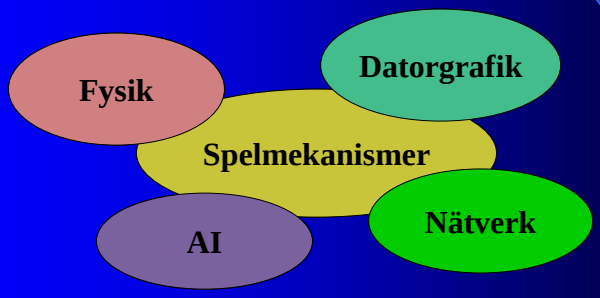
TSBK 10

Teknik för avancerade datorspel

Fö 10: Nätverk, Peter Johansson, ISY



Spelaspekter i vetenskaplig litteratur

M. Mauve, J. Vogel, V. Hilt, W. Effelsberg ”Local-Lag and Timewarp: Providing Consistency for Replicated Continuous Applications”, IEEE Transactions on Multimedia, vol 6, no 1, February 2004

Replicated Applications

- Replicated Application – en lokal kopia av tillståndet upprätthålls samtidigt av flera instanser
- Tillståndet förändras av (användarinitierade) operationer
- En operation utfärdas av en instans
- Effekterna av operationen på tillståndet behöver kommuniceras till alla instanser så deras respektive tillstånd kan uppdateras

Discrete/Continuous

- Discrete – tillståndet ändras endast av (användarinitierade) operationer
- Continuous – tillståndet kan också ändras av att tiden går, ingen kommunikation
  - Instanserna behöver hålla reda på aktuell tid
  - Klockorna är inte perfekt synkroniserade
  - När vi talar om tiden, t, menar vi enligt någon av dessa klockor, den tiden uppnås inte nödvändigtvis samtidigt för alla instanser!

Definitioner

- $s_{i,t}$ tillståndet som instans i har vid tiden t
- $o_{i,t^o,t*}$ operation som utfärdas av instans i vid tiden t^o och ska utföras vid tiden t*
- Mängden av alla operationer kallas O
- $R_i(t, o_{j,t^o,t*})$ mottagningsfunktionen, är sann om $o_{j,t^o,t*}$ tas emot av instans i innan tiden t och falsk annars

Konsistens

- Om det för alla t,i,j gäller (för alla t* <= t och $o_{w,t^o,t*}$ i O så att $R_i(t, o_{w,t^o,t*})$ och $R_j(t, o_{w,t^o,t*})$) medför ($s_{i,t} = s_{j,t}$) så har vi ett konsistent system
- Vid varje tid t ska tillståndet vid i och j vara samma, om både i och j har tagit emot alla operationer som ska ha utförts före t

Konsistens, kommentarer

- Instanser som inte tagit emot all information för att beräkna korrekt tillstånd påverkar inte om systemet är konsistent! (utan detta skulle inget system med fördröjningar kunna vara konsistent)
- Om alla instanser har tagit emot alla operationer vid någon tidpunkt t så måste tillståndet i alla instanser vara samma om systemet är konsistent.

Synkronisering

- Notera att definitionen av konsistens är oberoende av synkronisering av olika instansers klockor.
- Kriteriet kräver bara att de uppnår samma tillstånd vid samma avläsning av en fysisk klocka (om alla operationer tagits emot)
- Samma tillstånd behöver alltså inte uppnås samtidigt (samtidighet existerar inte?) enligt någon absolut referens-tid

Korrekthet

- Konsistens säger bara att alla instanser har samma tillstånd givet att de fått alla operationer, det säger inte att det tillståndet är det som uppnås av en instans som får alla operationer i rätt tid och ordning
- Definiera en virtuell perfekt instans P, som för varje $o_{i,t^o,t*}$ i O har $R_P(t*, o_{i,t^o,t*})$=sann, dvs alla operationer har tagits emot vid den tidpunkt då de ska utföras

Korrekthet

- Vidare ska P beräkna sitt tillstånd genom att för varje $o_{i,t^o,t*}$ i O utföra den vid tiden t* och uppdatera tillståndet enligt alla tidsberoende regler i systemet
- Om för alla t,i gäller (för alla t* <= t, och $o_{w,t^o,t*}$ i O så att $R_i(t, o_{w,t^o,t*})$) medför ($s_{i,t}$ = $s_{P,t}$) så är systemet konsistent och korrekt

Korrekthet

- Det betyder att om en instans i, vid tiden t, har fått alla operationer som ska ha utförts före t, så måste den ha samma tillstånd som den virtuella perfekta instansen P

Tillfällig inkonsistens

- Om det för en operation o_{i,t^o,t^*} och en instans j gäller att $R_j(t, o_{i,t^o,t^*})$ är falsk så sägs operationen ha orsakat en tillfällig inkonsistens vid instans j
- Hur länge en tillfällig inkonsistens varar beror på skillnaden mellan klockorna, fördröjning i överföring och hur mycket tidigare operationen utfärdades (t^o) än den ska utföras (t^*)

Tillfällig inkonsistens

- Given en operation o_{i,t^o,t^*} och en instans j, låt T^*_k beteckna en absolut tid då klockan vid instans k når t*, och $d_{i,j}$ fördröjningen i nätverket vid kommunikation från i till j
- Om $I_j(o_{i,t^o,t^*}) = d_{i,j} + T^*_i - T^*_j - (t^*-t^o)$ är negativ så har operationen inte orsakat tillfällig inkonsistens
- Responstiden för en operation är t^*-t^o
- Ideal responstid är 0, vi ser att det är en avvägning mellan responstid och risk för tillfällig inkonsistens

