

Matematikos taikymas kompiuteriniams žaidimams

Dan Goodman

Danas Goodmanas studijuoja matematiką Kembridže. Jis pradėjo programuoti žaidimus jau nuo 12 metų. Žaidimų programavimu jis tebesidomi ir dabar. Vieną iš jo sukurtų žaidimų galite rasti Internetu adresu <http://www.fcobob.demon.co.uk/tag.html>. Šį straipsnį apie matematikos taikymą kompiuteriniams žaidimams autorius parašė elektroniniam matematikos žurnalui „Mathematics Enrichment“ (<http://nrch.maths.org>), kurį Kembridžo universiteto matematikai leidžia moksleiviams ir visiems matematikos mėgėjams. Jo redaktoriai leido išversti straipsnį mūsų skaitytojams ir pakvietė aplankyti žurnalo svetainę, kurioje rasite daug įdomių matematinių rašinių bei uždavinių.

1. Įvadas

Šio straipsnio tikslas — žvilgterti, kaip matematika taikoma kompiuteriniams žaidimams. Panagrinėsime tuos žaidimus, kuriuos jūs jau tikriausiai žaidėte. Yra daug įvairių rūšių kompiuterinių žaidimų, aš papasakosiu, kaip matematika taikoma šaulio tipo, taip pat strateginiams ir imitaciniams žaidimams.

Šaulio tipo žaidimuose jūs lakstote po trijų matavimų lygius su dideliu šautuvu ir šaudote. Tokių žaidimų pavyzdžiai: *Doom*, *Quake*, *Unreal* ir *Goldeneye*. Yra panašių žaidimų, kurie nepriklauso šiam tipui, pavyzdžiui, *Zelda: Ocarina of Time* arba *Mario 64*.

Strateginiai žaidimai skirstomi į du pagrindinius tipus: realiojo laiko strateginius (*Real Time Strategy*) ir kintamos strategijos žaidimus (*Turn Based Strategy*). Šiuose žaidimuose reikia statyti ir valdyti miestus arba civilizacijas, taip pat kariauti ir vadovauti būriams. Realiojo laiko strateginių žaidimų pavyzdžiai: *Age of Empires*, *Command & Conquer*, *Tiberian Sun*. *Civilization* ir *Alpha Centauri* yra kintamos strategijos žaidimai.

Imitaciniuose žaidimuose tam tikrą situaciją bandoma imituoti kiek įmanoma realistiškiau. Pavyzdžiui, kompiuteriniuose žaidimuose *Flight Sims* bandoma realistiškai imituoti lėktuvo arba malūnsparnio valdymą. Du šios rūšies žaidimų pavyzdžiai: *Microsoft Flight Simulator* ir *Red Baron*. Žaidimai *Space Sims* panašūs į *Flight Sims*, tik vietoje lėktuvų naudojami erdvėlaiviai. Pavyzdžiui, *Wing Commander* ar *W-Wing*, *Tie Fighter*. Lenktynių žaidimuose imituojamas įvairių rūšių automobilių vairavimas. Pavyzdžiui, *Ned for Speed*, *NASCAR Racing*, *Gran Turismo* arba *Driver*.

Straipsnyje pateikta keletas pratimų, kuriuos galite (jei tik norite) atlikti. Atsakymai pateikiami straipsnio gale, tačiau iš pradžių geriausia pabandyti išspręsti juos patiems. Atsiprašau, jeigu kai kas

straipsnyje jums pasirodys per daug paprasta. Mano straipsnis skirtas įvairaus amžiaus žmonėms, todėl gali būti, kad kai kas jau yra jums žinoma. Jūs galite paprasčiausiai kai ką praleisti ir skaityti nuo tos vietos, kur jums įdomu.

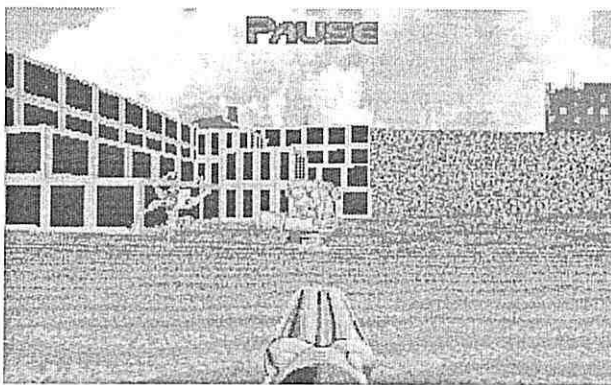
2. Šaulio tipo žaidimai

Šių žaidimų įdomiausias dalykas — neįtikėtina jų grafika. Jie atrodo beveik realūs, be sudėtingos matematikos taikymo tai nebūtų įmanoma. Štai keletas paveikslėlių pradedant pirmaisiais žaidimais (*Wolfenstein*) ir baigiant paskutiniais (*Quake III Arena*).

