

Quadern de Programació creativa

Autoria: Joan Soler-Adillon, Anna Carreras Sales i Alfredo Calosci

L'encàrrec i la creació d'aquest recurs d'aprenentatge UOC han estat coordinats pel professor: Joan Soler-Adillon

PID_00286423

Primera edició: setembre 2022

1. Visualització de dades

1.1. Introducció: Més que mil paraules

1.1.1. De què parlem quan parlem de «bones imatges»

1.1.2. Dibuixar el que sabem

1.1.3. Antecedents històrics: esquemes antics, diagrames i notacions

1.1.4. La representació esquemàtica i el desenvolupament de les ciències

1.1.5. La visualització de dades en el disseny de la informació: des d'Isotype als Design Systems

1.2. La visualització de dades en el nostre entorn digital

1.2.1. Introducció

1.2.2. Processing i el seu entorn

1.2.3. Treballar amb textos

1.2.4. Píxel per píxel

1.2.5. *Data scraping* i *data parsing*

1.2.6. API i temps real

1.2.7. Diagrames interactius

1.2.8. Simuladors

1.2.9. Visions de síntesi de dades estructurades

1.2.10. Poètica i estètica de les dades

Bibliografia

2. Art generatiu

2.1. Introducció: Què és l'art generatiu?

2.2. Art digital

2.2.1. Introducció

2.2.2. Precedents de l'art digital

2.3. Espais artístics generatius

2.3.1 Literatura

2.3.2. Dansa

2.3.3. Gràfic: Pioners de l'art digital generatiu

2.4. Artistes generatius destacats

Bibliografia

3. Joc i videojoc

3.1. Introducció

3.2. Joc i videojoc: aspectes bàsics

3.2.1. Què és un joc?

3.2.2. Regles i mecàniques

3.2.3. Altres aspectes rellevants

3.3. Disseny iteratiu

3.4. Joc obert i joc tancat

3.5. Estudi de cas: art digital i videojoc

3.5.1. Introducció

3.5.2. *Hacking*

3.5.3. El joc com a forma artística

Bibliografia

4. Més Processing

4.1. Introducció

4.2. Vectors i forces

4.3. Objectes

4.4. ArrayLists

4.5. Coordenades polars i cartesianes

4.6 Translació i rotació

Bibliografia

1. Visualització de dades

1.1. Introducció: més que mil paraules

1.1.1. De què parlem quan parlem de «bones imatges»

Fins i tot en la cultura popular, la idea que una bona imatge pot valdre més que mil paraules està ben arrelada. La visualització de dades és part del que solem definir com a disseny de la informació, un àmbit de la comunicació visual que sembla establir una aparent polaritat entre les paraules (escrites, parlades, etc.) i les imatges, reconeixent a aquestes últimes una suposada major eficàcia en el pla de la comunicació.

Si el nostre propòsit és aprendre a crear aquestes bones imatges (i estalviar-nos, així, més de mil paraules), potser val la pena començar a preguntar-se per què, i en quins contextos, la comunicació visual es demostra tan eficaç. Podem reemplaçar la polaritat entre paraules i imatges amb una altra, de caràcter més ampli, entre llenguatges i la realitat; els llenguatges són la nostra principal eina d'abstracció conceptual, ens ajuden a observar, descriure, comparar i entendre els fenòmens que ens envolten, però a l'hora d'operar en el nostre entorn necessitem una altra classe –completament diferent– d'eines analògiques. Des d'aquesta perspectiva, podem entendre el disseny de la informació com una activitat que se situa a mig camí entre els dos extrems i que actua en totes dues direccions amb la finalitat d'ajudar-nos a abstreure's de la nostra experiència quotidiana i d'ofrir-nos indicacions tangibles per poder actuar sobre el nostre entorn immediat. Vegem-ne alguns exemples.

En algun moment de la nostra vida escolar és molt probable que hagim hagut d'aprendre aquesta fórmula:

$$(a+b)^2 = a^2 + b^2 + 2ab$$

la qual expressa –en notació matemàtica– el quadrat de la suma; observant la il·lustració que apareix a continuació, el seu significat pot resultar encara més clar i veiem perfectament els seus diferents elements (els dos quadrats a i b + els dos rectangles ab):

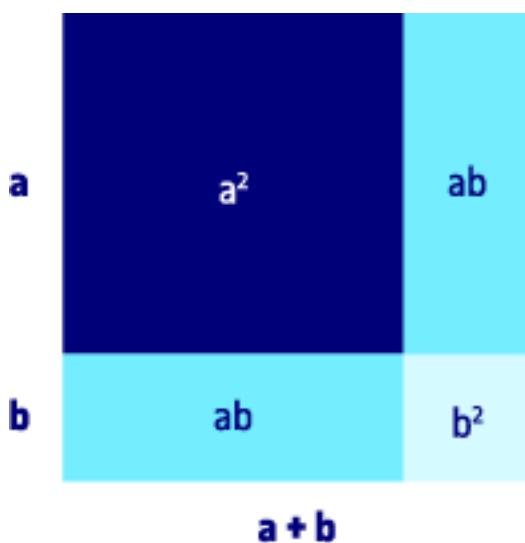


Figura 1. Quadrat de la suma

La fórmula, formalitzada com a notació, presenta un nivell d'abstracció més gran, ens permet operar-hi en termes matemàtics per inferir altres propietats, a més de poder-se aplicar per resoldre casos pràctics; la seva il·lustració expressa el mateix significat en un llenguatge més col·loquial i pròxim fins al punt que pot servir-nos com a recurs mnemònic.

Vegem un altre exemple, aquesta vegada sense l'auxili de cap imatge. S'estima que l'edat de la Terra pot rondar els 4.500 milions d'anys, una magnitud de temps que tots podem entendre en termes abstractes, però que s'escapa completament a la nostra experiència, tant és així que el que probablement recordem d'aquesta dada, quan ja no la tenim al davant, és que... són molts, molts, molts anys.

El físic i divulgador científic Fritjof Capra cita en un dels seus assajos (2009) un curiós recurs retòric ideat per l'ambientalista estatunidenc David Brower per tal que puguem entendre millor aquesta magnitud: en el seu relat, Brower fa coincidir l'edat de la terra amb els sis dies de la creació del món segons la tradició bíblica.

A l'escenari de Brower, la Terra es crea el diumenge a mitjanit. La vida en forma de primeres cèl·lules bacterianes apareix al voltant de les 8:00 del matí del dimarts. Durant els dos dies i mig següents, el microcosmos evoluciona fins a estar completament establert el dijous a mitjanit, tot regulant el sistema planetari per complet. El divendres cap a les 4:00 de la tarda, els

microorganismes inventen la reproducció sexual i el dissabte, l'últim dia de la creació, evolucionen totes les formes visibles de vida. Cap a la 1:30 de la matinada del dissabte es formen els primers animals marins, i cap a les 9:30, les primeres plantes surten a terra ferma, seguides, unes hores més tard, per amfibis i insectes. A les 4:50 de la tarda apareixen els grans rèptils, que vaguen durant cinc hores per selves tropicals exuberants i moren després sobtadament a les 9:45 de la nit. Mentrestant, pels volts de les 5:30 de la tarda, han arribat a la Terra els mamífers, i al capvespre, cap a les 7:15, els ocells. Poc després de les 10:00 de la nit, alguns dels mamífers habitants d'arbres en els tròpics evolucionen i donen pas als primers primats; una hora després, alguns d'aquests es converteixen en micos i, pels volts de les 11:40 de la nit, apareixen els grans simis. Vuit minuts abans de mitjanit, els primers simis del sud s'alcen i caminen sobre dues potes. Cinc minuts després, desapareixen de nou. La primera espècie humana, l'*Homo habilis*, apareix quatre minuts abans de la mitjanit, evoluciona fins a convertir-se en l'*Homo Erectus* mig minut després i en les formes arcaiques d'*Homo sapiens* trenta segons abans de la mitjanit. Els Neanderthals dominen Europa i Àsia des de quinze fins a quatre segons abans de la mitjanit. La moderna espècie humana apareix per fi a Àfrica i Àsia onze segons abans de la mitjanit, mentre que a Europa ho fa sis segons més tard. La història humana escrita comença uns dos terços de segon abans de la mitjanit.

La imatge mental evocada pel relat de Brower resulta tan eficaç com la millor infografia perquè aconsegueix traduir en una cosa tangible una informació abstracta, encara que això ocorri per mitjà d'un text breu i no a través d'una representació gràfica.

La creació d'aquestes imatges que valen més que mil paraules no pot, en la nostra opinió, abordar-se només servint-nos de les eines i dels conceptes de la comunicació visual. La visualització de dades no s'hauria d'entendre com un simple conjunt de mètodes per convertir números en alguna classe d'elements gràfics, sinó que requereix un esforç constant de codificació / descodificació que transcendeix l'àmbit del visual i ens obliga a considerar una multiplicitat de llenguatges. Llavors es pot entendre com una activitat de caràcter interdisciplinari en la qual intervenen semiòtica, retòrica, ontologia i ciències cognitives, a més d'estadística, arquitectura de dades i –ens n'ocuparem més detalladament– la nostra capacitat de traduir certes instruccions als llenguatges formals de la programació.

1. Visualització de dades

1.1. Introducció: més que mil paraules

1.1.2. Dibuixar el que sabem

Definir exactament què és una infografia i en què es diferencia de la resta d'imatges pot resultar menys fàcil del que podem imaginar, en el fons pràcticament totes les imatges porten una mica d'informació i estan fetes amb algun propòsit.

Vegem, per exemple, les imatges següents:



Figura 2. Són infografies?

La primera pertany a un herbari i representa una determinada espècie vegetal de manera realista. Una imatge d'aquest tipus, degudament emmarcada, podria utilitzar-se com a simple objecte de decoració, malgrat que el seu origen té una finalitat científica determinada. Les il·lustracions creades per als volums de botànica solen oferir imatges d'aquest tipus. Només els experts saben que no es tracta d'una reproducció realista qualsevol –com la que podríem obtenir amb la càmera dels nostres telèfons–, sinó d'una construcció meticulosa en la qual han d'aparèixer tots aquells elements que puguin tenir interès per un botànic: òrgans de reproducció, forma de les fulles, variacions al llarg de l'any, llavors, etc., representats sovint de manera simultània.

La imatge del centre pertany a un atles d'anatomia (concretament, a la *Historia de la composición del cuerpo humano* de Juan Valverde) i continua sent en bona part realista –el cos representat està amb els peus a terra–, però resulta més evident que la seva construcció també té un component artificial i que s'esforça per mostrar-nos certs detalls alhora que n'oculta molts d'altres deliberadament.

La tercera il·lustració ens mostra un mapa *orbis terrarum*, una representació utilitzada amb freqüència durant l'Edat Mitjana des de la seva primera aparició en les *Etimologies* d'Isidor de Sevilla. Es tracta d'un mapamundi que representa el món conegut a través d'una curiosa síntesi de coneixements empírics i teològics. La il·lustració esquemàtica es presta a diverses lectures: des del punt de vista estrictament geogràfic apareixen els tres continents envoltats de l'oceà i separats per la mar Mediterrània –el nord apareix girat 90 graus en sentit antihorari respecte les convencions actuals–. Al mateix temps, podem interpretar la il·lustració com un monograma on la lletra T (terra) apareix inscrita en una O (orbis: món) –una simplificació sens dubte excessiva, fins i tot pels coneixements empírics de l'època, que reflecteix d'alguna manera la necessitat de buscar algun tipus d'ordre de caràcter teològic en la representació d'allò conegut.

Les tres imatges pertanyen a diferents moments històrics de la il·lustració científica i comparteixen un mateix criteri: ens mostren el que sabem sobre determinats objectes més que els objectes en si.

Dibuixar el que sabem sovint ens obliga a buscar una manera de representar conceptes abstractes, la qual cosa no podríem veure de cap altra manera.

Vegem, per exemple, la imatge següent:



Figura 3. Comenius: *Anima Hominis*

Es tracta de la representació de l'ànima i forma part de l'*Orbis sensualium pictus*, la que avui podríem definir com una enclopèdia il·lustrada –dirigida a un públic infantil– publicada el 1658 per Comenius (Jan Amós Comenius), a qui molts consideren un dels precursors de la pedagogia. A falta d'altres recursos iconogràfics, la il·lustració ens presenta una cosa etèria que continua evocant les formes d'un ésser humà malgrat estar amagat darrere d'una tela.

La possibilitat de representar gràficament conceptes abstractes fa una funció essencial en la visualització de dades. Amb independència que decidim utilitzar una tipologia de representació més rigorosa i convencional o que explorem noves formes per representar magnituds d'una manera més lliure i poètica, el que volem representar no són les dades en si, sinó algun tipus d'abstracció que ens ajudi a interpretar el fenomen que estem observant. Les dades, tinguin el format i l'origen que tinguin, no formen part de la realitat, són sempre el resultat d'un procés d'observació d'un fenomen que ens interessa analitzar. A través de la visualització volem mostrar la seva evolució en el temps, la correlació entre variables, la seva estructura de relacions, etc. Alguna cosa que validi les nostres hipòtesis i la nostra narració o que ens suggereixi nous models d'interpretació.



Figura 4. Ferdio: dataViz project

A la web [Dataviz Project](#) podem observar com els seus autors, l'agència de comunicació danesa [Ferdio](#), a més de recopilar un ampli nombre de tipologies de representació de dades, han decidit classificar-les en funció de diferents criteris, com ara la seva funció, l'abstracció que s'evoca a través de cada classe de representació. A continuació intentarem esbrinar alguna cosa més sobre com –i en quin moment històric– han anat apareixent moltes de les diferents tipologies de representació que s'hi cataloguen.

