi iga tulba laius ja kõrgus.

3.2. Riist- ja tarkvara nõuded

Tarkvara kasutab *Flash Player 9* versiooni. Igale operatsioonsüsteemile on järgmised minimaalsed *riistvara nõuded [8]:*

- Microsoft Windows®: Intell® Pentium® II 450MHz või kiirem protsessor (või sarnane), 128 MB operatiivmälu.
- Macintosh: Power PC® G3 500MHz või kiirem protsessor, Intell Core Duo 1.33 GHz, 128 MB operatiivmälu.
- Linux®: Kaasaegne protsessor 800MHz, 512 MB operatiivmälu, 128 MB videomälu.

Flash Player 9 ise on toetatud järgmise operatsioonsüsteemide ja brauserite poolt:

Microsoft Windows:

Platvorm	Brauserid
Microsoft Windows Vista	Microsoft Internet Explorer 7, Firefox 2.0, AOL 9, Safari 3.x või vanem
Microsoft Windows XP	Microsoft Internet Explorer 6 või vanem, Firefox 1.x, Firefox 2.x, Mozilla 1.x või vanem, Netscape 7.x või vanem, AOL 9, Opera 7.11 või vanem, Safari 3.x või vanem
Microsoft Windows Server 2003	Microsoft Internet Explorer 6 või vanem, Firefox 1.x, Firefox 2.x
Microsoft Windows 2000	Microsoft Internet Explorer 5.x, Firefox 1.x, Firefox 2.x, Mozilla 1.x, Netscape 7.x või vanem, AOL 9, Opera 7.11 või vanem
Microsoft Windows Millenium	Microsoft Internet Explorer 5.5, Firefox 1.x, Mozilla 1.x, Netscape 7.x või vanem, AOL 9, Opera 7.11 või vanem
Microsoft Windows 98	Microsoft Internet Explorer 6.0 või vanem, Firefox 1.x, Mozilla 1.x, Netscape 7.x või vanem, AOL 9, Opera 7.11 või vanem

Macintosh:

Platvorm	Brauserid
Mac OS X v10.1 või vanem (Power PC)	Firefox 1.x, Mozilla 1.x, Netscape 7.x või vanem, AOL for Mac OS X, Opera 6 või vanem, Safari 1.x või vanem
Mac OS X v10.4.x või vanem (Intel)	Firefox 1.5.0.3 või vanem, Opera 6, Safari 2.x või vanem

Linux:

Platvorm	Brauserid
Red Hat Enterprise Linux 3 update, RHEL 4 update 4	Firefox 1.5.0.7 või vanem, Mozilla 1.7.x või vanem, SeaMonkey 1.0.5 või vanem
Novell Suse 9.x või 10.1	Firefox 1.5.0.7 või vanem, Mozilla 1.7.x või vanem, SeaMonkey 1.0.5 või vanem

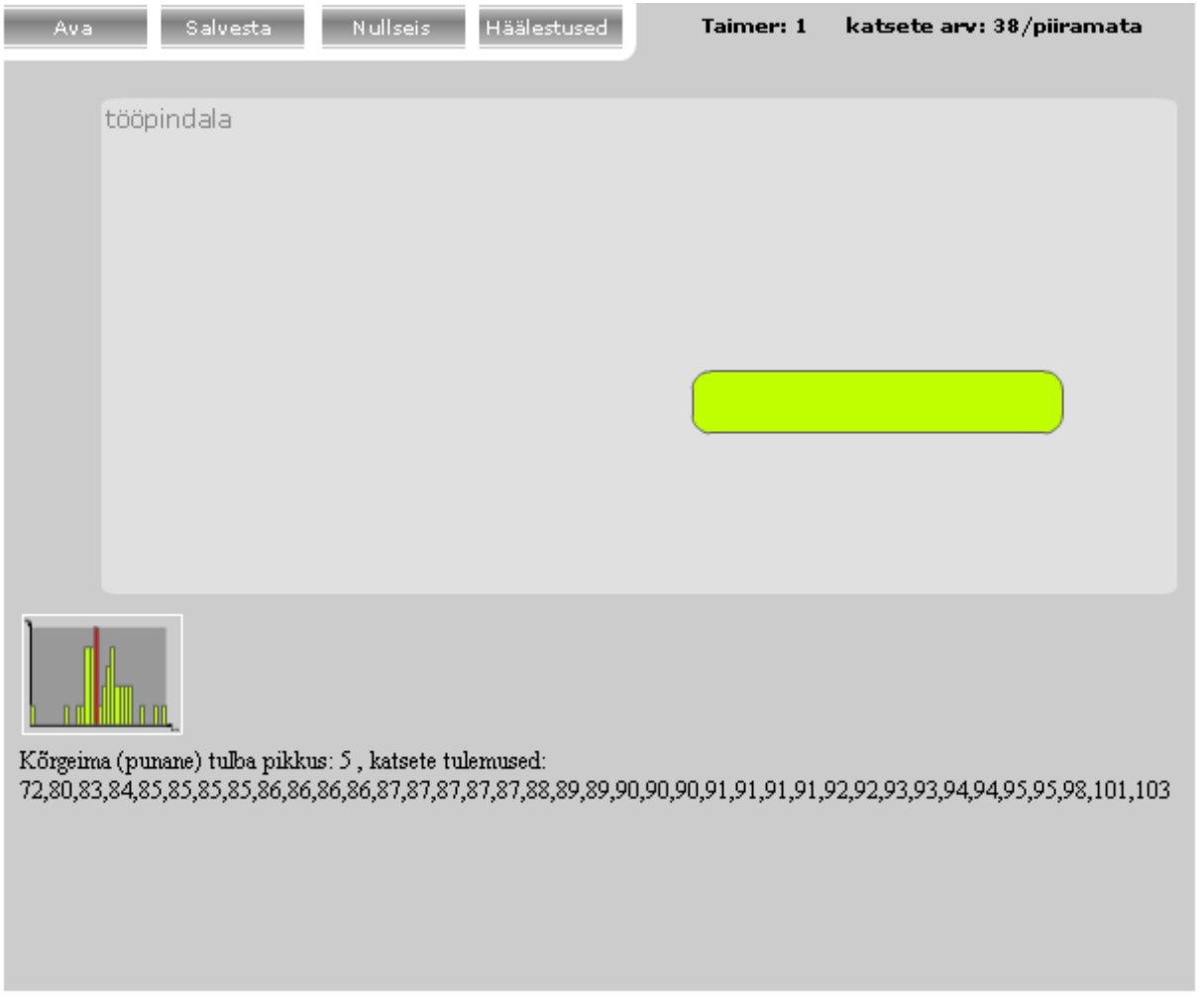
Solaris:

