Matemaattisten animaatioiden laatimisen tekniset ratkaisut

Simo K. Kivelä, Henna Komi, Ossi Mauno

2.6.2005

Matemaattisia käsitteitä voidaan usein havainnollistaa graafisilla esityksillä. Monissa tapauksissa syntyy lisäarvoa, jos esitys on muuntuva, jolloin esimerkiksi parametrien arvojen vaikutus voidaan nähdä, kolmiulotteista kohdetta voidaan katsoa eri suunnista jne. Tällaisia esityksiä on alettu kutsua *matemaattisiksi animaatioiksi*. Niiden liittäminen digitaalisiin dokumentteihin, esimerkiksi web-dokumentteihin on houkutteleva ajatus.

Matemaattisille animaatioille on tarpeen asettaa eräitä vaatimuksia, jotka erottavat ne monista muista animaatioista:

- Oikean mielikuvan saamiseksi animaatioiden tulee yleensä olla mittatarkkoja.
- Koska kyseessä on matemaattisen käsitteen tai ilmiön havainnollistaminen, lukijan/käyttäjän täytyy voida hallita animaation etenemistä ja halutessaan kelata sitä taaksepäin.
- Animaatioon tulee liittyä selittävä teksti, josta ilmenee, mihin piirteisiin on tarkoitus kiinnittää huomiota.
- Jotta animaatioiden katselu aktivoisi matemaattista pohdiskelua, niihin tulee liittyä harjoitustehtäviä, jotka voidaan ratkaista animaatiota tarkkailemalla.

Vaatimus mittatarkkuudesta merkitsee yleensä, että animaatio on luotava laskemalla. Tarjolla on useita erilaisia tekniikkoja sekä laskemiseen että animaation interaktiiviseen hallintaan. Tämä artikkeli on tiivis esitys niistä mahdollisuuksista, jotka on testattu MatTaFi-projektissa (`http://matta.hut.fi/mattafi/`). Tarkoituksena on antaa kuva tarjolla olevista vaihtoehdoista ja niiden ominaisuuksista, mutta yksityiskohtaisia ohjeita animaatioiden tekemiseen artikkeli ei sisällä. Nämä on haettava kunkin ohjelmistotyökalun omasta dokumentaatiosta.

Artikkelissa käsitellyt esimerkit löytyvät MatTaFi-projektin sivuilta osoitteesta `http://matta.hut.fi/mattafi/anim/anim.html`. Myös uusia animaatioita on tarkoitus ajan mukana lisätä.

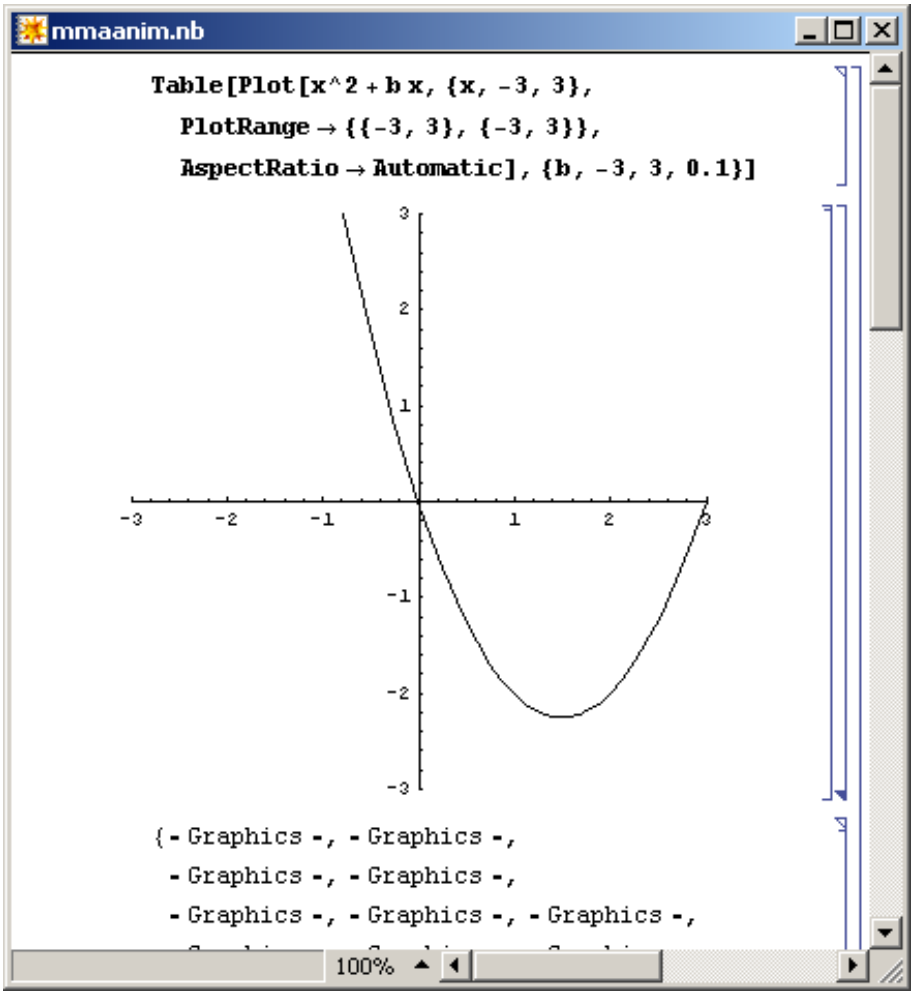
1 Laskentaohjelmat

Laskentaohjelmat (Mathematica [12], Maple [15], Matlab [14], Mathcad [11] etc.) tarjoavat yleensä työkalut animaatioiden muodostamiseen. Periaatteena on tällöin tuottaa ensin laskemalla sarja vähitellen muuntuvia kuvia ja tämän jälkeen esittää se katsojalle suhteellisen nopeasti. Kuvien vaihtumisnopeutta voidaan yleensä säätää.

Esimerkkinä olkoon Mathematican käsky, jolla tehdään sarja kuvia funktion $f(x) = x^2 + bx$ kuvaajista, missä kerroin b vaihtelee välillä $[-3, 3]$ askelena 0.1.

```
Table[Plot[x^2 + b*x, {x, -3, 3},
  PlotRange -> {{-3, 3}, {-3, 3}}, AspectRatio -> Automatic],
  {b, -3, 3, 0.1}]
```

Mathematican funktio `Table` tekee tarvittavan kuvasarjan (Mathematican terminologialla listan) ja funktio `Plot` piirtää kuvaajan. Määreet `PlotRange` ja `AspectRatio` ovat tarpeen asemoimaan ja skaalaamaan kaikki kuvat samalla tavoin. Kun sarja kuvia on muodostettu, varsinainen animointi käynnistetään valikosta (tai painamalla ctrl-y). Animointi tapahtuu Mathematica-ikkunassa.



Mathematica-dokumentti, jossa animaatio on konstruoitu, voidaan tallettaa, jolloin animaatiota voidaan käyttää myöhemmin uudelleen uusissa yhteyksissä. Haittana on tällöin, että käyttäjä tarvitsee Mathematican. Kuvasarja voidaan kuitenkin Mathematican keinoin (komento `Export`) tallettaa useisiin tiedostomuotoihin, esimerkiksi gif-kuvasarjaksi (yhteen tyyppiä gif olevaan tiedostoon), jolloin se voidaan Mathematicasta riippumatta esittää esimerkiksi www-dokumentissa. Animaation muuntelu ei tällöin kuitenkaan enää ole mahdollista. Esityksen hallinta puolestaan riippuu esitysmuodosta ja esitykseen käytetystä ohjelmasta.

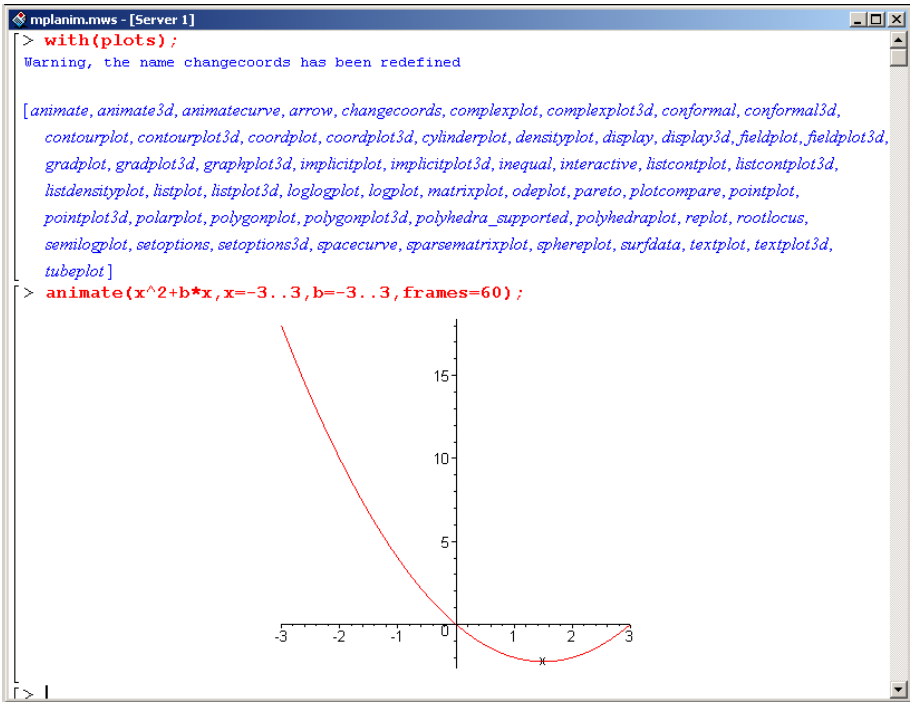
Edellä laadittu animaatio on nähtävissä gif-kuvasarjana: `http://matta.hut.fi/mattafi/anim/mmaanim.gif`.

