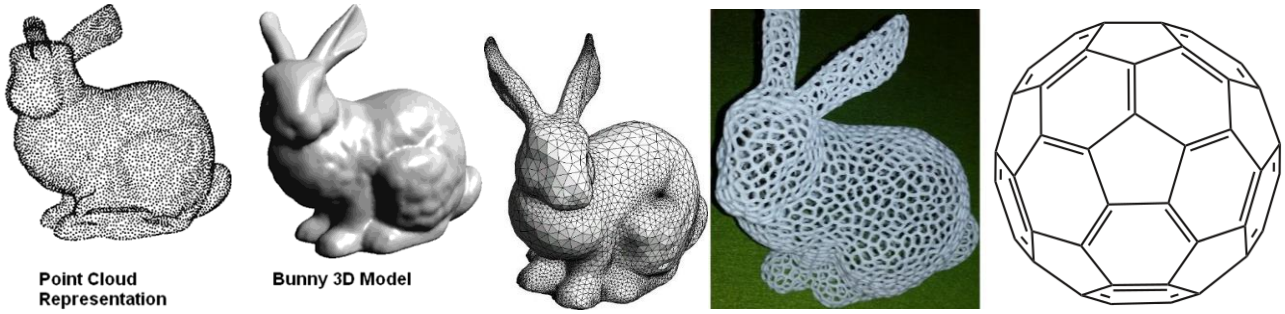
Podpora grafických poznámok IV

Nevyhnutnou etapou pred grafickým zobrazením je **geometrické modelovanie**. V počítačovej grafike sa používa 5 až 7 geometrických súradnicových systémov, ak nerátame ďalšie, ktoré treba na farby alebo pre multimédiá (zvuk, interakcia). Nasleduje charakteristika 5 súradnicovych systémov pre 2D a 3D grafiku podľa ISO noriem CGRM, PREMO, PHIGS (Ružický, kapitoly 16-20):

- modelové (pre objekty, lokálne súradnice pre každý objekt)
- svetové (scéna, kde sa objekty kombinujú)
- normalizované, interné pre grafický systém
- pohľadu, kamery (view, eye)
- zariadenia (DC, device coordinates)

Šiesty súradnicový systém sa označuje TLC (text local coordinates), v ktorom návrhári fontu reprezentujú krivky pre true type fonty. Tieto súradnice sú obdobou modelových pre grafický výstupný prvok text. Siedmy súradnicový systém sa v prípade potreby používa na textúrovanie povrchu objektov a pre tieto súradnice sa zaužívalo označenie (u, v). Okrem toho sa používajú aj súradnice na vyjadrenie polohy svetelného zdroja pre výpočet viditeľnosti či osvetlenia, alebo sa môže zaviesť pre potreby interakcie rozdelenie obrazovej alebo pracovnej plochy do nejakých vhodných častí. Existujú aj ďalšie, tzv. multimediálne súradnice (napr. na vyjadrenie polohy záznamu v excelovskej tabuľke, spreadsheet). Pri každom súradnicovom systéme treba zvoliť počiatok, osi, orientáciu a pomer strán. Po zavedení súradníc možno už geometriu scény a jej transformácie vhodne vyjadriť pomocou lineárnej alebo nelineárnej algebry a maticového vyjadrenia afinných transformácií a projekcií.



Obrázky z internetu ilustrujú oblak nameraných bodov, Phongom osvetlený povrchový model, Delaunayov mesh a k nemu duálne Voronoiove oblasti. Drôtový model molekuly fullerénu C60 naznačuje, že aj „matka Príroda“ pozná Voronoiov diagram v mikrosvete. Dizajn nevedomky prevzali aj futbalisti...

Objekty v scéne sa v počítačovej grafike reprezentujú ako oblak bodov, drôtový model, hraničná reprezentácia (najčastejšie sa povrch objektu aproximuje nepravidelnou trojuholníkovou sieťou, ktorej sa hovorí mesh) alebo objemová reprezentácia (volume graphics). Na všetky tieto reprezentácie treba používať efektívne dátové štruktúry. Priemyselným štandardom pre väčšinu aplikácií počítačovej grafiky a vizualizácie je hraničná reprezentácia a reprezentatívnym kódovaním 3D scén je formát VRML (Virtual Reality Modeling Language, norma ISO, i novšia verzia

X3D), v ktorom môžu byť do scény zahrnuté aj objekty, adresované na inom počítači po internete použitím URL (Uniform Resource Locator). Na zložitejšie aplikácie treba objekty vyjadrovať pomocou povrchov s viacerými úrovňami detailu (level of details). VRML podporuje 8 úrovní. Pre niektoré operácie ako morphing býva najvhodnejšie mať objekt reprezentovaný funkcionálne (F-rep).