Platvorm	Brauserid
Solaris 10	Firefox 1.5.x või vanem, Mozilla 1.7.x või vanem

3.3. Installeerimine, käivitamine

Programm ei vaja arvutile lisapaigaldamist, kui enne oli juba installeeritud mingi nõuetes nimetatud operatsioonsüsteem brauseriga, kus on installeeritud *Flash Player 9* või selle vanem versioon. Juhul kui brauseris enne ei olnud instaleeritud *Flash Player*, siis selle kontroll toimub automaatselt ja pakutakse kohe värske versiooni paigaldada. Kui on soov kasutada programmi „offline“ režiimis, siis tuleks arvutile alla laadida failid programm.swf, see on kompileeritud fail, ja programm.html, mille käivitamisega avaneb veebileht, mis ei vaja Internetiga ühendust. HTML lehe käivitamine oli kasutatud selliseks, et kasutajale ei peaks paigaldama arvutile *Flash Player* versiooni, mis on suuteline ilma brauserita kompileeritud *.swf* faile tööle panna.

3.4. Graafililine liides



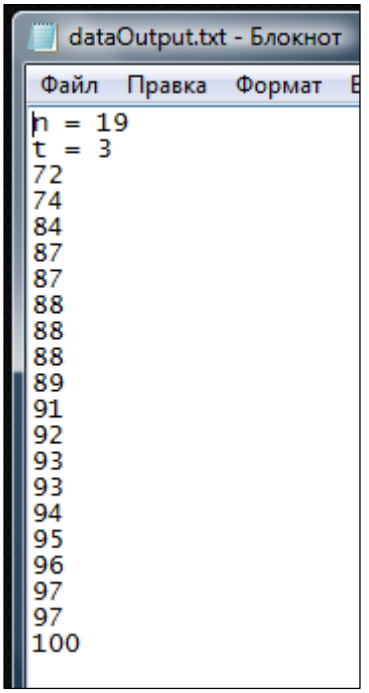
Joonis 3.2 Programmi ülevaade

Graafiline liides on ehitatud niimoodi, et kogu kasutamise protsess oleks lähedane laua arvuti rakendustele ja kasutajal ei tekki raskusi omandada kasutamist intuitiivselt mõistliku aja piiri raames. Sihina oli samuti tööprotsessi toimumine ühes kohas, et kasutajale ei oleks vajalik andmeid otsida mitmest kohast ja tulemuste hulk oleks kohe nähtav ja piisavalt kirjeldatav. Kohe pärast programmi käivitamist on peidetud võimalus näha mõõtmiste tulemusi ja praeguse maksimumi. Selle põhjus seisneb selles, et kasutajale ei anna vihjet hetkeseisu kohta ja selle pärast ei korrigeeri ta oma tööprotsessi. Tulemusi on võimalik kohe nähtavateks teha *Häälestused* osa kaudu.

Graafilises liideses on kasutusel järgmised nupud:

Joonis 3.3 Nupud

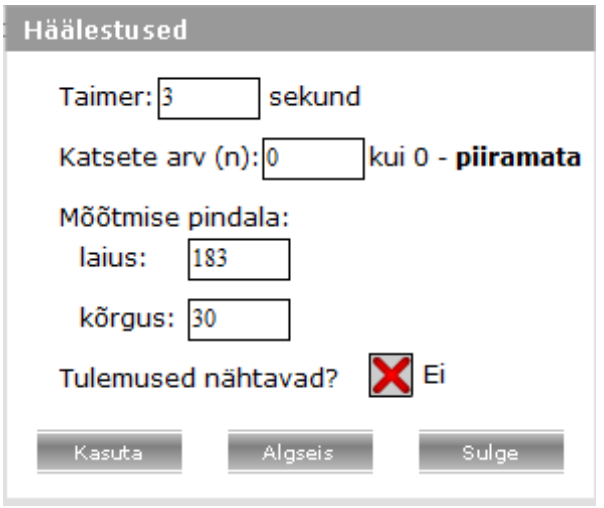
- ***Ava*** – vajutades avaneb hetkese operatsioonsüsteemi põhine aken, kus pakutakse valida andmete sisaldava failit. Aega säästamiseks on sisseehitatud filter, et näidatakse ainult failid *.txt* laiendusega. Kui kasutaja valib ühte faili ja vajutab *Ava* nuppu, siis rakendus võtab kasutusele tekstifaili ja loeb sisse andmeid. Nende andmete põhjal tehakse kõiki arvutusi, et histogrammi joonistada. Faili sisu peab olema formateeritud nii, nagu kirjeldatud järgmises osas. Šabloonina saaks kasutada ka joonisel 3.4 kujutatud faili sisu. Selle töö lisana on pannud üks näidefail, mis oli saadud programmi kasutades.
- ***Salvesta*** – vajutades pakutakse kasutajale salvestada kõiki andmeid tekstifailina laiendusega *.txt***.** Tekstifaili ehitus on järgmine: *n* – katsete arv, *t* – taimeri näide ja pärast seda tulevad kõike mõõtmiste tulemused, kus iga tulemus on eraldatud uue reaga. Pildil antud juhul tulemused on numbrid vahemikus *72 - 100*. Reavahetuse põhjus on loomulik – kasutajal võib tekkida vajadus kasutada andmed teises analüütilises tarkvaras, näiteks *Microsoft Excel*. Kuid tavalised komaga eraldatud andmed võivad vajada transponeerimist, mis ei ole mugavaimaks töötlemise viisiks. Sisse loev fail peab olema moodustatud sarnasel viisil, struktuur on vigane, siis selle töö raames loodud programm ei ole suuteline korrektselt kõik andmeid kasutada.