Vastaavalla tavalla animaatioita voidaan tehdä lähes kaikissa laskentaohjelmissa. Tarvittavat komennot vaihtelevat ja usein animaatioita voidaan tehdä edellä olevaa mallia suoremminkin. Periaate ilmenee kuitenkin edeltä.

Esimerkkinä animaation tekemisestä yhdellä komennolla on seuraava Maplen komentorivi (vastaava työkalu löytyy Mathematicastakin):

```
with(plots);animate(x^2+b*x,x=-3..3,b=-3..3,frames=60);
```

Animointi voidaan käynnistää Maplen dokumentissa (napauttamalla hiiren kakkosnappia kuvan päällä), mutta se voidaan myös tallettaa gif-kuvasarjaksi: `http://matta.hut.fi/mattafi/anim/mplanim.gif`.



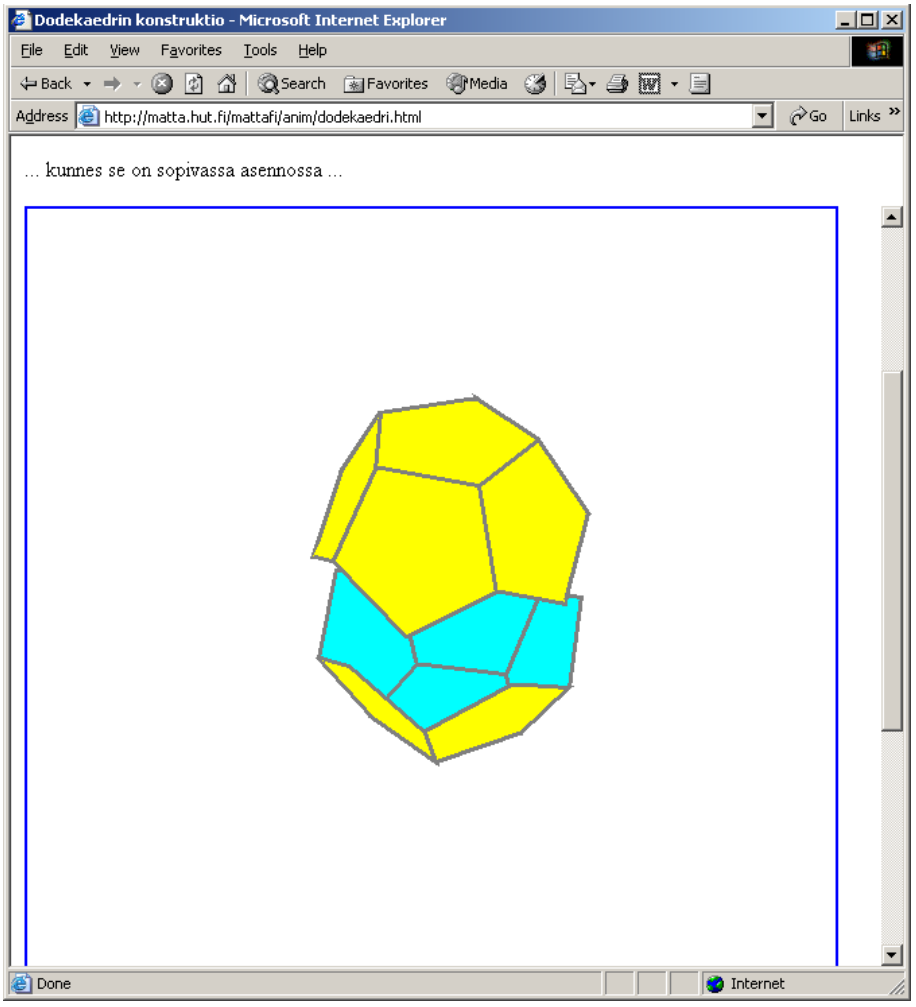
Mahdollisuudet hallita animointia laskentaohjelman omassa dokumentissa vaihtelevat ohjelman mukaan. Joissakin voidaan vain säätää esitysnopeutta ja suuntaa, toisissa voidaan luoda esimerkiksi liukusäätimiä monien asioiden hallintaan. Jos kuvasarja tulostetaan gif-tiedostoksi, voidaan tulostusvaiheessa valita esimerkiksi haluttu nopeus ja suunta, mutta käyttäjä ei näihin enää voi vaikuttaa.

Laskentaohjelmilla on myös edellä kuvattua laajempi merkitys animaatioiden laatimisessa. Vaikka animaation esitys ei tapahtuisikaan laskentaohjelmadokumentissa tai gif-kuvasarjana, on mittatarkat kuvat tai kolmiulotteiset mallit usein helpointa laskea ohjelman avulla. Esitys voi tämän jälkeen tapahtua monellakin eri tavalla, kuten seuraavat kohdat osoittavat.

2 JavaScript ja gif-kuvasarjat html-dokumentissa

Jos html-dokumentissa halutaan esittää gif-kuvasarja, jonka esitystä käyttäjä voi hallita, on yhtenä mahdollisuutena sisällyttää dokumenttiin *JavaScript-koodi*, joka vaihtaa kuvia sen mukaan, mitä käyttäjä osoittaa hiirellä. Edellytyksenä on, että skriptien ajo on sallittua selaimessa.

Lähtökohtana on tällöin sarja erillisiä gif-kuvia, kukin omassa tiedostossaan. Nämä on yleensä helpointa tehdä laskentaohjelman avulla. Useimmiten kannattaa kirjoittaa laskentaohjelman tarjoamalla kielellä ohjelmakoodi, joka tekee kuvat. Koodia muuttamalla voidaan tällöin suhteellisen helposti hakea juuri sellaiset kuvat kuin halutaan.



Esimerkkinä on dodekaedrin kokoamista esittävä animaatio `http://matta.hut.fi/mattafi/anim/dodekaedri.html`, jonka kuvat vaihtuvat käyttäjän siirtäessä hiirtä vaakasunnassa kuvan päällä. Ylimpänä on vaihtuvasisältöinen kokoamisprosessia selittävä tekstirivi.

Animaation perustana on 27 Mathematicalla tehtyä gif-kuvaa dodekaedrin kokoamisen eri vaiheista. Hiiren asema kuvan päällä määrää, mikä näistä näytetään. Html-dokumentin alussa on seuraavantyyppinen JavaScript-koodi:

```
<script>
kuva01= new Image();
kuva02= new Image();
...
kuva01.src='dode01.gif';
kuva02.src='dode02.gif';
...
kuva01.txt='<html><body>
  Lähtökohtana viisikulmion muotoinen pohjatahko.
  </body></html>';
kuva02.txt='<html><body>
  Pohjatahkon ympärille lisätään ensimmäinen sivutahko ...
  </body></html>';
...
function vaihda(kuva){
ani.document.images['dodepict'].src=kuva.src;
seli.document.open('text/html');
seli.document.writeln(kuva.txt);
seli.document.close();
}
</script>
```

Tässä määritellään aluksi kuvat ja tiedostot, joissa ne sijaitsevat, sekä selitystekstit. Lopussa määritellään funktio, joka saa argumentikseen kuvan nimen ('kuva') ja joka vaihtaa tämän 'dodepict'-nimiseen 'ani'-kehyksessä olevaan kuvaan. Samalla vaihdetaan vastaava selitysteksti 'seli'-kehykseen.

Funktiota kutsutaan 'ani'-kehyksen koodissa:

```
<img name="dodepict" src="dode27.gif" usemap="#dodemap">
<map name="dodemap">
<area onmouseover="parent.vaihda(parent.kuva01);return true;"
shape="rect" coords="0,0,20,600">
<area onmouseover="parent.vaihda(parent.kuva02);return true;"
shape="rect" coords="20,0,40,600">
...
</map>
```

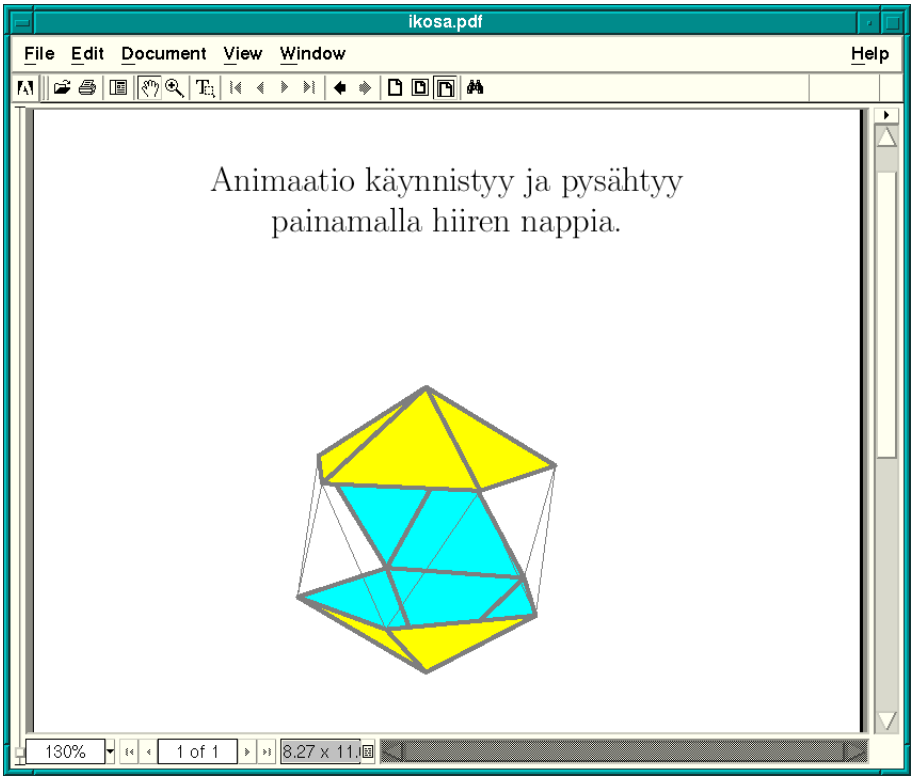
Tässä ensimminen rivi luo kuvan nimeltään 'dodepict', jossa dokumenttia avattaessa esitetään tiedoston `dode27.gif` kuva. Lisäksi ilmoitetaan, että käytetään seuraavilla riveillä määriteltyä karttaa 'dodemap'. Tässä kuva on jaettu koordinaattien avulla suorakulmioihin ja hiiren ollessa tietyn suorakulmion päällä, kutsutaan funktio 'vaihda' vaihtamaan 'dodepict'-kuvan sisältö.