Time-warp

- Artikeln fortsätter med att definiera en formell algoritm för time-warp, som hanterar vad man ska göra om man får en operation för sent, det är väsentligen att backa tiden till tiden innan operationen skulle utförts, sen utföra den och alla andra operationer efter den som man fått
- De bevisar sen att time-warp ger korrekthet och konsistens

Död räkning

- I artikeln jämför de även med död räkning (dead reckoning), som är det vi behandlat tidigare med prediktion av tillståndet och sen interpolation till korrekt tillstånd när man väl fått det
- Samma typ av avvägning mellan respons och tillfällig inkonsistens finns här

Jämförelse

- Både död räkning och time-warp ger konsistens
- Bara time-warp ger korrekthet
- Död räkning har komplexitet O(n) medan timewarp har komplexitet $O(n^2)$
- Se artikeln för längre diskussion om för- och nackdelar med respektive metod

Interpolation

- Vi har nämnt att man kan behöva göra interpolation eller extrapolation i olika situationer (tappade paket, lag-kompensering, etc)
- Som exempel har jag nämnt positionen för en spelare
- Andra saker vi kan behöva interpolera är orientering/rotation

Hur representerar vi rotation?

- Eulervinklar, $Y=R_{yaw}R_{pitch}R_{roll}X$
  - Interpolation av vinklarna, problem, ryckighet, gimbal lock
- Ortogonal Matris, $Y=RX$
  - Interpolation? Enkelt att göra succesiva rotationer, men hur hittar vi rotationssteget?
- Rotationsaxel och vinkel, $Y=f(v,\mathbf{n},X)$
  - Hur interpolerar vi mellan olika rotationsaxlar?

Kvaternjoner

- Generalisering av komplexa tal, en skevkropp (skewfield), divisionsalgebra, funkar som vanliga tal (reella, komplexa) förutom att vi inte har kommutativitet $xy=yx$
- Tre komplexa enheter i, j, k där $i^2=j^2=k^2=ijk=-1$
- En kvaternjon $q=w+xi+yj+zk$, där w,x,y,z är reella tal, skrivs även $q=(w,\mathbf{n})$ där **n** är en tredimensionell vektor

Kvaternjoner

- $ij=k$, $jk=i$, $ki=j$, $ij=-ji$, $jk=-kj$, $ki=-ik$, funkar som kryssprodukten!
- $q_1=(w_1,\mathbf{n}_1)$, $q_2=(w_2,\mathbf{n}_2)$, $q_1q_2=(w_1w_2-\mathbf{n}_1.\mathbf{n}_2, w_1\mathbf{n}_2+w_2\mathbf{n}_1+\mathbf{n}_1 x \mathbf{n}_2)$
- Konjugat $q^*=(w,-\mathbf{n})$, norm $|q|^2=qq^*=w^2+|\mathbf{n}|^2$
- $(q_2q_1)^*=q_1^*q_2^*$
- Enhetskvaternjon q har $|q|=1$, kan skrivas $q=(\cos(v/2),\sin(v/2)\mathbf{n})$, där $|\mathbf{n}|=1$

Rotationer med kvaternjon

- En enhetskvaternjon $q=(\cos(v/2),\sin(v/2)\mathbf{n})$ representerar en rotation med vinkeln v kring axeln **n** om x är en vektor (realdel 0) så roteras den med $y=qxq^*$
- Sammansättning av rotationer är multiplikation av motsvarande kvaternjoner!
- Kvaternjoner kan interpoleras med slerp (spherical linear interpolation), ger den kortaste vägen på sfären (i 4 dimensioner) av enhetskvaternjoner

Slerp

- För att interpolera från q_1 till q_2 med en parameter t som går från 0 till 1, gör man $q=(q_2q_1^{-1})^tq_1=q_1(q_1^{-1}q_2)^t=q_2(q_2^{-1}q_1)^{1-t}=(q_1q_2^{-1})^{1-t}q_2$
- Där vi definierar q^t för $q=(\cos(v/2),\sin(v/2)\mathbf{n})$ som $q^t=(\cos(tv/2),\sin(tv/2)\mathbf{n})$
- För $q=(w,t\mathbf{n})$, där $|\mathbf{n}|=1$, är $\exp(q)=e^w(\cos(t),\sin(t)\mathbf{n})$

Slerp

- Från definitionen av exp ser vi att en kvaternjon q kan även skrivas q=Rexp((0,**n**)t) med |**n**|=1, då är log(q)=(log(R),**n**t)
- Kunde även definierat q^t=exp(t log(q)), blir samma sak som förra definitionen
- Slerp interpolerar med konstant hastighet

Squad

- Vid interpolation mellan flera kvaternjoner med slerp blir det diskontinuitet i hastigheten eftersom den är konstant mellan varje par av kvaternjoner i slerp
- Vi kan göra något som liknar bezier eller hermite-splines
- Med slerp(t,q_1,q_2)=$(q_2 q_1^{-1})^t q_1$ får vi squad(t,a,p,q,b) = slerp(2t(1-t),slerp(t,a,b),slerp(t,p,q))

Squad

- För att få kontinuitet när vi interpolerar mellan punkterna a_i ska vi välja $q_i=a_i\exp(-(\log(a_{i+1}a_i^{-1})+\log(a_{i-1}a_i^{-1}))/4)$ och interpolera med $\mathrm{squad}(t,a_i,q_i,q_{i+1},a_{i+1})$