Neljän kilotavun taide

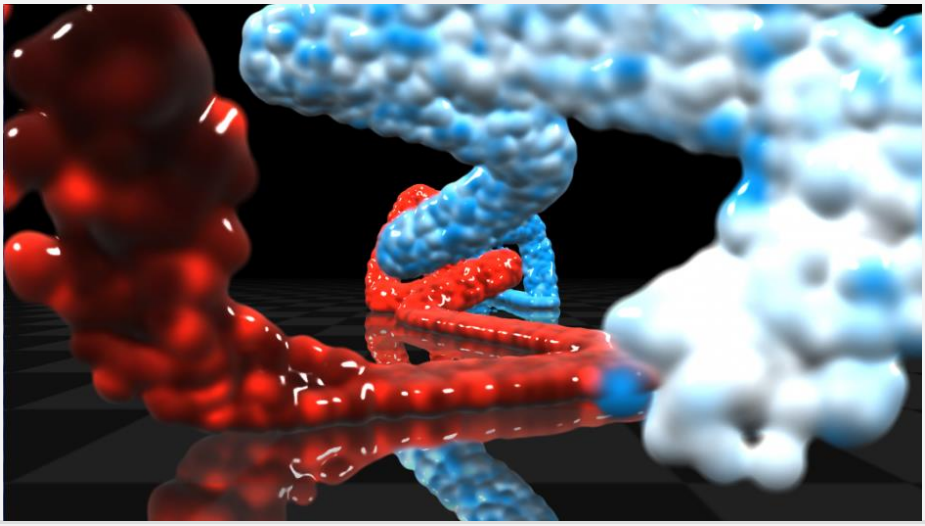
Markku Reunanen
Lehtori
Aalto-yliopiston taiteiden ja suunnittelun korkeakoulu, median laitos.
Turun yliopiston digitaalisen kulttuurin jatko-opiskelija

4k-introt ovat neljän kilotavun tiedostotilaan ahdettuja reaaliaikaisia audiovisuaalisia esityksiä, joita on tehty etenkin 1980-luvun puolivälissä syntyneen demokulttuurin eli demoskenen piirissä. Artikkelissa tarkastellaan 4k-introjen historiaa ja kulttuurista merkittävyyttä marginaalisena tietokonetaiteen muotona.

Kaksi potenssiin kaksitoista eli 4096 tavua ei ole paljon: yksi pikselirivi näytöllä tai arkillinen tekstiä sisältää suunnilleen saman verran dataa. Näinä päivinä, kun ohjelmien ja mediatiedostojen käyttämää levytilaa mitataan gigatavuissa, voi olla vaikea ajatella, että neljään kilotavuun mahtuisi jotakin merkityksellistä, saati että moiselle tiivistämiselle olisi edes tarvetta.

Digitaaliajan miniatyyrejä ovat *4k-introt*, reaaliaikaiset audiovisuaaliset esitykset, joita tehdään etenkin 1980-luvun puolivälissä syntyneen demokulttuurin eli demoskenen piirissä. Tässä artikkelissa introja tutkitaan yhtäältä kulttuurihistoriallisina artefakteina, jotka peilaavat tietoteknisen kentän ja demokulttuurin muutoksia, ja toisaalta taideteoksina, joiden kautta tekijät ovat tuoneet esille taituruuttaan. Historiallisen kehityskaaren lisäksi valotetaan käytännön tasolla niitä työkaluja ja lähestymistapoja, joilla neljän kilotavun haasteita on ratkottu. Käsittelyn keskiössä ovat PC-tietokoneille tehdyt introt, paitsi yleisyytensä vuoksi, myös siksi, että niissä ilmenee selvimmin kotitietokoneiden laskentatehon ja grafiikkaominaisuuksien kehitys 1990-luvun alusta asti.

Tarkemmin määriteltynä 4k-intro on ajettava ohjelmatiedosto, joka saa olla kooltaan enintään 4096 tavua sisältäen niin ohjelmakoodin, grafiikan kuin äänetkin. Häviävän pienestä koostaan huolimatta parhaat 4k-introt ovat varsin edistyneitä, sisältäen useita efektejä sekä kokoaan suuremmalta kuulostavaa musiikkia, jotka on synkronoitu toisiinsa ehjäksi kokonaisuudeksi (kuva 1). Avainsanoja näin rajallisen tilan kanssa toimittaessa ovat *generatiivisuus* ja *tiivistys*: niin visuaalit kuin äänetkin luodaan pääosin algoritmisesti, minkä lisäksi ohjelmakoodin koko optimoidaan tarkoitukseen kehitetyillä apuohjelmilla.



Kuva 1. Portal Processin ja TCB:n Nucleophile (2008).

Siinä missä varsinaiset demot ovat vuosi vuodelta kasvaneet suuremmiksi, ovat pikkuintrot menneet pikemminkin toiseen suuntaan. Perinteiset 40k- ja 64k-introt ovat saaneet 4k:n lisäksi rinnalleen edelleen pienempiä kategorioita kuten 1k-introt, joissa koko ohjelman pitää mahtua 1024 tavuun. Kahden potenssit seuraavat toisiaan: demoja arkistoivan *Pouet.netin* (http://www.pouet.net) luokituksesta löytyvät myös 512, 256, 128, 64 ja viimeisenä luokkana 32 tavun introt. Näissä äärimmäisen pienissä tuotoksissa ei ole tyypillisesti enää kuin yksi, tavu tavulta optimoitu ja kokoonsa nähden lähes mahdottomalta vaikuttava visuaalinen efekti. Tarkkaan määritellyt kategoriat kertovat osaltaan myös siitä, kuinka tärkeää teosten luokittelu on demokulttuurissa. (Reunanen 2010, 52–57.)