3 Pdf-animaatiot

Pdf-dokumentteihin voidaan upottaa JavaScript-koodia, joka mahdollistaa erilaisten toiminnallisuuksien lisäämisen dokumentteihin. Tätä voidaan käyttää animaatioiden toteuttamiseen asettamalla yhdelle painikkeelle useita eri taustakuvia ja hoitamalla näiden vaihtaminen JavaScriptin avulla. Vaihtamista voidaan kontrolloida JavaScriptin keinoin monella eri tavalla: hiiren liikkeen tai napin painallusten avulla, tai animaatiokuvasarjan näyttäminen voidaan aloittaa automaattisesti, kun dokumentti avataan.

Taustakuvien vaihtamiseen perustuvien yksinkertaisten animaatioiden teko onnistuu kirjoittamalla dokumentti LaTeX-koodina ja käyttämällä Jochen Skupinin pakettia `pdfanim.sty` [19]. Edellytyksenä on, että animaatio muodostuu pdf-muodossa olevasta kuvasarjasta. LaTeX-koodi ajetaan pdflatexilla, jolloin syntyy pdf-tiedosto, johon animaatio on upotettu.

Tarvittavat pdf-muotoiset kuvat on ensin tuotettava jollakin tavalla, esimerkiksi laskentaohjelmalla. Parasta on, jos kuvat voidaan suoraan tallettaa pdf-mutoon. Toisena vaihtoehtona on kuvien tallettaminen esimerkiksi eps-mutoon ja konvertointi pdf:ksi sopivalla ohjelmalla (esimeriksi ImageMagick [8]).



Esimerkkinä tällä tekniikalla toteutetusta animaatiosta on ikosaedrin kokoaminen: `http://matta.hut.fi/mattafi/anim/ikosaedri.pdf`.

Skupinin paketti otetaan käyttöön määreellä `\usepackage{pdfanim.sty}` ja animaatiot liitetään dokumenttiin komennolla

`\PDFAnimLoad[`*optiot*`]{`*nimi*`}{`*ruutu*`}{`*ruutujen lkm*`}`,

missä *nimi* on animaatiolle annettava nimi, jolla siihen viitataan myöhemmin, *ruutu* viittaa osakuvien pdf-tiedostojen nimiin ja viimeinen parametri kertoo osakuvien lukumäärän. Esimerkiksi `\PDFAnimLoad{ani}{ruutu}{3}` määrittelee animaation, jonka ruudut ovat tiedostoissa `ruutu0.pdf`, `ruutu1.pdf` ja `ruutu2.pdf`.

Animaatio sijoitetaan dokumenttiin omalle paikalleen komennolla `\PDFAnimation{`*nimi*`}`.

Alla olevasta esimerkistä selviävät tarvittavan LaTeX-koodin oleelliset osat.

```
\usepackage{pdfanim}

\PDFAnimLoad[remember,interval=100,defaultframe=157,width=\textwidth]%
{ani}{frm}{158}

\begin{document}

\begin{center}
\PDFAnimation{ani}
\end{center}

\end{document}
```

Näin tuotettu animaatio käynnistyy painamalla hiiren nappia animaation päällä ja pysähtyy painettaessa nappia uudelleen. Komennolle `\PDFAnimLoad` annettu optio `remember` saa aikaan sen, että pysäytettäessä animaatio jää näkyville senhetkinen ruutu. Optio `interval` määrää peräkkäisten ruutujen välin millisekunteina ja `defaultframe=157` asettaa ennen animaation ensimmäistä käynnistystä näkyväksi ruuduksi animaation viimeisen ruudun eli tiedoston `frm157.pdf`.

Paketti `pdfanim.sty` on vielä hieman keskeneräinen ja animaation käyttäytymisen kontrollointi optioiden avulla on jonkin verran puutteellista. Esimerkiksi animaatio alkaa aina alusta pysäytyksen jälkeen. Tämän ongelman voi tosin kiertää paketissa olevan JavaScript-koodin editoimisella.

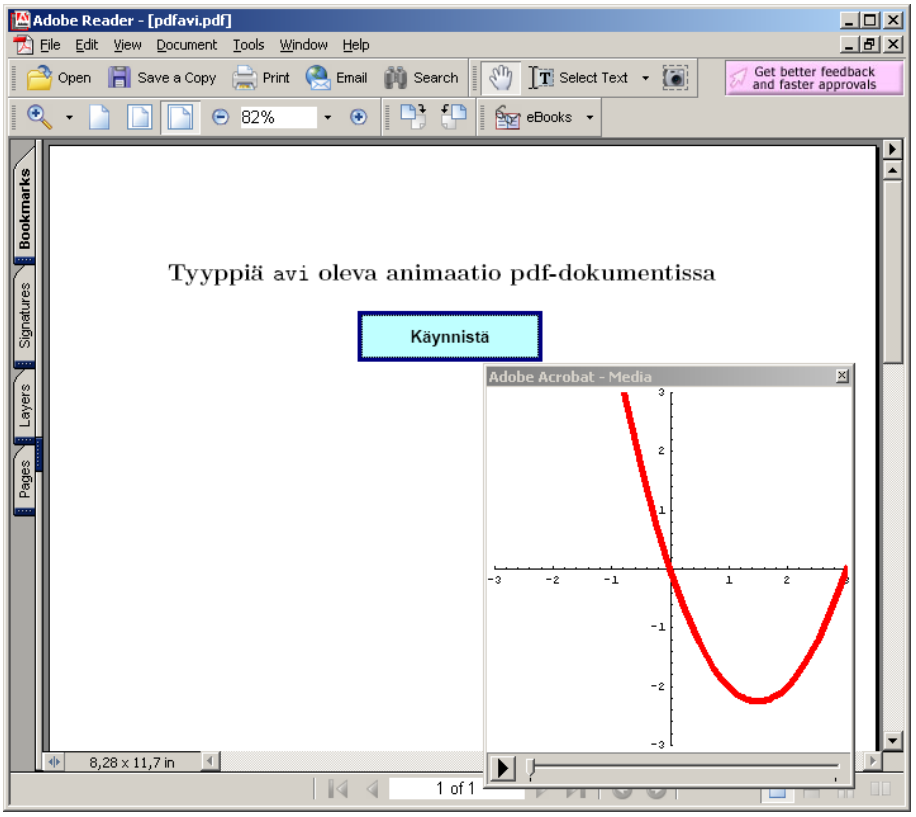
Animaatioita voidaan muokata myös Adobe Acrobat Professional -ohjelmalla. Esimerkiksi animaation käynnistymismekanismin voi vaihtaa editoimalla JavaScript-koodia, mutta muokkaaminen käy helposti työlääksi, jos muutettavia kohtia on useita.

4 Videotyyppiset animaatiot

Videoesityksiin perustuvia matemaattisia animaatioita on luonnollisesti myös mahdollista tehdä. Ongelmana tällöin on yleensä mittatarkan laskemalla tehdyn kuvasar-

jan laatiminen ja videoesityksen muodostaminen tämän pohjalta. Valmiin esityksen liittäminen html- tai pdf-dokumenttiin on suhteellisen ongelmatonta. Varsinainen esittäminen edellyttää yleensä Media Playerin, QuickTime Playerin, Macromedia Flash Playerin tms. käyttöä.

Seuraavassa esimerkissä animaation ruudut on laadittu Mathematicalla ja ne on Mathematican `Export`-komennolla talletettu yhdeksi avi-tiedostoksi. Tämän käynnistäminen voi tapahtua paitsi html- myös pdf-dokumentista: `http://matta.hut.fi/mattafi/anim/anim.avi`, `http://matta.hut.fi/mattafi/anim/pdfavi.pdf`. (Esimerkkitiedostot ovat niiden yksinkertaisesta sisällöstä huolimatta huomattavan suuria: n. 15 MB.)