Simplificant una mica les coses, intentarem recórrer aquesta evolució seguint tres grans línies de desenvolupament: l'evolució de la representació esquemàtica i dels sistemes de notació, els sistemes de representació en estadística i, més en general, en el context científic, i –finalment– la seva utilització i influència en el context més general de la comunicació visual i en les expressions més artístiques.



Figura 5. El bestiari d'Aberdeen

1. Visualització de dades

1.1. Introducció: més que mil paraules

1.1.3. Antecedents històrics: esquemes antics, diagrames i notacions

El principal recurs per expressar conceptes abstractes a través d'imatges ha estat tradicionalment el de recórrer a associacions metafòriques. Al llarg del temps, en tots els continents, les metàfores visuals han estat un dels mitjans més eficaços per arribar a una població majoritàriament analfabeta.

Entre la gran quantitat d'exemples possibles ens podem fixar en un bestiari medieval –com el d'Aberdeen, al qual pertanyen les imatges del punt anterior–, autèntics diccionaris iconogràfics que representaven criatures reals, exòtiques o imaginàries que, amb el temps, havien anat consolidant un significat convencional –com l'ocell fènix, capaç de renéixer de les seves cendres– i que en la cultura visual de l'època podien utilitzar-se per desenvolupar al·legories, l'extensió en el pla narratiu d'una metàfora.

La capacitat de fixar significats utilitzant la nostra memòria visual i espacial formava part de la retòrica clàssica que, a més d'ocupar-se de les diferents estratègies per millorar la capacitat de persuasió, incloïa tècniques de memorització les quals consistien, bàsicament, a associar els conceptes a una imatge –o a un lloc– convertint els discursos en alguna cosa semblant a una pel·lícula (o a un recorregut) que caldia recordar i, en el seu moment, tornar a descodificar.



Figura 6. Esquemes figuratius

Algunes imatges, més que pel seu significat metafòric, han estat utilitzades amb freqüència per la seva capacitat de suggerir –per si mateixes– una estructura de continguts o un índex (Bolzoni, 2004). És el cas del drac amb set caps –un per cada pecat capital, oportunament retolat al costat– d'un manuscrit de Joaquim de Fiore del segle XII, de les moltes referències a arbres per il·lustrar genealogies o branques del coneixement (Lima, 2014), com l'arbre de la ciència de Ramon Llull del segle XIII, o dels diversos esquemes cíclics i concèntrics, com l'*Integra naturae speculum artisque imago* de Robert Fludd –del segle XVI– que apareixen en la imatge anterior.

A més d'aquestes metàfores visuals, amb un fort component figuratiu, des de l'antiguitat s'han anat desenvolupant formes més abstractes i simbòliques de representació que solen utilitzar elements alfabètics i simples formes geomètriques. És el cas, per exemple, del *Liber Figurarum*, una espècie de teologia il·lustrada elaborada per l'abat cistercenc Joaquim de Fiore, al segle XI, al qual pertany la il·lustració dels cercles trinitaris que apareix a continuació en la qual molts reconeixeran una certa analogia amb els diagrames moderns de Venn o del cristograma JHS utilitzat com a suport mnemònic per Sant Bernardí de Siena durant les seves predicacions. Sense voler entrar en una descripció detallada d'aquests exemples, és important observar com, en tots dos casos, es tracta d'imatges que es presten a múltiples lectures i que intenten articular conceptes molt abstractes, en aquest cas de caràcter teològic, plasmant-los en una cosa tangible.

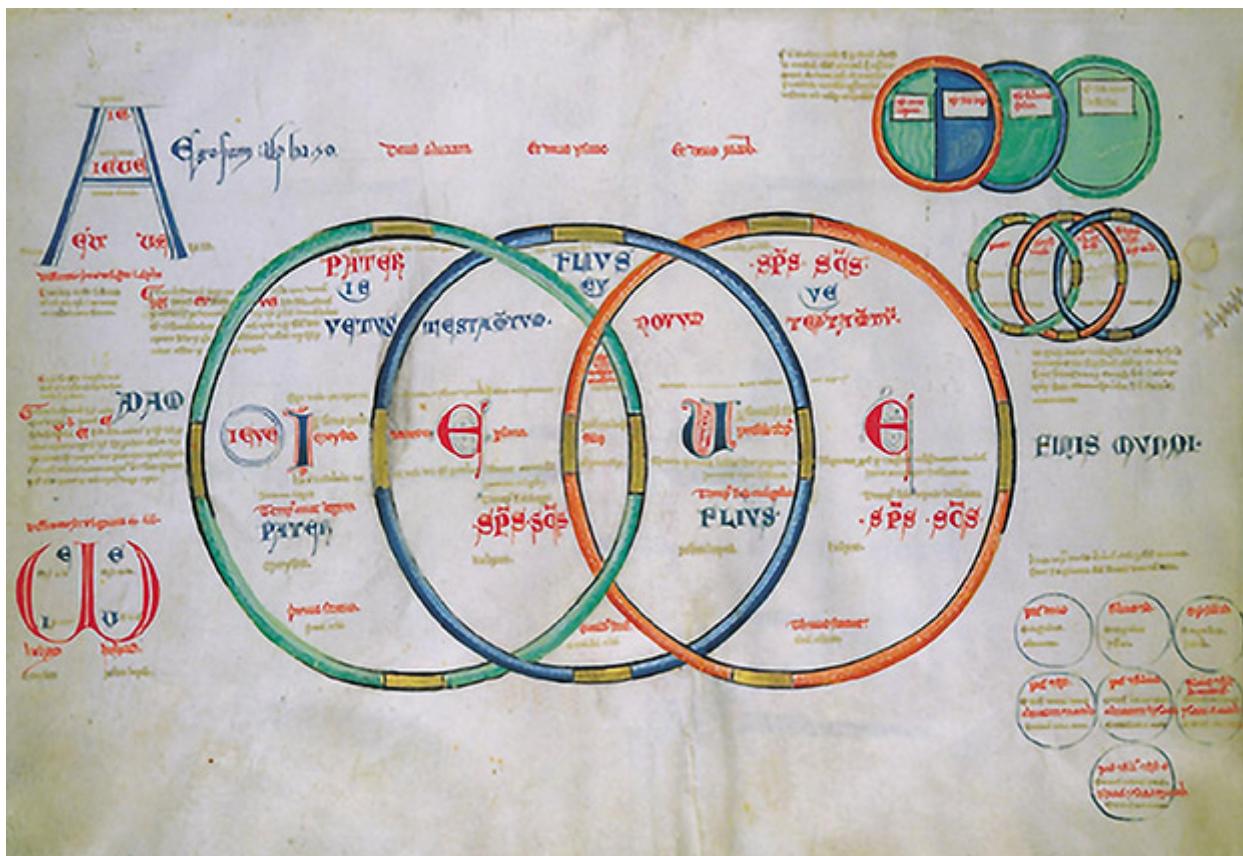


Figura 7. Símbols, esquemes i notacions (1)



Figura 8. Símbols, esquemes i notacions (2)

Ramon Llull va fins i tot més enllà i en el seu *Ars Magna Generalis* arriba a idear un esquema dinàmic, una espècie de màquina lògica, que a través de combinacions i permutacions basades en determinats atributs divins (identificats per lletres) es proposava facilitar l'enteniment i el diàleg interreligiós. Gràcies a Galileu, uns segles més tard, tenim una mostra de com aquest tipus de representació s'ha utilitzat també per expressar coneixements de naturalesa empírica. Al començament del procés que portarà al desenvolupament de la ciència moderna, Galileu elabora un sistema de notació gràfica per sintetitzar les seves observacions astronòmiques i l'utilitza tant en les seves anotacions personals com en les seves relacions epistolars i en els seus textos de divulgació.

1. Visualització de dades

1.1. Introducció: més que mil paraules

1.1.4. La representació esquemàtica i el desenvolupament de les ciències

A partir d'aquest moment, amb la consolidació del pensament científic, comencen a prendre forma les representacions més rigoroses com les que es continuen utilitzant actualment en l'àmbit de l'estadística.

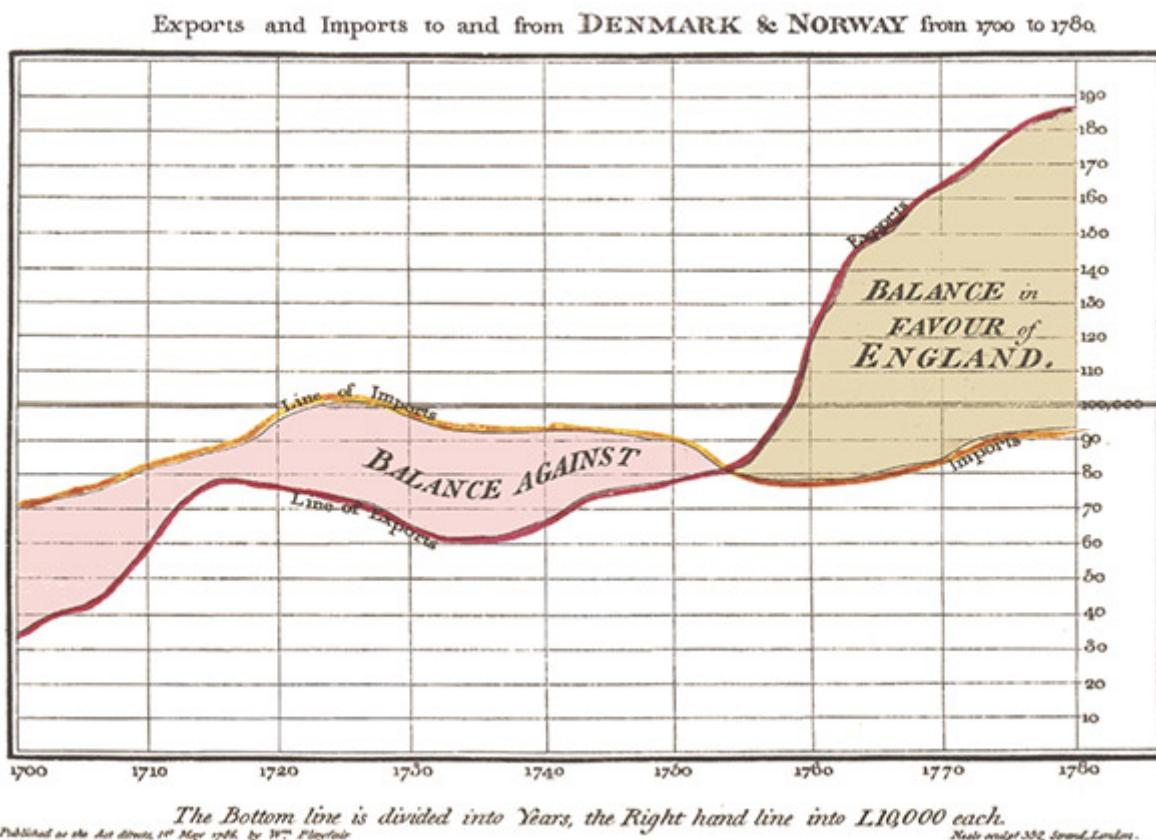


Figura 9. William Playfair: Atles comercial i polític

Entre els pioners cal destacar la tasca de l'escocès William Playfair, precursor en el seu *Atles comercial i polític*, de 1786, de molts dels gràfics, (de barres, de sectors, etc.) que des d'aleshores solen acompanyar les taules de dades. Gairebé al mateix temps, el britànic Joseph Priestley, en una de les seves moltes obres de divulgació (escriurà també sobre òptica i electricitat), utilitzava una sofisticada línia de temps, aquesta vegada en l'àmbit humanístic, per crear una singular taula sinòptica de les principals èpoques històriques i establint les bases per tot un gènere d'il·lustració que donarà lloc –entre altres– a la *Histomap* –publicada el 1931 per John B. Sparks– o a la més recent *Genealogy of Pop/Rock Music* del musicòleg Julio Garofalo, inclosa el 1977 a *Rock 'N' Roll is Here to Pay: The History and Politics of the Music Industry*.