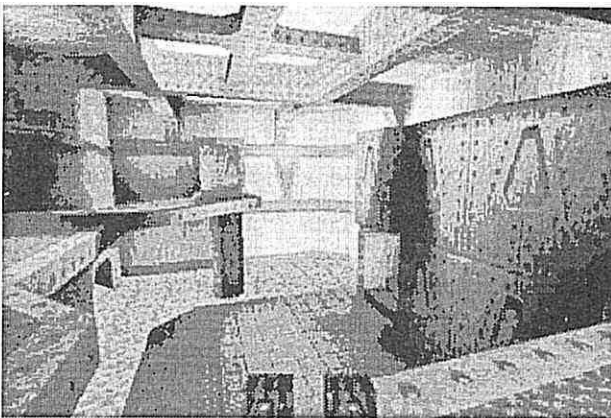
Wolfenstein — vienas pirmųjų šaulio tipo žaidimų



Doom — naujos kartos grafika



Quake — tai didžiulis kokybinis šuolis kompiuterinių žaidimų grafikoje

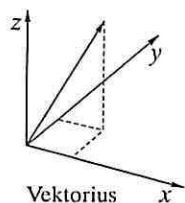
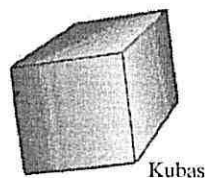
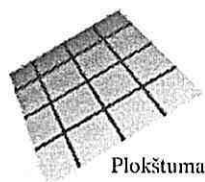
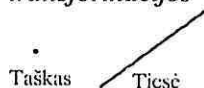


Quake III — vienas naujausių šaulio tipo žaidimų



Kad galėtume pradėti aiškintis, kaip šie žaidimai veikia, reikia trupučio žinių apie geometriją, vektorius ir transformacijas.

2.1. Geometrija, vektoriai ir transformacijos



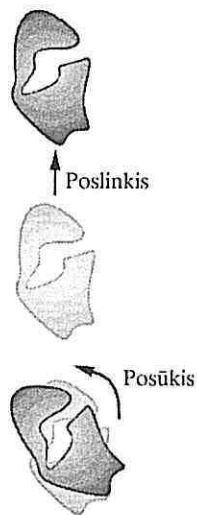
Geometrija tiria įvairias figūras. Paprasčiausia figūra yra taškas. (Nelengva paaiškinti, kas yra taškas, iš esmės tai tam tikra vieta, pavyzdžiui, pats jūsų nosies galiukas yra taškas). Kita paprasta figūra yra tiesė. Tiesė yra paprasčiausia figūra, einanti per du taškus. Plokštuma yra sudėtingesnis darinys, galima ją vaizduoti kaip plokščią popieriaus lapą ar sieną. Yra daug sudėtingesnių darinių, jie vadinami kūnais, pavyzdžiui, kubas arba rutulys. Nubraižykime šias paprastas geometrines figūras.

Jeigu turite tiesę ir plokštumą, galite rasti tašką, kuriame tiesė kerta plokštumą. Iš tiesų kartais susikirtimo taško negalima rasti, nes jos nesikerta, o kartais tiesė yra plokštumoje, taigi jos „kertasi“ kiekviename tiesės taške, tačiau mes tokiais atvejais nesidomėsime. Pavaizduokime tiesę, kuri kerta plokštumą.

Vektorius tai matematinis būdas vaizduoti taškus. Vektorius tai trys skaičiai, paprastai žymimi x , y ir z . Jūs galite įsivaizduoti, kad šie skaičiai nurodo, kokius atstumus reikia nueiti trimis skirtingomis kryptimis, kad pasiektumėte tašką. Pavyzdžiui, įsivaizduokime, kad viena kryptis rodo į dešinę, kita — tiesiai prieš mus, o trečia — į viršų. Jei aš nurodysiu jums vektorių, jūs galėsite rasti tašką. Pavyzdžiui, tegu $x = 2$, $y = 3$, $z = 5$, tada jums reikia nueiti du metrus į dešinę, tris metrus priešais ir susiradus kopėčias palypėti penkis metrus į viršų.

Vektoriai užrašomi (x, y, z) , taigi, pavyzdžiui, užrašas $(1, 2, 3)$ reiškia, kad x kryptimi reikia nueiti vieną, y kryptimi du ir z kryptimi tris ilgio vienetų.

Kartais vektoriai naudojami vaizduoti taškams, o kartais kryptims. Vektorius $(1, 0, 0)$ gali reikšti „kad patektumėte į reikalingą tašką, jums reikia nuo pradžios taško pasislinkti per ilgio vienetą x kryptimi“, tačiau gali reikšti nurodymą „iš tos vietos, kur dabar esate, pasislinkite per ilgio vienetą x ašies kryptimi“.



Poslinkiai ir posūkiai

1 pratimas (vektorių sudėtis). Jeigu iš taško, kurį vaizduoja vektorius $(0, 0, 0)$, judėsime, kaip nurodo vektorius $(1, 2, 3)$, o po to — kaip vektorius $(4, 5, 6)$, į kokią tašką pateksime? Šį tašką vaizduojantis vektorius vadinamas vektorių suma. Raskite $(1, 2, 3) + (4, 5, 6) = (?, ?, ?)$.