5 Java-sovelmat

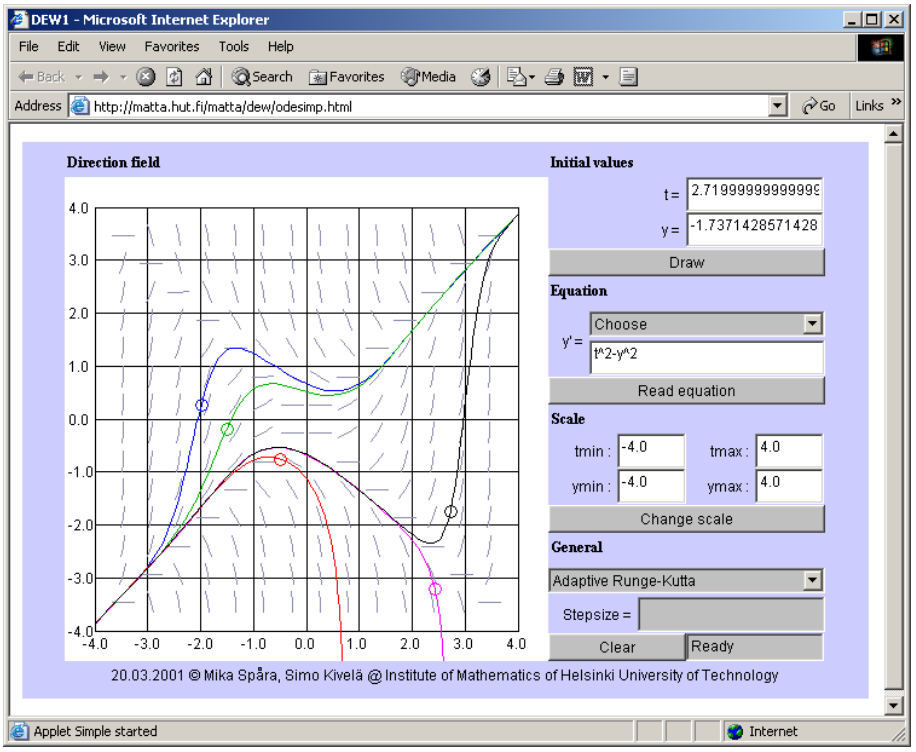
Html-dokumenttiin voidaan sisällyttää *Java-sovelma* (applet), joka sisältää Java-kielellä kirjoitetun ohjelmakoodin. Tämän avulla sivulle voidaan liittää animaatio, mutta sen luomisesta tulee ohjelmointitehtävä: tarvittavat kuvat ja toiminnallisuus on ohjelmoitava Javalla.

Valmiin sovelman käyttö html-dokumentissa on yksinkertaisimmillaan seuraavan näköinen:

```
<applet code="simple.class" archive="dew.jar" width="700" height="460">
</applet>
```

Tällöin www-palvelimella tulee olla (pakattu) tiedosto `dew.jar`, joka sisältää Java-koodin `simple.class`. Tämä siirretään html-dokumentin mukana käyttäjälle ajettavaksi selaimen Java-koneessa (Java Runtime Environment [9], Internet Explorerissa myös Microsoftin oma). Jotta ajo onnistuisi, selaimen asetusten tulee sallia ajo.

Esimerkkinä animaatiosta olkoon sovelma, jolla voidaan piirtää differentiaaliyhtälöiden ratkaisukäyriä: `http://matta.hut.fi/mattafi/dew/odesimp.html`. Ohjelmointiin on tarvittu Java-koodia noin 1200 riviä.



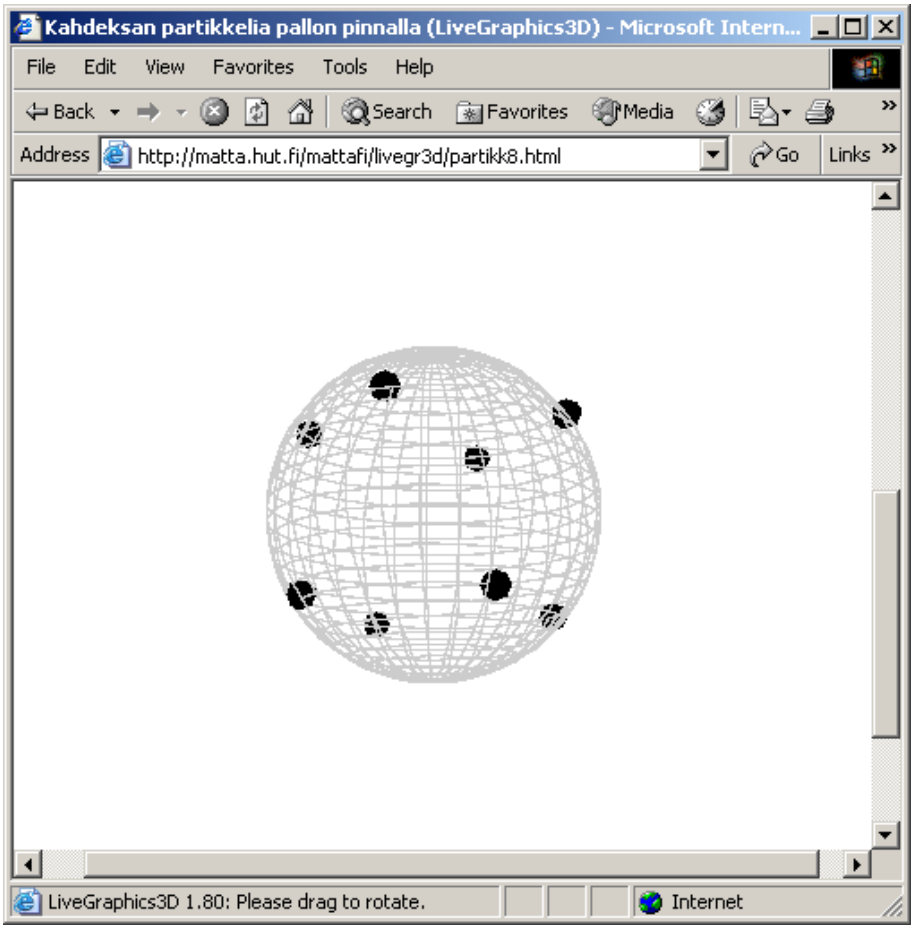
Tarvittavaa ohjelmointityötä voidaan pienentää, jos kyseessä olevat animaatiot ovat siinä määrin samantyyppisiä, että voidaan laatia yhteinen Java-koodi niiden esittämiseen. Eri animaatiot määritellään tällöin Java-ohjelmalle annettavana datana. Tämä saattaa olla varsin laajakin, mutta sen generoimisessa voidaan pyrkiä käyttämään esimerkiksi laskentaohjelmistoja.

Seuraavassa esitettävät ohjelmistot LiveGraphics3D ja Cabri ovat esimerkkejä tällaisesta Java-sovelmien käytöstä.

5.1 LiveGraphics3D

LiveGraphics3D [10] on ei-kaupalliseen käyttöön tarkoitettu Java-sovelma, jolla voidaan näyttää Mathematicalla tuotettuja kolmiulotteisia kuvia ja animaatioita html-dokumentissa. LiveGraphics3D mahdollistaa muun muassa objektien pyörittelyn ja zoomauksen.

Yksinkertaisimmassa LiveGraphics3D-sovelluksessa esitetään kolmiulotteinen malli, jota voidaan hiiren avulla kääntää tai saattaa pyörimään: `http://matta.hut.fi/mattafi/livegr3d/partikk8.html`.



Tällöin html-dokumentti sisältää seuraavantyyppisen Java-sovelman (appletin):

```
<APPLET ARCHIVE="live.jar" CODE="Live.class" WIDTH="500" HEIGHT="500">
<PARAM NAME="MAGNIFICATION" VALUE="0.7">
<PARAM NAME="INPUT_FILE" VALUE="partikk8.grf">
</APPLET>
```

Tässä `live.jar` on pakattu tiedosto, joka sisältää Java-koodin `Live.class`. Syöttötiedosto (partikk8.grf) sisältää kyseessä olevan kolmiulotteisen mallin Mathematican Graphics3D-objektina.

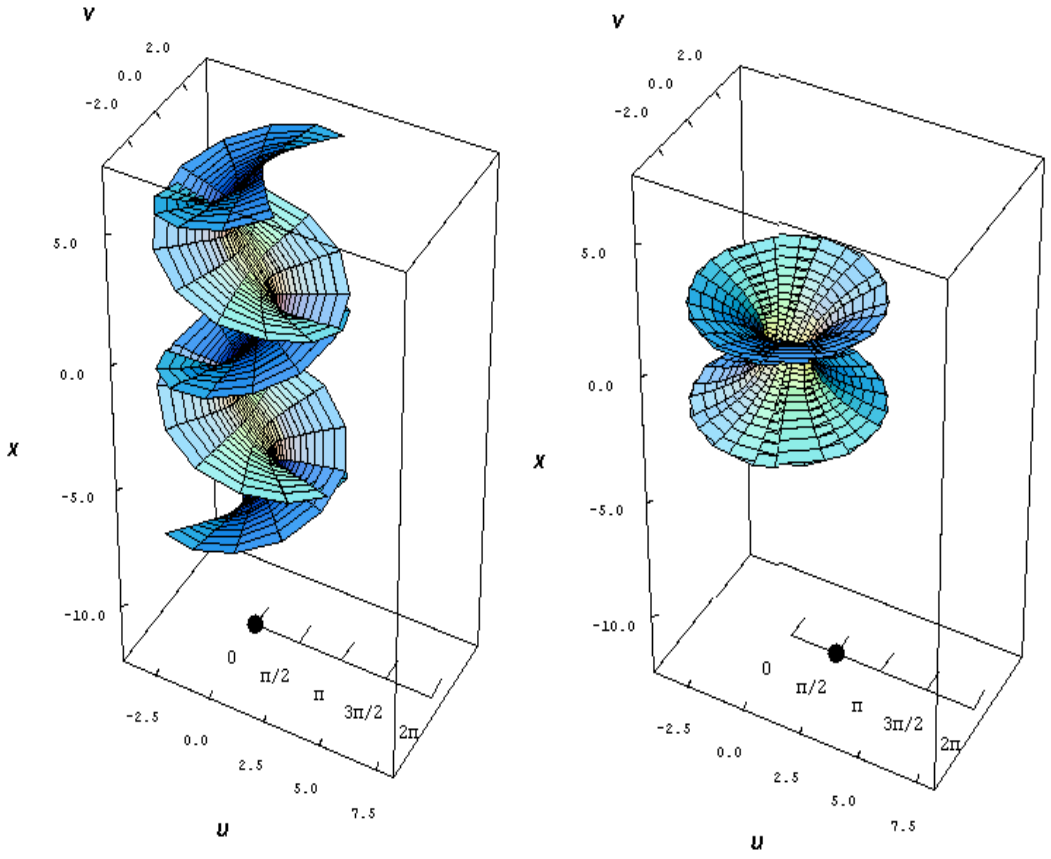
LiveGraphics3D:llä voidaan tuottaa myös varsinaisia animaatioita, joita käyttäjä voi hallita. Näitä on kolmea eri tyyppiä.