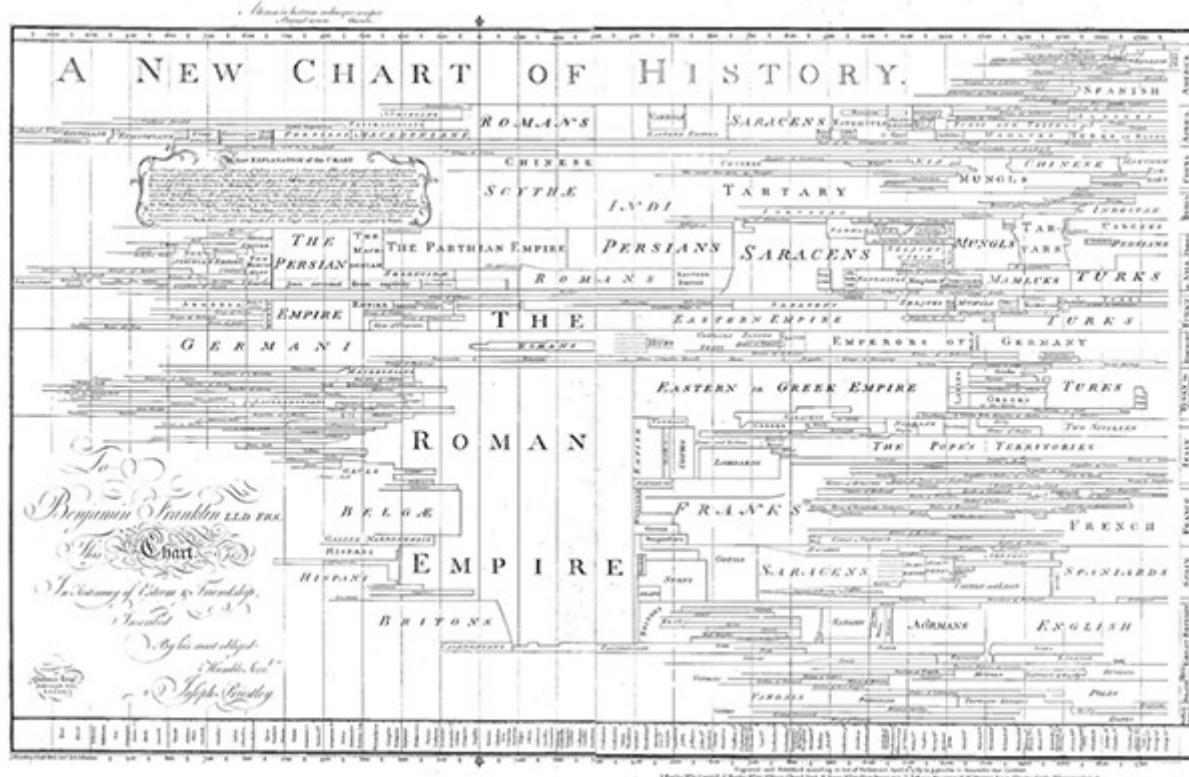


Figura 10. Joseph Priestley: *A new chart of History*

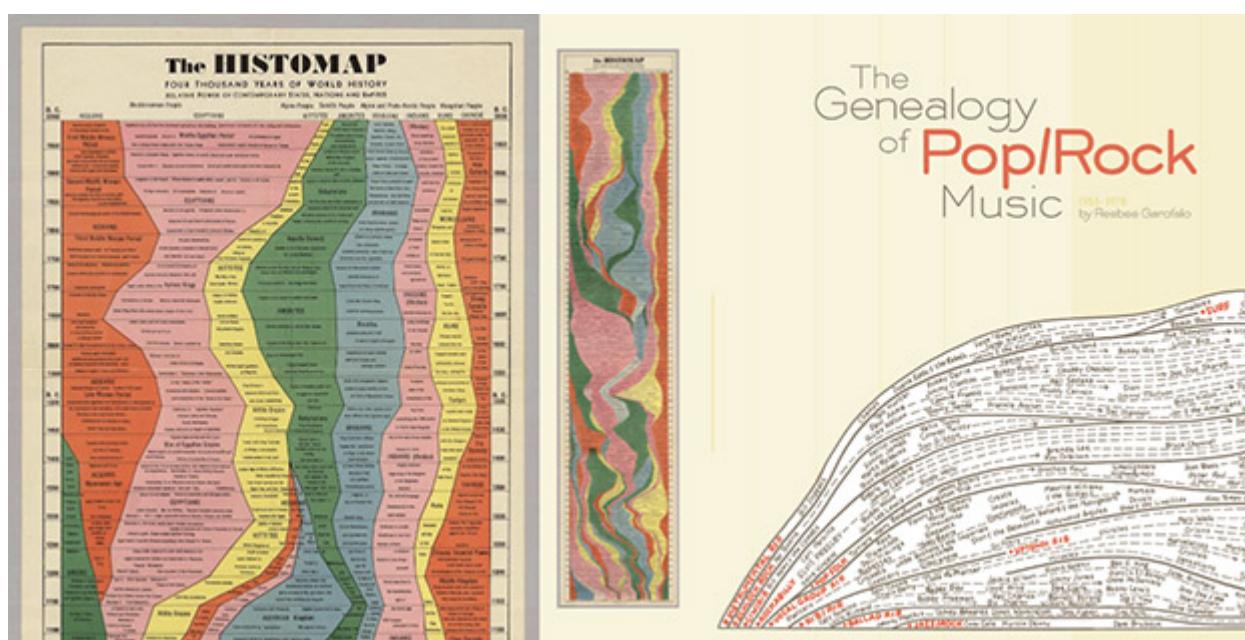


Figura 11. The Histomap, de Rand McNally (1931) i Genealogy of Pop/Rock Music, de Reebbee Garofalo (1978)

Ens trobem ja a les portes del que molts defineixen com la primera edat daurada de la visualització de dades (Rendgen, 2019). El 1854, durant una epidèmia de còlera que va afectar Anglaterra, John Snow –considerat avui com un dels pares de l'epidemiologia– fa redactar un mapa d'un barri de Londres en el qual es posa en evidència la densitat de morts a prop d'una font, fet que li va permetre argumentar el que en aquell temps era només una hipòtesi sobre les possibles vies de transmissió del bacil i així convèncer les autoritats que calia intervenir sobre la xarxa d'aigües potables.

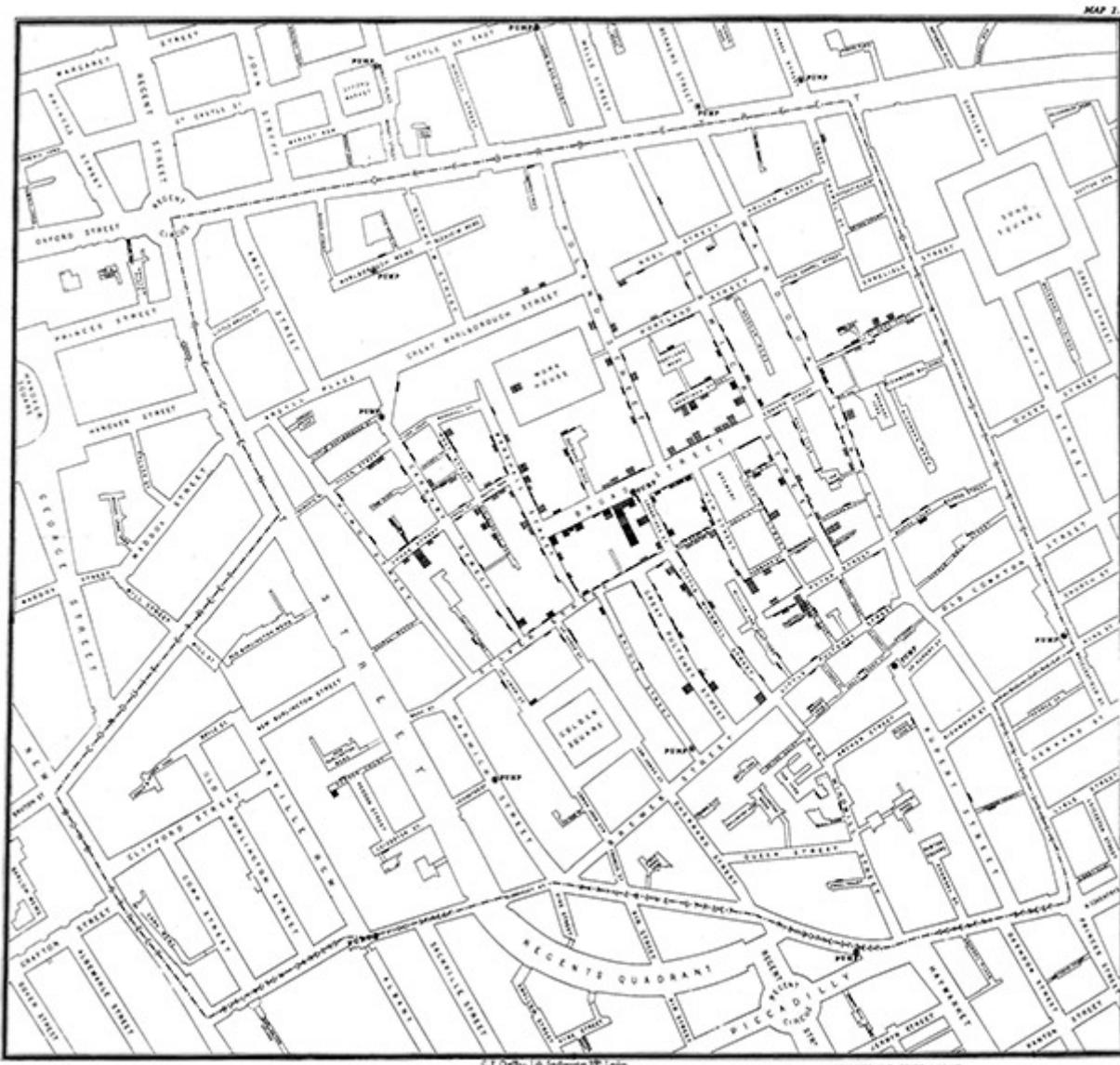


Figura 12. John Snow: Londres 1854

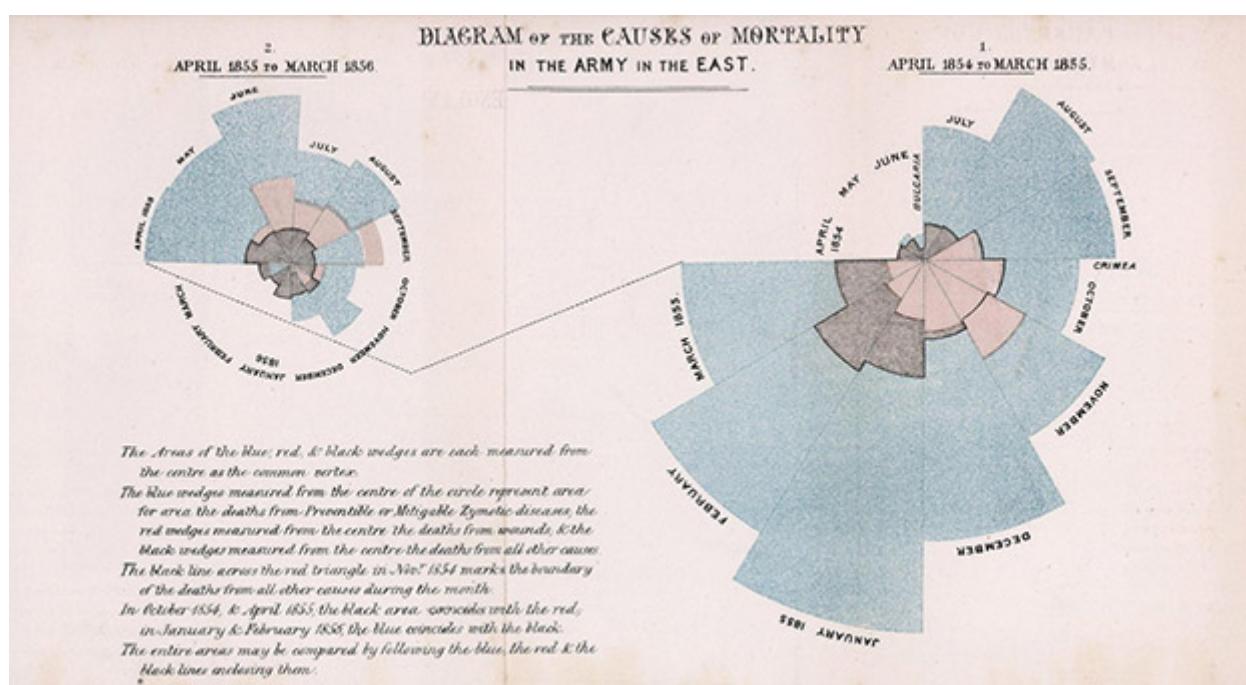


Figura 13. Florence Nightingale: causes de mortalitat... 1858

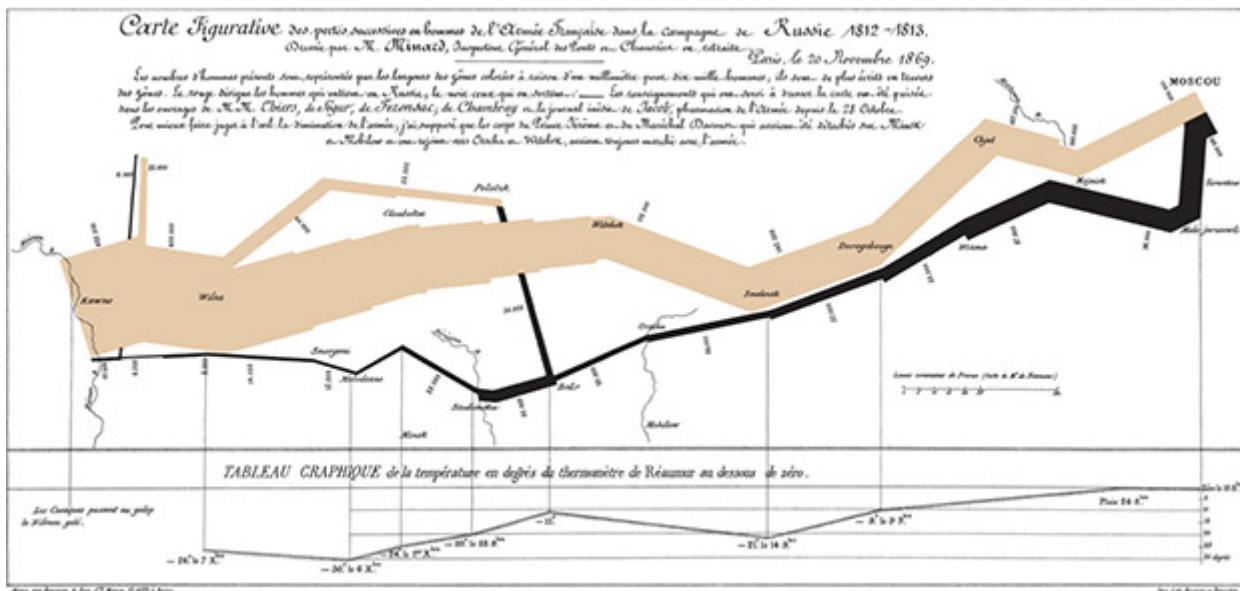


Figura 14. Charles Joseph Minard: Mapa figuratiu de la campanya de Napoleó a Rússia

