**VILNIAUS UNIVERSITETAS**

**MATEMATIKOS IR INFORMATIKOS FAKULTETAS**

**PROGRAMŲ SISTEMŲ KATEDRA**

**LĖKTUVO TŪPIMO IR KILIMO TRAJEKTORIJŲ  
TRIMATIS VIZUALIZAVIMAS**

**3D VISUALISATION OF  
PLANE LANDING AND TAKE-OFF TRAJECTORIES**

**Bakalaurinis darbas**

Martynas Žuravliovas (parašas)

Darbo vadovas:

doc. Kristina Lapin (parašas)

Vilnius, *2016*

Turinys

[Įvadas 3](#_Toc479609087)

[Motyvacija 3](#_Toc479609088)

[Tikslas 3](#_Toc479609089)

[Uždaviniai: 3](#_Toc479609090)

[1. Idealios trajektorijos integracija 5](#_Toc479609091)

[2. Trajektorijos rizikos valdymas 7](#_Toc479609092)

[3. Nukrypimo nuo idealios trajektorijos atvaizdavimas 8](#_Toc479609093)

[4. Kameros 9](#_Toc479609094)

[5. Realaus reljefo pritaikymas terrain objektui 10](#_Toc479609095)

[Šaltiniai 11](#_Toc479609096)

[Rezultatai ir išvados 12](#_Toc479609097)

# Įvadas

Daugelyje kompiuterinių sistemų galima atrasti trūkumų, netikslumų, nepatogumų, tačiau tai kas liečia žmogaus, o šiuo atveju, kai analizuojami lėktuvų skrydžiai, daugelio žmonių saugumą, klaidų ir netikslumų pasitaikyti negali. Tokiuose oro uostuose, kur nėra galingų lėktuvo tūpimą valdančių sistemų, skrydžio vadovas yra žmogus reguliuojantis eismą. Kuriama sistema yra skirta tam, kad šis žmogus galėtų lengviau stebėti kaip leidžiasi lėktuvai, kuriems jis paskyrė vienokią ar kitokią nusileidimo procedūrą. Sistema turėtų aiškiai pateikti vykdomo skrydžio vaizdą.

## Motyvacija

Pagal žinomas koordinates vizualizuoti trajektoriją, kai visi trajektorijos duomenys yra žinomi iškart. Duomenys – koordinačių seka. Šiuo metu skrydžio vadovas duoda leidimą pilotui užimti tūpimo-kilimo erdvę. Patį tūpimą vykdo pilotas. Leidimas apima tūpimo procedūrą, kuri nustato visus tūpimo parametrus. Kuriama sistema turi parodyti skrydžio vadovui ar pilotas laikosi nurodytos procedūros. Sprendimas turi reikalauti kuo mažiau mintinių pastangų, turi būti paremtas atpažinimu, o ne mintiniais skaičiavimais. Ši sistema yra kuriama mažiems oro uostams, neturintiems instrumentinės tūpimo įrangos. Sistema turi veikti lidaro ir radaro pagrindu. Radaras neteikia duomenų mažame aukštyje, todėl lėktuvai yra nematomi skrydžių vadovui, tačiau aukštis netrukdo lidarui, taip atsiranda galimybė patikrinti, ar pilotas laikosi nurodymų.

## Tikslas

Sukurti programos prototipą, kuri leistų skrydžių vadovui ekrane stebėti lėktuvo nusileidimo ir pakilimo trajektorijas. Ruošiamas prototipas yra 2-ojo lygio skalėje iš 9, pagal NASA technologijų parengimo tvarką.

## Uždaviniai:

* Oro uosto tūpimo ir kilimo procedūrų (toliau darbe – ideali trajektorija) analizė.
* Idealios trajektorijos integravimas į esamą prototipą.
* Trajektorijos rizikų valdymas.
* Išanalizuoti būdus, kaip pažymėti nukrypimą nuo trajektorijos ir parodyti skrydžio vadovui atsiradusius netikslumus (Informacinės lentelės, su aiškiu būsenos atvaizdavimu, taip pat galimybė išjungti vieną ar kitą vizualų komponentą).
* Kameros buvimo vietos analizė bei galimybė persijungti į kitą perspektyvą, siekiant sumažinti skrydžio vadovo dėmesio poreikį.