Yksinkertaisimmassa varsinaisessa animaatiossa on syöttötiedostossa lista Graphics3D-objekteja ja listaan on kohdistettu funktio `ShowAnimation`. Listan alkiot määrittävät animaation yksittäiset kuvat. Animaatio käynnistyy, kun hiiri siirretään kuvan päälle. Animaatio voidaan pysäyttää missä vaiheessa tahansa hiiren kaksoisklikkauksella, minkä jälkeen objektia voidaan hiiren avulla käännellä. Animaatio käynnistyy uudella kaksoisklikkauksella. Esimerkki: `http://matta.hut.fi/mattafi/livegr3d/aallot.html`.

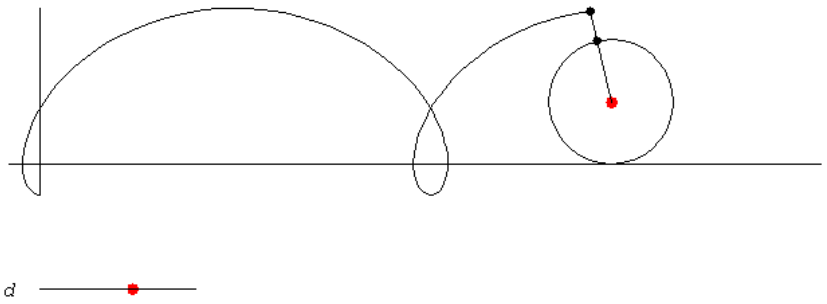
Toinen LiveGraphics3D:n tukema animaatiomuoto perustuu ns. *parametrisoituun grafiikkaan*. Käyttäjä voi tarttua hiirellä johonkin Graphics3D-objektin pisteeseen,

joka määrittää ns. riippumattoman muuttujan, ja siirtää sitä. Tästä riippuva muu grafiikka muuttuu tällöin vastaavasti.

Esimerkkinä on kompleksifunktion $\sin z$ kuvaaja neliulotteisessa avaruudessa, joka on yhdensuuntaisprojektiolla projisioitu kolmiulotteiseksi pinnaksi: `http://matta.hut.fi/mattafi/livegr3d/Sinz.html`. Janalla oleva piste määrittää riippumattoman muuttujan, josta riippuu kuvaajan asema neliulotteisessa avaruudessa: kun käyttäjä siirtää janalla olevaa pistettä hiirellä, pintaa kierretään (neliulotteisessa avaruudessa).



Toisena esimerkkinä LiveGraphics3D:n parametrisoiduista animaatioista on sykloidi `http://matta.hut.fi/mattafi/livegr3d/sykloidi.html`. Ympyrän keskipisteen siirtäminen aiheuttaa vierimisliikkeen, joilloin varrella oleva piste piirtää sykloidin. Kyseessä on tasoanimaatio, mutta se on koodattava kolmiulotteisena, ts. asetettava kaikkialla kolmannelle koordinaatille vakioarvo.

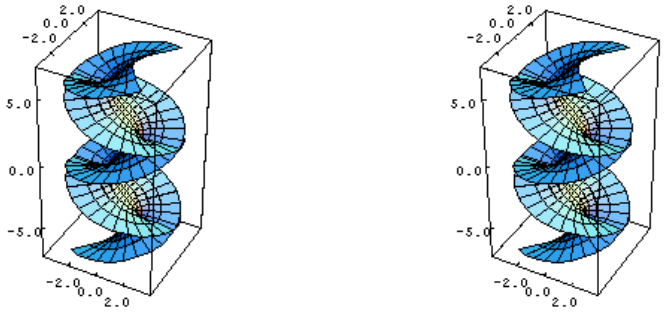


Kolmantena animaatiomuotona on yhdellä tai useammalla parametrilla varustettu Graphics3D-objekti, jossa yksi parametri toimii aikaparametrina (komentona `Animate` samalla tavoin kuin Mathematicassa). Näin saadaan animaatio, joka voidaan ajaa läpi siirtämällä hiiri LiveGraphics3D-objektin päälle. Käyttäjä voi pysäyttää animaation, tarttua mahdollisten riippumattomattomien muuttujien avulla määriteltyihin pisteisiin ja siirtää niitä. Näin koko näkymä päivittyy ja animaatiota voidaan ajaa jälleen läpi.

Parametrisoidun grafiikan tapauksessa html-sivuille sijoitettavassa koodissa tulee määritellä lisäksi riippumattomat muuttujat. Grafiikan riippuvuus näistä sisältyy mahdollisiin erikseen määriteltyihin riippuviin muuttujiin ja syöttötiedoston (alla olevassa esimerkissä sinz.grf) Graphics3D-objekteihin, jotka sisältävät muuttujia. Sinifunktioesimerkissä koodi on seuraava:

```
<applet archive="live.jar" code="Live.class" width="150" height="100">
<param name=independent_variables value="{xx->0}">
<param name=dependent_variables value="{t->(2*Pi/8)*xx}">
<param name=input_file value="sinz.grf" >
</applet>
```

LiveGraphics3D sisältää muitakin ominaisuuksia. Esimerkkinä muista toiminnoista on stereokuvapari kompleksifunktiosta $\sin z$. Tämä saadaan aikaan näppäimellä 's'. Steoreovaikutelmaa voidaan myös muuttaa.



5.2 Cabri ja muut dynaamisen geometrian ohjelmistot

Cabri Géomètre [1] on ns. *dynaamisen geometrian* ohjelmisto. Tällaisissa voidaan hiiren avulla konstruoida geometrisia kuvioita periaatteessa euklidisen geometrian tapaan. Valmiin kuvion lähtökohtina olevia objekteja (pisteitä, suoria, ympyröitä etc.) voidaan siirtää tai muuntaa, jolloin konstruktion periaate säilyy ja kuvio muuntuu tätä vastaavasti. Ohjelmisto toimii tällöin geometrian tutkimisen työkaluna.

Cabrin tyyppisiä dynaamisen geometrian ohjelmistoja on useita. Tunnetuimmat Cabrin kilpailijat ovat Geometer's Sketchpad [7] ja Cinderella [4]. Myös kolmiulotteisia versioita alkaa olla tarjolla, Cabri 3D:n lisäksi Calques 3D [3].

Kuvioiden konstruointi edellyttää, että käyttäjällä on ohjelma itsellään. Valmiin kuvion muunteluun sen sijaan on tarjolla myös Java-sovelmien käyttöön perustuvia ratkaisuja. Valmis Java-koodi on saatavissa tasogeometriatapauksessa kaikkiin edellä mainittuihin ohjelmiin. Kolmiulotteisissa ohjelmissa tällaisia ei ainakaan toistaiseksi liene olemassa.

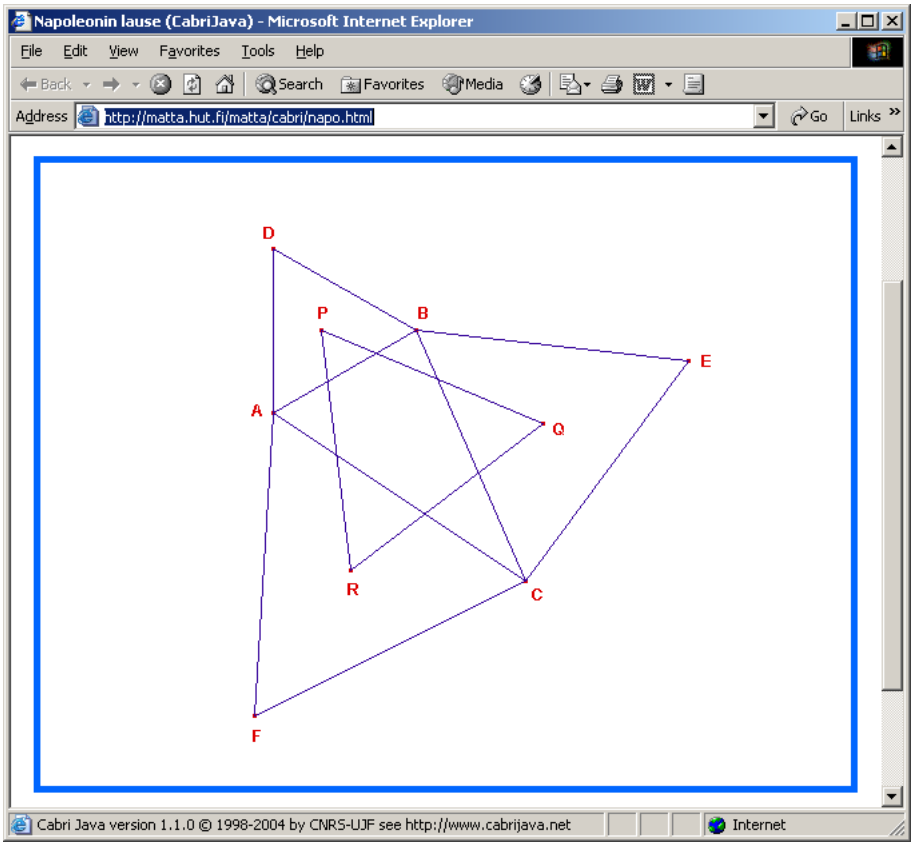
Cabrin tapauksessa html-sivulle sijoitettava koodi näyttää yksinkertaisimmillaan seuraavalta:

```
<APPLET CODE="CabriJava.class" WIDTH="617" HEIGHT="477"
 ARCHIVE="CabriJava.jar">
<PARAM NAME="file" VALUE="napo.fig">
</APPLET>
```