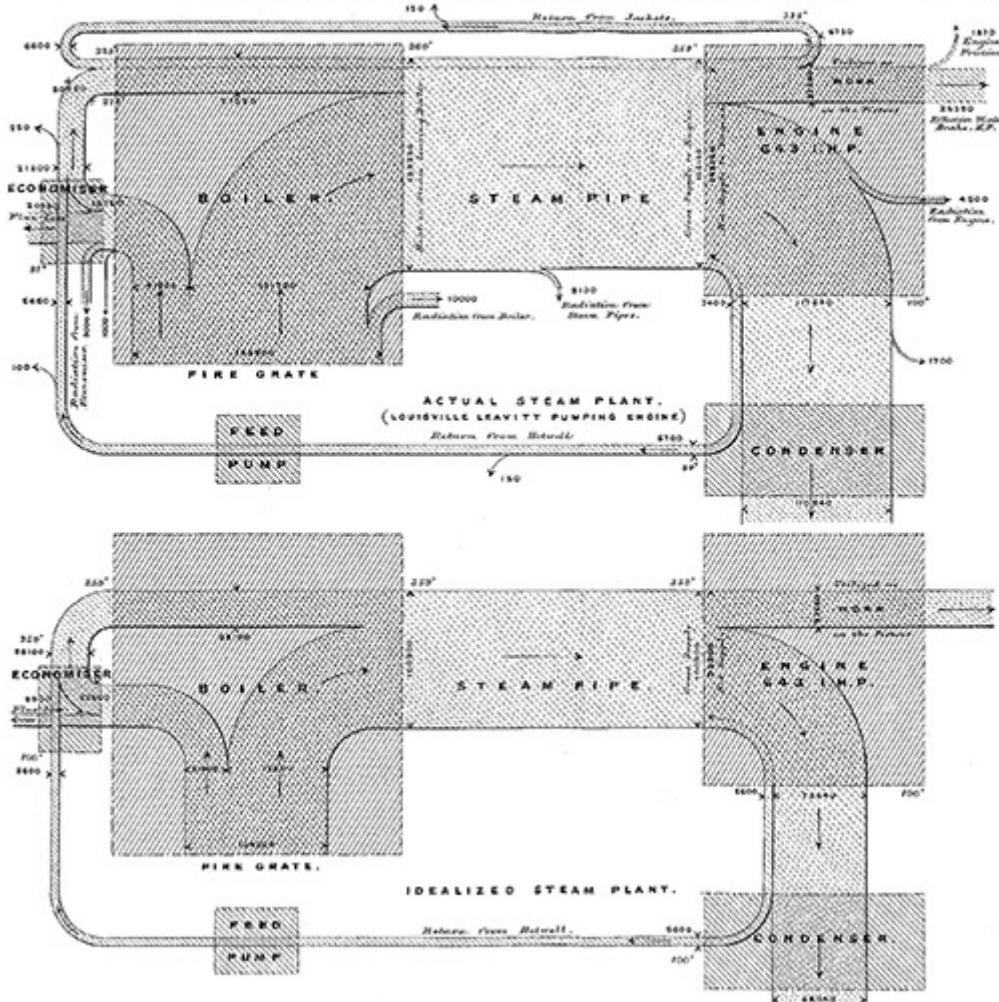
Gairebé al mateix període, el 1858, Florence Nightingale –una de les poques dones que apareixen en aquests relatius sobre pioners de la visualització– elabora el seu célebre diagrama sobre les causes de mortalitat a l'exèrcit desplaçat a les colònies britàniques en què s'evidencia l'elevat nombre de defuncions per causes no bèl·liques i que portarà a una profunda reorganització de la sanitat militar.

El 1869, l'enginyer francès Charles Joseph Minard publica el seu mapa figuratiu de la campanya napoleònica a Rússia, de principis d'aquest segle: una autèntica fita en la infografia, en la qual, a un mapa geogràfic convencional se sobreposa una línia de gruix variable en funció del nombre d'efectius amb els quals comptava l'armada francesa en els seus desplaçaments i un gràfic de temperatures, a la part inferior de la il·lustració, que explica el progressiu aprimament de la línia també en la fase de retirada –moment en què la línia canvia de color– després de la derrota en la batalla del riu Moskvà.

Es poden observar molts dels elements gràfics que tornarem a trobar ja al 1898 en els diagrames de fluxos ideats per Matthew H. Sankey per il·lustrar l'eficàcia energètica de les màquines de vapor i posar en evidència pèrdues i dispersió. L'aplicació d'aquesta mena de representació –que avui denominem amb el nom del seu autor dins de la categoria més general dels diagrames alluvials– s'ha popularitzat els últims anys gràcies a la seva automatització en plataformes en línia com ara RawGraph o Flourish Studio.

THE THERMAL EFFICIENCY OF STEAM-ENGINES.

PLATE 5.



Minutes of Proceedings of The Institution of Civil Engineers, Vol. CIX, Session 1897-98, Part IV.

Figura 15. Matthew H. Sankey: eficiència tèrmica de una màquina de vapor – 1898

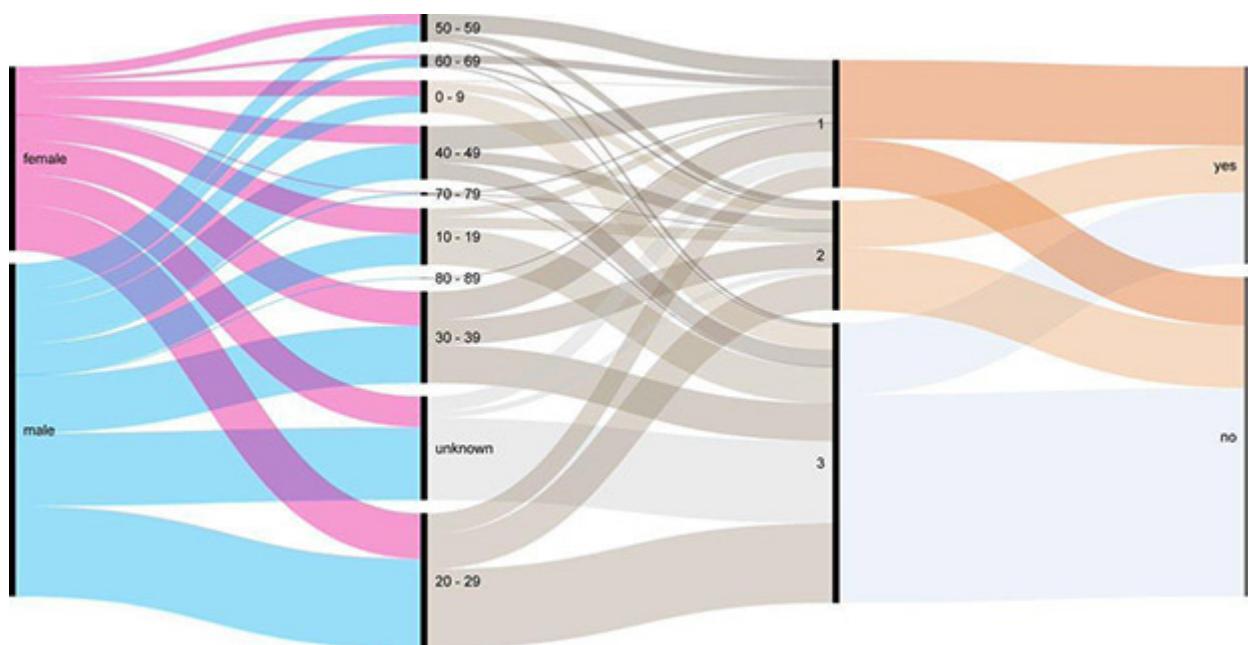


Figura 16. RawGraph: els passatgers del Titanic – exemple de diagrama alluvial

L'evolució de les tècniques fotogràfiques, des de la introducció dels daguerreotips el 1839, juntament amb altres evolucions tècniques, fan que al llarg de tot el segle XIX es multipliquin els aparells per mesurar i observar científicament la realitat. Es consolida la idea que existeix un mètode gràfic per conèixer el món tan rigorós com altres activitats experimentals. Étienne-Jules Marey, creador d'instruments per mesurar la pressió sanguínia o per transcriure fonemes i pioner, al costat d'Eadweard Muybridge, de la cronofotografia, descriu aquesta nova actitud en el seu assaig *La Méthode graphique dans les sciences expérimentales* (Marey, 1878), en què recopila nombrosos exemples, com el ja esmentat mapa figuratiu de Minard o l'horari de trens de 1847, atribuït a Charles Ibry, que apareix a continuació.



Fig. 6. Matin d'espion du lac photographique.

Figura 17. Étienne-Jules Marey: pistola cronofotogràfica – 1882

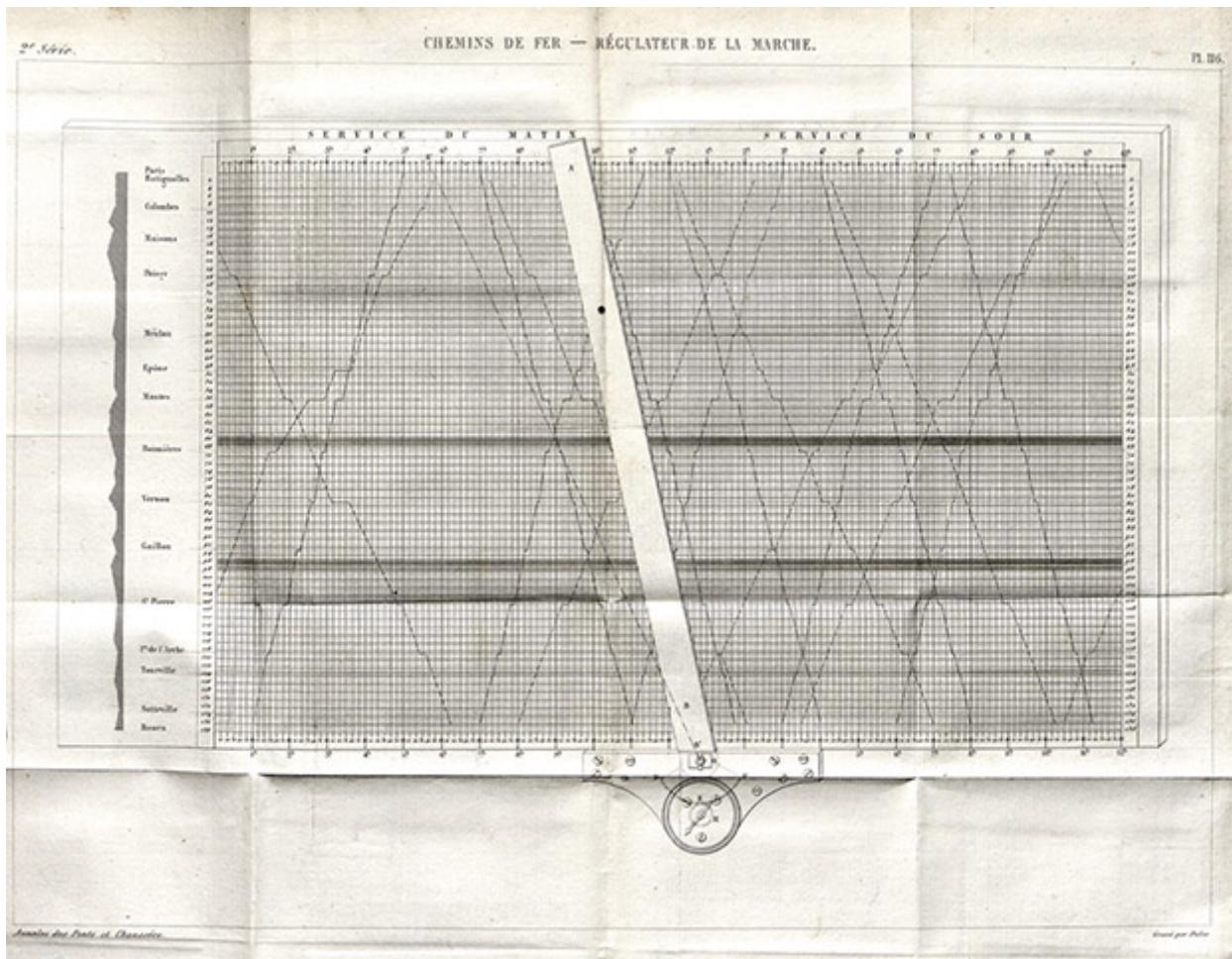


Figura 18. Xerris Ibry: Horaris de la línia ferroviària París – Rouen

Les xarxes ferroviàries eren una altra de les innovacions de l'època i les modalitats amb les quals s'oferia el servei de transport estaven encara en fase de definició. Durant diversos anys, les línies del nord de França utilitzaven per als seus horaris aquest original diagrama en el qual cada tren es representa amb una línia inclinada que connecta les diferents estacions –en l'eix vertical– al llarg de les hores de dia –en horitzontal.

La versió d'aquest horari publicat per Marey apareix a la portada de *The Visual Display of Quantitative Information*, l'assaig d'Edward Tufte (1983) –el primer d'una sèrie interessant– que té entre els seus nombrosos mèrits el d'haver tornat a proposar, traient-los dels arxius, molts d'aquests exemples històrics en una tasca de divulgació que ha sabut arribar al món del disseny. Tufte, que ha estat professor d'estadística a la Universitat de Yale, és hereu d'una tradició que podríem definir com a minimalisme pragmàtic i ha sabut expressar de manera concisa i eficaç aquesta sensibilitat de la qual continua sent un dels defensors més actius en el debat actual.