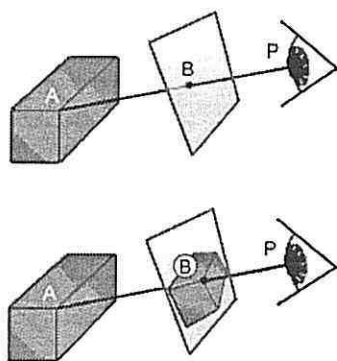
Transformacija perkelia tašką (arba objektą, o gal ir visą pasaulį) iš vienos vietos į kitą. Pavyzdžiui, aš galiu jį pastumti keturis metrus į dešinę — šios rūšies transformacija vadinama poslinkiu. Kita transformacijų rūšis — posūkiai. Jeigu jūs laikote rankoje objektą (pavyzdžiui, rašiklį) ir pasukate ranką per riešą, jūs pasukate objektą. Pavaizduokime poslinkį ir posūkį brėžiniu.

2 pratimas (poslinkiai). Jeigu paslinksime vektorių $(1, 2, 3)$ penkiais vienetais y ašies kryptimi, tai kokių vektorių tada gausime? Apskritai, jeigu T žymi poslinkį, o v — vektorių, tai paslinktas vektorius žymimas Tv . Pavyzdžiui, jei T reiškia poslinkį x ašies kryptimi per vienetą, o $v = (0, 0, 0)$, tai $Tv = (1, 0, 0)$.

2.2. Trimatė grafika

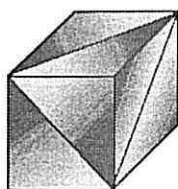
Esminė trimatės grafikos idėja — paversti matematinį pasaulio aprašymą paveikslu, kuris rodytų, kaip tas pasaulis atrodo kažkam esančiam pasaulio viduje. Matematinis aprašymas gali būti duotas tam tikru sąrašu, pavyzdžiui: yra kubinė dėžė, jos kraštinių ilgiai lygūs 3, centras yra taške $(2, 4, 7)$, dėžė nuspalsvota pilkai su melsvu atspalviu. Kad paverstume šį aprašymą paveikslu dar turime žinoti, kur yra asmuo ir kokia kryptimi jis žvelgia. Pavyzdžiui, asmuo yra taške $(10, 10, 10)$, jis žiūri tiesiai į dėžės centrą. Pasinaudoję šiomis žiniomis, jau galime sukonstruoti vaizdą, kurį mato tas asmuo.

Įsivaizduokime dailininką, kurio akys yra taške P . Tarkime, prieš jį yra stiklo lakštas, ant kurio jis ruošiasi piešti. Tame pačiame kambaryje yra medinė dėžė. Vienas iš šios dėžės kampų yra taške A , o dailininkui reikia nuspręsti, kurioje stiklo paviršiaus vietoje jis turi nupiešti šį kampą. Jis elgiasi taip: iš taško P veda tiesę tiesiai į dėžės kampo tašką A , randa tašką B , kuriame tiesė kerta stiklo paviršių. Taigi jam reikia rasti tiesės ir plokštumos susikirtimo tašką. Šis taškas B ir yra tas taškas, kuriame ant stiklo reikia pavaizduoti dėžės kampą. Šią taisyklę jis taiko kiekvienam dėžės taškui ir gauna teisingą dėžės vaizdą. Pirmasis iš mūsų brėžinių vaizduoja tą momentą, kai ant stiklo nupieštas tik dėžės kampas, antrasis — kai nupiešta visa dėžė.

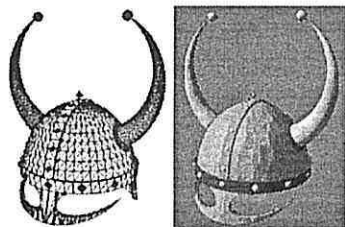


Projekcija į plokštumą

Veiksmus, panašius į mano ką tik aprašytuosius, ir atlieka kompiuteris (50 kartų per sekundę!), kai jūs laktote ir šaudote siaubingas pabaisas žaidime *Quake*, nors detalės ir skiriasi. Kompiuteriniuose žaidimuose (šiuo metu) pasaulio aprašymas yra tiesiog trikampių ir spalvų sąrašas. Naujieji kompiuteriai naudoja sudėtingesnius pasaulio aprašymus kreivais paviršiais, *NURBS* ir kitais keistai skambančiais objektais, tačiau galų gale visi jie suvedami į



Iš trikampių sudaryta dėžė



Vikingo šalmas

trikampius.¹ Pavyzdžiui, galima įsivaizduoti, kad dėžė sudaryta iš trikampių, kaip parodyta brėžinyje.

Trikampiai naudojami todėl, kad jie yra labai paprastos figūros, be to, jeigu jūs žinote, kad viskas yra sudaryta iš vienos rūšies figūrų, kuriant žaidimą jums jau nebereikia rašyti kiekvienam figūrų tipui atskiros programos.