Z matematického hľadiska vyjadrujeme objekt najčastejšie parametricky, no známe objekty zapisujeme aj analyticky alebo implicitne. Informaticky môžme objekt deklarovať, guľa v jazyku VRML Sphere, v X3D červená guľa a rotovaný modrý hranol, napr. https://www.web3d.org/x3d/content/examples/Basic/X3dSpecifications/RedSphereBlueBoxIndex.html

Modely získame pomocou krokov, ktoré sumarizuje **metodika matematického modelovania**:

1. Analýza problému, ktorej výsledkom je exaktný popis modelu (napr. modelové rovnice).
2. Analýza modelu problému, ktorej výsledkom je algoritmus na riešenie problému.
3. Analýza algoritmu, ktorej výsledkom je program.
4. Výpočet zvolených úloh, ktorých výstupom sú údaje-výsledky.
5. Analýza výsledkov, ktorej syntézou by malo byť riešenie problému.

Z každej nasledujúcej etapy sa možno v prípade potreby vrátiť do niektorej predchádzajúcej a spresniť model, algoritmus, program, výstupné údaje.)

Metodika zobrazovania má 4 etapy:

1. Určenie objektov, ktoré sa budú zobrazovať. (Nie všetko z modelu sa vždy zobrazuje!)
2. Geometrický popis symbolov, ktoré budú dané objekty reprezentovať.
3. Vyjadrenie geometrického popisu symbolov v grafických prvkoch daného grafického systému.
4. Samotné zobrazenie.

Metodika návrhu dialógu (interakcie) má tiež 4 časti (vrstvy). Jazyk dialógu má byť pohotový a úplny a má umožňovať spätnú väzbu.

1. pojmový (používateľský aplikačný model)
2. sémantický (činnosť a stav systému, napr. aké akcie možno zadať)
3. syntaktický (pravidlá dialógu, napr. ktoré akcie sa vylučujú);
4. lexikálny (ako sformovať symbol jazyka z hardverových primitívov, napr. ktorá klávesa znamená danú akciu operátora, napr. CTRL-ALT-DEL).

Postupom v rámci uvedených metodík vytvárame jednak model (dáta), jednak interaktívny grafický program (funkčnosť, súbor metód) na prácu s konkrétnym aplikačným modelom. Po modelovaní možno objekt zobraziť. Ako na to?

Vo všeobecnosti pracujeme v v 8-rozmernom priestore parametrov (x,y,z,t,r,g,b, α) pre grafické objekty, prezentované v danej scéne, minimalizujeme chyby geometrické, rádiometrické a kinematické modelovaním a/alebo meraním. Vo virtuálnej realite pristupuje naviac kritérium uveriteľnosti. Parameter α vyjadruje priehľadnosť v intervale [0,1], (r, g, b) predstavuje hodnoty farieb v modeli RGB, t znamená čas a (x, y, z) označujú priestorové súradnice v rozšírenom euklidovskom priestore. Ideálna by bola univerzálna dátová reprezentácia, no za grafický objekt volíme takú úroveň abstrakcie, aby sa daná scéna efektívne modelovala a renderovala. Pri animácii sa používa aj charakteristika objektu stupňom voľnosti, bod má dva, štvorec v rovine štyri (polohu stredu, dĺžku strany, natočenie). Matematicky je objekt svojou geometriou podmnožinou roviny alebo priestoru a svojím vzhľadom nosičom atribútov ako farba či priehľadnosť. Na rýchle zobrazenie ho však musíme stripifikovať, štvorec dvomi, kocku 12 šikovne usporiadanými trojuholníkmi, jediným pásom. Pracujeme teda s objektami, ktoré majú nekonečný počet bodov, no ktoré vieme konečne reprezentovať a v konečnom čase zobraziť. Trojuholník zadáme tromi bodmi a vykresľujeme v rastri.