Figura 19. Edward Tufte: *The Visual Display of Quantitative Information* i altres publicacions

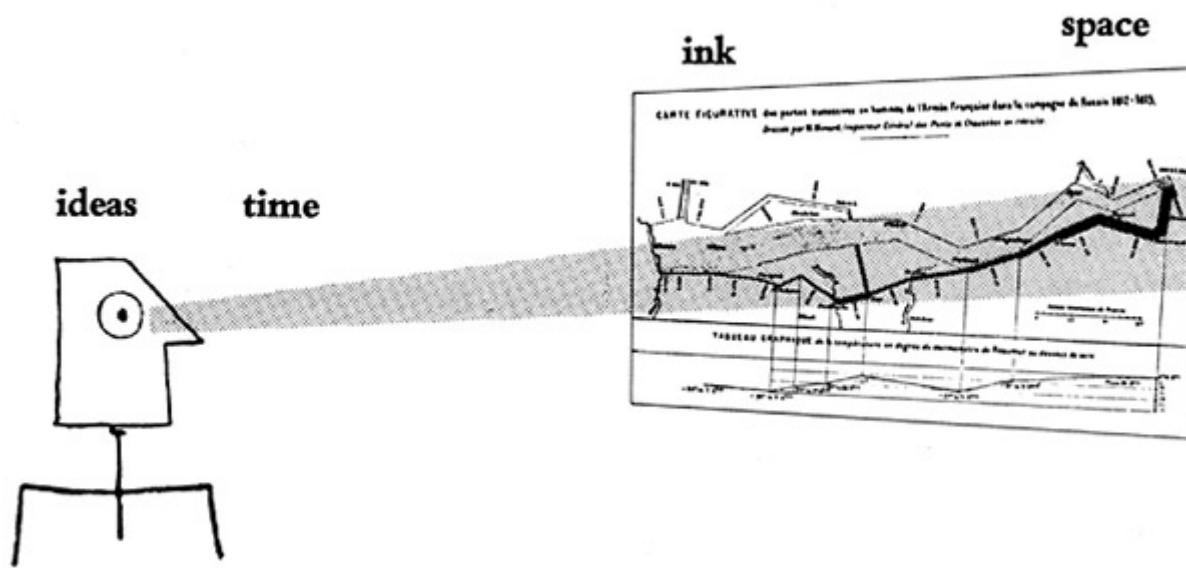


Figura 20. Edward Tufte: Principis d'excellència gràfica

Des d'aquesta perspectiva l'excellència gràfica s'aconseguiria en transmetre el màxim nombre d'idees possible, en el mínim temps possible, estalvant espai i utilitzant els recursos gràfics imprescindibles.

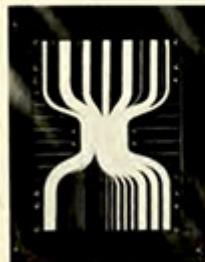
Abans que ell, els primers anys del segle xx, Willard Brinton havia tingut una actitud menys estricta i havia inclòs en els seus manuals (Brinton, 1914, 1939) exemples més pròxims al món de la il·lustració. L'obra de Brinton, sobretot en la versió ampliada de 1939, representa un dels tractats més exhaustius sobre la producció d'infografies, tant des del punt de vista conceptual com en els aspectes tècnics, que continua sorprendent per la seva capacitat de mantenir-se actual i que es pot consultar lliurement a archive.org.



International Business Machines Corp., N. Y. C.

A. The Use of a Cosmograph to Make a Flow Chart.

1. The "Cosmograph" is a flow chart made by using the device shown above. One thousand strips of paper are set on edge to represent 100%, and are separated into component parts of 100%.
2. These two illustrations give two steps in making a "Cosmograph." The first shows the process of locating and firmly clamping the strips of paper into position. The second shows wedge spacers and bar spacers being inserted between groups of strips of paper.



International Business Machines Corp., N. Y. C.

B. The Completed Cosmograph.

1. Border guides are placed in position to block out excess ends of the paper strips and the Cosmograph is ready for photostatting.
2. The negative photostatic print appears at the right. Note that all black portions of the device fail to reproduce. Of the one thousand strips of paper, twenty are red and are set at each 5% mark. In the negative photostat, these red strips of paper reproduce as white.

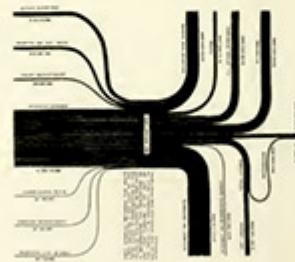


International Business Machines Corp., N. Y. C.

SCALE .6

A. Cosmograph Showing Distribution of German Reparation Payments.

1. The left side of the chart shows the total amount of reparations, and the countries by which they were received. The center of the chart shows the amounts retained by each country, indicated by the broken portions of the branches. The right side of the chart shows the amounts paid in turn by the several countries to the United States. The extreme right shows the total amount received by the United States.
2. The effect of the broken branches is obtained by sliding the paper strips backward until their ends lie at the center of the chart. The remaining strips are held in position at the center by the insertion of wedges.



International Business Machines Corp., N. Y. C.

SCALE .6

B. Cosmograph Showing Simple Income and Outgo.

1. In setting up such a chart, the center trunk is clamped in the usual manner. The income side of the chart is set up and clamped, the board is turned and the expenditure side is arranged and clamped.
2. A short strip of black paper is pasted across the trunk to provide a white block on the negative photostatic print. The total money value is noted in type on this white block.

Figura 21. Willard Brinton: *Graphic presentation* – 1939

Com hem vist fins ara, molts dels mètodes gràfics desenvolupats en l'àmbit científic es caracteritzen per la seva versatilitat i poden ser aplicats a disciplines i contextos diferents. D'altres han sorgit per respondre a exigències més específiques, per descriure fenòmens que tenen lloc en una dimensió espacial i temporal una mica més complicada, com les interaccions entre elements químics, l'estructura tridimensional de les proteïnes, la geometria dels cristalls o les interaccions subatòmiques objecte de la física quàntica.

James Elkins (1999) s'ha ocupat de moltes d'aquestes representacions amb la intenció d'establir una relació amb els processos que s'estaven produint simultàniament en la producció d'imatges dins del món de l'art. L'imaginari iconogràfic de cada època ha influït, i ha estat al seu torn modificat, per les imatges –gràfiques i mentals– que la ciència ha anat produint. L'anècdota sobre el químic alemany August Kekulé és prou coneguda: va explicar que havia intuiti l'estructura química circular del benzè després d'un somni en què va veure la imatge ancestral d'un uròbor, una serp que es mossega la cua. Elkins tracta en el seu assaig –entre altres exemples– les projeccions estereogràfiques, utilitzades en cristallografia per descriure la geometria i el desenvolupament dels minerals; la tècnica consisteix essencialment a representar cada faceta d'un cristall com un punt en una superfície esfèrica i dona lloc a descripcions que –si bé són exhaustives– presenten un elevat grau d'abstracció i disten molt de les imatges més convencionals obtingudes amb projeccions paral·leles. L'autor associa aquesta capacitat de representar simultàniament diferents plans de projecció amb l'estètica proposada en els mateixos anys pel moviment cubista.

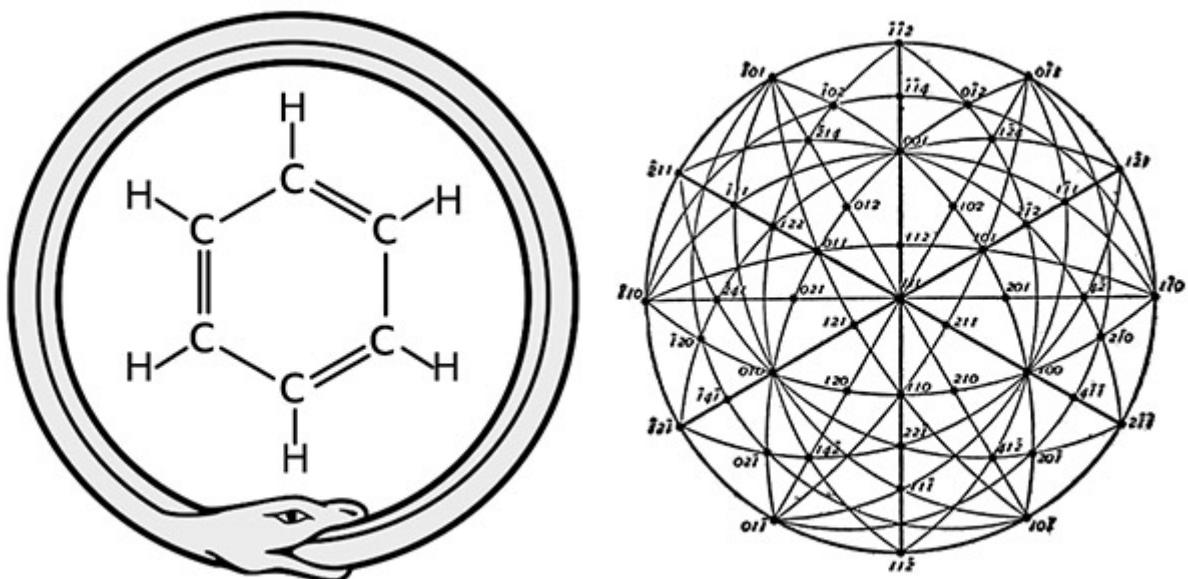


Figura 22. August Kekulé: estructura del benzè | projecció estereogràfica

La creació de noves notacions i mètodes gràfics continua acompanyant en els nostres dies el desenvolupament de les ciències; en un cèlebre article de 1948, el físic Richard Feynman donava a conèixer els seus célebres diagrames, un sistema per representar i calcular gràficament les trajectòries de les partícules subatòmiques. En temps encara més recents –el 1971– el físic i matemàtic Roger Penrose va elaborar un sistema de notació, un diagrama de tensors normalitzat, que permet simplificar la reproducció de fórmules matemàtiques complicades i que ha estat àmpliament adoptat per la comunitat científica en les seves publicacions.

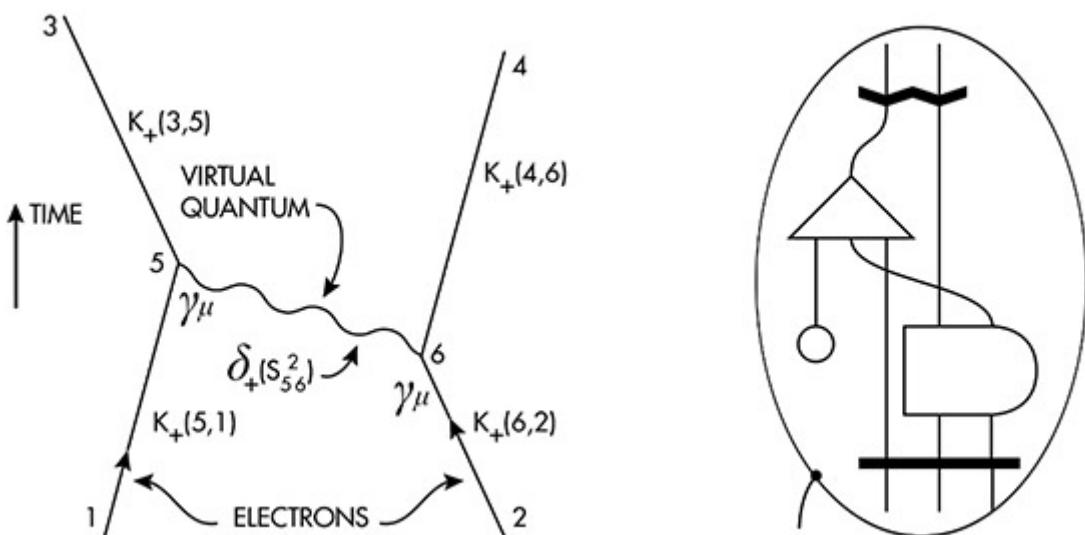


Figura 23. Richard Feynman: Diagrames | Roger Penrose: Notació gràfica

Els exemples que hem comentat en aquest subapartat comparteixen un tret comú: els seus creadors –tot i que podem considerar-los dissenyadors amb caràcter general– són experts en les seves respectives àrees de coneixement que han sabut adoptar i integrar la cultura visual a la seva feina. Per completar aquest breu recorregut històric, ens queda ocupar-nos del paper dels professionals que s'han especialitzat en la comunicació mitjançant imatges.

1. Visualització de dades

1.1. Introducció: més que mil paraules

1.1.5. La visualització de dades en el disseny de la informació: des d'Isotype als Design Systems

Un dels antecedents més clars d'aquest procés d'especialització el trobem a Viena a principis dels anys 20 del segle passat. En un moment de grans transformacions econòmiques i socials –i en un clima de grans expectatives pel que fa a la modernitat– la ciutat, governada aleshores pel partit socialdemòcrata, vivia una època de gran fervor cultural.



Figura 24. Isotype: exposició a l'ajuntament de Viena

Al voltant d'Otto Neurath –ja membre del cercle de Viena– es va començar a formar un grup de treball interdisciplinari del qual formarien part –entre d'altres– Rudolf Modley i els dissenyadors Gerd Arntz i Marie Reidemeister (Marie Neurath a partir del seu casament el 1941), que amb el suport de l'administració municipal organitzarien cicles d'exposicions dedicades a grans temes d'actualitat, industrialització, transformacions en el sector agrari, creixement urbà o moviments migratoris, per exemple, amb la finalitat de sensibilitzar el gran públic sobre aquests fenòmens globals. Aquesta activitat va dur a la creació del Siedlungsmuseum (Museu de l'Habitatge i de la Planificació Urbana) el 1923 i successivament a la seva transformació –el 1925– al Gesellschafts und Wirtschafts Museum (Museu de la Societat i de l'Economia) –encara en actiu– del qual Neurath va ser el primer director.