Tietokonedemot ovat kaiken kaikkiaan marginaalinen tutkimusaihe, ja 4k-introt ovat kaikesta päätellen marginaalin marginaalissa. Edes muuten mittavat skenekirjat *Freax* (Polgar 2005) ja *Kunst, Code und Maschine* (Botz 2011) eivät käsittele aihetta, vaan molempien painotus on täysikokoisissa demoissa. Omassa lisensiaattityössäni (Reunanen 2010, 52–57) on omistettu joitakin sivuja rajatun koon introille, mutta kaikkiaan pikkuintroista ei ole olemassa sanottavasti tutkimusta. Eräs kiintoisimpia kirjoituksia aiheesta on Sebastian Gerlachista (Minas/Calodox) *SCEEN*-lehteen tehty haastattelu, jossa useiden menestyneiden 4k-introjen ohjelmoija valottaa omia työskentelytapojaan (SCEEN #2 2007, 72–75). Aiheen kannalta merkittäviä lähdekokonaisuuksia on demotietokanta *Pouet.netin* lisäksi *IN4K* (http://in4k.northerndragons.ca/), johon on kerätty työkaluja sekä vinkkejä 4k-introjen tekijöille.

Tutustuin itse 4k-introjen tekemiseen vuosina 2003–2005, jolloin ohjelmoin Antti Silvastin kanssa kolme introa. Ensimmäinen niistä, *Yellow Rose of Texas* (2003), vaati selvästi eniten teknistä taustatyötä, siinä missä *Je Regrette* (2004) ja *Make It 4k* (2005) ovat pikemminkin samalle pohjalle rakennettuja kokeiluja. Kaikki kolme julkaistiin ensin Linuxille, minkä jälkeen niistä muokattiin versioita lukuisille muille alustoille. Osan käännöksistä tekivät muut harrastajat avoimen lähdekoodin hengen mukaisesti. Näiden projektien kautta hahmottuivat konkreettisesti ne lukuisat haasteet, joita kokorajoitus asettaa efektien ja musiikin ohjelmoinnille sekä käytettäville työkaluille.

4k-introt ennen ja nyt

1960-luvun ja 1970-luvun alun varhaisten tietokoneiden vaatimaton laskentateho ja muistimäärä rajoittivat monin tavoin ensimmäisten tietokonetaiteilijoiden ilmaisukeinoja, ja aikakauden digitaalista taidetta leimaakin minimalismi. Teknisten reunaehtojen lisäksi kyse oli tietoisista valinnoista: mediateoreetikko Lev Manovichin (2007, 342–344) mukaan tietokonegrafiikan pelkistetyn estetiikan juuret ovat 1800-luvun lopun ja 1900-luvun alun modernismissa, jonka myötä visuaalinen taide kulki minimalistiseen suuntaan. Ensimmäisen katsauksen nuoreen taidemuotoon teki Herbert W. Franke (1971) kirjassaan *Computer Graphics – Computer Art*. Kirjassa esiteltyjen pioneerien kuten Charles Csurin ja John Whitneyn työt näyttävät arkaaisuudestaan huolimatta kovin tutuilta, kun niitä vertaa 4k-introihin; matemaattisin kaavoin ja pienin resurssein generoitu grafiikka johti samankaltaiseen estetiikkaan myös 25 vuotta myöhemmin (ks. myös Saarikoski 2011).

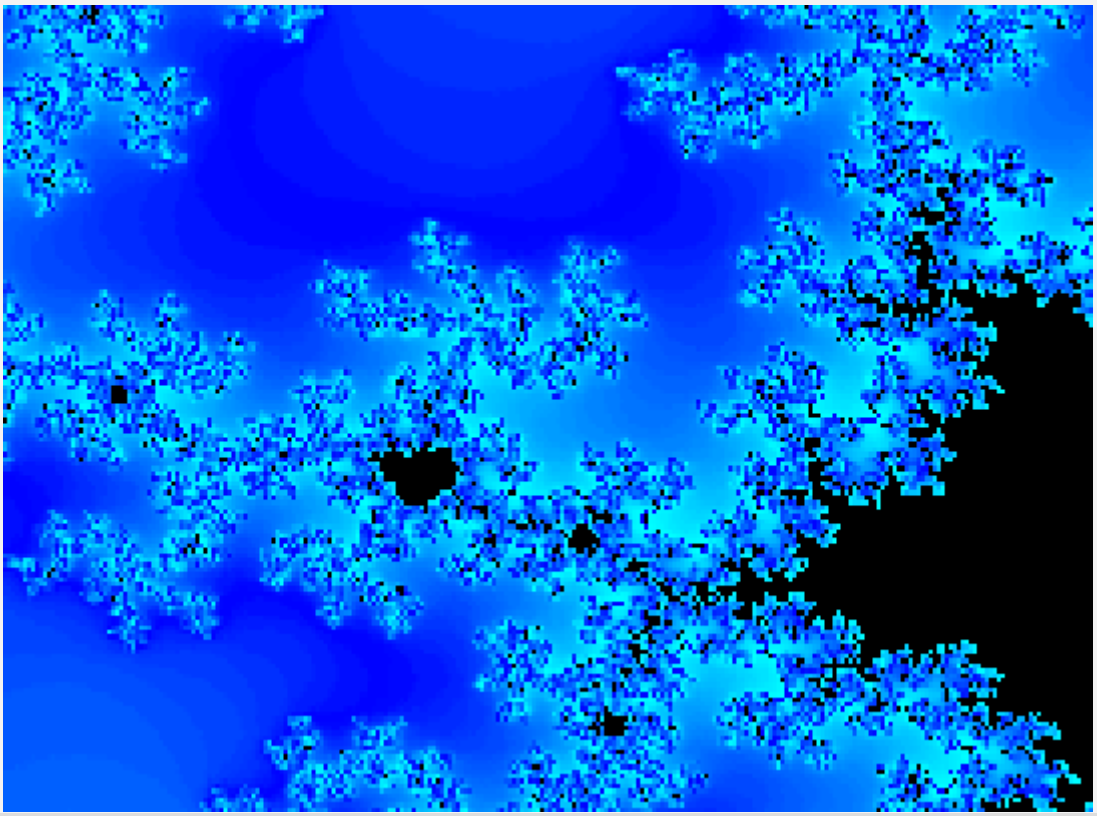
Siinä missä pikkuintrojen tekijöille kokorajat ovat lähinnä yhteisön noudattama käytäntö, olivat ensimmäisten kolmen sukupolven peli- ja kotitietokoneiden vastaavat rajoitukset hyvinkin konkreettinen seuraus laitteiden teknisistä ominaisuuksista. Kirjassaan *Racing the Beam* Nick Montfort ja Ian Bogost nostavat esiin tutulta kuulostavia lukemia Atari VCS -pelikonsolista: 6507-suoritin pystyi käsittelemään kerralla kahdeksan kilotavun muistiavaruutta, useat pelit mahtuivat kahteen kilotavuun ja pelimoduulin maksimikoko oli neljä kilotavua (Montfort ja Bogost 2009, 25–26). Kuten erikokoisten introjen tapauksessakin, ollaan jälleen kahden potenssien äärellä. Matemaattisen ulottuvuutensa lisäksi kyseiset luvut ovat tapa hahmottaa tietokoneen sisäistä maailmaa, minkä vuoksi ne toistuvat sellaisissakin yhteyksissä, joissa niiden käyttöön ei edes olisi teknisiä perusteita.