Scéna sa zapisuje najčastejšie grafom scény (ako jednotlivý objekt CGS-stromom) a **animácia** či **interakcia** modifikáciou grafu scény, napr.

3 minúty https://www.youtube.com/watch?v=NH3J6Z5jKmM

15 minút, pre programátorov hier https://www.youtube.com/watch?v=ktz9AlMSEoA

Asi treba priznať, že vedome modelujeme a zobrazujeme s chybami: „chyba nie je vždy zlá. Vskutku, akceptovateľná chyba môže byť Vaším priateľom, poskytuje nám priestor na vyladenie a voľbu kde čo simulovať“ (voľne preložené z Hughes et al. s. 989, tu je aj celý obsah 38 kapitol našej „grafickej biblie“ https://ptgmedia.pearsoncmg.com/images/9780321399526/samplepages/0321399528.pdf,). Ako teda grafický objekt zobraziť? Navzorkovať na jeho viditeľnom povrchu dosť husté body s dosť presnou farbou a premietnuť na dosť presný displej, tojuholník po trojuholníku, pás po páse, postupne všetko, čo nájdeme v grafe scény v danom pohľade kamery. Cieľom je fotorealizmus (s výnimkou kapitoly 34 Expressive Rendering).

Fotorealizmus

Výklad sledovania lúča uvádza v 2D začiatok videa Coding Challenge #145: 2D Raycasting, stačí prvých 3 a pol minúty
https://www.youtube.com/watch?v=TOEi6T2mtHo

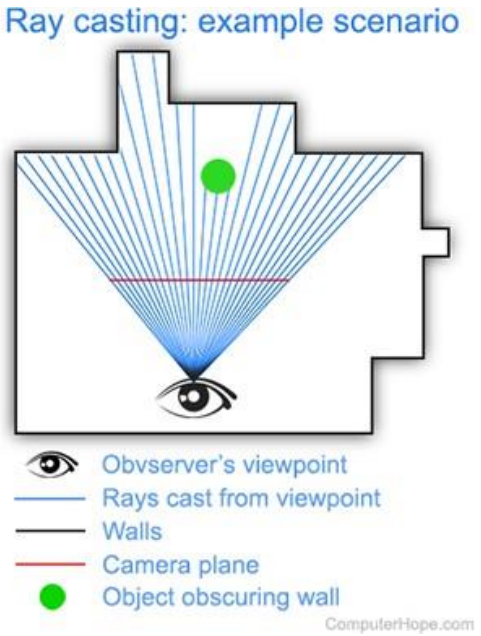
Ružický, s. 207n

Po modelovaní možno objekt zobraziť. Na **fotorealistické zobrazovanie** empirické osvetľovacie modely a tieňovanie (konštantné, Gouraudovo, Phongovo) nepostačujú,

treba prejsť k fyzikálne adekvátnejším: **globálne osvetľovacie modely**. Alternatívou **z-buffer** je viditeľnosť pomocou **Ray-casting**, sledovanie lúča z oka cez pixel do scény.

Algoritmus pre sledovanie lúča

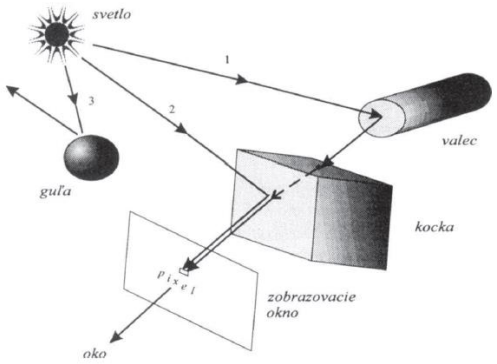
```
Program Ray-casting;
function ray_trace (lúč): farba ;
begin
2.1    Intersection (lúč; var bod, objekt);              { zisti prienik lúča s objektom }
2.2    if objekt > 0 then farba := Illumination (lúč, bod, objekt, zdroj_svetla)
2.3    else farba := farba_pozadia;
end;

begin
   for (všetky body_obrazovky) do
   begin
1.     lúč := Gener_ray (pozorovateľ, pixel);          { vytvor lúč od pozorovateľa }
2.     farba := ray_trace (lúč);                       { zisti intenzitu bodu}
3.     plot (pixel, farba);                            { zobraz bod obrazovky intenzitou }
   end
end.
```



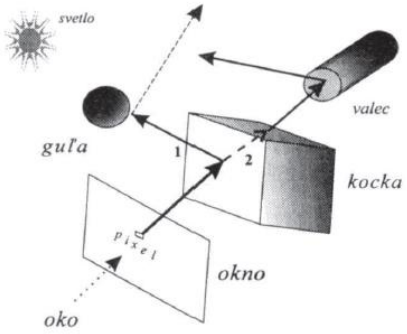
Na presnejšie, zložitejšie fotorealistické výpočty použijeme rekurzívne aj prenos svetla cez odraz a lom lúča.