3 pratimas. *Naudodami tik trikampus, nupieškite dėžę, ant kurios uždėta kita mažesnė dėžė.*

Kiekvieną kartą, kai kompiuteris piešia pasaulio vaizdą, jis atlieka tokius žingsnius: iš pradžių transformuoja pasaulį (naudodamas poslinkius ir posūkius), kad asmuo atsidurtų taške $(0, 0, 0)$, o stiklo lapo centras (kompiuterio ekrano centras) — taške $(1, 0, 0)$. Tai palengvina tolesnius skaičiavimus. Po to pašalina visus trikampus, kurių jūs negalite matyti, todėl jis gali jais nesirūpinti; pavyzdžiui, trikampus, kurie yra už jūsų, arba tuos, kurie taip toli, kad jų nesimato. Pagaliau kiekvienam iš likusiųjų trikampių kompiuteris nustato, kaip jie atrodytų nupiešti ant stiklo (arba pavaizduoti kompiuterio ekrane). Pagaliau savo sukurtą vaizdą kompiuteris perkelia į ekraną. Mūsų laikų kompiuteriai yra tokie greiti, kad per sekundę gali nupiešti šimtus tūkstančių trikampių ir sukurti vaizdus, vis labiau panašius į realybę.

Žinoma, tam reikia ne tik trikampių, bet ir daugybės kitų dalykų: šviesos efektų, animacijos, tekstūros ir dar šimtų kitų. Beveik viskam prisireikia matematikos ir fizikos, tačiau svarbiausia yra tai, kad trimatė grafika atrodytų tikroviškai.

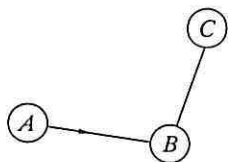
3. Strateginiai žaidimai

Paprastai strateginių žaidimų grafika yra daug paprastesnė negu šaulio tipo žaidimų, todėl aš papasakosiu apie kitus įdomius strateginių žaidimų ypatumus. Kas vyksta kompiuteryje, kai bakstelėjate pelės strėlę mažą strateginio žaidimo kareivį ir po to bakstelėjate į kokią nors kitą vietą šitaip nurodydami, kad jis turi ten nueiti? Iš kur kompiuteris žino, kaip priversti kareivį persikelti iš dabartinės į naująją vietą? Prisiminkime, kad kompiuteriai nemoka galvoti (vis dar!), todėl jiems reikia tiksliai nurodyti, ką daryti. Taigi jūs negalite paprasčiausiai tarti: „žiūrėk į žemėlapij ir keliauk geriausiu keliu į nurodytą vietą“. Kiekviename kelionės žingsnyje jam reikia tikslios instrukcijos. Šis uždavinys vadinamas kelio radimo problema, apie ją aš papasakosiu kitame skyrelyje.

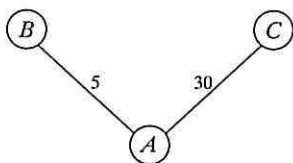
3.1. Viršūnės, briaunos, grafai

Kad suprastumėte, kaip kompiuteris randa geriausią kelią, turite žinoti, kas yra viršūnės, briaunos ir grafai. Galbūt jūs žinote apie grafus iš matematikos, bet čia jie bus kiek kitokie. Paprasčiausias grafo pavyzdys — miestų ir juos jungiančių kelių planas (arba metro

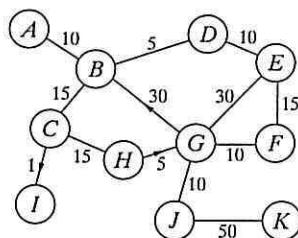
¹ Apie matematikos taikymą kompiuteriniam modeliavimui rašyta mūsų žurnalo 1997 metų 1-ajame numeryje (vertėjo pastaba).



Grafas su orientuota briauna



Grafas su kainomis



Sudėtingas grafas

stotelių ir kelių planas). Kiekvienas miestas yra viršūnė, paprastai vaizduojama nedideliu skrituliuku. Kiekvienas kelias yra briauna, ji jungia dvi viršūnes (miestus), paprastai briaunos vaizduojamos atkarpomis. Viršūnių ir briaunų visuma (miestų ir kelių visuma) vadinama grafu. Kartais kelias būna tik vienos krypties, tada jis vadinamas orientuota briauna, tada kryptį, kuria galima keliauti nurodome strėlę. Pavyzdžiui, jeigu yra du miestai A ir B ir juos jungianti briauna nukreipta iš A į B , tai galima keliauti iš A į B , bet ne priešinga kryptimi. Brėžinyje pavaizduotas grafas, kuriame negalima keliauti iš B į A , bet galima iš A į B . Taip pat galima keliauti iš B į C ir iš C į B .

Kartais tenka kiekvienai briaunai priskirti kainą. Kaina rodo, kiek kainuotų kelionė išilgai briaunos. Štai paprastas pavyzdys.

Šiame grafe dauguma žmonių iš miesto A nori vykti į miestą C , tik nedaugelis nori vykti į B . Nelaimei, abu keliai yra to paties pločio, taigi kelyje iš A į C nuolat susidaro spūstys, todėl kelionė šiuo keliu trunka apie 30 minučių. Nuvykti į B miestą yra kur kas paprasčiau, todėl kelionė trunka tik 5 minutes. Skaičiai parašyti prie briaunų kaip tik ir rodo, kiek teks sugaišti kelionėje. O štai dar vienas, daug sudėtingesnis pavyzdys.