Demoskenen juuret ovat sitä muutamaa vuotta vanhemmassa kräkkeriskenessä, jonka tunnuspiirteenä olivat pelien alkuun lisätyt kuvat, jotka kehittyivät myöhemmin 1980-luvun puolivälissä visuaalisesti näyttäviksi *crack-introiksi*. Pelin alkuun sijoitettu crack-intro toimi kopiosuojaukset murtaneen ryhmän käyntikorttina, jolla nostettiin omaa statusta sekä luotiin ja ylläpidettiin harrastajien omia sosiaalisia verkostoja. (Reunanen 2010, 22–23; Polgar 2005, 40–70.) Kapasiteetiltaan pienet tallennusvälineet sekä joidenkin kymmenien kilotavujen muistimäärä asettivat tiukat puitteet introjen koolle, mikä johti jälleen omanlaiseensa minimalismiin. Murrettuja pelejä myös pakattiin pienempään tilaan kalliiden levykkeiden säästämiseksi (Wasiak 2012). Toinen esimerkki pieneen tilaan ahdetusta koodista ovat BBS-järjestelmien mainoksina toimineet *purkki-introt*, joita lisättiin 1990-luvulla järjestelmien kautta kulkeneisiin ohjelmapaketteihin (Reunanen 2010, 52). Voidaan siis perustellusti sanoa, että tilaoptimointi on ollut demoskenen keskiössä jo sen ensimetreiltä asti.

Varsinainen 4k-introjen genre syntyi *Pouet.netin* mukaan 1990-luvun alussa. Atari ST -harrastajien STNICC-tapahtumassa vuonna 1990 järjestettiin 3,5 kilotavun ohjelmointikilpailu nostalgisella nimikkeellä ”VIC Times Revisited” (Commodore VIC-20 ilmoittaa käynnistyessään vapaan muistin määräksi 3583 tavua). Osa kilpailutöistä oli pelejä ja osa introja, joten käytännöt eivät tuolloin olleet vielä täysin mukautuneet nykyiseen muotoonsa. Eräs osallistujista oli laama- ja kameliaiheisilla peleillään tunnetuksi tullut omintakeinen britti Jeff Minter. STNICC:n jälkeen kategoria jäi muutamaksi vuodeksi lähes unohduksiin, kunnes se otettiin osaksi demopartyjen kilpailuja eli *kompoja*.

4k-introt muuttuivat satunnaisesta kuriositeetista keskeiseksi kilpailusarjaksi etenkin suurimpien partyjen joukkoon nousseen Assembly’94:n myötä. Jo aiemmin samana vuonna oli järjestetty 4k-kompo Bush Partyssä, mutta viimeistään Assembly nosti pikkuintrot näkyviksi. Ajan hengen mukaisesti kilpailut oli jaoteltu tarkkaan laitealustan perusteella, ja niinpä sarjan nimi oli ”PC 4k”. Tuloksissa (*Assembly 94 results*, pouet.net) on näkyvissä vain yhdentoista yleisölle näytetyn intron tiedot, mutta kilpailuun osallistujia oli enemmän. Voittajaksi selviytyi saksalaisen Dust-ryhmän *Stoned*, jossa oli efekteinä ajalle tyypillinen kuvanpyöritys ja tunneli. Grafiikan generointia edusti aikakauden musiikkivideoistakin tuttu

Mandelbrotin fraktaali (kuva 2). Seuraavana vuonna loputkin suuret partyt, The Gathering ja The Party, ottivat 4k-introt mukaan ohjelmaansa, ja niin niistä tuli pysyvästi osa demoskenen kaanonia.



Kuva 2. Dust-ryhmän Stoned (1994)

Laitteiden markkinatilanteiden heilahtelut ja tekninen kehitys vaikuttavat 4k-introihin, mutta tietyllä viiveellä, sillä demoskene ei omaksu uutta teknologiaa nopeasti eikä kritiikittömästi. Tarkemmin aiheesta ovat kirjoittaneet Reunanen ja Silvast (2009), jotka mainitsevat eräiksi muutosvastarinnan syiksi kriittisen suhtautumisen kaupalliseen valtavirtaan sekä nopeampien tietokoneiden mukanaan tuoman kuvitteellisen helppouden. *Pouet.netin* tallentamien kilpailutulosten perusteella Amigan asteittainen katoaminen markkinoilta johti 1990-luvun loppuun tultaessa MS-DOS:lle tehtyjen introjen vahvaan asemaan suurissa partyissä. Erilliset Amiga- ja PC-kompot saivat väistyä, ja ne korvattiin yhdistetyillä sarjoilla. Samaan aikaan myös retrokoneiden parissa omaksuttiin 4k-introt ja niitä alkoi ilmestyä esimerkiksi Commodore 64:lle ja Sinclair ZX Spectrumille. Pieniä introja oli niille toki tehty aiemminkin, mutta ei samalla tavoin tietoisesti luokiteltuna.