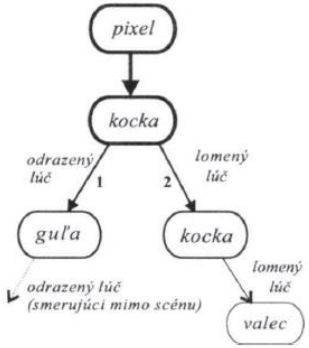
Výborne to vysvetľuje žijúci klasik Eric Haines na YouTube v dvoch častiach

https://www.youtube.com/watch?v=gBPNO6ruevk
https://www.youtube.com/watch?v=ynCxnR1ioQY



Obr. 14.1 Priamy a nepriamy lúč pre sledovanie lúča





Obr. 14.2 Strom pre sledovanie lúča

Rekurzívna funkcia *ray_trace*

```
function ray_trace (lúč) : farba ;
begin
    Intersection (lúč, bod, objekt);
    if objekt > 0 then
    begin
        farba1 := model (lúč, bod, objekt, zdroj_svetla){ farba1 sa určí podľa modelu }
        tieň_lúč := lúč (bod, zdroj_svetla);
        Intersection (tieň_lúč, tieň_bod, tieň_objekt);
        if tieň_objekt ≠ 0 then                    { zatienenie }
            farba1 := ambient;                     { farba1 sa určí ambientným osvetlením }
        zrkad_lúč := reflex (lúč, bod, objekt);
        if zrkad_lúč ≠ 0 then                      { odraz }
            farba2 := ray_trace (zrkad_lúč);       { farba2 zohľadňuje hodnotu k_r }
        lom_lúč := transp (lúč, bod, objekt);
        if lom_lúč ≠ 0 then                        { lom }
            farba3 := ray_trace (lom_lúč);         { farba3 zohľadňuje hodnotu k_t }
        farba := farba1 + farba2 + farba3;
    end
    else farba := farba_pozadia;
end;
```

ANIMÁCIA (neuvedená v učebnici Ružický)

Základné techniky

Medzinárodná norma PREMO (Presentation Environment for Multimedia Objects) [PREM94] definuje animáciu ako postupnosť obrázkov usporiadaných v čase na zobrazenie ako video. Toto pokrýva všetky zmeny, ktoré majú vizuálny efekt, t. j. v čase sa mení pozícia, tvar, farba, priehľadnosť, štruktúra a textúra objektu, ako aj osvetlenie, poloha kamery, jej orientácia a ohnisková vzdialenosť, ba aj technika renderingu. Mimochodom, táto definícia uvádza všetky možnosti, ktoré môže režisér použiť medzi dvoma susednými políčkami animovaného filmu.

Treba poznamenať, že postupnosť obrázkov môžeme získať aj videokamerou, napr. snímaním jednotlivých políčok nakreslených v klasickom animačnom štúdiu. V takom prípade hovoríme o klasickej animácii. Ak sme pri získaní políčok nepoužili počítač ani kreslenie, hovoríme o videozázname. Medzi animáciou a videom existujú aj ďalšie možnosti, napr. odpaľovanie náloží počítačom ako vo filme Bonnie and Clyde alebo kolorovanie starých Č/B PA3 filmov (Casablanca, Laurel a Hardy...).