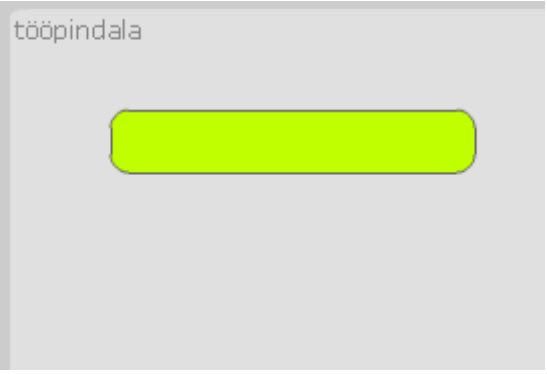
Joonis 3.4 Programmi tulemusfail

- ***Nullseis*** – see nupp on tehtud selliseks juhuks, kui tekkib vajadus kõike hetkeseisu tulemusi nullida ja alustada töid uuesti. Lisaks andmete kustutamisele joonistatakse järele ka jooksva histogrammi. Turvalisuse tagamise hoolduses on sisseehitatud mehhanism, et küsida kasutajalt, kas ta on kindel omal soovil nulliseisule viimise tegemises.

- ***Häälestused*** – programmi eraldine ja väga oluline osa, kus kõik seadused on võimalik muuta vastavalt oma soovile. Häälestuse liigid on järgmised: *taimer*, *katsete arv (n)*, *mõõtmise pindala laius ja kõrgus*, *tulemuste nähtavus,* nupud *Kasuta, Algseis* ja *Sulge. Taimer* määrab iga mõõtmise pindala ümberpaigaldamise tsükli, kus väärtuse ühikuna on võtnud 1 sekund. Kui kasutaja määrab ise uut aega, siis toimuv juhuslik objekt ümberpaigaldamise protsess seisatakse ja pannakse uuesti tööle ainult sel juhul, kui kasutaja teeb uut hiirega vajutamist objektile. *Kastese arv (n)* on abivahend, kus kasutaja määrab arvuna sobiva katsete arvu, et kasutaja ise ei peaks selle jälgima. Kui määratud arv on täis, siis antakse vastav teade, et töö on lõppenud ja pannakse taimerit seisma. Mõõtmise pindala ühikuna on võetud pikslid. Antakse võimalus muuta nii objekti laiust, kui ka kõrgust. Vastavad muudatused kehtivad kohe peale *Kasuta* nuppu vajutamist. Viimane punkt on *tulemuste nähtavus*, see osa kohe aktiveeritakse, ilma *Kasuta* nuppu vajutamiseta ja programm teeb nähtavaks kogu mõõtmiste tulemusi. Kõik ülejäänud osad vajavad aktiveerimiseks nuppu *Kasuta* vajutamist. Ise *Häälestused* paneel pannakse kinni sama *Häälestused* või *Sulge* nuppu kasutades. *Algseis* nupu kasutades on võimalik programmi viia seisule, millises ta asub kohe pärast käivitamist.



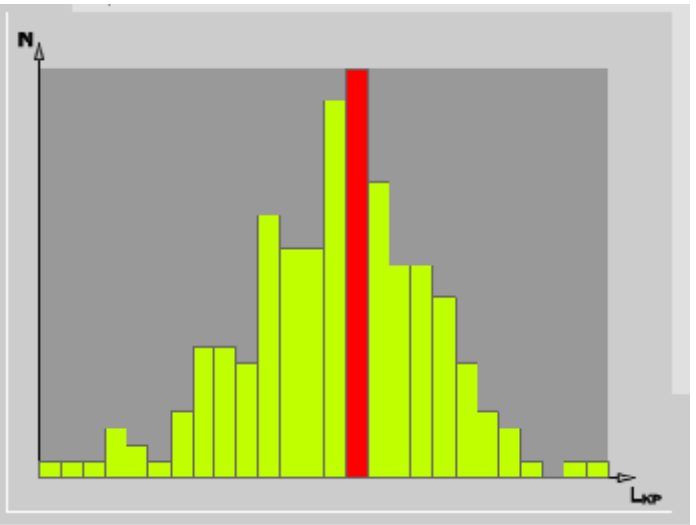
Joonis 3.5 Häälestused



Joonis 3.6 Tööpindala ja objekt

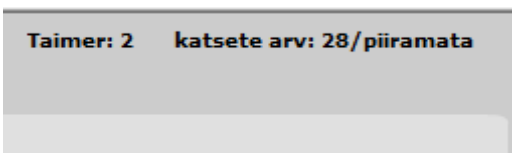
Mõõtmiste tegemiseks on kasutuses elemendid nagu mõõtmise- ja tööpindala. Üldiselt nad on riiskülikud ja esimene nendes on element (antud juhul roheline) mis määrab sihi kasutajale. Kasutaja hiire nuppu vajutamisega peab määrama pindala tsentrit. Vaikimisi tema parameetrid on valitud niisugusel viisil, et kõrgus oleks tunduvalt väiksem kui laius, sellega kasutajale on tunduvalt raskem otsustada, kus asub tõeline tsenter. Ümmardatud nurgad on

implementeeritud sama idee arvestades. Aga soovi korral *Häälestused* osa kaudu on võimalik muuta laiuse ja kõrguse parameetreid, mis pakub lisa võimalusi uurimiseks. Teine element - tööpindala (antud juhul helehall) määrab piire, milliste sees mõõtmisepindala muudab oma positsiooni.