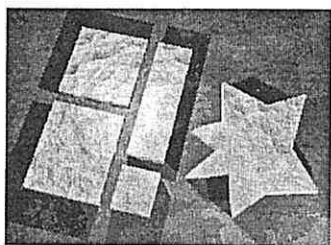
4 pratimas. Raskite pigiausią (arba mažiausiai laiko reikalaujantį) kelią iš A į K .

3.2. Kelio paieška

Dabar jūs jau žinote viską, ko reikia suprasti kelio paiešką. Jeigu atlikote paskutinį pratimą, jūs jau išsprendėte vieną iš tos rūšies uždavinių, kokius kompiuteriui tenka spręsti vedant būrius per sudėtingus žemėlapius.

Kaip visi šie grafai padeda kompiuteriui vesti būrius? Kompiuteris sudaro grafą, kurio viršūnes sudaro visos įdomios vietos, o kiekvienas kelias yra briauna, jungianti vieną viršūnę su kita, po to jis sprendžia uždavinį, panašų į jūsų išspręstąjį. Tačiau viskas nėra taip paprasta. Pirmiausia, kas tie įdomūs taškai? Galima manyti, kad kiekviena žaidimo vieta yra įdomi, tačiau daugelyje žaidimų tokių vietų būtų šimtai tūkstančių ir kelio paieška truktų net metus. Todėl žmonės, kuriantys žaidimus, nusprendžia, kur yra įdomios vietos. Pavyzdžiui, jeigu yra didelė atvira laukymė (galbūt didelis laukas), visai nereikia, kad viršūnė būtų kiekviename taške, nes būriai gali keliauti per lauką tiesiomis trajektorijomis. Paprastai reikia, kad viršūnės būtų pažymėtos apie kliūtis. Štai vieno lygio žemėlapio vaizdas, kai žiūrima iš viršaus.

Jeigu jau pasirinkote savo įdomius taškus, jums reikia sujungti juos briaunomis. Pakanka jungti tik tas viršūnes, tarp kurių nėra kliūčių.



Žaidimo lygio žemėlapis

5 pratimas. Nubraižytame žemėlapyje pasižymėkite įdomius taškus, sujunkite juos briaunomis, atsimindami, kad galima viršūnes jungti tik tiesiomis atkarpomis, kurios negali eiti per kliūtis.

Jeigu jūs jau sukūrėte žemėlapiui atitinkamą grafą, tai vesdamas būrius kompiuteris atlieka tokius žingsnius. Iš pradžių randa viršūnę, į kurią galima patekti tiesiu keliu. Tai jo starto viršūnė. Antra, jis turi rasti viršūnę, kuri yra arčiausiai tikslo vietos (jis taip pat įsitikina, kad gali patekti iš tos viršūnės į tikslo tiesią liniją). Tai tikslo viršūnė. Tada jis randa trumpiausią kelią, jungiantį starto viršūnę su tikslo viršūne. Dabar būriams lieka nueiti iki starto viršūnės, o po to kompiuterio rastu keliu nuo viršūnės prie viršūnės keliauti tol, kol pasieks tikslo viršūnę, nuo kurios jau galima baigti kelionę iki nurodytos vietos.

Kad būtų įdomiau, briaunoms galime priskirti kainas. Pavyzdžiui, jeigu tarp dviejų viršūnių yra pilnas krokodilų griovys, tai kelionė kertant šį griovį gali brangiai kainuoti, todėl gali būti protingiau keliauti aplinkui, nors kelias per krokodilų griovį yra trumpesnis. Kita vertus, jei aplinkinis kelias truktų 3 dienas, o per krokodilų griovį tik 15 minučių, jie gali nuspręsti, kad verta rizikuoti, nes mūšis gali persikelti į kitą vietą. Tenka kruopščiai įvertinti kainą, kad šios problemos būtų kuo geriau išspręstos.

Aš dar nepasakojau, kaip padaryti, kad kompiuteris rastų geriausią kelią nuo vienos viršūnės prie kitos. Visa kelio, vedančio iš A_1 į A_m per $A_2, A_3, \dots, A_{m-2}, A_{m-1}$, kaina randama sudedant kelių iš A_1 į A_2 , iš A_2 į $A_3, \dots$, iš A_{m-1} į A_m kainas. Svarbu rasti tas viršūnes, kad visa kaina būtų mažiausia. Vienas sprendimo būdas — rasti visus galimus kelius iš vienos viršūnės į kitą, rasti jų kainas ir išrinkti mažiausią. Deja, toks būdas net greičiausiems kompiuteriams būtų per lėtas. Yra būdas labai greitai rasti kelią, tačiau ne taip lengva apie tai papasakoti. Tačiau jeigu jums įdomu, galite paieškoti medžiagos Internete apie A^* algoritmą arba pavartyti straipsnius, nurodytus mūsų straipsnio pabaigoje.

4. Imitaciniai žaidimai

Svarbiausias imitacinių žaidimų bruožas — siekiama, kad žaidimas atrodytų kaip realus pasaulis. Paprastai naudojamas fizikinis modeliavimas. Ši straipsnio dalis gali būti kiek sunkesnė tiems, kas nesimokė fizikos mokykloje.