Ja en els seus primers anys de funcionament va prendre forma el que al principi es va conèixer com a Wiener Methode der Bildstatistik (mètode vienès d'estadística per imatges) i que aviat va passar a denominar-se ISOTYPE (International System of Typographic Picture Education) i que va evidenciar encara més la seva vocació universal i pedagògica. Simplificant molt podríem dir que el mètode es caracteritza per ser gairebé exclusivament il·lustrat i per l'ús de pictogrames per representar –per mòduls– quantitats discretes en lloc de la tradicional comparació per àrea. La disposició en el pla d'aquests elements sol oferir, a més, altres possibles lectures i interpretacions, com en els exemples següents.

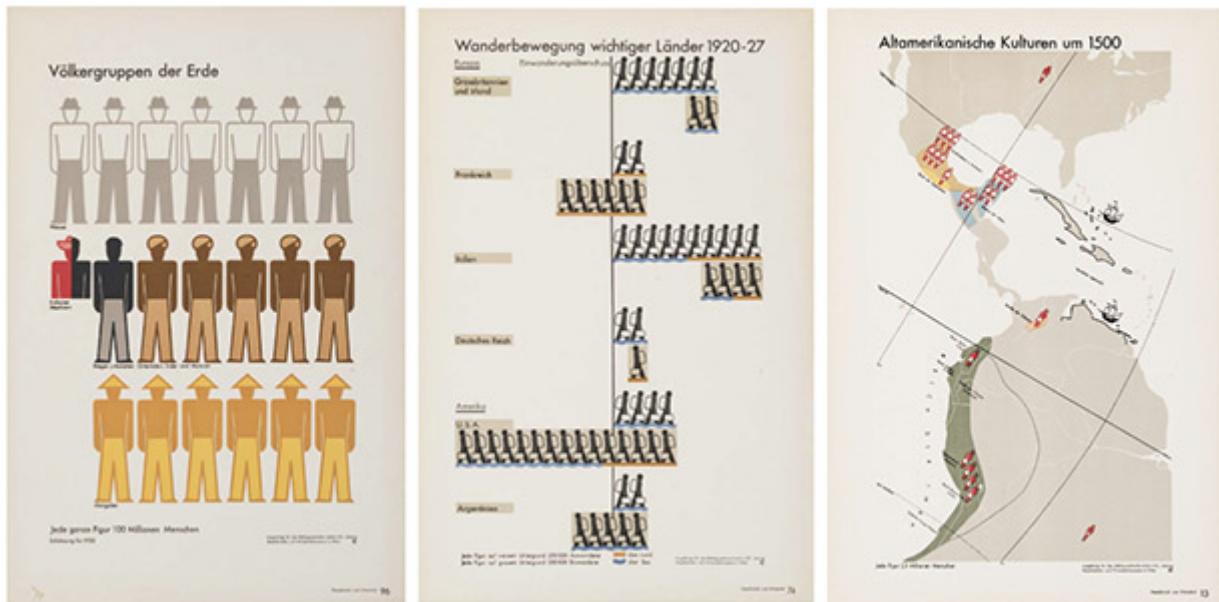


Figura 25. Isotype: composició ètnica de la població mundial | fluxos migratoris entre 1920 i 1927 | mapa de cultures precolombines

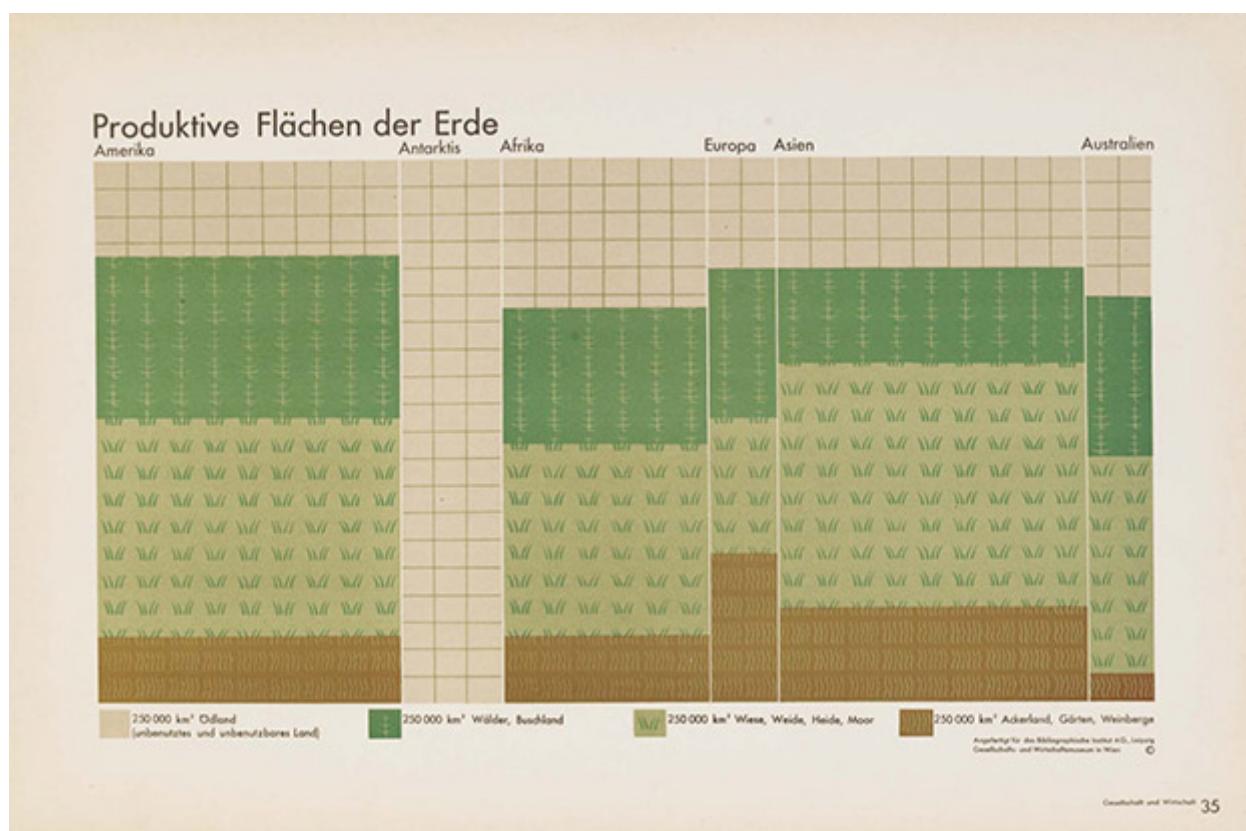


Figura 26. Isotype: productivitat agrícola en els continents

En la il·lustració sobre la composició ètnica de la població mundial podem observar, per exemple, com la població caucasiana representava aproximadament una mica més d'un terç de la població mundial; en la il·lustració següent els pictogrames que representen cada 250.000 migrants estan ordenats i orientats a partir d'una línia mitjana que evidencia el saldo migratori de cada àrea geogràfica representada. Una altra característica del treball d'Isotype va ser sens dubte la gran atenció prestada a la representació cartogràfica, en una època en què la producció de mapes continuava sent una tasca tècnicament complicada; les il·lustracions produïdes intenten adaptar-se a les necessitats de cada tema tractat, s'allunyen de projeccions convencionals –com la de Mercator– i s'obren a l'experimentació de nous cartogrames, com en la il·lustració sobre la productivitat agrícola.

Isotype ha representat, tal vegada, el primer exemple d'integració vertical en el sector de la divulgació dirigida a un públic generalista. El procés integral –que anava des de la identificació dels temes, el tractament de les dades i la conceptualització dels objectius de la comunicació fins a la producció gràfica i material dels elaborats– està descrit minuciosament a *The transformer principles of making Isotype charts* (Neurath, Kinross, 2009) per una de les seves integrants, Marie Neurath, i per l'assagista Robin Kinross que, en la seva època d'estudiant, va treballar en la catalogació del llegat d'Isotype a la Universitat de Reading.

L'experiència a Viena va arribar a la seva fi, en un ambient ja prebètic, el 1934. Després d'una breu etapa a Rotterdam, Isotype es va establir a Oxford el 1940. En aquest nou ambient l'activitat va abordar altres temes d'actualitat, com ara la publicació d'*«Only an ocean between»*, part d'una campanya d'opinió per enaltir les relacions entre el Regne Unit i els Estats Units amb la finalitat de propiciar un clima favorable a la intervenció d'aquests últims en el conflicte mundial.

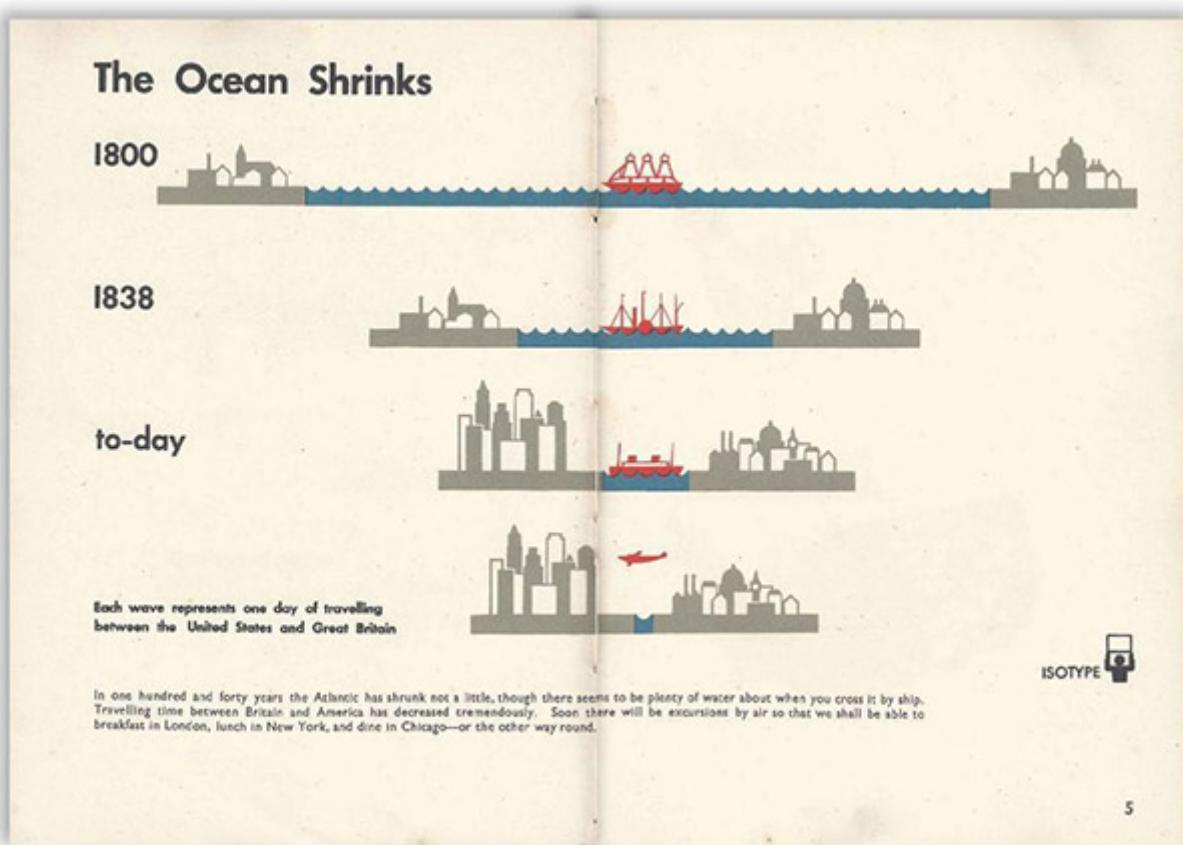


Figura 27. Only an ocean between – 1943

Després de la mort d'Otto Neurath el 1945 la tasca d'Isotype va seguir al Regne Unit, sota la direcció de la seva dona Marie, i als Estats Units, gràcies a l'activitat de Rudolf Modley, i amb una influència rellevant en la cultura visual del segle xx.

No és fàcil trobar altres exemples, més propers en el temps, de grups de treballs interdisciplinaris que hagin contribuït significativament a la creació de mètodes i eines per tractar els problemes de comunicació de masses del nostre temps. Potser valgui la pena fer esment de la redacció gràfica del *New York Times*, que en els últims 20 anys ha anat redefinint gèneres i formats del periodisme en línia amb experiments editorials com *The Snow Fall* (2012), a la tasca del grup coordinat per Amanda Cox que ha resultat en la publicació de *The Upshot*, o bé al desenvolupament, per mèrit entre altres de Mike Bostock, de d3.js, un dels frameworks per la visualització de dades més populars actualment.

Amb les proporcions adequades podríem assenyalar, també, el treball de dues estructures acadèmiques com el DensityDesign Lab del Politecnico de Milà o el Urban Complexity Lab de la Fachhochschule de Potsdam. A més de la seva tasca docent i de recerca, tots dos laboratoris s'han dedicat –des de la universitat– a la realització d'encàrrecs professionals per entitats públiques i privades.