Eräs merkittävimpiä käänteitä demojen historiassa on se, kun matalan tason laiteohjelmoinnista siirryttiin, lähinnä Windowsin myötä, rajapintoihin ja kirjastoihin perustuvaan järjestelmäystävälliseen ohjelmointiin 2000-luvun taitteessa: perinteinen laitteiston suoraan ohjaaminen ei yksinkertaisesti ollut enää mahdollista. Tämäkään muutos ei ollut kivuton, vaan se vaati useita vuosia ja mukautumista. Eräänä keskeisenä tekijänä voi nähdä edullisten 3D-kiihdyttimien yleistymisen, sillä niiden kiinnostavia ominaisuuksia ei pystynyt käyttämään kuin uusien käyttöjärjestelmien kautta. (Reunanen 2010, 92–96.) Muutkin käyttöjärjestelmän tarjoamat palvelut, kuten äänirajapinnat, muuttivat 4k-introjen luonnetta oleellisesti, sillä kaikkea ei enää kannattanut ohjelmoida itse, jos sopiva komponentti oli jo valmiina olemassa.

Vuonna 2013 4k-introt ovat edelleen voimissaan. Bent Stamnesin (2013) keräämien, *Pouet.netiin* perustuvien tilastojen perusteella uusia introja ilmestyy vuosittain noin sata. Määrä on säilynyt suunnilleen samana viimeiset kymmenen vuotta, minkä perusteella pikkuintrot eivät ole katoamassa ainakaan lähitulevaisuudessa. 4k-introjen audiovisuaalinen taso on noussut jo niin korkeaksi – ”4k is the new 64k” – että yhteisön mielenkiinto näyttää osittain siirtyneen seuraavaan, vielä mahdottomampaan kokoluokkaan eli 1k-introihin, joissa on edelleen uutta löydettävää. Esimerkiksi Assembly 2012 -tapahtumassa 1k-sarjassa oli huomattavasti enemmän osallistujia kuin 4k-introjen sarjassa (*Assembly 2012 results*, pouet.net). Löytämisen ilon lisäksi kyse lienee siitäkin, että 4k-introon vaadittava työmäärä on kasvanut riman ollessa entistä korkeammalla. Juurille paluuta edustaa niin ikään uusi kilpailusarja ”procedural graphics”, jossa on tarkoitus luoda mahdollisimman näyttävä staattinen kuva ohjemallisesti pienessä tilassa.

Pienet efektit

Neljään kilotavuun on käytännössä mahdotonta sisällyttää mediatiedostoja kuten videota, tallennettua ääntä, mallinnusohjelmalla tehtyjä 3D-kappaleita tai edes kuvia, joten tavalliset digitaalisen median työtavat eivät toimi tässä kokoluokassa. Pääasiallinen keino näyttävien efektien luomiseen on niiden ohjelmallinen *generointi*: luovasti käytetyillä matemaattisilla funktioilla, satunnaisluvuilla ja fraktaaleilla saadaan aikaan esineitä, kuvioita ja liikettä. Tällaisen tekotavan ytimessä on ohjelmoija, jonka luovuus ja osaaminen määrittävät lopputuloksen laadun.

Omia 4k-introjamme tehdessä nousi nopeasti esiin toinenkin käyttökelpoinen strategia, joka jalostui jo silloin, kun osittain samoja efektejä käytettiin musiikin visualisointiin konserteissa ja klubeilla muutamaa vuotta aikaisemmin. Tämä strategia oli *parametrisointi*. Musiikin visualisoinnissa parametrisointi tarkoittaa, että samaa materiaalia näytetään monta kertaa progressiivisesti vähän kerrassaan muunnellen, mikä mahdollistaa pidemmän esityksen luomisen samasta materiaalista. Vastaavasti 4k-introjen tapauksessa kallisarvoista dataa ja ohjelmakoodia voidaan käyttää useisiin eri tarkoituksiin, jolloin katsojalle syntyy harha siitä, että efektejä on useita. Yleisiä parametrisoinnin keinoja ovat esimerkiksi värien vaihtaminen, kuvioiden peilaaminen, 3D-kappaleiden muokkaus ja monistaminen kaavoja käyttäen, vaihtuvien kuvakulmien käyttö sekä luodun grafiikan suodatus eri tavoin. Visuaalista esitystä ei siis rakenneta kiinteäksi (”kovakoodata”), vaan sen ominaisuudet jätetään joustaviksi ja niitä muokataan ohjelman ajon aikana.

Grafiikan ohjelmallinen ja matemaattinen generointi tuottaa omanlaistaan, tunnistettavaa estetiikkaa, jonka kenties keskeisin piirre on abstraktisuus: orgaanisia hahmoja, kuten ihmismalleja on vaikeaa tuottaa uskottavasti. Niinpä 4k-introt eivät usein edes pyri esittämään mitään todellisen maailman ilmiöitä, vaan kuvaruudulla nähtävät muodot ovat rehellisen abstrakteja. Esimerkiksi voidaan nostaa lukuisat 3D-maastot, jotka voivat parhaimmillaan olla hyvinkin realistisen näköisiä. Muita hyvin generoitaviksi sopivia tosimaailman esineitä ja ilmiöitä ovat esimerkiksi säännölliset kasvit (Prusinkiewicz ja Lindenmayer 1990), lainehtiva vesi, pilvet, rakennukset ja mekaaniset koneet, joita kaikkia onkin usein nähty pikkuintroissa ja proseduraalisissa kuvissa.

Tietokonegrafiikan historiasta löytyy lukuisia esimerkkejä siitä, kuinka vastaavilla menetelmillä on pyritty luomaan uskottavia synteettisiä kuvia etenkin 1980-luvulta lähtien (esim. Goodman 1987, 102–164; Foley et al. 1996, värikuvat 12–41). Karl Simsin animaatiot kuten *Particle Dreams* (1988) ja *Panspermia* (1990) perustuvat ohjelmallisesti luotuun grafiikkaan, eikä niiden vähintään epäsuoraa vaikutusta demoihin ole vaikea huomata. Algoritmista taidetta edustavat niin ikään Laurent

Mignonneaun ja Christa Sommererin työt, joissa on tuotu yhteen taidetta ja teknologiaa jo 1990-luvun alusta lähtien. Mediataiteessa usein nähty roolijako, jossa ohjelmoijan tehtävä on lähinnä mahdollistaa taiteilijan luova toiminta, on vieras niin Mignonneaulle ja Sommererille kuin demoskenellekin: ohjelmakoodin merkitystä teoksissa ei pyritä piilottelemaan, vaan se nähdään pikemminkin kokeellisena ja luovana työkaluna (ks. Mignonneau ja Sommerer 2006) .