Joonis 3.7 Histogramm

Katsete tulemuste abil ehitatakse histogrammi. On kasutatud eeldus, et iga tulba laius on võimalik proportsionaalselt histogrammi pindalaga arvutada, sest kõik väärtused on diskreetsed ja kogu mõõtmise protsessi ajal ei tekkitakse liiga suurt variatsiooni. Sama eeldus tehtib ka iga tulba pikkuse jaoks. Iga tulba kõgrus määrab kui palju liikmeid on iga mõõtmisegruppi sees, kus punase värviga on märgendatud tulp maksimaalse pikkusega. Tulpade arv määrab mõõtmisegruppide elementide arvu. Telgjoontel asuv tähistus *N* näitab erinevate tulemuste korduste arvu ja L_{KP} on keha laiuse keskpunkt. Kasutajale näidetakse histogrammi arvutuste tulemusena, kus midagi muuta ei ole võimalik, sellega on tagatud võimalus kontrollida ja võrrelda arvutusi tehtud paberil *Juhuvea uurimise juhendi* [6] abil ja selle töö raames loodud programmi kasutusel.



Joonis 3.8 Abielemendid

Programmi paremas ülevas nurgas asub abiinfo paneel, kus näidetakse järgmisi parameetreid: *Taimeri* näitarv ja tehtud/jäänud *katsete arv (n)*. Taimeri osa uueneb iga sekund, et anda kasutajale vihjet, millal toimub järgmine mõõtmisepindala ümberpaigutamine. Katsete arvu vasaku poolel (joonis 3.8) näidetakse hetkel tehtud katsete arv, paremal poolel on kogu katsete arv (joonisel 3.8 piiramata). Kui katsete arv *Häälestused* osas on valitud 0, siis kirjutatakse „piiramata“ ja programmi teade, mis kajastab, et mõõtmiste arv on täis, mitte kunagi ei näideta.

3.5. Programmis kasutatavad funktsioonid

Programmi kood on jagatud funktsionaalselt. Iga funktsioon on vastutav kindla tegevuse eest, et tagada vajadusel võimalusi suhteliselt mugavalt ja kiiresti laiendada funktsionaalsust.

Olulised meetodid ja osad on:

Main class *extends MovieClip* – klassi nimi on pannud Main, kuna ajalooliselt on loodud niisugune kord, kui peaklass alati on Main nimega, sellega programmeerijale on lihtsam otsustada, millega tegu on. *Extends MovieClip* osa tähendab, et klass laiendab *MovieClip* ülemklassi, aga ülemklass ise tagab baas graafikaga, animatsiooniga ja objektidega töö toestust [13].

Konstruktorid:

- *Main()*, kasutus: **Main()** - peakonstruuktor, tema nimi peab klappima klassi nimega, vastasel juhul kompileerimise ajal antakse veateadet, see on standaartne nõude *ActionScript 3.0* jaoks.

Olulised meetodid:

- *drawRect,* kasutus: **function drawRect():void** – meetod vastab mõõtmisepindala joonistamise protsessi eest. Kasutati standartsed võtmed nagu *beginFill* ja *drawRoundRect*, et tagada parameetriliselt joonistamist. Kasutatakse eeldefineeritud muutujatena väärtusi - asukoht, kõrgus, laius, nurkade ümardus ja tausta värv.
- *histOut,* kasutus**: function histOut(event:MouseEvent):void** – on väike abimeetod, mis tegeleb ainult histogrammi peegeldamisega. Selle põhjuseks on järgmine – *Flash* maailmas koordinaadid (0,0) asuvad ülevas vasakus nurgas, selleks et histogrammi koordinaadid oleksid rohkem loomulikul kujul, tehakse vastav peegeldus. Järgmine meetod *histOve*r on väga sarnane ja teeb ainult pöörd operatsiooni.
- *fillHistogramm,* kasutus: **function fillHistogramm():void** - dünaamiliselt loob histogrammi, uuendatakse iga kord, kui tehakse kasutajapoolne nuppu vajutamine mõõtmisepindalale (kutsutakse välja meetodi *overFunction* poolt). Esimesena tehakse kontroll, kas histogramm oli enne pannud stseenile või mitte. Positiivse vastuse juhul

eemaldatakse vanemat versiooni, et ekraanil oleks alati värskeim histogramm. Järgmisena võetakse kõik kasutajapoolseid andmeid massivina, sorteeritakse neid suurenemise suunas ja tehakse järgmised arvutused: minimaalne, maksimaalne mõõtmiste väärtused ja nende vahe, iga histogrammi tulba proportsionaalselt pikkuse ja laiuse ühikud, otsustatakse, kui palju andmeid kuulub igale tulbale, ja moodustatakse vastavalt sellele uus andmete massiiv. Viimase massiivi abil välja arvutatakse iga tulba tegelik kõrgus. Kui kõik arvutused on tehtud, alustatakse vahetult histogrammi graafiliselt joonistamist. Selliseks luuakse uus objekt tüübiga *MovieClip* [13] ja tsükliliselt kujutatakse vajalik tulpade hulk. Graafika elemendid nagu telgjooned on eeldefineeritud ja neid hoiakse failis *.fla* laiendusega. Siin tuleb mainida, et koodis on jäänud vana algoritm histogrammi joonistamiseks. Vanas versioonis kogu arvutus toimus täpselt nii, nagu kirjeldatud *Juhuvea uurimine laboritöö* [6] sees. Miks kasutatav algoritm oli muudetud on seletatud histogrammi kirjelduse osas, samas peatükis.