4.1. Šiek tiek daugiau apie vektorius

Fizikos taikymas kompiuteriniams žaidimams taip pat remiasi vektoriais. Apie juos mums reikia dar šiek tiek daugiau žinių. Pirmiausia vietoje žymens (x, y, z) , kurį naudojame, kai tik kalbame apie vektorius, dabar rašysime tik v , tai reiškia tą patį kaip žymuo (v_x, v_y, v_z) . Rašydami v_x mes turime galvoje atstumą, kurį reikia nueiti x kryptimi keliaujant prie v . Du vektorius galime sudėti

$$\alpha + \omega$$

$v+w = (v_x+w_x, v_y+w_y, v_z+w_z)$, vektorių galima dauginti iš skaičiaus $av = (av_x, av_y, av_z)$. Pavyzdžiui, $3 \times (1, 2, 3) = (3, 6, 9)$. Vektoriaus ilgis yra atkarpos, jungiančios $(0, 0, 0)$ ir v , ilgis. Jis žymimas $|v|$. Jį galima rasti iš formulės $|v| = \sqrt{v_x^2 + v_y^2 + v_z^2}$. Pavyzdžiui, $|(1, 2, 2)| = \sqrt{1^2 + 2^2 + 2^2} = \sqrt{9} = 3$. Vektoriaus, gauto padauginus vektorių iš teigiamo skaičiaus, ilgis lygus pradinio vektoriaus ilgiui, padauginamam iš to skaičiaus, taigi $|av| = a|v|$. 6 pratimas. Kam lygūs vektoriai $(1, 2, 3) + (3, 2, 1)$, $10 \times (9, 5, 2)$? Kam lygus vektoriaus $(4, 2, 4)$ ilgis?

4.2. Fizika

Kiekvienas objektas turi savo vietą, nusakomą vektoriumi x , greitį, kuris vaizduojamas vektoriumi v , nukreiptu jo judėjimo kryptimi. Kiekvienas objektas taip pat turi pagreitį, rodantį, kaip greitai keičiasi greitis. Pagreitis yra vektorius a . Galų gale kiekvienas objektas turi masę, kuri rodo, koks sunkus yra tas objektas. Masė paprastai žymima m , ji nėra vektorius. Jeigu objektas startuoja iš padėties x ir juda pastoviu greičiu v , tai po t sekundžių objekto padėtis bus $x_t = x + v \times t$. Taigi x_t reiškia objekto padėtį laiko momentu t . Analogiškai v_t reiškia greitį laiko momentu t . Jeigu objektas juda pastoviu pagreičiu a , o pradinis jo greitis yra v , tai po t sekundžių objekto greitis bus $v_t = v + a \times t$.

O štai pavyzdys, kaip visa tai gali būti pritaikyta. Tarkime Bilis-Džo kybo erdvėje, jo padėtis yra $(100, 150, 150)$ ir jis sviedžia uolos gabalą greičiu $(5, 2, 0)$. Kur bus ta uola po 10 sekundžių? Pritaikę formulę $x_t = x + v \times t$ gausime, kad uolos padėtis bus $(100, 150, 150) + (5, 2, 0) \times 10 = (100, 150, 150) + (50, 20, 0) = (150, 170, 150)$.

7 pratimas. Mirtinas Bilio-Džo priešas Artūras Blenkinsonas skaito laikraštį, būdamas padėtyje $(240, 640, 143)$ ir nežinodamas, kad Bilis-Džo jį stebi. Bilis-Džo nusprendė sviesti į savo priešą akmenį. Bilis-Džo teikia magiškos reikšmės skaičiui 7, todėl nori, kad akmuo pasiektų Artūrą lygiai po 7 sekundžių. Kokiu greičiu jis turėtų sviesti akmenį?

Jeigu Bilis-Džo nėra kažkur erdvėje, o ant Žemės paviršiaus, problema tampa dar sudėtingesne, nes gravitacijos jėga traukia kiekvieną objektą prie Žemės, suteikdama pagreitį. Šis pagreitis kiekvienam objektui ant Žemės paviršiaus yra vienodas, jeigu y ašį nukreipsime aukštyn, tai pagreitis bus $g = (0, -9.81, 0)$. Padėties nustatymo problema dabar yra sunkesnė todėl, kad greitis keičiasi, todėl ankstesnės lygybės jau nebegalima naudoti padėčiai įvairiais laiko momentais rasti. Kompiuteriu problemą galime spręsti ir naudodamiesi šia lygtimi, tačiau ne visiškai tiksliai. Galime tarti, kad greitis keičiasi tik po labai mažo laikotarpio ir tas pokytis yra nedidelis. Pavyzdžiui, galime manyti, kad greitis pasikeičia tik kas 0,02 sekundės. Taigi iš lygties $x_{t+s} = x_t + v_t \times s$ galime gana tiksliai nustatyti padėtį po labai mažo laikotarpio s , galbūt $s = 0,02$.