Pouet.netin tällä hetkellä suosituin 4k-intro, TBC:n ja rgba:n vuonna 2009 julkaisema *Elevated*, sopii monessa suhteessa esimerkiksi grafiikan generoinnista ja parametrisoinnista. Virtuaalista kameraa lennätetään pitkin uskottavaa lumista vuoristomaisemaa, jota näytetään monista eri kuvakulmista (kuva 3). Laaksoissa on vettä, taivaalla pilvimassaa, ilmassa sumua ja aurinko heijastuu pinnoista – tyypillisiä tietokonegrafiikan keinoja kohentaa realismia, mutta tässä tapauksessa minimaaliseen tavumäärään toteutettuna. Taustalla soi musiikki, jonka kaiku ja tuuli tukevat omalta osaltaan tilavaikutelmaa. Iñigo ”iq” Quilez, toinen *Elevatedin* ohjelmoijista, valotti introssa käytettyä tekniikkaa ja sen tekoprosessia Function 2009 -tapahtumassa pitämässään esitelmässä (Quilez 2009).

Kuva 3. TBC:n ja rgba:n Elevated (2009)

Laitteiden ja ohjelmistojen kehitys on väistämättä vaikuttanut siihen, kuinka paljon ja millaista sisältöä 4k-introihin saa mahdutettua. Perinteisten MS-DOS- ja Amiga-introjen täytyy käytännössä sisältää kaikki efekteistä musiikkiin yhdessä tiedostossa, kun taas modernien PC-käyttöjärjestelmien mukana seuraa lukuisia tekemistä helpottavia komponentteja kuten ohjelmakirjastoja, fontteja ja pakkausohjelmia. Esimerkiksi 3D-grafiikan osalta ero on dramaattinen: siinä missä ennen vaadittiin työlästä omaa ohjelmointia, nyt saman tai paremman toiminnallisuuden saa käyttöönsä järjestelmän mukana tulevan valmiin ohjelmointirajapinnan kautta. Ulkoisten komponenttien käyttö on muokannut 4k-introjen luonnetta ja johtanut toisinaan kitkeräänkin rajanvetoon, kun vastakkain ovat olleet demoskenelle tyypillinen itse tekemisen etiikka ja genren rajojen venyttäminen (*Some thoughts on 4k competition rules*, pouet.net).

Pieni ääni

4k-introt olivat lähes koko 1990-luvun ajan äänettömiä ja siinä suhteessa muihin demoihin verrattuna varsin askeettisia; onhan juuri kuvan ja äänen vuorovaikutus monien teosten kantava voima. Vielä Assembly'98:n kilpailusäännöissä (*Assembly'98 Official Invitation Text*, ftp.scene.org) perusteltiin ristiriitaista tilannetta seuraavasti:

> NO MUSIC or other sound is allowed (this is because this a coders' competition, not musicians')

Keinotekoiselta vaikuttavan rajanvedon taustalla oli ilmeisesti tarkoitus tarjota sekä ohjelmoijille, muusikoille että graafikoille kaikille yksilölaji – täysikokoisia demojahan tehdään yleensä ryhmätyönä. 2000-luvun taitteessa musiikista alkoi lopulta tulla keskeinen osa 4k-introja, mikä asetti jo ennestään haastavalle kategorialle lisävaatimuksia: efektien lisäksi samaan tilaan piti nyt saada mahtumaan sekä ääntä toistava koodi että itse sävellys.

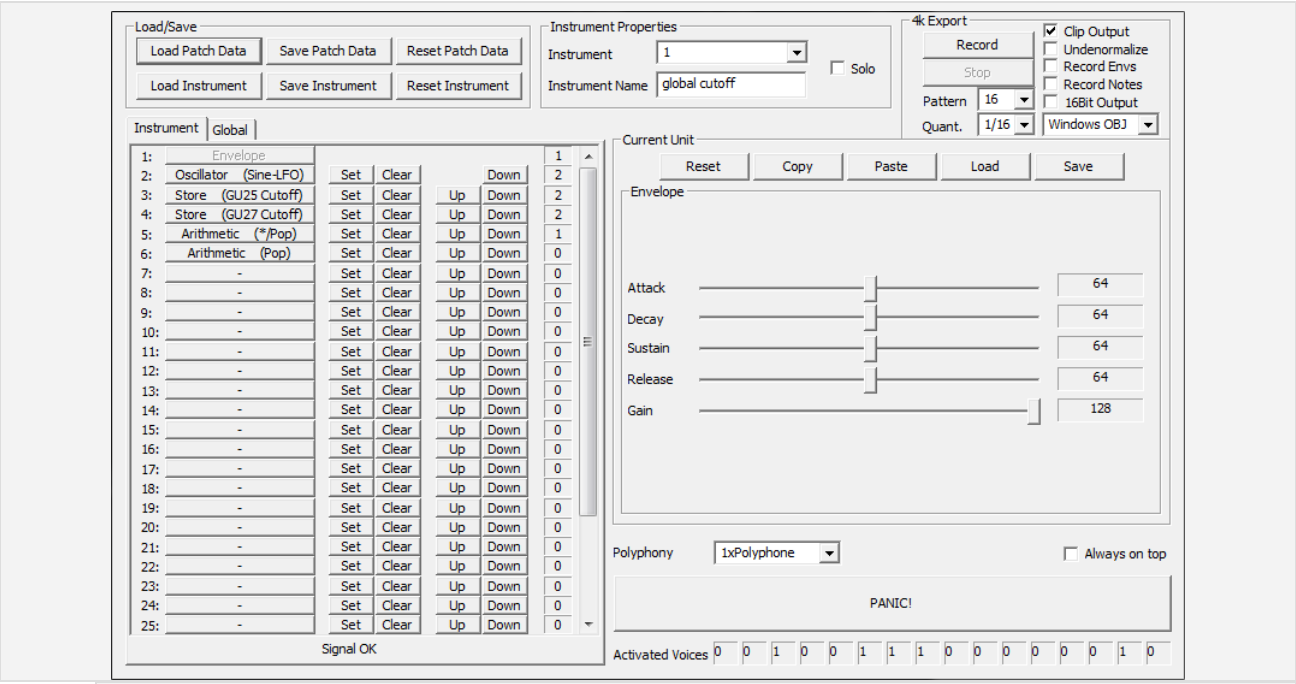
Musiikin toteuttaminen pieneen tilaan riippuu pitkälti käytettävästä ohjelmisto- sekä laitealustasta. 1980-luvun alun 8-bittisissä laitteissa on yleensä muutaman eri aaltomuodon ja kanavan sisältävä äänipiiri, jonka puitteissa musiikkia tehdään. Commodore Amigan ja PC-laitteiden tapauksessa ääni puolestaan muodostuu digitaalisista näytteistä, jolloin aaltomuodot on tuotettava tavalla tai toisella. Äänipiirien ominaisuuksia säveltämisen näkökulmasta on käsitellyt tarkemmin Anders Carlsson (2010) pro gradu -tutkielmassaan. Keskityn tässä etenkin äänen omalla koodilla syntetisointiin, vaikka se ei ainoa lähestymistapa olekaan: osassa 4k-introista on käytetty käyttöjärjestelmän tarjoamia puhesynteesi- ja MIDI-palveluja, joiden avulla tuotettu ääni on tilatehokasta, mutta samalla tunnistettavaa ja väritöntä.