- *onOpenClick,* kasutus: **function onOpenClick(event:MouseEvent):void** – käivitatakse „kuulaja“ [13] poolt juhul, kui kasutaja oli vajutanud *Open* nuppu, et avada eelsalvestatud faili. Esiteks pannakse tööle filter, mis võtab vastu ainult failid laiendusega *.txt*, sellega kiirendatakse otsing failisüsteemis ja kontrollitakse, et kasutatakse õige fail. Luuakse uus „kuulaja“ funktsiooniks *selectHandler* ja kasutatakse käsku **.***browse*, et kasutajale avatakse operatsioonsüsteemipõhine aken, kust on võimalus valida sobivat faili. Kuulajat *.SELECT* pannakse tööle juhul, kui kasutaja valis faili alla laadimiseks. Pärast seda käivitatakse järgmine funktsioon *selectHandler*.
- *selectHandler,* kasutus: **function selectHandler(event:Event):void –** kui see meetod oli käivitatud ülalnimetatud funktsiooni poolt, pannakse tööle sisseehitatud meetodit *.load()* ja toimub vahetult faili sisse lugemine, kui see on edukalt tehtud käivitatakse järgmine meetod *completeHandler*.
- *completeHandler*, kasutus: **function completeHandler(event:Event):void** – funktsioon saab sisseloetud faili eelmise meetodi *selectHandler* poolt. Ise meetod *completeHandler* vastab operatsioonide eest, et töödelda andmeid, mis on sisseloetud faili sees. Esimesena kasutatakse töötlemise meetodit *readUTFbytes*, et avatud fail oleks tekstina. Pärast ka standartse meetodi *split()* abil võetakse vajalikud teksti read. Nende ridade põhjal salvestatakse programmi muutujadena taimeri nädendit ja tehtud katsete arvu, aga ülejäänud väärtused kasutatakse histogrammi ümberjoonistamiseks. Viimasena

kutsutatakse funktsioon *fillHistogramm()* vahetult histogrammi ümbermoodustamiseks.

- *onOptionsClick*, kasutus: **function onOptionsClick(event:MouseEvent):void** – kutsutatakse välja juhul, kui kasutaja vajutab *Häälestused* nuppu. Meetod ise kasutab *Flash Tween* klassi, et *Häälestuse* akna sujuvalt ekraanile tuua. Järgmisel kutsul pannakse *Häälestused* aken kinni. See funktsionaalsus on realiseeritud *Boolean* tüübi muutujate kaudu.
- *onCloseClick,* kasutus: **function onCloseClick(event:MouseEvent):void** – väike abimeetod, et *Häälestused* osa kinni panna. Töötab eelmise meetodiga *onOptionsClick* sarnasel viisil.
- *onTimerComplete,* kasutus: **function onTimerComplete(event:TimerEvent):void** – kutsutatakse ainult sellisel juhul kui taimeri tsükkel on lõppenud. Kui see juhtus nullitatakse taimerit ja pannakse seda uuesti tööle meetodite *timer.stop()* ja *timer.start()* abil. Pärast käivitakse mõõtmisepindala ümberpositsioneerimise meetodit – *positionRandomisator()*. Sellega on tagatud tsükliline mõõtmisepindala ümberpositsioneerimine.
- *onResetClick,* kasutus*:* ***function onResetClick(event:MouseEvent):void*** – paneb tööle järgmist meetodit *reset()*. Loodud selliseks, et meetodit *reset()* oleks võimalik kutsuda mitte ainult hiire sündmuste kaudu, aga ka teiste funktsioonide poolt. Käivitatakse juhul, kui kasutaja vajutab *Nullseis* nuppu.
- *reset(),* kasutus: **function reset():void** – viib programmi algseisu, teiste sõnadega nullitakse mõõtmiste arvu, kõike arvutusi mis oli enne tehtud, massiivid andmetega ja käivitatakse meetodit, et värsket histogrammi ekraanile tuua - *fillHistogramm()*. Sellisel juhul võib olla kasutatud, kui töö käigus tekkis mingi viga või takistus, selleks et uuesti töid alustada.
- *checkClick,* kasutus: **function checkClick(event:MouseEvent):void** – kontrollimise meetod, jälgib *Häälestused* osas asuvat *kontrollkasti.* Kui kasutaja muudab selle seisundit, siis muudetakse vastavat muutujat, mille abil otsustatakse, kas kujutada ekraanile mõõtmiste tulemusi, või mitte.
- *onSaveClick,* kasutus: **function onSaveClick(event:MouseEvent):void** - käivitatakse juhul, kui kasutaja vajutab *Salvesta* nuppu. Pärast tekkitakse dialoogaken, kust on võimalik valida salvestatava faili asukoha ja nimi. Fail on *.txt* laiendusega ja sisaldab *katsete arvu n*, *taimeri väärtust t* ja kõike mõõtmiste tulemusi uue reaga eraldatud.

Salvestamine toimub funktsiooni *.save()* kaudu, andes argumentidena sobiva faili nime ja laiendust.