Pirminis darbų eigos planas:

1. Pirmiausia reikėtų išanalizuoti idealios trajektorijos duomenų lentelę. Iš pateiktų duomenų išgauti trajektorijos koordinates (Sugalvoti kaip tai padaryti, turima informaciją reikia paversti koordinatėmis, o galbūt tinkamas variantas leisti naudotojui nuskaityti idealios trajektorijos parametrus ir atitinkamai pagal juos suformuoti objektų pozicijas, pvz: pakilimo taką padėti per centrą, paskutiniajai idealios trajektorijos koordinatei).
2. Išgavus koordinates, bus nupiešiamas tunelis (aptvarkyti nusileidimo taką, pagal idealią trajektoriją jeigu jo pozicija nebus dinaminė trajektorijos atžvilgiu).
3. Tuomet, nustatyti maksimaliai leidžiamo nukrypimo nuo trajektorijos atstumą (ieškoti informacijos), kuris bus mūsų piešiamo tunelio spindulys R. Šis dydis, nebūtinai turi būti konstanta, galima padaryti reguliuojamą tunelio spindulį, jeigu nėra žinoma, koks jis turi būti iš tikrųjų.
4. Nupiešus teisingą trajektoriją, sugalvoti, kokiu būdu bus nustatomas nukrypimas nuo trajektorijos (pirma mintis: kiekviename koordinatės taške tikrinti su trajektorijos tunelio rėžiais, pvz: idealios trajektorijos taškui pridedame spindulį ir lyginame ar tai nėra daugiau už realios trajektorijos tašką. Ir taip tikriname į visas.
5. Gavus požymį, ar lėktuvas skrenda teisingoje trajektorijoje pradėti analizuoti nukrypimo atvaizdavimą, pvz: papildomas skydelis ekrane, su spalvinėmis žymomis (išsiaiškinti, ką Unity siūlo naudoti 3D scenose).
6. Taip pat, įgalinti kitos kameros perspektyvą. Esamame skydelyje, paprastas mygtukas kameros vaizdui keisti.

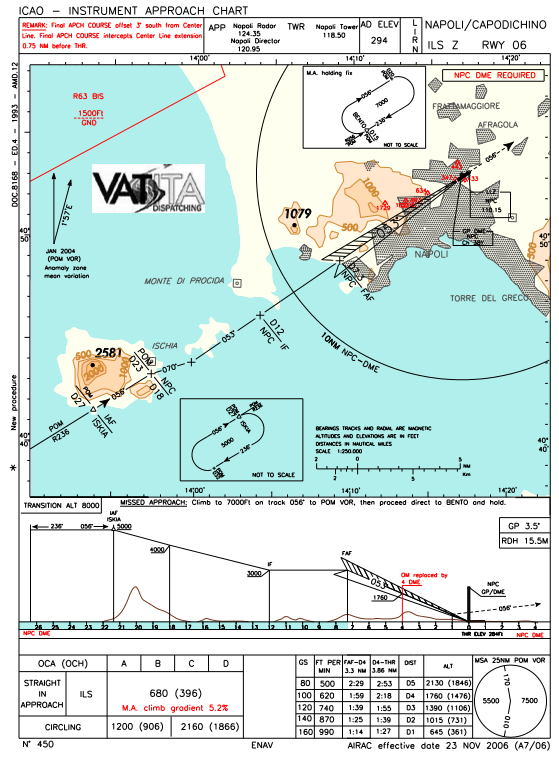
Kuriama sistema

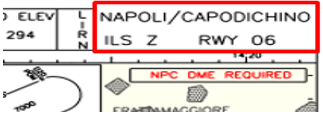
Eismo ore valdymo taisyklės

Oro uosto procedūros

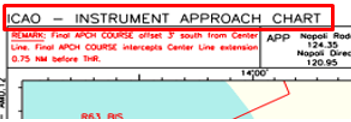
Esama lėktuvo pozicija, rizikos, skrydžio būsena