Platí tu presne tá istá analógia ako medzi syntetickým obrázkom a prirodzeným obrazom. Pre animáciu poznáme aj 2D alebo 3D model, video podobne ako skener iba sníma. Model nemusíme poznať. V oboch prípadoch máme napokon animovanú sekvenciu, postupnosť políčok. Autor audiovizuálneho diela môže kombinovať obrazový materiál nezávislo od jeho pôvodu. (Výpoveď alebo príbeh môže aj vizualizačne vyrásť z dát, napr. From Visual Exploration to Storytelling and Back Again https://www.youtube.com/watch?v=r1DZGIYIFRo)

Animačná sekvencia sa navrhuje v nasledujúcich krokoch [Hear94, s. 584]:

* Scenár (*storyboard layout*)
* Definície objektov (*object definitions*)
* Špecifikácie políčok (*key-frame specifications*)
* Generovanie medzipolíčok (*generation of in-between frames*)

Scenár je náčrtom akcie. Definuje postupnosť pohybu (pohybov) ako súbor základných udalostí, ktoré treba predstaviť, ktoré sa musia stať. V závislosti na type animácie môže byť scenár súborom nákresov alebo zoznamom základných nápadov pre pohyb.

Definícia objektu sa robí pre každého účastníka akcie. Objekty možno definovať ako základné tvary (polygóny, splajny). Navyše môže byť pre každý tvar stanovený súbor pohybov.

Políčko (key frame, kľúčová snímka) predstavuje detailné vykreslenie statickej scény v danom čase niekde v animačnej sekvencii. Každý objekt sa lokalizuje na tom mieste, kde má byť v danom políčku. Niektoré políčka sú pre postupnosť zásadné, iné sa vyberajú na doplnenie tak, aby ich vzdialenosti v čase neboli priveľké. (Pripomeňme, že bežná televízna norma PAL/SECAM vyžaduje 25 políčok za sekundu.) Viac políčok si vyžadujú zložité a rýchle pohyby, kým na pomalé a jednoduché pohyby postačuje menej políčok.

Medzipolíčka (*in-betweens*) sa používajú na doplnenie potrebnej frekvencie zrakového vnemu u ľudí. Film používa 24 políčok, kým grafické obrazovky až 60 (včela sníma až 300 políčok za sekundu). V závislosti na rýchlosti pohybu možno niektoré políčka zdvojiť. Na minútový film bez zdvojenia potrebujeme 1440 políčok (*frames*), kým pri použití piatich medzipolíčok vystačíme iba s 288 políčkami. Ak navyše pohyb nie je priveľmi komplikovaný, môžeme políčka rozmiestniť v čase ešte vzdialenejšie.

Okrem horeuvedených 4 úloh treba na dokompletovanie sekvencie alebo celého audiovizuálneho diela riešiť ešte viaceré ďalšie, napr. verifikáciu pohybu, editovanie a produkciu a synchronizáciu zvukovej nahrávky.

Všeobecné funkcie pre počítačovú animáciu (výber)

Niektoré etapy animácie sa výnimočne hodia na počítačové spracovanie. Vytváranie a správa databázy objektov, pohybov, vykresľovanie políčok, generovanie medzipolíčok... Špičkový profesionálny softver (Softimage, Maya) poskytuje aj špeciálne funkcie na návrh animácie.

Funkcie na riadenie kamery zabezpečujú štandardné pohyby ako zoom, švenk a pod. Veľmi žiadané je automatické generovanie medzipolíčok.

Špeciálnym a niekedy potrebým spôsobom animácie je animácia v reálnom čase (*real-time animation*), kedy musí výpočet a zobrazenie políčok stihnúť vzorkovaciu frekvenciu. Horeuvedené celovečerné filmy sa však vytvárali off-line animáciou. Výpočet jediného políčka môže trvať aj na výkonnej platforme desiatky minút.

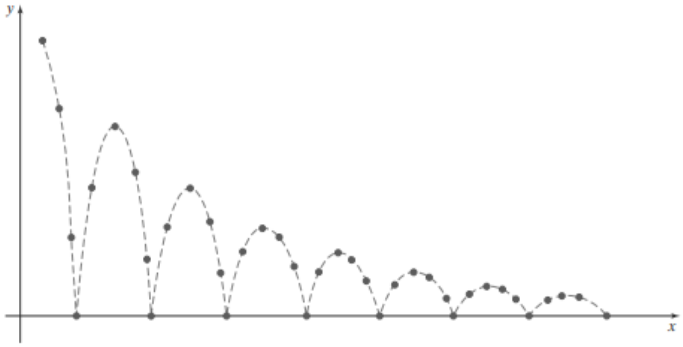


FIGURE 16
Approximating the motion of a bouncing ball with a damped sine function (Eq. 10).