- *overFunction,* kasutus: **function overFunction(event:MouseEvent):void** - meetod käivitatakse iga kord, kui kasutaja vajutab hiire nuppu mõõtmise pindala sees, et arvutada suhtelist hiire kursori positsiooni. Koordinaatide kättesaamine toimub sisseehitatud meetodi *mouseY* kaudu. Pärast toimub mõõtmise arvu loendajat *n* ühe ühiku võrra suurendamine ja pannakse tehtud mõõtmise tulemust andmete massiivi sisse. Viimasena kutsutatakse meetodeid taimeri uuendamiseks - *reset()* ja histogrammi ümberjoonistamiseks - *fillHistogramm().*
- *onKasutaClick,* kasutus: **function overFunction(event:MouseEvent):void** – see funktsioon käivitatakse juhul, kui kasutaja *Häälestuse* osas vajutab *Kasuta* nuppu. Tema ülesandeks on võtta uuesi kasutajapoolseid väärtusi ja panna neid tööle. Siin tehakse ka kontroll, kas uued andmeid oleksid mingi piiride sees ja juhuslikult ei olnud pannud valesti.
- *positionRandomisator,* kasutus: **function positionRandomisator():void** - meetod juhuslikult paigutab ümber mõõtmise pindala. Uued koordinaadid arvutatakse *Math.random( )*meetodi kaudu, mis on *Flash* meetodite kogumi osa ja annab välja juhuslike arve piirides 0 kuni 1.
- *onAlgClick,* kasutus: **function onAlgClick(event:MouseEvent):void** – meetod viib programmi algseisundisse. Esiteks see kasutab *.reset()* meetodit, väärtustakse kõik parameetrit ja kasutatakse uued väärtused programmi tööle.
- *frameEvent,* kasutus: **function frameEvent(event:Event):void** - meetod värskendab kogu stseeni ja sellega kaasa ka andmeid, milliste nähtavus peab toimuma reaalses ajas. *Flash* kutsub selle funktsiooni iga kaadri muutmisel (antud programmi juhul 24 korda sekundis). Niimoodi garanteeritakse kohest informatsiooni värskendamist ekraanil ja vajadusel ka värskendamist. Loodud programmi juhul uuendatakse mõõtmiste tulemuste väärtuseid, hetkeseisu mõõtmiste väärtuste maksimumi ja taimeri näidendit. Eraosana on kommenteerinud välja koodiread, milliste abil kujutatakse ekraanil hiire kohalike ja globaalseid koordinaate. Need on olulised väärtused vigade kotrollimiseks, millised on jäänud võimaliku tuleviku programmi testimiseks.

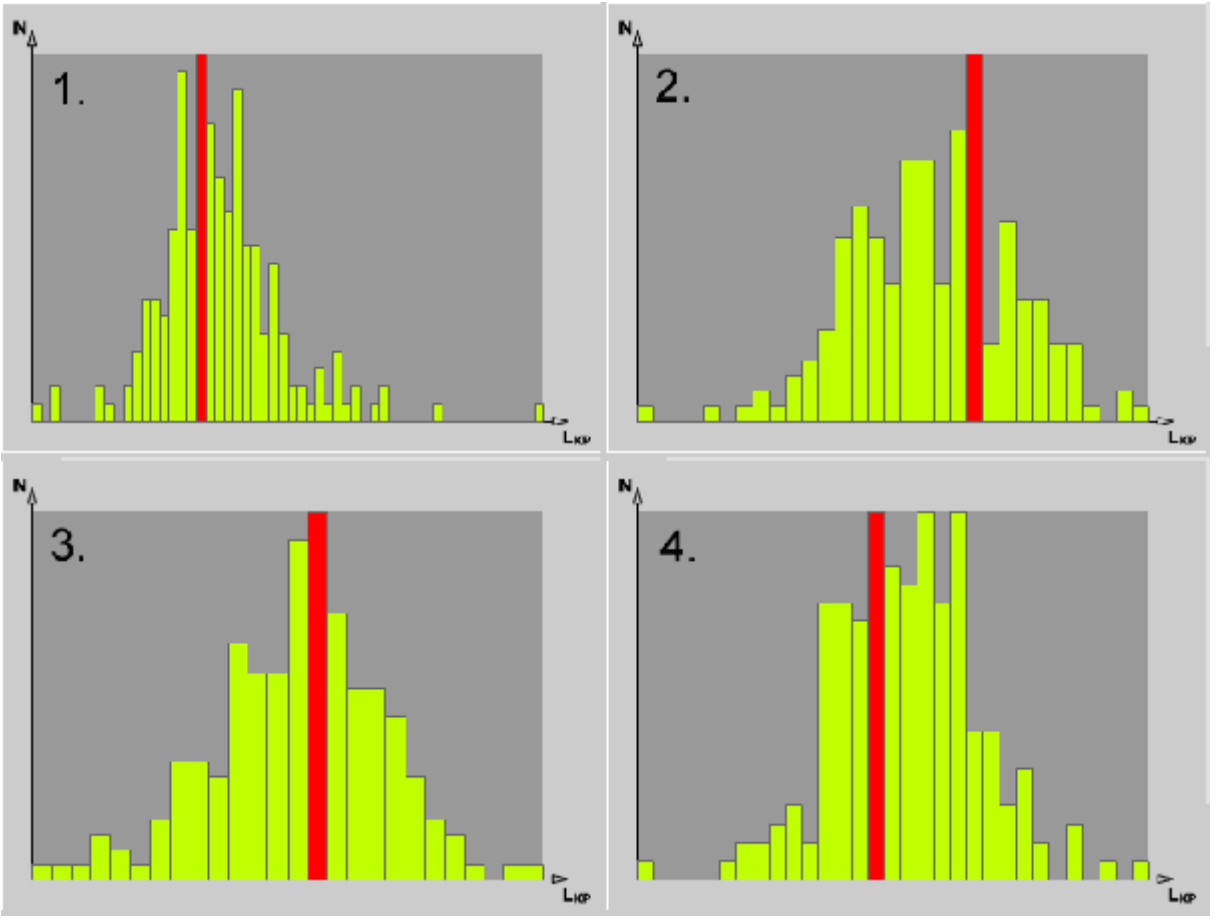
Kasutatud parameetritena on palju muutujaid ja kõike nende kirjeldamine ei ole mõistlik selle

töö raames, kuid huvi juhul on võimalik neid leida lähtekoodi failis. Siin tuuakse ainult olulisimaid parameetreid, milliste väärtused on võimelised funktsionaalsust muuta:

- *bgColor:uint=0xBFFF00* – objekti tausta värvi väärtus;
- *cornerRadius:uint=18* – objekti nurkade ümmarduse koefitsient;
- *myTimer = new Timer(1000, appearanceTimer)* – taimeri väärtus, kus 1000 tähendab, et alusena on võtnud 1000 millisekundit;
- *docFilter:FileFilter = new FileFilter("Documents", "*.txt")* – filter, mis toetab ainult *Documents* faili tüüpi ja avab neid laiendusega *.txt*.