# Oro uosto tūpimo ir kilimo procedūrų analizė

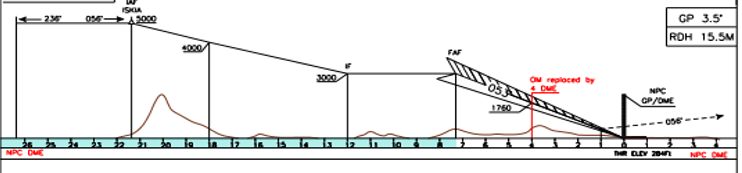
Procedūra yra sudėtinga, o sistemai automatiškai ją perskaityti yra dar sudėtingiau, todėl, net ir kuriant galutinį prototipo produktą, bandyti iš šios procedūros išgauti idealią trajektoriją yra netikslingas ir daug laiko kainuojantis darbas. Todėl nuspręsta idealios trajektorijos koordinates leisti pateikti pačiam naudotojui.

1. Oro uosto pavadinimas, paskirtos nusileidimo procedūros kodas –ILS Z bei nusileidimo kampas RWY 06 (24 pav.).

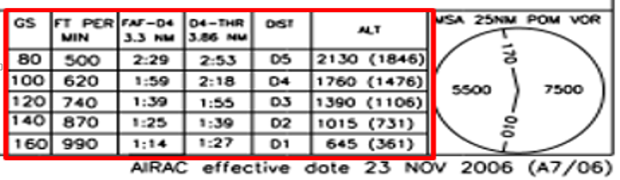
1 pav. Oro uosto pavadinimas, nusileidimo procedūros kodas, nusileidimo kampas.

2. Oro uoste esanti instrumentinė sistema (25 pav.).

2 pav. Instrumentinė oro uosto sistema

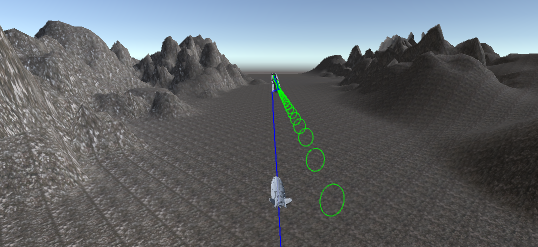
3. Paskirtos procedūros nusileidimo trajektorijos nustatymai. Šioje dalyje atvaizduojami atstumai iki tūpimo taško pradžios, bei kokiame aukštyje (matavimo vienetas – pėdos) tam tikru atstumu (matavimo vienetas – jūrmylės) iki tako pradžios turi būti lėktuvas (26 pav.).

3 pav. Idealios trajektorijos ašis ir jos nustatymai: aukštis, atstumas, nusileidimo kampas.

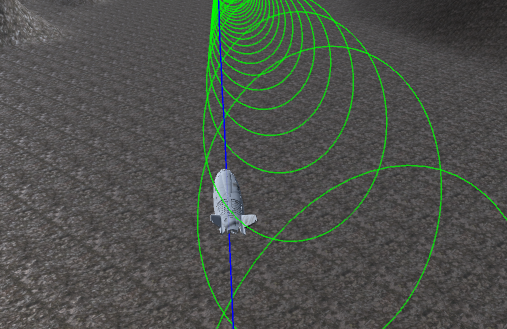
4. Detali lėktuvo judėjimo, paskirta procedūra, lentelė (27 pav.). GS(Ground speed) – horizontalus greitis matuojamas mazgais, FT PER MIN(Foot per min) – vertikalus greitis matuojamas pėdomis, FAF-04 3.3 NM – laikas, kada bus pasiektas FAF taškas judant konkrečiu greičiu.

4 pav. Lėktuvo teisingo judėjimo duomenys.