Šioje lygtyje x_{t+s} yra padėtis laiko momentu $t + s$, x_t yra padėtis laiko momentu t , v_t yra greitis laiko momentu t . Greitį galime rasti remdamiesi tuo, kad pagreitis nesikeičia. Atlikdamas šiuos skaičiavimus pakartotinai, kompiuteris gali rasti akmens vietą bet kuriuo laiko momentu, iš pradžių suradęs padėtį po 0,02 sekundės, tada po 0,04, 0,06 sekundės ir taip toliau, kol pasieks reikiamą laiko momentą.

8 pratimas. Bilis-Džo yra padėtyje $(0, 0, 0)$ ant Žemės paviršiaus, jis meta akmenį greičiu $(0, 4, 0)$. Kur bus akmuo po 1 sekundės? Akmens pagreitis yra $(0, -9.81, 0)$, skaičiuojant aptartuoju metodu, kai $s = 0,2$, skaičiavimus tektų atlikti penkis kartus.

4.3. Imitaciniai žaidimai

Kaip visa tai panaudojama kompiuteriniams žaidimams? Štai paprastas pavyzdys. Kaskart, kai jūs iššaukiate į taikinį skrisdami lėktuvu skrydį imituojančiame žaidime, jūsų kompiuteris, nustatydamas kulkos padėtį, atlieka skaičiavimus, panašius į tuos, kurių reikia Bilio-Džo mesto akmens padėčiai nustatyti. Kompiuteriui reikia kas sekundę atlikti 50 skaičiavimų kiekvienos lekiančios kulkos padėčiai rasti, be to, daug sudėtingesnius skaičiavimus paties lėktuvo padėčiai nustatyti. Deja, viskas nėra taip paprasta kaip su Bilio-Džo akmenimis. Viskas yra sudėtingiau, nes realiame pasaulyje yra ir vėjas, ir oro pasipriešinimas. Štai šitaip skaičiuodami jūs galite atsižvelgti į vėją. Tarkime, vėjas, pūsdamas pastoviu greičiu, sukelia papildomą kulkos pagreitį w . Dabar kulkos pagreitis yra $g + w$, taigi situacija nėra keblesnė nei anksčiau. Tačiau atsižvelgti į oro pasipriešinimą yra sudėtingiau, nes jos poveikis priklauso nuo kulkos greičio. Faktiškai trintis sukelia papildomą pagreitį $-kv$, čia k yra tam tikras teigiamas skaičius, o v — greitis. Taigi dabar pagreitis yra $g + w - kv$, dabar ne tik greitis, bet ir pagreitis keičiasi! Tačiau skaičiuoti galima tuo pačiu metodu kaip ir anksčiau, padarius prielaidą, kad dydžiai labai mažais laikotarpiais lieka pastovūs.

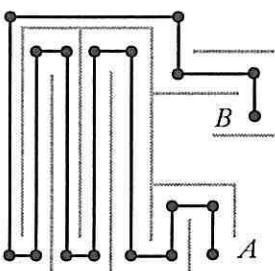
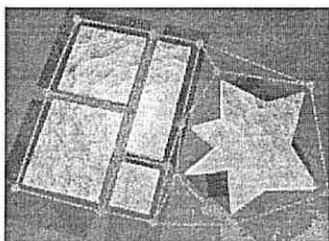
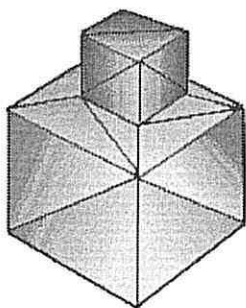
Imituoti lėktuvo skrydį yra dar sudėtingiau. Jeigu jūsų lėktuvas skrenda labai aukštai, o po to neria žemyn, tai gali atsitikti, kad jo sparnai nutrūks (jeigu žaidžiate naujuosius lėktuvus imituojantį žaidimą, to neatsitiks, nes dabar lėktuvai yra tobulesni nei anksčiau). Tai įvyksta todėl, kad oras labai stipriai spaudžia sparnus, o jie nėra pakankamai tvirtai pritvirtinti prie lėktuvo liemens.

Kuriant kompiuterinius žaidimus, fizika yra kiečiausias riešutas. Mažai žaidimų teisingai imituoja fizikinius reiškinius, daug žaidimų modeliuojami maždaug taip, kaip čia aprašyta. Kaip realistiškai modeliuoti fizikinius reiškinius kompiuteriniuose žaidimuose, yra sudėtinga problema, nors ją žmonės ir tyrinėja, tačiau iki sprendimo dar labai toli. Taigi jei jūs domitės žaidimų programavimu bei fizika, galbūt ir jūs šioje srityje kada nors pasidaruosite.

5. Pabaiga

Keletu paprastų pavyzdžių aš pabandžiau parodyti, kaip matematika taikoma kuriant kompiuterinius žaidimus. Tačiau yra daugybė kitokių pavyzdžių. Be matematikos faktiškai neįmanoma sukurti jokio žaidimo. Net paprasčiausiems iš jų reikia kad ir nesudėtingos matematikos.

6. Atsakymai



1 *pratimas.* $(1, 2, 3) + (4, 5, 6) = (5, 7, 9)$.

