Animacija:

* Temelji animacije:
  + Krčenje in raztezanje(med udarcem)
  + Pričakovanje(Karakter v opravi za baseball, lahko pričakujemo da bo metal zogo)
  + Predstavitev(kako karakter izgleda, če je na primerni sceni ipd)
  + Tekoče akcije, kjer pri karakterju poskušamo čim bolje predstaviti pot(pijanec)
  + Pozne akcije (iz ene poze v drugo, kjer lahko pričakujemo kej se je zgodilo vmes)
  + Akcije follow through(akcije po koncu)(met zoge razbije okno)
  + Prekrivanje akcij(oseba na mono ciklu žonglira in se pogovarja po telefonu, druga oseba pijana).
  + Pospeševanje in upočasnevanje, da objekti dobijo tezo(brc zoge)
  + Gibanje po krivulji zaradi sklepov.
  + Sekundarne akcije, in časi(oseba se pojavi ravno v pravem času)
  + pretiravanje(hiter tek)
  + Osebnost osebka
  + Animacija gibanja (contact recoil passign highpoint contactm recoil ..), lahko predstavimo z preprostimi krivuljami.
* Kardanska zapora: predstavlja problem z eulerjevo enačbo, če se dve osi naenkrat poravnata potem obračamo obe naenkrat namesto ene, računalnik za rotacijo uporablja eulerjevo enačbo ali quetrnijevo. Zgodi se med dvema key framoma, ko računalnik zracuna vmesne korake. To se ne zgodi z queternijevo enačbo ampak nimajo podpore za animation curve. Quaterni uporablja pozicijske vektorje.
* Inverzna kinematika kako en vertex vplivan a druge pri okončinah
* Animacija gibanja delcev, sistemi delcev, jate, umetno življenje

Teksture:

* Polja vrednosti tekstur teksturne koordinate(u,v)
* Posamezni pikesl v teksturi se imenuje texel
* Več različnih vrst lepljenja tekstur, kot so ravninsko lepljenje, kjer se koordinati u in v preslikata na objekt po njegovih x,y koordinatah.
* Sferično okoli objekta je tekstura v obliki sfere, tako se u in v preslikata v theta in omega.x,y,z -> r theta, omega.
* Cilindrično lepljenje, okoli objekta je tekstura v obliki cilindra, kjer se koordinati u in v preslikata v theta in beta.x,y,z -> r,theta, beta.
* Prostorsko, objekt obkrožimo z kvadratom plošč.
* Naravno, da poznamo preslikavo iz x,y,z -> u in v kordiante,
* Kožno, primerno za živali .itd.
* Kadar so koordinate zunaj meja u in v € [0,1], se potem recimo vse, kar ja nad 1 slika ravno obratno kot narobe obrnjena slika.
* Različne vrste interpolacije:
* Linearna, hiperbolična. Blinearna , trilinearna
* Izračun Barve, Najbližji teksel.

Podatkovne strukture: ([vec](http://goo.gl/iSDmv1))

* Hierarhična podatkovna struktura, kar pomeni da mizi lahko dodamo otroka kot je naprimer luč zvezek ipd.
* Preproste hierarhije: so kot datoteke na os, za kategorizacijo objekto, kako so objekti povezani.
* Hierarhija je pogosto v obliki stars otrok, otroka lahko prosto premikamo ne da bi spremenili starsa, to pa ne velja za starsa, ce spremenimo starsa se spremenijo tudi vsi njegovi otroci. Temu pravimo da otrok podeduje lastnosti oceta.
* Folder hierarhije, se uporabljajo predvsem kadar hočemo da otroci podedujejo transformacije.
* Node hierarhije pa podpirajo še več, enačbe, omejitve, barve…
* V programih se pogosto uporablja Null ki je oče vsem delom. (recimo v blenderju koordinatni sistem ).
* Večnivojska hierarhija: recimo miza na tleh, če premaknemo tla se premakne tudi miza in vse kar je otrok mize, medtem ko premaknemo mizo, premaknemo samo mizo in njene otroke tla so nespremenjena.
* Pri hierarhijah je pomenbno da pravilno nastavimo imena objektov in njihove povezave (stars otrok).

Skeletne strukture:

* Lahko predstavimo, kot večnivojsko hierarhijo. Pomenbno predvsem pravilno nastavljanje hierarhije

Rendering Optimisation:

* Del ki ga kamera lahko vidi se imenuje frustum culling.(kot piramida v neskončnpost).
* Prva optimizacija je da izrišemo samo objekte,ki so v vidnem območju. (piramido omejimo z dvema kvadratoma(near and far clipping plane), da izrisuje samo dele ki so vmes.)
* Druga optimizacija predstavlja samo izris planov ki jih vidimo(backface culling), primer če imamo kocko(vidimo prve 3 plati, medtem ko zadnjih treh ne). Deluje tudi ko so objekti pred drugimi objekti.
* 3 optimizacija LOD(Level of detail), bližje ko smo objektu več poligonov se izrisuje. Za world pa uporabljamo discrete lod.
* MegaTexturing predstavlja podobno kot LOD, samo na teksturah, teksture ki so v našem blizjem območju so “boljse”, medtem ko dlje kot so slabse so.(zmanjsa porabo spomina)
* Pvs (potentiali visability sets) nalozimo samo tisto kar vidimo.
* HBV (octree), space partitioning.(za bounding box?)