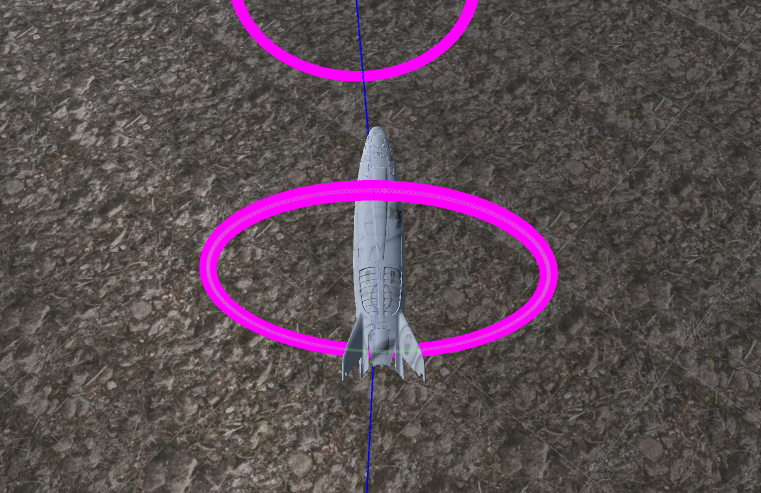
# Idealios trajektorijos integracija

Idealios trajektorijos nuskaitymui buvo panaudotas tas pats metodas, kuris nuskaito realios trajektorijos koordinates. Sukurtas atskiras idealios trajektorijos koordinačių masyvas. Pakoregavus esamą „PathFollower“ skriptą taip, kad tunelis būtų piešiamas naujai nuskaitytai idealiai trajektorijai. Taip gaunamas pirminis, realios skrydžio trajektorijos palyginimo su idealia, vaizdas (1 pav.).

5 pav. Pirminis realios skrydžio trajektorijos palyginimo su idealia vaizdas

Tačiau atskyrus šias dvi trajektorijas pasimato nauja problema – idealios trajektorijos tunelis piešiamas naudojant Gizmos objektus, kurie matomi tik programavimo rėžime ir yra pirminiai, t.y. lėktuvas, būdamas išorėje tunelio atrodo, lyg būtų po tuneliu, nes Gizmos objektai uždengia lėktuvą (2 pav.). Todėl šis tunelio piešimo būdas naudojant Gizmos yra netinkamas.

6 pav. Neaiški lėktuvo buvimo vieta, idealios trajektorijos atžvilgiu

Šiai problemai išspręsti, buvo sugalvota, vietoje Gizmos objektų naudoti realius žaidimo objektus. Tam prireikė sukurti naują objektą – 3D apskritimą. Unity standartinių žaidimo objektų bibliotekoje tokio objekto nėra, todėl jį tenka pasigaminti. Unity bendruomenės puslapyje yra patalpintas būtent tokio objekto sukūrimo skriptas[[1]](#footnote-1). Šio skripto pagalba, sukuriamas, vadinamasis, Torus objektas. Tačiau šis objektas yra tik nematomi kontūrai mums reikalingo apskritimo. Pasinaudodami Unity įrankiais, apvelkame kontūrus matoma medžiaga ir gauname tinkamą objektą idealios trajektorijos tuneliui formuoti. Suformavus tunelį, lėktuvo pozicija idealios trajektorijos atžvilgiu tapo aiškiai matoma (3 pav.).