2 *pratimas.* $(1, 7, 3)$. Apskritai, poslinkis T reiškia, kad prie vektoriaus reikia pridėti kitą tam tikrą vektorių

$$T(x, y, z) = (x, y, z) + (u, v, w) = (x + u, y + v, z + w).$$

Posūkį apie x ašį kampų θ galima nusakyti matrica

$$M_\theta = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Taigi $M_\theta(x, y, z) = (x \cos(\theta) - y \sin(\theta), x \sin(\theta) + y \cos(\theta), z)$.

3 *pratimas.* Vienas, tačiau ne vienintelis būdas sudaryti dvi dėžes iš trikampių, parodytas paveiksle.

4 *pratimas.* Trumpiausias kelias iš A į K yra $ABCHGJK$.

5 *pratimas.* Vienas iš būdų sudaryti grafą parodytas paveiksle.

Jeigu jūs žaidime kuriate atsitiktinius lygius, tai negalite fiksuoti, kokie taškai yra įdomūs. Tada jums tenka pagalvoti, kaip išmokyti patį kompiuterį tai nuspręsti. Yra įvairių būdų. Pagalvokite, kaip jūs tai darytumėte. Tenka atsižvelgti į du dalykus: a) ar iš tam tikros viršūnės įmanoma į visus žemėlapiu taškus patekti tiesiu keliu? b) Ar nėra tokių kvailių padėčių, kaip parodyta paveiksle, kur iš A į B galima nueiti tik labai kvailu keliu?

6 *pratimas.*

$$(1, 2, 3) + (3, 2, 1) = (4, 4, 4);$$

$$10 \times (9, 5, 2) = (90, 50, 20);$$

$$|(4, 2, 4)| = \sqrt{4^2 + 2^2 + 4^2} = \sqrt{36} = 6.$$

7 *pratimas.* Tegu Bilio-Džo padėtis yra x o Artūro — y . Jei Bilis-Džo mes akmenį greičiu v , tai po t sekundžių akmens padėtis bus $x + v \times t$. Kad akmuo pataikytų į Artūrą, turi būti $x + v \times t = y$. Pertvarkę lygybę, gausime $v = (y - x)/(t)$. Įstatę x, y ir $t = 7$, gausime $v = ((240, 640, 143) - (100, 150, 150))/7 = ((140, 490, -7))/7 = (20, 70, -1)$.

8 *pratimas.* Kadangi visų vektorių x ir z komponentės lygios nuliui, galime apsiriboti tik y . Kai $t = 0$, akmuo yra padėtyje 0, o jo greitis 4, todėl laiko momentu $t = 0,2$ akmens padėtis bus $0 + 4 \times 0,2 = 0,8$, o greitis be $4 + 0,2 \times (-9,81) = 2,038$. Taigi, kai $t = 0,4$, akmuo padėtis bus $0,8 + 2,038 \times 0,2 = 1,2076$, o greitis $4 + 0,4 \times (-9,81) = 0,076$. Skaičiuodami toliau gauname, kad akmuo padėtis po 1 sekundės bus $(0, 0, 0,812, 0)$.

7. Įdomūs tinklapiai

Norintys išmokti programuoti C++ kompiliatorių DJGPP gali nemokamai rasti svetainėje

<http://www.delorie.com/djgpp>

Specialią „biblioteką“ Allegro žaidimų kūrėjams galima rasti adresu

<http://www.talula.demon.co.uk/allegro>

Kelios svetainės apie žaidimų programavimą:

<http://www.gamasutra.com> <http://www.gamedev.net/gamedev.asp>

Puiki svetainė, skirta kelio paieškos problemai,

<http://www-cs-students.stanford.edu/~amitp/gameprog.html>

Java apletų, skirtų A* algoritmui iliustruoti, rasite svetainėje

<http://www.ccg.leeds.ac.uk/james/aStar/>

Išsamios informacijos apie trimatę grafiką galima rasti svetainėje

<http://www.cc.gatech.edu/gvu/multimedia/nsfmmedia/cware/graphics/toc.html>



G. Strazdo uždavinių (*Alfa plus omega*, 2001, 1) atsakymai:

1) $5544\dots = 33\dots$

Ką įrašytumėte daugtaškių vietoje, kad lygybė būtų teisinga?

Atsakymas. $5544 \text{ valandos} = 33 \text{ savaitės}$.

2) Iš nesikartojančių skaitmenų 0, 1, ..., 9 sudarykite skaičius, kuriuos susumavę gautumėte 1881.

Atsakymas. $1854,97 + 26,03 = 1881$.

3) Sugretinus skaičių n ir $2n - 1$ užrašus, be galo daug palindromų galima gauti taip: jei $n = 70\dots04$, tai

$2n - 1 = 140\dots07$ ir $70\dots04140\dots07$ — palindromas.

ALFA PLUS OMEGA, 2001, Nr. 2(12)

2001 08 20. 13,5 sp. l. Užs. Nr. 1278
Leidykla TEV, Akademijos g. 4, LT-2600 Vilnius
Spausdino UAB „Mokslo aidai“
A. Goštauto g. 12, LT-2600 Vilnius
Viršelį spausdino P. Kalibato II „Petro ofsetas“
Žalgirio g. 90, LT-2600 Vilnius