Tässä `CabriJava.jar` [2] on valmiina saatava tiedosto, joka sisältää tarvittavan Java-ohjelman. Tälle annetaan syötteeksi parametrimäärittelyssä ilmoitettu (palvelimella sijaitseva) tiedosto `napo.fig`, joka sisältää kyseessä olevan kuvion konstruktion määrittelyn. Tiedostotyyppi `fig` on Cabrin dokumenttityyppi luotujen konstruktioiden tallettamiseen.

Java-sovelmien käyttö edellyttää siten, että laatijalla on Cabri käytettävissään `fig`-tiedoston luomiseen, mutta html-sivulla kuviota voidaan muunnella ilman erityisohjelmistoja. Uusia kuvioita ei html-sivulla siten voida luoda.

Esimerkkinä on ns. Napoleonin lausetta esittävä Java-sovelma: `http://matta.hut.fi/mattafi/cabri/napo.html`. Konstruktion lähtöelementteinä ovat olleet pisteet A, B ja C, joita voidaan selainikkunassa hiiren avulla siirrellä.

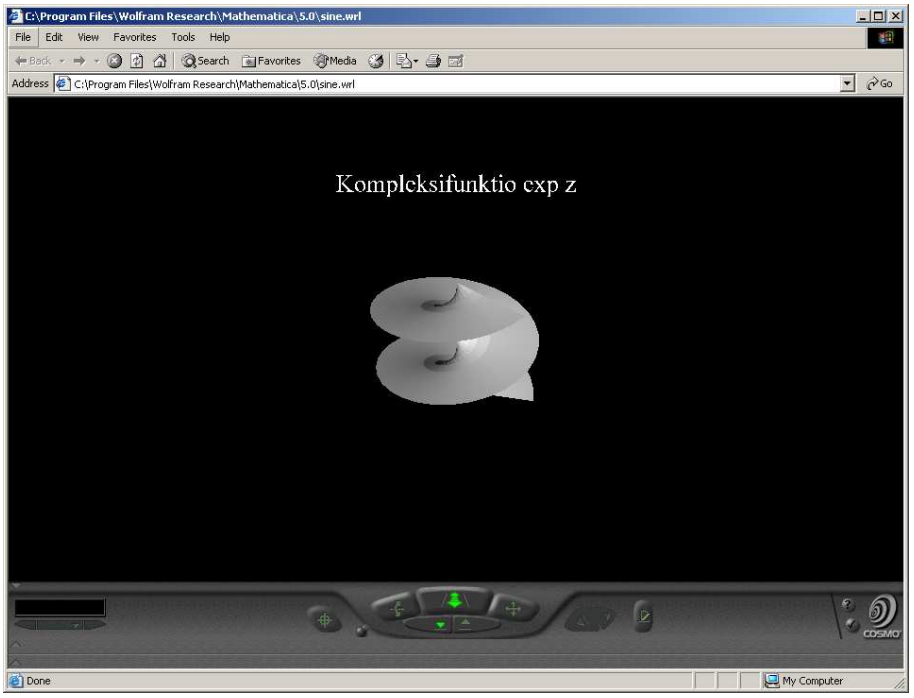


6 VRML

VRML (Virtual Reality Modeling Language) mahdollistaa www-selaimen kautta katsottavien kolmiulotteisten ympäristöjen suunnittelun. Tiedostotyyppi on wrl. Ympäristöjä voidaan katsoa (ja niissä liikkua) selaimen sopivan lisäosan (plug-in) avulla. Tällaisia ovat esimerkiksi Cosmo Player [6] ja Cortona [5].

Matemaattisia VRML-ympäristöjä voidaan luoda Mathematican lisäpaketilla VRMLConvert [17], joka muuntaa Mathematicalla tuotettuja Graphics3D-objekteja VRML:n version 1 mukaisiksi wrl-tiedostoiksi. MathGL3d [13] on toinen, varsin monipuolinen Mathematican lisäpaketti, jossa kolmiulotteiset mallit voidaan muuntaa VRML:n version 2 mukaisiksi wrl-tiedostoiksi.

Esimerkkinä on kompleksifunktion exp z neliulotteisen kuvaajan projektio kolmiulotteiseen avaruuteen (Riemannin pinta): `http://matta.hut.fi/mattafi/anim/exp.wrl`.



Kuva on Cosmo Playerin näytöstä. Selainikkunan alareunassa näkyy työkaluja pinnan tutkimiseen.

Mathematican grafiikka, esimerkissä funktion $\exp z$ Riemannin pinta voidaan muuntaa VRML-muotoon seuraavasti:

```
ekspfkt=ParametricPlot3D[{E^x Cos[y],E^x Sin[y],y},
   {x,-2,2},{y,-6,6}, PlotRange->All]
Needs["VRMLConvert`"]
VRMLConvert[ekspfkt, "exp.wrl",PlotLabel->"Kompleksifunktio exp z"]
```

Toisena esimerkkinä on kahdeksan partikkelin asettaminen pallon pinnalle tasaisesti (sopivalla kriteerillä mitattuna): `http://matta.hut.fi/mattafi/anim/partikk8.wrl`.

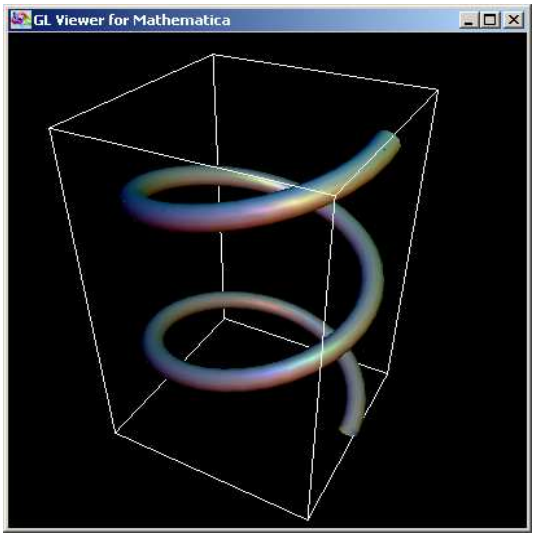
7 MathGL3d

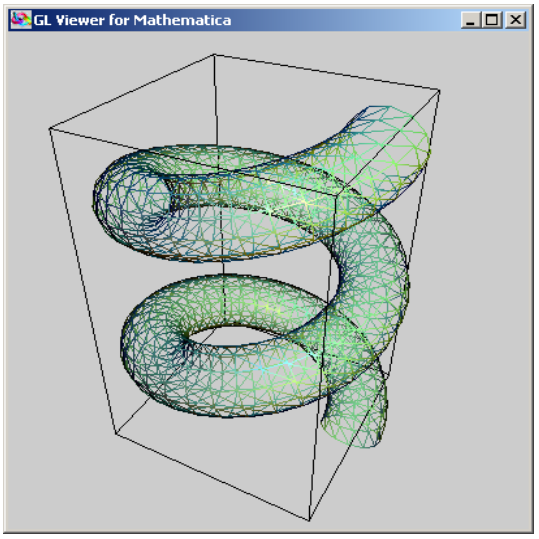
MathGL3d [13] on OpenGL-pohjainen ohjelma, joka mahdollistaa Mathematicalla tuotettujen kolmiulotteisten kuvien katselun ja pyörittelyn sekä yksinkertaiset animaatiot. Teknisesti kyseessä on Mathematican lisäpaketti, joka asennetaan Mathematican tiedostojärjestelmään.

Paketti ladataan komennolla `Get["MathGL3d`OpenGLViewer`"]`, jolloin aukeaa erillinen ikkuna paketin komennoilla tehtäviä graafisia esityksiä varten. Nämä voivat olla myös animaatioita (esimerkiksi komennot `MVTranslate[]` ja `MVRotate[]`). Hiiren kakkosnapin painaminen MathGL3d-ikkunassa avaa paletin, jonka avulla kuvan

ominaisuuksia voidaan muuttaa ja animaatioita hallita. Graafisten esitysten pohjana ovat kolmiulotteiset mallit ja ne voidaan tallettaa paitsi kuvina myös mm. VRML-muotoon.

Seuraavassa on käyrä $(\cos t, \sin t, t/4)$ esitettynä MathGL3d-ikkunassa. Käyrän paksuudeksi on asetettu 0,1. Toisessa kuvassa on saman käyrän ympärille muodostetu ruuviputki muokkaamalla mallin esitystä MathGL3d-ikkunassa.