Myös oma lähestymistapani vuonna 2003 oli äänen syntetisointi. Assemblerilla kirjoitettu, *Syna*-nimellä kulkenut minimalistinen syntetisaattori mahtui musiikkikappaleen kanssa 1,5 kilotavuun, mikä oli juuri ja juuri hyväksyttävää, sillä mukaan piti toki mahtua paljon muutakin. Lähtökohtaisesti *Syna* mahdollistaa neljän tyypillisen aaltomuodon (kantti-, saha- ja siniaalto sekä kohina) soittamisen eri nuottien korkeudelta, mikä on riittävä karun piippausmusiikin tekemiseen. Väriä luotuihin instrumentteihin saa lisättyä oikeiden soittimien käyttäytymistä jäljittelevillä verhokäyrillä sekä ääntä pehmentävällä alipäästösuotimella (vrt. Tolonen et al. 1998). Näillä keinoin luotu musiikki kuulostaa edelleen jokseenkin lattealta, minkä vuoksi lopputulos syötetään vielä delay-tyyppisen kaikuefektin läpi tilavaikutelman luomiseksi.

Muusikon osalta *Synan* käyttäminen vaatii teknisen ymmärryksen lisäksi viitseliäisyyttä ja suunnitelmallisuutta, sillä säveltäminen tapahtuu kirjoittamalla nuotteja tekstitiedostoon. Rajalliset ominaisuudet pakottavat samalla kekseliäisyyteen ja rajoitusten kiertämiseen, kuten *Synaa* käyttäneet muusikot myös osaltaan demonstroivat. Esimerkiksi särön lisääminen äänenvoimakkuutta kasvattamalla tai samojen melodioiden kierrättäminen eri instrumenteilla ovat ominaisuuksia, joita en toteuttanut tietoisesti, vaan jotka syntyivät vasta todellisen käytön ja sen tarpeiden myötä.

Tällä hetkellä 4k-musiikin yleisimmin käytetty työkalu, Alcatraz-ryhmän kehittämä *4klang*, sisältää vaikkapa *Synaan* verrattuna lukuisia teknisiä parannuksia kuten vapaammin muokattavat ja suodatettavat aaltomuodot. Eräs kantava ajatus on se, että yksinkertaisia rakennuspalikoita, oskillaattoreita ja suotimia joustavasti yhdistelemällä muusikko voi luoda monimutkaisia instrumentteja, mikä on periaatteena monissa muissakin ns. modulaarisissa syntetisaattoreissa. Kiinteän, mahdollisesti

tarpeettomiakin ominaisuuksia sisältävän lopputuloksen sijasta *4klang* tuottaa optimoidun, vain ja ainoastaan kappaleen soittamiseen vaadittavan ohjelmakoodin, jota ohjelmoija voi käyttää suoraan omissa introissaan. Muusikkojen työtä on helpotettu tekemällä *4klangista* VSTi-tyyppinen laajennus (kuva 4), jota voi käyttää osana tavallisia musiikkiohjelmia. (Zine #14, 2010)



Kuva 4. 4klangin graafinen käyttöliittymä

Vaiheittain parantuneet työkalut ja noussut vaatimustaso ovat johtaneet siihen, että parhaiden 4k-introjen äänet kuulostavat – kuten niiden on tarkoituskin – yhden tai kahden kilotavun kokoonsa nähden varsin massiivisilta. Efektit ja musiikki myös synkronoidaan tarkasti toisiinsa yhdeksi kokonaisuudeksi, joka etenee vaihe vaiheelta hallitusti. Äänen generointi sekä käytössä oleva niukka tila rajaavat tyylilajien kirjon lähinnä toisteisiin elektronisen musiikin genreihin, jotka ovat muutenkin suosittuja demoissa. Toisinaan musiikin tilalla saattaa olla puhtaasti ambientti äänimaisema, jonka tarkoituksena on vahvistaa visuaalisin keinoin luotua tilavaikutelmaa.

Tilaoptimoinnin työkalut

Audiovisuaalisen annin lisäksi 4k-introilla on puhtaasti tekninen ulottuvuutensa, joka on välttämätöntä tuntea pienessä tilassa toimittaessa. Työkalujen kuten kääntäjien ja pakkausohjelmien oikein käyttäminen säästää kallisarvoisia tavuja varsinaiselle sisällölle ja vähentää monimutkaista ohjelmointityötä. Työkalut ja lähestymistavat ovat kohentuneet vaiheittain; eri ohjelmoijien kehittämät parannukset ovat seuranneet toinen toistaan, ja tällä hetkellä pikkuintrojen tekemisen prosessi on kehittynyt jo varsin virtaviivaiseksi omine tarkoitukseen soveltuvine työtapoineen sekä apuohjelmineen. Aloittelijan tietä tasoittavat valmiit esimerkkiohjelmat, joiden pohjalle voi ryhtyä rakentamaan omia kokeiluja.