Density, durant molts anys sota la direcció de Paolo Ciuccarelli, i ara amb Michele Mauri, ha mantingut una línia de recerca i producció d'extrema coherència malgrat les contínues rotacions a un equip on han fet els seus primers passos reconeguts professionals que han seguit amb les seves trajectòries professionals en el món acadèmic, en estudis com *The Visual Agency* o *Calibro*, o com Giorgia Lupi, fundadora de l'agència *Accurat* i ara *partner* de Pentagram.

Gairebé el mateix es podria dir d'**UCLAB**, que ha orientat bona part del seu treball cap a la visualització de col·leccions i que ha tingut entre les seves files professionals avui de reconegut prestigi com Moritz Stefaner. Entre treballs didàctics, projectes de recerca i encàrrecs professionals, el laboratori ha aconseguit una producció completa i ha mantingut una evolució fins ara molt coherent i reconoscible.

Altres àmbits de la cultura visual, com la il·lustració i la fotografia, han contribuït al disseny de la informació, potser de manera menys sistemàtica però sens dubte rellevant, i han ampliat els registres del llenguatge gràfic utilitzat més enllà del minimalisme formal. En aquest text, per raons de brevetat, esmentarem només dues de les moltes referències possibles començant pel metge i divulgador d'origen alemany, Fritz Kahn.

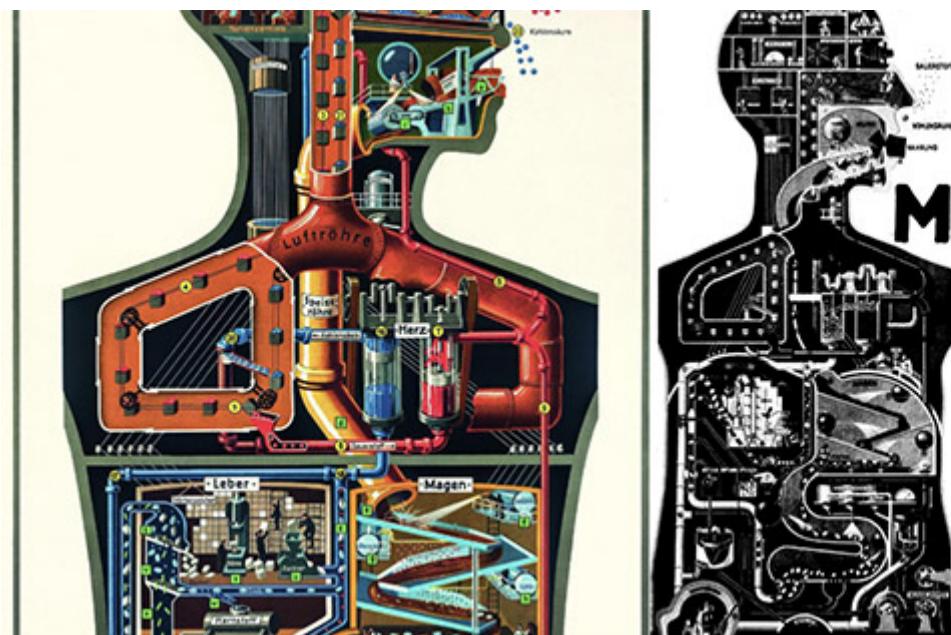


Figura 28. Fritz Kahn: *Das Leben des Menschen* – 1922

Kahn és conegut per una sèrie de publicacions, iniciades el 1922 amb *Das Leben des Menschen*, que utilitzen la metàfora de la màquina per il·lustrar el funcionament del cos humà, com si es tractés d'un seguit d'activitats industrials, i que van tenir un gran èxit editorial. El metge alemany, que no era il·lustrador, va actuar com a assessor científic i director artístic i va coordinar la tasca d'un petit equip d'experts contractats per l'editorial Franckh'sche Verlagshandlung.

Les il·lustracions realitzades sota la supervisió de Kahn van representar un dels primers exemples en la utilització d'un llenguatge molt informal en la divulgació científica. Seguint la màxima de «falsch ist es schon, aber verständlich!» (no és cert, però s'entén!) inicia un nou gènere d'il·lustració que trobarà el seu entorn natural de difusió en el naixent sector editorial de les revistes de divulgació científica, com *Scientific American* –publicada ininterrompidament des de 1845 fins avui– i s'obre un debat, encara en curs, sobre quantes concessions es poden (s'han d') admetre perquè certs continguts agradin, i siguin comprensibles, per un públic interessat, però no especialista.

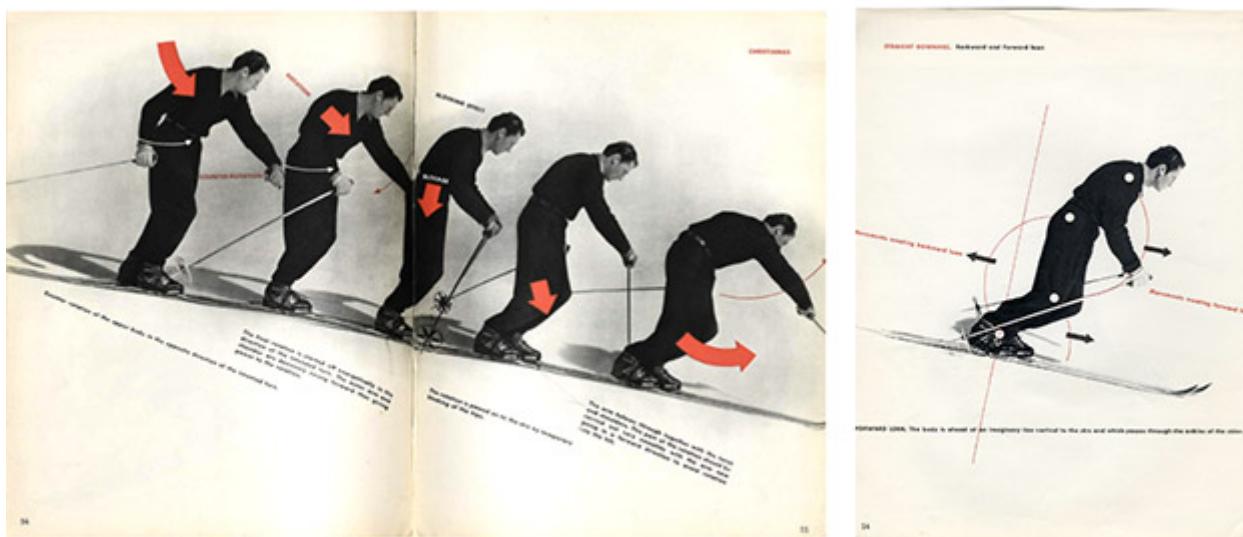


Figura 29. Emile Allais, Pierre Boucher: *How to ski by the french method* – 1947

Un altre moment clau en l'evolució del disseny de la informació va ser, sens dubte, la publicació el 1947 de *How to ski by the french method* en el qual el fotògraf Pierre Boucher il·lustra el mètode desenvolupat pel célebre esquiador Emile Allais. Boucher porta habíliment el llenguatge formal desenvolupat per les avantguardes del segle xx, tècniques de fotomuntatge, eixos inclinats i formes geomètriques al terreny de la comunicació tècnica, amb una sèrie d'il·lustracions en dos colors –negre i taronja– que barregen de manera molt eficaç seqüències fotogràfiques, textos i elements gràfics.

Al llarg de la segona meitat del segle passat, la visualització de dades es consolida com un dels gèneres del disseny gràfic i és objecte d'atenció específica per part de la disciplina; el 1974, Walter Herdeg (1974) coordina per l'editorial suís Graphis un dels

seus cèlebres monogràfics dedicats a la visualització de dades abstractes on es recull la diversitat de llenguatges i sensibilitats que s'havien anat desenvolupant sobre mapes, diagrames i esquemes.

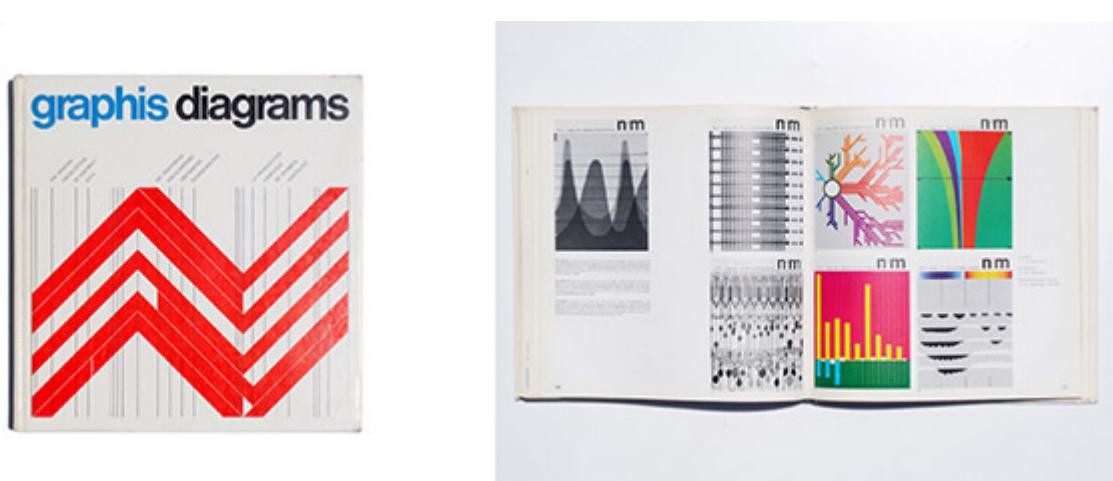


Figura 30. Walter Herdeg: Graphis *The graphic visualization of abstract data* – 1974

Poc després, el 1976, en una conferència a l'American Institute of Architecture, el dissenyador i arquitecte Richard Saul Wurman comença a referir-se a aquesta activitat com a disciplina utilitzant el terme *information architecture* –en lloc de *design*– tot afirmant:

“Information architecture (IA) is a professional practice and field of studies focused on solving the basic problems of accessing, and using, the vast amounts of information available today».

Wurman, que té entre els seus mèrits el d'haver estat l'iniciador, al costat de Harry Marks, de les populars TED Talks (*technology, entertainment and design*) el 1984, transmet, a més, una sensació que s'anirà consolidant amb el temps: cada vegada més hi ha més informació disponible. Uns quants anys abans de la difusió generalitzada dels ordinadors personals –a partir dels anys 80– i de la popularització d'internet a través de la www, que modificarà radicalment l'accés a la informació a partir dels anys 90, al voltant de la IA comencen a definir-se els conceptes que caracteritzaran l'evolució de la visualització de dades en l'actual panorama digital.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.1. Introducció

Han estat diversos els processos que ens han portat a la situació actual de gran abundància, per alguns fins i tot d'excés, en la disponibilitat de dades de tot tipus –el que coneixem com a fenomen de les dades massives (*big data*).

Simplificant molt, podríem dir que hi ha hagut tres grans vectors: d'una banda, el que coneixem com a *media fusion*; la versatilitat demostrada pels ordinadors que han passat de ser simples màquines de càlcul, pensades per operar amb xifres, a poder abastar i descriure tota classe d'informació a través d'un comú denominador digital que ens permet operar en els diferents mitjans (Alpaydin, 2016). Paral·lelament, gràcies a processos de miniaturització, més eficiència de càlcul i abaratiment dels costos de producció –l'evolució de la qual ha estat brillantment sintetitzada a través de la llei de Moore– hem arribat a l'actual situació en la qual la presència de microprocessadors i dispositius digitals s'ha fet pràcticament ubiqua. I, per acabar, l'expansió d'una xarxa, que inicialment incloïa només algun ordinador, capaç de connectar tots aquests dispositius a la que avui denominem la internet de les coses (IoT, *the internet of things*).

En aquest escenari, cadascun de nosaltres deixa inevitablement un rastre digital en la immensa majoria de les seves activitats quotidianes i alimenta, així, un volum de registres que no té precedents històrics. Una de les primeres conseqüències d'aquest procés, encara relativament recent, ha estat modificar radicalment la nostra visió de la intel·ligència artificial, del que sabem fer a través d'algorismes. Fins fa relativament poc, la intel·ligència d'un algorisme estava en la seva estructura lògica: els programes per jugar als escacs valoraven les posicions i coneixien els escacs, els que componen música contemplaven regles d'harmonia, acords, melodia, contrapunt, etc. Actualment, bona part del que considerem intel·ligència artificial es desenvolupa per mitjà de tècniques de *machine learning* i es basa, en última instància, en l'anàlisi estadística de grans volums de mostres digitals. Podem crear música per piano simplement observant i registrant patrons de comportament en com, quan i on els pianistes experts posen els seus dits, sense que l'algorisme conegui les notes associades a cada tecla d'un piano MIDI –com a [TensorFlow Performance RNN](#).

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.2. Processing i el seu entorn

En un context en què la comunicació es crea i es distribueix en suport digital, la programació ha estat tradicionalment, i continua sent, l'eina per excel·lència. En aquest escenari, l'ecosistema de programari que ha crescut els últims 20 anys al voltant del Processing s'ha consolidat com una de les portes d'accés al món de la programació més amigables per als qui procedeixin d'una formació visual.

Més enllà del que puguem fer amb les aplicacions que ens permeten editar els diferents mitjans digitals, i al marge que aquestes siguin productes comercials o desenvolupaments en codi obert, els entorns de programació creativa –com Processing o p5js– ens permeten arribar a una profunditat incomparable i ens proporcionen una enorme llibertat d'acció per barrejar mitjans diferents.