7 pav. Aiškiai matoma lėktuvo pozicija, idealios trajektorijos tunelio atžvilgiu

# Trajektorijos rizikos valdymas

Besileidžiantis ar kylantis lėktuvas turi judėti pagal jam paskirtą procedūrą. Procedūroje yra pateikiama trajektorijos nustatymų lentelė, kurioje atvaizduojami atstumai iki tūpimo ar kilimo taško, bei kokiame aukštyje tam tikru atstumu iki tako pradžios ar pabaigos turi būti lėktuvas. Šie duomenys yra pateikiami idealiu nusileidimo ar kilimo atveju, tačiau jiem gali būti taikomos paklaidos, t.y. lėktuvo reali buvimo vieta gali skirtis nuo procedūroje nurodytos per tam tikrą toleruojamos rizikos paklaidą. Trajektorijų rizikos valdymui buvo sukonstruotas koordinačių paklaidos matavimo metodas. Šiam metodui paduodama dinaminė reikšmė – koordinatės paklaida. Šią paklaidą pridėjus bei atėmus iš idealios trajektorijos koordinatės gauname rėžius, kuriuose privalo atsidurti realios trajektorijos koordinatė konkrečiu skrydžio momentu. Įteruojant per realaus skrydžio koordinates, kiekvienoje iteracijoje iškviečiamas šis metodas, kuris atlieka paprastą patikrinimą ar reali koordinatė priklauso šiems rėžiams ar ne. Metodas grąžina reikšmę, kuri padeda nuspręsti ar lėktuvo skrydis vykdomas kaip numatyta. Tačiau problema atsiranda tada, kai reikia apskaičiuoti koordinatės paklaidą konkrečiam trajektorijos taškui, nes paklaidos dydis skiriasi idealios trajektorijos pradžioje ir pabaigoje. Taip pat, kiekviena koordinačių ašis turi savo paklaidos dydi, todėl atliekami skaičiavimai tampa dar sudėtingesni. Galbūt įmanoma trajektoriją padalinti į keletą etapų (procedūroje yra DST stulpelis su 5 reikėmem), kur kiekvienas iš jų turėtų savo rizikos reikšmes. Taip būtų galima nesudėtingai apsibrėžti, kada ir kokias paklaidas perduoti rizikos skaičiavimo metodui.

# Nukrypimo nuo idealios trajektorijos atvaizdavimas

Nukrypimo atvaizdavimui sukuriamas UI dialogas (performuluoti) su spalviniu indikatoriumi (kodėl būtent toks sprendimas). Pasinaudojant anksčiau minėtu metodu, kuris grąžina požymį apie lėktuvo nukrypimą nuo trajektorijos, keičiame šio indikatoriaus spalvą. Esant nukrypimui užsidega raudona spalva, priešingu atveju dega žalia. Taip skrydžio vadovui nereikia didelių pastangų, kad suprasti ar skrydis vykdomas teisingai.

# Kameros

Pagrindiniam lėktuvo objektui sukuriamos papildomos, kito vaizdo perspektyvos kameros, kurias būtų galima keisti skrydžio metu norint matyti kitas skrydžio detales. Kamerų keitimo funkcionalumas pasiekiamas paspaudus klaviatūros klavišą „C“ (Nes didžioji dauguma žaidimų ir kitų simuliatorių naudoja būtent šį klavišą kameros keitimui), kuris leidžia paprastu būdų pakeisti kamerą. Paspaudus klavišą, iškviečiamas kameros keitimo metodas, kuris pagal eilę, deaktyvuoja esamą kamerą bei aktyvuoja sekančią. Kamera galėtų pasikeisti pagal konkrečią koordinatę, kuri indikuoja riziką, bet kaip suvaldyti kamerų pasikeitimą, jeigu rizikos atsiranda kelios iškart, arba labai greitai keičiasi?

# Realaus reljefo pritaikymas terrain objektui

# Šaltiniai

[1] Oficialus „Unity“ puslapis: https://unity3d.com

[2] Kristina Lapin, Vytautas Čyras, Laura Savičienė. Visualization of Aircraft Approach and Departure Procedures in a Decision Support System for Controllers. In: Barzdins, J., Kirikova, M. (eds.) Databases and Information Systems VI, Selected Papers from the Ninth International Baltic Conference, DB&IS 2010, Vol. 224 Frontiers in Artificial Intelligence and Applications, ISBN 978-1-60750-687-4, IOS Press Amsterdam, 2011, pp. 408-421.

[3] Projekto „Sky-Scanner“ duomenų šaltinis: http://www.mif.vu.lt/~moroz/SKY.html

# Rezultatai ir išvados

1. <http://wiki.unity3d.com/index.php?title=CreateTorus> [↑](#footnote-ref-1)