1980-luvun alun tietokoneiden ohjelmatiedostot olivat yksinkertaisia: esimerkiksi MS-DOS:n COM-tyyppiset ohjelmat, jotka ovat edelleen suosittuja kaikkein pienimmissä kategorioissa, sisältävät vain ajettavan ohjelmakoodin ja datan ilman yhtäkään ylimääräistä tavua. Primitiivinen COM-formaatti

periytyy jo 1970-luvulta, jolloin se oli käytössä CP/M-käyttöjärjestelmässä (ks. Digital Research 1983). Uudempien käyttöjärjestelmien ohjelmat ovat puolestaan huomattavasti mutkikkaampia ja niissä on erilaisia otsikkotietoja (headers), jotka ovat kaikki turhaa painolastia varsinaisen sisällön suhteen. Eräs tavujen *pummaamisen* keino onkin ollut kaiken ylimääräisen karsiminen ohjelmatiedostoista. Brian Raiterin *A Whirlwind Tutorial on Creating Really Teensy ELF Executables for Linux* (http://www.muppetlabs.com/~breadbox/software/tiny/teensy.html) on havainnollinen esimerkki paitsi tilasäästöstä, myös siitä viitseliäisyydestä, jolla harrastajat tekemiseensä suhtautuvat.

Perehdyin järjestelmä- ja kooditason optimointiin vuonna 2003, jolloin Linuxin 4k-introt olivat vasta lapsenkengissään ja Windowsillakin vielä pääosin vaatimattomia. Tuossa vaiheessa elettiin murrosaikaa, jolloin introja oli vasta alettu tehdä moderneille PC-käyttöjärjestelmille perinteisen MS-DOS:n sijasta. Myös *Yellow Rose of Texasin* ohjelmointiin käytetystä ajasta kului suuri osa puhtaaseen insinöörityöhön kuten sopivien kääntäjäparametrien kokeiluun, ulkoisten kirjastojen käytön optimointiin ja koodille sopivan tiivistystavan etsimiseen. Seuraavien introjen kohdalla työnkulku oli jo sujuvampi ja pystyimme käyttämään pääosan ajasta itse sisällön luomiseen.

Kenties mystisin ja vaikeimmin kontrolloitava tekijä tavujen pummaamisessa on tiivistetyn koodin kanssa toimiminen. 4k- ja 1k-introt toimivat tyypillisesti niin, että käynnistettävän tiedoston alkuun sijoitettu pieni koodinpätkä purkaa tiedoston lopussa olevan varsinaisen koodin ja sitten käynnistää sen. Alkuperäinen tiivistämätön ohjelma voi siten olla huomattavasti suurempi kuin neljä kilotavua – esimerkiksi *Yellow Rosen* tapauksessa 7632 tavua. Jo varhaisissa MS-DOS:lle tehdyissä introissa oli käytössä näin toimiva *PKLITE*, jonka tunniste löytyy helposti ohjelmatiedostojen alkua tarkastelemalla. Matemaattisen luonteensa vuoksi kompression lopputulos on kuitenkin vaikeasti ennustettava, ja pienetkin muutokset vaikuttavat kokoon lähes satunnaisesti: yksittäisen numeron muuttaminen tai jopa luontevalta vaikuttava koodin poistaminen saattavat kasvattaa lopullisen tiedoston kokoa, millä on merkitystä siinä vaiheessa, kun tavoitteesta uupuu vain muutamia tavuja.

IN4K-sivustolle (http://in4k.northerndragons.ca/) on kerätty vinkkejä 4k-introjen tekemiseen soveltuvista työkaluista ja lähestymistavoista. Syvällistä aiheeseen paneutumista edustaa Rune L. H. ”Mentor” Stubben ja Aske Simon ”Blueberry” Christensenin *Crinkler*, jota on kehitetty Windowsille vuodesta 2005 alkaen erityisesti pienten introjen tarpeisiin. Pelkän tiivistämisen lisäksi *Crinkler* tekee introille lukuisia muitakin matalan tason optimointeja, kuten lataa niiden käyttämät kirjastot tilatehokkaasti. (*The Crinkler executable file compressor*, http://www.crinkler.net/.) Ohjelman versiohistoria paljastaa myös sen, kuinka järjestelmäriippuvaista äärimmilleen viety tilaoptimointi on: uusien käyttöjärjestelmien ja päivitysten myötä aiemmat mekanismit voivat lakata toimimasta, mikä näkyy käyttäjille vanhojen teosten toimimattomuutena.

Lopuksi

4k-introjen tutkiminen nosti esiin ilmiöitä, joille löytyy luontevasti vertailukohtia myös demoskenen ulkopuolelta. Introt ovat esimerkki siitä, kuinka alkujaan teknisistä rajoituksista on muotoutunut ajan myötä käytäntö ja perinne, jolla on merkitystä enää lähinnä yhteisölle itselleen. Nykyisillä laitteilla ei ole mitään syytä rajoittaa ohjelmien kokoa neljään kilotavuun, vaan niin tehdään yksinomaan yhteisön luomien ja ylläpitämien sääntöjen vuoksi. Demokulttuuria leimaa vahvasti teknisen osaamisen sekä luovuuden arvostus, ja näyttävän teoksen pieneen tilaan ahtaminen vaatii niistä kumpaakin.

Demoskene tuottaa omiin tarpeisiinsa tee-se-itse-hengessä työkaluja, joista mainittiin esimerkkeinä *Crinkler* ja *4klang*. Vaivalla ohjelmoidut apuohjelmat annetaan usein vapaaseen

levitykseen, jolloin ne hyödyttävät muitakin yhteisön jäseniä. Työkalujen tekeminen on samalla tilaisuus osoittaa osaamistaan, ja niiden kehitys on vienyt genreä eteenpäin mahdollistamalla aikaisempaa laajemman ja monimutkaisemman sisällön mahduttamisen samaan tilaan. Toinen yhteisöllisen jakamisen muoto ovat valmiit esimerkkiohjelmat, jotka helpottavat alkuun pääsyä. Kaiken kaikkiaan kehitys on kulkenut kohti triviaalien sekä ihmisen hankalasti hallittavien työvaiheiden automatisointia, jolloin ohjelmoija on voinut keskittyä oleelliseen, eli sisältöä koskevien ongelmien ratkomiseen.