4. Programmi testimine, komplikatsioonid ja arutelu

Hea tava kohaselt programmi testitakse konkreetselt kasutajate peal fikseerides kõik tekkinud komplikatsioonid või ebamugavused kasutamisel. Et antud töö ülesanne ei olnud massilise testimise korraldamine, siis seda pole tehtud. Et leida üles kõige jämedamad vead, siis sel eesmärgil sai antud programmi testitud nelja inimesega. tingimusena oli teha 200 katseid, vaikimisi väärtuste kasutusele võttes. Mõõtmiste tegemiseks kulunud aeg oli 2-5 minutit ja mitte keegil ei tekinud probleemid programmi tööle panemisega. Igal kasutajal oli *Windows* pere operatsioonsüsteemid ja tulemusena olid saadud failid laiendusega *.txt.* Nende failide põhjusel koostatud histogrammid on toodud järgmisena joonistusena.



Joonis 4.1 Histogrammid

Küsitlus näitas, et inimene, kelle histogramm on märgendatud numbriga *1*. mängib palju

mänge, kus on vaja tihti tulistada ja kiiresti reageerida. Saadud objekti tsentri määramise täpsus on võrdeliselt kõrge, see karakteriseeritakse maksimaalse tulba kõrgusega. Teiselt poolt on olemas ka väärtused, milliste asukoht on liiga kaugel tsentrist, võrreldes teiste histogrammi elementidega. See on hästi nähtav nii äärmise väärtustega, kui ka tulpade mastaabiga. Kõike tõenäoliseks põhjuseks on - kiirustamine. Inimene, kelle hostigrammon märgendatud numbriga *4.* ausalt ütles, et ei tahtnud jõupingutusi teha ja tegi kõik 200 katseid hoolimata täpsusest. Uurimise raames saadud tulemus on väga kirjeldav. Histogrammi tsentriline osa on liiga lai, kus on palju tulpasid sama kõrgusega. Lisaks sellele mõned tulbad ei erine maksimumist märgatavalt, tegelikult on olemas kolm maksimumi.

Arendamise ajal tekkisid erinevad raskused, suur hulk nendest oli võimalik lahendada *ActionScript 3* dokumentatsiooni kasutades. Aga mõned nendest tekitasid tõelisi probleeme. Näiteks, *Windows* pärane tekstifaili salvestamine. Probleem on sellises, et reavahetus määratakse kahe reavahetuse võtmetega *–' \r\n'*. Selle lahendamine võttis tunduvalt palju aega, sest ei olnud võimalik aru saada, miks fail salvestatakse korrektse võtiga *'\r'* või *'\n'*, aga lõpus saadud tekstifaili sees reavahetust ei esine. Teiseks suureks probleemiks oli üleminek pidevatest väärtustest diskreetsetele. Kui õpilane sooritab *Juhuvea uurimise laboritööd* [6], siis kõik väärtused on pidevad, aga antud juhul, kui tegu on arvutipärase koordinaatidega - sama lähenemine ei ole enam võimalik. Selle lahendamiseks olid tehtud ja testitud eeldused histogrammi tulpade laiuse, kõrguse ja mastaapi kohta. Testide tulemusena saadud histogramm on korrektne.

Perspektiivis programmi saab vaadelda lähtepunktina ja selle funktsionaalsus saab olla ulatuslikult laiendatud. Uurimiseks on mõistlik lisada veel objekte niisugusel viisil, et nende vorm ja kujutus tooks lisaraskuse kasutajale. Selle sorti takistusi on võimalik uurida histogrammiga ja kui piisavalt mõõtetulemusi korjata, siis on võimalik teha mahukama analüüsi. Autori arvamusel programmile saab olla lisatud ka võimsam andmete analüüsi ja visualiseerimise mehhanism. Näiteks histogrammile oleks kirjeldav lisada normaaljaotuse tihedusefunktsiooni graafikut. Veel huvi saaks pakkuda standartse hiire kursori asendamine. Kui näiteks see on mitte nool, aga punkt, kuidas siis muudab tulemus? Parema arusaamiseks programmile oleks mõistlik lisada ka õppematerjalle või isegi lühikest demot, millises seletakse kogu tööprotsessi ja antakse vihjet, kuidas programmi õigesti kasutada.

Kokkuvõte

Juhuvea uurimine [3] on lai valdkond, mille uurimiseks ja millest arusaamiseks on võimalik kasutada palju erinevaid meetodeid. Üks lähenemistest on laboritööde tegemine. Tartu Ülikoolis tuleb tudengitel, kes soovivad saada füüsikuks või füüsika õpetajaks tegema laboritööd sama nimega – „*Juhuvea uurimine*“ [6].Nimetatud laboritöö annab küll hea ülevaate probleemist, kuid on mahukas ja vajab palju täpsust. Selle töö eesmärgiks oli simuleerida juhuvea tekkimist arvuti programmiga, võttes aluseks kirjeldatud laboritöö.

Käesolevas töös on antud ülevaade ja põhjendatakse, millist keelt ja platvormi kasutada. Tehti otsus, et antud töö raames on mõistlikum kasutada *Adobe Flash* [11] tehnoloogiat ja programmeerimiskeelt *ActionScript versiooniga 3* [12].