(Hearn-Baker, 2014, p. 376)

Špecifikácie pohybu (úvodný príklad)

Pohyb možno niekedy špecifikovať priamo, napr. pre skákajúcu loptu pre 2D animáciu vieme napísať vzťah pre tlmené kosínusové kmity:

$y(t) = A \cdot abs(\cos(\omega t - T)) \exp(-kt)$,

kde A je výška amplitúdy, ω je uhlová frekvencia, T je fázový uhol a k tlmiaca konštanta. Modifikujeme absolútnou hodnotou známu funkciu sínus tak, aby lopta skákala nad zemou do výšky A a aby rýchlo klesajúca exponenciálna funkcia tlmila výšku y skackania v čase t podobne ako gravitácia. Tento jednoduchý model napodobuje jednoduchý pohyb, ale je nepresný, lebo sínusoida nepozostáva z parabol, ktoré by sme mali vyhodnocovať, keby sme počítali skutočný gravitačný model padania lopty a jej odrazov.

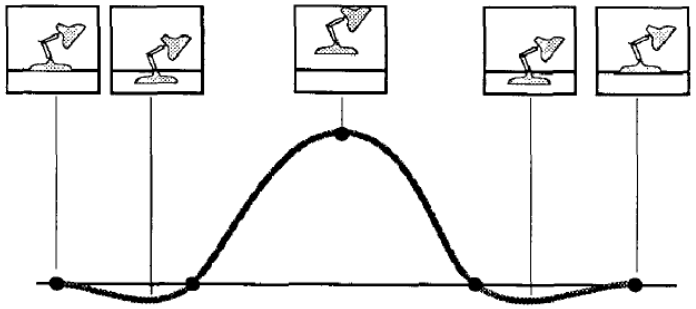


FIGURE 10a. This spline controls the Z (up) translation of Luxo Jr. Dips in the spline cause him to intersect the floor.

(Lasetter, 1987)

Všeobecným nástrojom na zachytenie pohybu sú krivky v trojrozmernom priestore, ktorými vieme dostatočne hladko pospájať požadované postupnosti bodov. Pri interpolácii rotačných pohybov sa niekedy okrem splajnov používajú aj zovšeobecné komplexné čísla, kvaternióny [Szir95]. Ak poznáme polohy, možno jednoduchú animáciu v 2D zapísať aj maticami posunutia, rotácie a škálovania a na dané miesto posunúť obrázok loptičky, doterajší obraz prekresliť novým. Inverznej matici tu rozumieme ako spätnému pohybu. Ak prekresliť aj s výpočtom stihneme 24-krát za sekundu, hovoríme o animácii v reálnom čase. Szirmay-Kalos to pre animačnú sekvenciu od času štartu po koniec zapisuje s označením t čas, o objekt, TM,o matica transformácie objektu o, TV matica pohľadu v nasledujúcom pseudokóde.

Initialize Timer(t_{start});
do
  t = Read Timer;
  for each object o **do** Set modeling transformation: $\mathbf{T}_{\mathbf{M,o}} = \mathbf{T}_{\mathbf{M,o}}(t)$;
  Set viewing transformation: $\mathbf{T}_{\mathbf{V}} = \mathbf{T}_{\mathbf{V}}(t)$;
  Generate Image;
while $t < t_{\text{end}}$;

Princípy/Disney
1. *Squash and Stretch* -- Defining the rigidity and mass of an object by distorting its shape during an action.
2. *Timing* -- Spacing actions to define the weight and size of objects and the personality of characters.
3. *Anticipation* -- *The* preparation for an action.
4. *Staging* -- Presentating an idea so that it is unmistakably clear.
5. *Follow Through and Overlapping Action* -- The termination of an action and establishing its relationship to the next action.
6. *Straight Ahead Action and Pose-To-Pose Action* -- *The* two contrasting approaches to the creation of movement.
7. *Slow In and Out* -- *The* spacing of the inbetween frames to achieve subtlety of timing and movement.
8. *Arcs* -- The visual path of action for natural movement.
9. *Exaggeration* -- Accentuating the essence of an idea via the design and the action.
10. *Secondary Action* -- The action of an object resulting from another action,
11. *Appeal* --- Creating a design or an action that the audience enjoys watching.

Slávna sekvencia s cválajúcim koňom a komentár: https://en.wikipedia.org/wiki/Twelve_basic_principles_of_animation