4k-introjen yli kahdenkymmenen vuoden historia heijastelee laitteiden, ohjelmien ja yhteisön käytäntöjen kehittymistä samalla ajanjaksolla. Historiallisesti tarkasteltuna uudet ohjelmisto- ja laitealustat on omaksuttu verrattain hitaasti, eikä suinkaan kritiikittä, uusimpia villityksiä seuraten. Etenkin ikonisiksi nousseista vanhoista koneista on haluttu ”ottaa kaikki irti” ja toisaalta ei ole haluttu menettää niillä kertynyttä kulttuurista ja sosiaalista pääomaa – osaamista ja tuttua yhteisöä. Suuret demotapahtumat, etenkin suomalainen Assembly, ovat olleet merkittävässä roolissa käytäntöjen aktiivisina muokkaajina sääntöjensä kautta: mitkä laitteet ja käyttöjärjestelmät ovat kulloinkin sallittuja, onko mukana saanut olla musiikkia, ja miten 4k-intro ylipäätään määritellään.

Laajemmassa tarkastelussa pienten introjen tekemisen voi rinnastaa muihin taidemuotoihin kuten miniatyyrimaalaukseen, haiku-runoihin, limerikkeihin tai pullolaivoihin. Ensi tarkastelulla satunnaisilta vaikuttavat säännöt vaativat niissäkin vastaavaa ongelmanratkaisua ja pakottavat keskittymään olennaiseen – taituruus ja luovuus voivat yhtä hyvin kummuta tiukoista rajoituksista kuin täydestä ilmaisukeinojen vapaudesta.

Kiitokset

Kiitokset Koneen säätiölle *Kotitietokoneiden aika ja teknologisen harrastuskulttuurin perintö* -tutkimusprojektin rahoittamisesta ja Yrjö Fagerille, Anna Haveriselle, Petri Isomäelle sekä Antti Silvastille hyvistä kommenteista.

Lähteet

Kaikki linkit tarkistettu 22.4.2013.

Lehdet

SCEEN #2, 2007

Zine #14, 2012, levykelehti

Verkkomateriaali

Assembly 94 results, http://www.pouet.net/results.php?which=7&when=94.

Assembly'98 Official Invitation Text, 20.7.1998, ftp://ftp.scene.org/pub/parties/1998/assembly98/info/asm98inv.txt.

Assembly 2012 results, http://www.pouet.net/results.php?which=7&when=12.

Main Page – IN4K, http://in4k.northerndragons.ca/.

Pouet.net :: your online demoscene resource, http://www.pouet.net/.

Quilez, Iñigo (2000). *Behind Elevated*. http://www.iquilezles.org/www/material/function2009/function2009.pdf.

Raiter, Brian. *A Whirlwind Tutorial on Creating Really Teensy ELF Executables for Linux*, http://www.muppetlabs.com/~breadbox/software/tiny/teensy.html.

Some thoughts on 4k competition rules, http://www.pouet.net/topic.php?which=9093.

Stamnes, Bent (2013). *(NEW) State of the demoscene: 2012*, http://blog.subsquare.com/state-of-the-demoscene-1991-2012-new.

The Crinkler executable file compressor, http://www.crinkler.net/.

Kirjallisuus

Botz, Daniel (2011). *Kunst, Code und Maschine – Die Ästhetik der Computer-Demoszene*. Bielefeld: Transcript Verlag.

Carlsson, Anders (2010). *Power Users and Retro Puppets. A Critical Study of the Methods and Motivations in Chipmusic*. Pro gradu -tutkielma. Lund: Lundin yliopisto.

Digital Research (1983). *CP/M Operating System Manual*. 3. painos. Pacific Grove: Digital Research.

Foley, James D., Andries van Dam, Steven K. Feiner, John F. Hughes & Richard L. Phillips (1996/1990). *Introduction to Computer Graphics*. Reading (Mass.): Addison-Wesley.

Franke, Herbert W. (1971). *Computer Graphics – Computer Art*. Lontoo: Phaidon Press.

Goodman, Cynthia (1987). *Digital Visions. Computers and Art*. New York: Harry N. Abrams.

Manovich, Lev (2007). ”Abstraction and Complexity”. Teoksessa *Media Art Histories*. Toim. Oliver Grau. Cambridge (Mass.): MIT Press, 339–354.

Mignonneau, Laurent & Christa Sommerer (2006). ”From the Poesy of Programming to Research as Art Form”. Teoksessa *Aesthetic Computing*. Toim. Paul A. Fishwick. Cambridge (Mass.): MIT Press, 169–183.

Montfort, Nick & Ian Bogost (2009). *Racing the Beam: The Atari Video Computer System*. Cambridge (Mass.): MIT Press.

Polgar, Tamas (2005). *Freax. The Brief History of the Demoscene. Volume 1*. Winnenden: CSW Verlag.

Prusinkiewicz, Przemyslaw & Aristid Lindenmayer (1990). *The Algorithmic Beauty of Plants*. New York: Springer.

Reunanen, Markku & Antti Silvast (2009). ”Demoscene Platforms: A Case Study on the Adoption of Home Computers”. Teoksessa *History of Nordic Computing 2*. Toim. John Impagliazzo, Timo Järvi and Petri Paju. New York: Springer, 289–301.

Reunanen, Markku (2010). *Computer Demos – What Makes Them Tick?* Lisensiaatintutkimus. Espoo: Aalto-yliopiston teknillinen korkeakoulu, Mediatekniikan laitos.

Saarikoski, Petri (2011). ”Kasarisukupolven teknoanimaation perintö”. *Wider Screen* 1–2/2011, http://www.widerscreen.fi/2011-1-2/kasarisukupolven-teknoanimaation-perinto/.

Tolonen, Tero, Vesa Välimäki & Matti Karjalainen (1998). *Evaluation of Modern Sound Synthesis Methods*. Raportti 8. Espoo: Teknillinen korkeakoulu, Akustiikan ja signaalinkäsittelyn laboratorio.

Wasiak, Patryk (2012). ”Illegal Guys. A History of Digital Subcultures in Europe during the 1980s”. *Zeithistorische Forschungen/Studies in Contemporary History* 9/2012, http://www.zeithistorische-forschungen.de/site/40209282/default.aspx.