Seuraava koodi luo em. käyrän ja kiertää sitä MathGL3d-ikkunassa hiljalleen kulman $\pi/10$ välein täyden kierroksen verran z-akselin ympäri.

```
MVShow3D[ParametricPlot3D[{Cos[t], Sin[t], t/4}, {t, 0, 4*Pi}],
   MVLineTubeSize -> 0.1];
Do[MVRotate3D[{0,0,1},Pi/10],{i,0,19}]
```

8 webMathematica

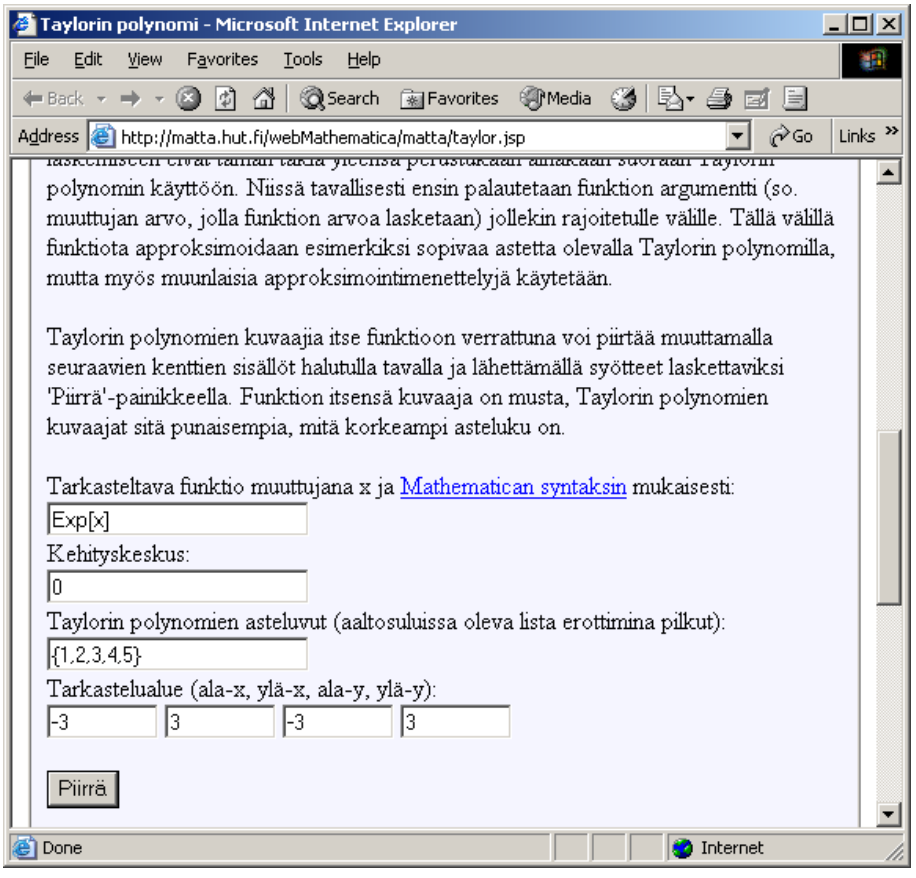
Mathematica-ohjelmistosta on saatavissa webMathematicaksi [18] kutsuttu versio, jota ajetaan www-palvelinkoneella. Tälle tarkoitetut syötteet voidaan antaa lomakkeella www-sivulla ja lähettää sen jälkeen sivu laskettavaksi, jolloin palvelinkoneessa käynnistyy Mathematica-prosessi. Täysi Mathematican laskentakapasiteetti on käytettävissä. Tulokset palautetaan käyttäjälle generoimalla halutunmuotoinen www-sivu.

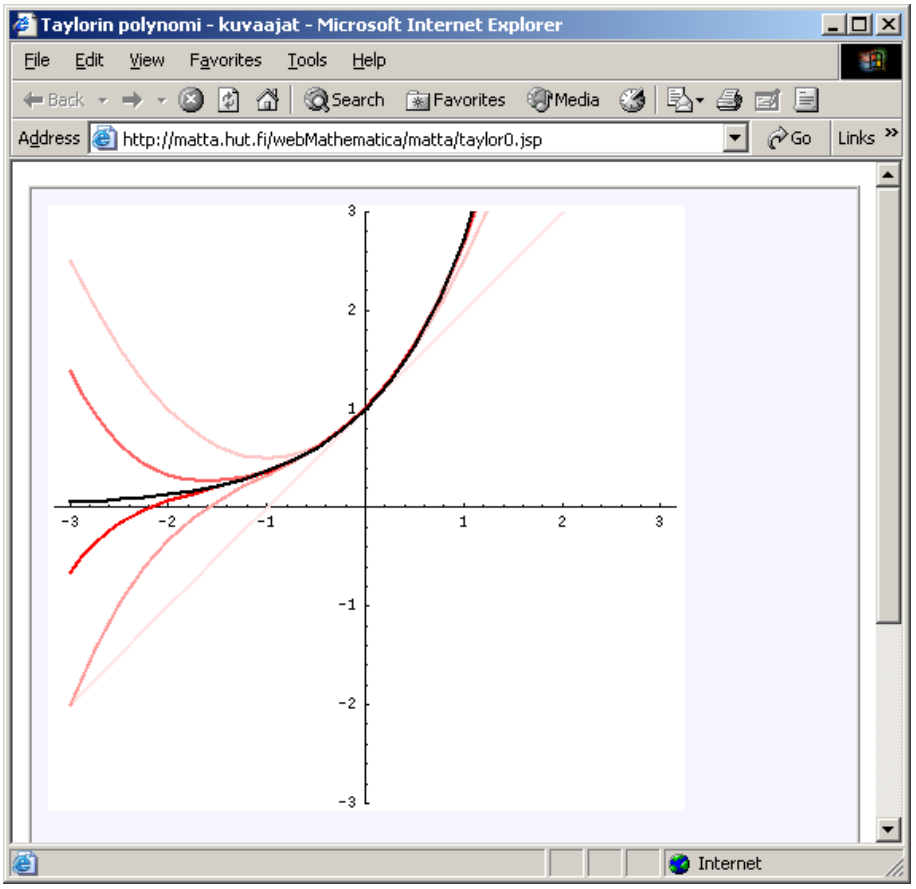
Prosessi on jossain määrin raskas. Palvelinkoneen kuormitusta on mahdollista rajoittaa asettamalla Mathematica-prosessille suurin sallittu ajoaika. Riskinä on luonnollisesti, että tällöin käyttäjän lähettämä laskenta katkeaa kesken.

webMathematica-sovellusten tekeminen on ohjelmointitehtävä: syötesivu on html-koodia, tulostesivu sisältää html-määrittelyjen lisäksi laskennassa tarvittavan Mathematica-koodin seuraavaan tapaan:

```
<msp:allocateKernel>
<msp:evaluate>
MSPBlock[{$$fkt,$$keskus,$$aste,$$alku,$$loppu,$$ala,$$yla},
 n= Length[$$aste];
 sarjat= Table[Series[$$fkt,{x,$$keskus,$$aste[[k]]}],{k,1,n}];
 ...]
</msp:evaluate>
...
</msp:allocateKernel>
```

Esimerkkinä on annetun funktion Taylorin polynomien kuvaajien piirtäminen: `http://matta.hut.fi/webMathematica/matta/taylor.jsp`. Syöte- ja tulostesivu näyttävät seuraavilta:





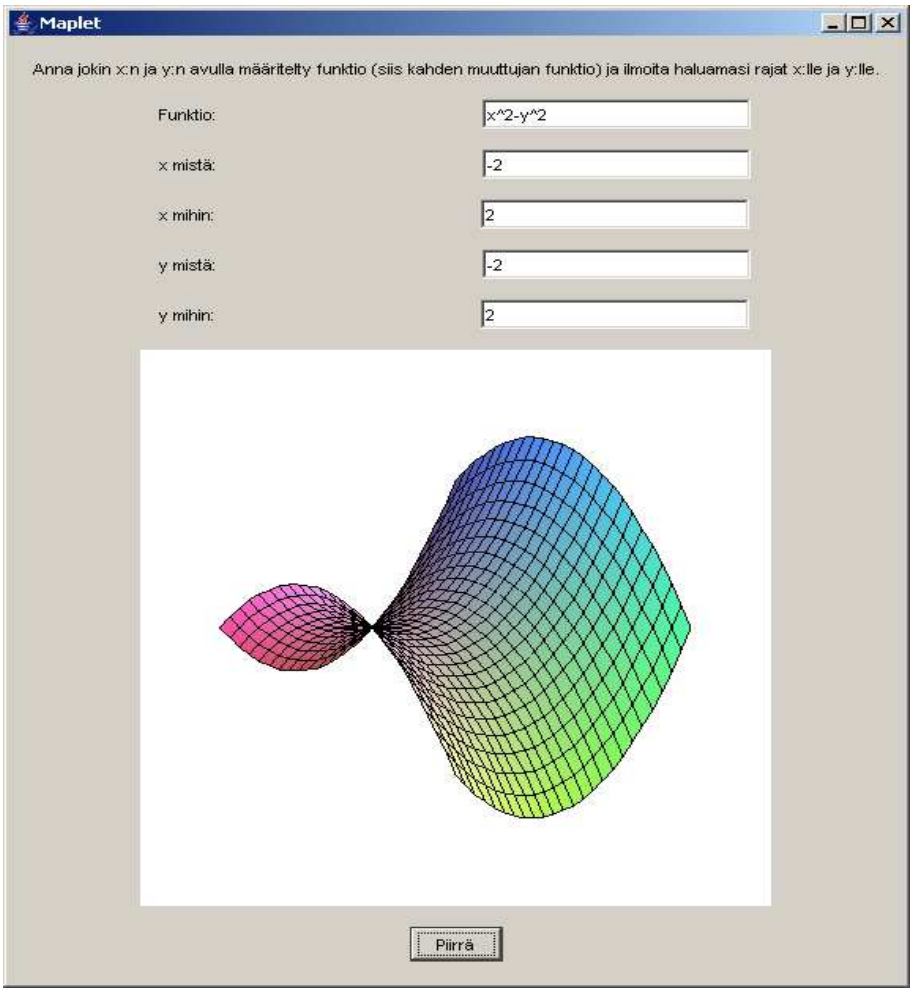
9 MapleNet ja mapletit

Maplella on mahdollista luoda graafisia käyttöliittymiä usealla eri tavalla, mm. Maplet-tekniikalla. Nämä ns. mapletit ovat joko selaimen kautta käytettäviä verkkosovelluksia tai paikallisesti Maplessa avattavia käyttöliittymiä. Verkkosovellus asennetaan palvelimelle, jossa ajetaan MapleNet-ohjelmistoa. Mapletti avaa ikkunan, joka voi sisältää tekstikenttiä, painikkeita ja valikoita. Käyttäjän ei tarvitse tuntea Maplea.

MapletNet [16] on web-palvelimella toimiva ohjelmakokonaisuus. Tällöin www-dokumentit voivat sisältää mm. Java-sovelmia, jotka kommunikoivat MapleNet-palvelimen kanssa. Yksinkertaisin vaihtoehto on luoda Java-sovelmat Maplen Mapletspaketin avulla. Toiminnallisuus voidaan tällöin testata Maplen sisällä ja sen jälkeen siirtää sovelma, ns. mapletti palvelimelle. Sovelman laatijan tulee osata käyttää Maplea, käyttäjän ei.

Laskennan tai grafiikan tarvitsemat syötteet annetaan www-sivulla mapletin kentissä ja ne lähetetään lomakkeelta palvelimelle laskettaviksi. Palvelimella ajetaan Maplea ja saadut tulokset palautetaan mapletille.

Esimerkkinä mapleteista on kahden muuttujan funktion kuvaajan piirtäminen annettujen syötteiden perusteella. Mapletti on osoitteessa `http://matta.hut.fi/mattafi/anim/mapletpinta.maplet`; avaamiseen käyttäjä tarvitsee MapletViewerin. Pintaa voidaan myös käännellä hiiren avulla MapletViewerin ikkunassa.



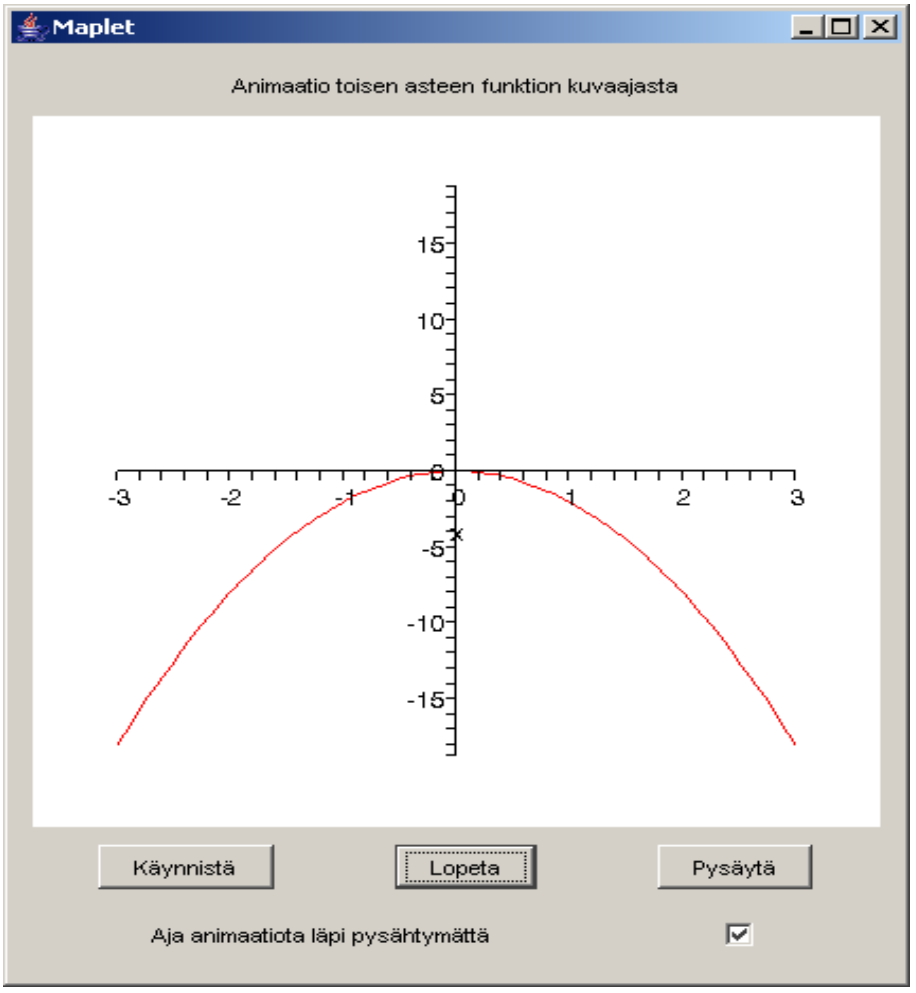
Mapletti on saatu aikaan seuraavalla Maplen määrittelyllä:

```
with(Maplets[Elements]);
mapletti:=Maplet([
["Anna jokin x:n ja y:n avulla määritelty funktio (siis kahden
  muuttujan funktio) ja ilmoita haluamasi rajat x:lle ja y:lle."],
["Funktio: ", TextField['funktio']()],
["x mistä: ", TextField['x1']()],
["x mihin: ", TextField['x2']()],
["y mistä: ", TextField['y1']()],
["y mihin: ", TextField['y2']()],
[Plotter['kuva']()],
[Button("Piirrä", Evaluate('kuva'='plot3d(funktio,x=x1..x2,y=y1..y2)'))]
]):
Maplets[Display](mapletti);
```

Ensimmäinen komento `with(Maplets[Elements])` lataa tarvittavan Maplen paketin. Komento `TextField['nimi']` määrittelee tekstikentän nimeltään ”nimi”, johon käyttäjä voi antaa syötteen. Funktiota `Display` käytetään mapletin näyttämiseen.

Toisena esimerkkinä nähdään animaation sisältävä mapletti. Kuvassa on toisen asteen funktion kuvaaja $f(x) = ax^2$. Kun animaatio käynnistetään, muuttuu vakion a

arvo arvosta -2 arvoon 2. Mapletti on osoitteessa `http://matta.hut.fi/mattafi/anim/mapletanim.maplet`.



Mapletin koodi on seuraava:

```
with(Maplets[Elements]);
kuvaaja:=plots[animate](a*x^2,x=-3..3,a=-2..2,frames=30):
animaatio := Maplet([["Animaatio toisen asteen funktion kuvaajasta"],
Plotter[P](kuvaaja,continuous=false),
[Button("Käynnistä", SetOption(P('play')=true)),
Button("Lopeta", SetOption(P(''stop'')=true)),
Button("Pysäytä", SetOption(P('pause')=true))],
["Aja animaatiota läpi pysähtymättä",
CheckBox[CONTINUOUS](value=false,
onchange=SetOption(target=P,'option'='continuous',
Argument(CONTINUOUS)))]]):
Maplets[Display](animaatio);
```

Viitteet

[1] Cabri Géomètre, http://www-cabri.imag.fr/

[2] Cabri Java Project, http://www.cabri.net/cabrijava/

[3] Calques 3D, http://www.psyc.nott.ac.uk/staff/nvl/Calques3D/

[4] Cinderella, http://www.cinderella.de/

[5] Cortona, http://www.parallelgraphics.com/products/cortona/

[6] Cosmo Player, http://cic.nist.gov/vrml/cosmoplayer.html

[7] Geometer's Sketchpad, Key Curriculum Press, http://www.keypress.com/sketchpad/

[8] ImageMagick, http://www.imagemagick.org/

[9] Java Runtime Environment, Sun Microsystems, Inc., http://www.java.com

[10] LiveGraphics3D, http://www.vis.uni-stuttgart.de/~kraus/LiveGraphics3D/

[11] Mathcad, Mathsoft Engineering & Education, Inc., http://www.mathcad.com/

[12] Mathematica, Wolfram Research, Inc., http://www.wolfram.com/

[13] MathGL3d, http://phong.informatik.uni-leipzig.de/~kuska/mathgl3dv3/

[14] Matlab, The MathWorks, Inc., http://www.mathworks.com/

[15] Maple, Waterloo Maple, Inc., http://www.maplesoft.com/

[16] MapleNet, http://www.maplesoft.com/products/maplenet

[17] VRMLConvert, http://www.ma.iup.edu/MathDept/Projects/VRMLConvert/

[18] webMathematica, Wolfram Research, Inc., http://www.wolfram.com/products/webmathematica/index.html

[19] Jochen Skupin, The PDFAnim package, http://www-user.uni-bremen.de/~skupin/pdfanim/