Tulemusena oli loodud programm, mis automatiseeris mõõtetulemuste kättesaamist ja simuleeris mõõtmise- ja juhuvea uurimise protsessi. Mõõtmine toimub arvuti hiire kasutamisel. Mõõdetava objekti parameetreid määrab kasutaja ainult silmadega. See on mugavam ja tõstab kogu töö kiirust. Mõõtmiste graafiline esitus toimub dünaamiliselt kogu programmi kasutamise ajal ja õpilastel on võimalus näha ja analüüsida tulemusi kohe. Kõige olulisem ja samal ajal ohtlikum oli eeldus, et diskreetseid väärtusi võttes on võimalik arvutipõhine mõõtmise protsess ja nende väärtuste põhjal saada mõõtmistulemuste jaotuvus, mis oleks sarnane normaaljaotusele. Tuli välja, et niisugune lähenemine on õige. Programm oskab andmeid töödelda, salvestada ja sisselugeda. Pärast sisselugimist tehakse kohene arvutusi. Selleks, et saadud andmeid visualiseerida programmis kujutakse histogrammi, mille dünaamiline uuendus toimub iga kord, kui kasutaja teeb uue mõõtmise. Vaatamata programmi võimalustele ei ole soovituslik kogu laboritöö asendamine, sest täielikuks arusaamiseks on parem, kui õpilane teeb kõik arvutused ja ka histogrammi ise *juhendi* [6] abil.

Testimiseks programm oli pakkutud neljale inimestele. Saadud tulemused olid visualiseeritud programmi abil ja oli tehtud lühike histogramme analüüsi. Selgus, et vaatamata sellele, et inimest oli neli, nende tulemused on kirjeldatavad ja piltlikud, andmete põhjal on võimalik teostada analüüsi ja teha järeldusi.

Selle töö autor eeldab - programmi funktsionaalsust on võimalik laiendada elemente lisades niisugusel viisil, et üliõpilane saaks proovida kätte saada tulemusi erinevatel viisil ja visualiseerida seda võimsamate vahenditega. Praegusel kujul on programm võimeline andmeid hankima ning kiiremini illustreerima andmeid *Juhuvea uurimise* [3] tööst aru saamiseks.

Computer application to automate physics lab work „Random error research“

Bachelor thesis (6 ECP)

Lavrenti Tšudakov

Summary

At present, in the field of *„Random error research“* several approaches are taken as a research methods. One of them is lab work doing. In this bachelor work is introduced review of such work, which is mandatory for some students. Given lab work needs a lot of time and accuracy, there is a big possibility that results will be incorrect. In a whole student have to measure body placed in a galss tube, make calculations and build a bar chart, in order to be able to makse some conclusions, research random error, dispersion, deviation.

In order to help studens with better understanding of a *random error* nature and prepare them for a lab work doing, have been analysed possibilities how some part of a lab work can be replaced with computer application. As a result was choosen *Adobe Flash* platform and *ActionScript 3* programming language. As a powerful tool *Adobe Flash* provides all needed functionality and possibilities to make such application.

As a result by author of this thesis was developed and tested small application with a set of base functions. Application itself simulates random error emergence – rectangle is drawn on a stage, user should define possible middle (only horisontal direction is taken in action) of a given rectangle and point it with a mouse cursor. After it necessarry calculations are done and bar chart is constructed. Data is calculated dynamically, results can be saved and uploaded for a following usage. Data is stored as *.txt* file. Programm uses standart GUI elements as buttons and windows so understanding of a working process should be intuitive and fast. In a testing meanings given application was proposed to a four volunteers. Results were recieved as a *.txt* files and time spent on work with application was in range from 2 to 5 minutes. None of the testers faced with a problems during testing process. Some prast of a *„Random error*

research" lab work were successfully replaced with a computer programm. But it is not recommended to replace a whole lab work, given programm should be considred as a prior preparation and helping material, to make knowledge more wide and understanding more clear.

Kirjandus

[1] Random error. http://en.wikipedia.org/wiki/Random_error

[2] A. Aarma, V. Vensel. *Statiskita teooria põhikursus*. „Külim“, 2005.

[3] Error analysis.
http://felix.physics.sunysb.edu/~allen/252/PHY_error_analysis.html

[4] Experimental error.
http://www2.volstate.edu/tfarris/PHYS2110-2120/experimental_error.htm

[5] Error analysis: Systematic Vs. Random error.
http://canu.ucalgary.ca/map/content/erroranalysis/sysran/explain/index.html

[6] H. Voolaid. *Füüsika praktikumi tööjuhend I*, Tartu, 1988.

[7] Flash Player penetration.
http://www.adobe.com/products/player_census/flashplayer/

[8] Flash Player System Requirements.
http://www.adobe.com/products/flashplayer/productinfo/systemreqs/flashplayer9/

[9] ECMA Script.
http://www.ecmascript.org/

[10] Top Browser Share Trend.
http://www.netmarketshare.com/browser-market-share.aspx?qprid=1

[11] Adobe Flash.
http://ru.wikipedia.org/wiki/Adobe_Flash

[12] Actionscript 3, Adobe help.
http://help.adobe.com/ru_RU/ActionScript/3.0_ProgrammingAS3/WS5b3ccc516d4fbf351e63e3d118a9b90204-7ec7.html

[13] C. Moock. *Essential Actionscript 3*, O`Reilly Media, 2007. ISBN: 978-0-596-52694-8

Lisad

Lisadena selle tööle on pannud järgmised failid (failid asuvad CD plaadil):

- *programm.swf* – kompileeritud programm
- *Main.as* – lähtekoodi fail
- *programm.fla* – konteiner graafikaga, kasutab ka lähtekoodi kompileerimise ajal
- *programm.html* – HTML leht programmiga, et ”offline” režiimis programmi kasutada
- *dataOutput.txt* – näitefail mõõtmistega