Pel que fa específicament al món de la visualització de dades, aquesta llibertat es reflecteix no només en el que podem crear a partir de dades estructurades, sinó també en la nostra capacitat de tractar com si fossin dades qualsevol tipus d'informació codificada en suport digital.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.3. Treballar amb textos

Ben Fry: *The Preservation of Favoured Traces*

En el procés que ha portat a la progressiva digitalització dels diferents mitjans, els textos han ocupat tradicionalment el primer esglao, just per sobre dels números. Durant una pila d'anys, abans que els ordinadors poguessin mostrar imatges, els textos han estat la intereficie entre aquests aparells i els seus usuaris.

L'exemple del qual ens ocuparem ara utilitza els textos com a dades i elabora informació a partir d'operacions senzilles de comparació entre paràgrafs. Es tracta de *The Preservation of Favoured Traces*, un treball pioner de Ben Fry –un dels desenvolupadors de Processing– en el qual s'analitzen les primeres sis edicions del célebre assaig de Charles Darwin *L'origen de les espècies*. Des de la seva publicació el 1859, les teories de Darwin van donar lloc a un intens debat que es reflecteix, d'alguna manera, en les modificacions introduïdes per l'autor en cada nova edició –n'hi va haver sis fins al 1872 a intervals d'entre un i tres anys–. El naturalista anglès va sentir el deure d'argumentar de manera diferent algunes de les seves tesis acceptant certes crítiques i matisant la seva posició o oferint nous arguments per mantenir la seva posició inicial.

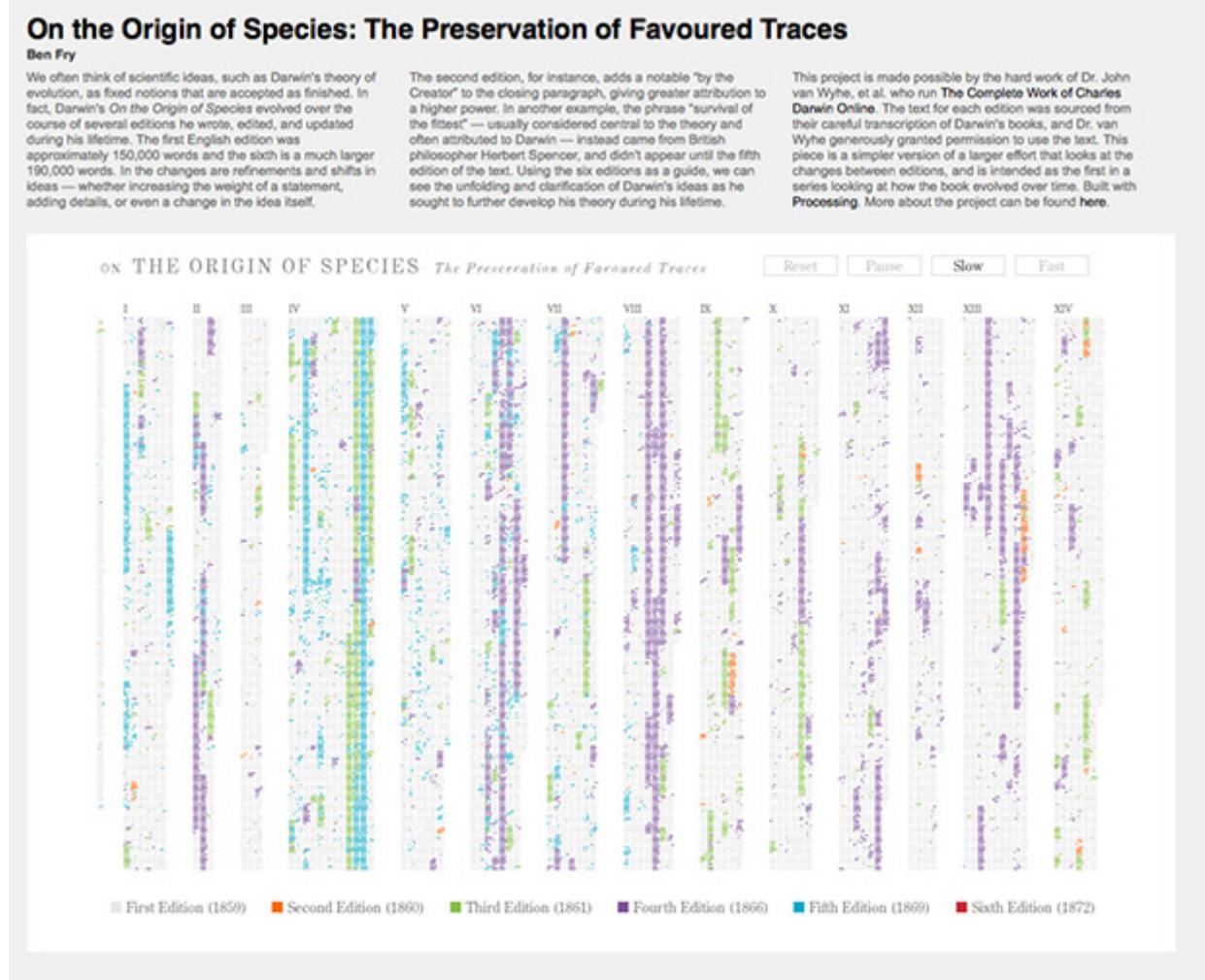


Figura 31. Ben Fry: *The Preservation of Favoured Traces*

En la visualització de Fry, els quinze capítols del llibre s'organitzen en columnes i cada paràgraf és representat per una petita àrea rectangular; l'aplicació presenta una animació inicial en la qual van apareixent progressivament les modificacions aportades al text inicial, edició rere edició, a través d'un codi de color. En moure el cursor per sobre de cada paràgraf és possible llegir-ne el contingut amb els textos modificats marcats amb el color corresponent a cada edició.

El resultat ens ofereix un significatiu mapa sinòptic dels passatges de l'obra que han suscitat més debat. Abans que el desenvolupament del processament del llenguatge natural (NLP, en les seves sigles en anglès) arribés als avenços que podem utilitzar avui, Fry va saber generar informació per mitjà d'unes quantes línies de codi –per comparar sistemàticament cada

paràgraf– i traduir-la de manera elegant i eficaç a un senzill llenguatge visual. El treball està disponible també en forma de pòster de 60 per 90 cm, amb el text integral de l'edició de 1872 en sis colors, i com a llibre.

The Preservation of Favoured Traces pertany, d'alguna manera, al camp més ampli de les humanitats digitals, l'àrea de recerca que, gràcies al tractament digital dels textos, ha anat implementant diferents tècniques d'anàlisi quantitativa per a la creació i l'estudi de corpus lingüístics. Iniciatives com el [Projecte Gutenberg](#), que publica en format editable els textos de domini públic, i la disponibilitat d'eines específiques per a l'escriptura generativa i el NPL com [Rita](#), fan de Processing i p5js un entorn particularment interessant per al desenvolupament d'aplicacions d'aquest tipus, com els d'aquesta col·lecció d'[exemples](#).

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.4. Píxel per píxel

A Cinemetrics, Frederic Brodbeck duu a terme una operació en molts aspectes similar a la que acabem de descriure: desagregar les dades que descriuen un determinat mitjà per utilitzar-les com a font d'informació en una inusual visió de síntesi.

Cinemetrics utilitza els píxels, la unitat mínima d'informació discreta amb la qual descrivim les imatges de tipus *raster*, dels fotogrames d'una pel·lícula per explicar i reordenar els colors en lloc de representar-los en una matriu bidimensional en les seves corresponents coordenades.

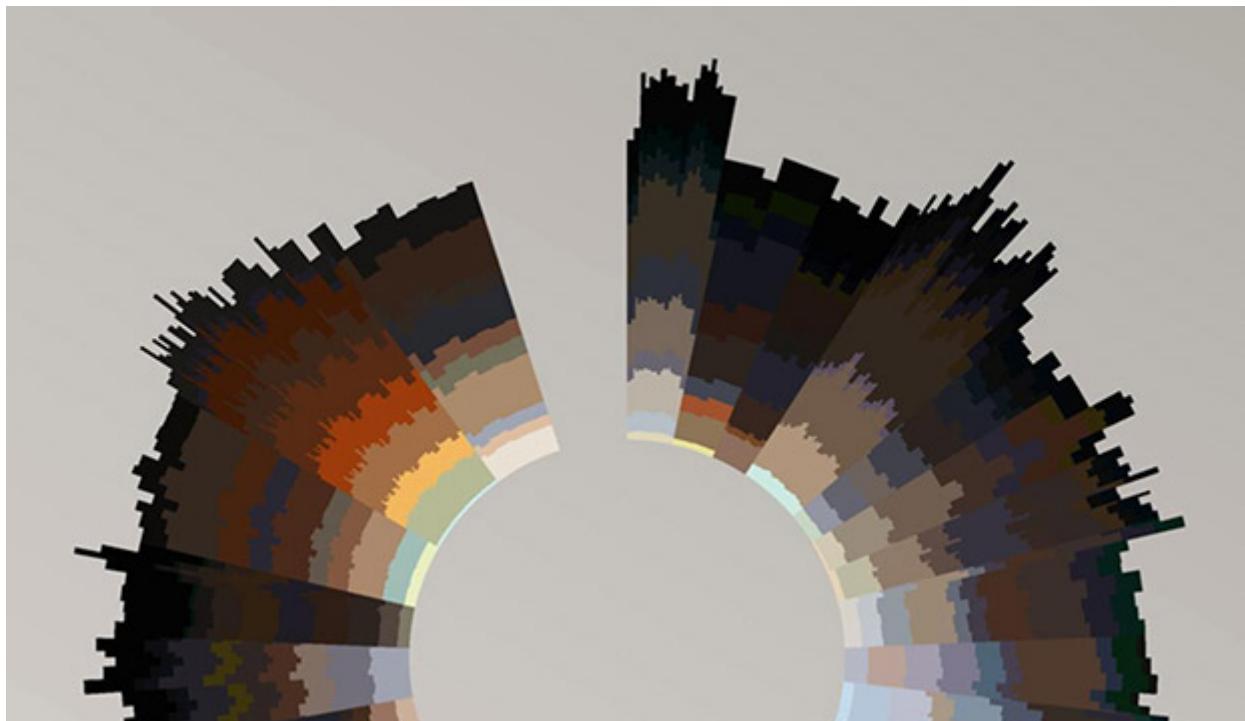


Figura 32. Frederic Brodbeck: Cinemetrics

Analitzant cada píxel de cada fotograma, un algorisme redueix la informació cromàtica de cada pla seqüència als 5 colors que li són més representatius; una operació anàloga permet definir els 5 colors més representatius del conjunt de la pel·lícula i –una vegada ordenades aquestes 10 mostres de colors, en funció del tot de la lluminositat (en l'espai HSB)– es tornen a comptar de nou els píxels de cada seqüència per assignar-los a la mostra de color més pròxima. L'algorisme per calcular la similitud entre colors és bastant senzill: només cal calcular la distància en l'espai tridimensional de dimensions R, G i B.

Llavors, amb aquestes dades es pot crear el que Brodbeck defineix com la petjada (cromàtica) d'una pel·lícula; es tracta d'un anell on cada sector representa 10 seqüències i ocupa un espai proporcional a la seva durada. Cada sector, al seu torn, està compost per 10 arcs concèntrics, un per cada mostra de color, amb un gruix proporcional al nombre de píxels representats. En versió multimèdia, els sectors oscil·len al llarg del seu radi amb una freqüència proporcional al dinamisme de les escenes (calculat observant quants píxels canvien respecte al fotograma anterior) i, en moure's, el cursor revela una miniatura de cada fotograma.

Tot i que en molts aspectes són arbitràries, les petjades de Cinemetrics aconsegueixen captar trets distintius de cada pel·lícula i es presten a comparacions, entre pel·lícules del mateix gènere, del mateix director o entre versions originals i *remakes*.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.5. Data scraping i data parsing

Lostalgic, el projecte de Santiago Ortiz dedicat a la sèrie *Lost* (Perduts) –produïda per ABC– és un exemple més d’una visualització creada a partir de dades, en suport digital, que havien estat estructurades inicialment amb una altra finalitat. El treball es basa en els guions dels 115 episodis de les set temporades –publicats a la web Lostpedia– dels quals s’han extret els diàlegs entre els diferents personatges; gràcies a la seva estructura predictable, composta per: nom del personatge + dos punts + frase entre cometes, ha estat possible crear una primera visualització: un índex general de la sèrie que ens mostra, en successió horitzontal, una miniatura de cada personatge per cada entrada en el diàleg.



Figura 33. Santiago Ortiz: Lostalgic

Els diàlegs permeten mesurar el nivell d’interacció entre personatges, en cada episodi i al llarg de la sèrie, evidenciant certs patrons en el desenvolupament narratiu de les set temporades; ens diuen qui hi apareix, quan ho fa, quant de temps està present en escena i amb quins altres personatges es relaciona. Lostalgic ens permet analitzar aquesta informació per mitjà de diferents diagrames: un graf dibuixat sobre una superfície esfèrica ens mostra la freqüència de diàlegs entre els personatges en cada episodi; la mateixa informació està disponible també en forma de matriu i des de totes dues s’accedeix a una fitxa del personatge on es mostra un histograma, amb la freqüència d’aparicions al llarg de les diferents temporades, i un treemap de les seves relacions amb altres personatges.



Figura 34. Lostalgic: relació entre personatges

En la secció «Reenactment», a més, es pot gaudir d'una recreació de la sèrie en un format molt poc convencional, una espècie de còmic multimèdia en el qual els diàlegs apareixen i s'esvaeixen en bafarades dinàmiques.

La manera amb la qual s'ha obtingut la informació utilitzada a Lostalgic es col·loca a mig camí entre dues tècniques molt comunes en la visualització de dades: el *data scrapping* i el *data parsing*. Normalment se sol definir l'activitat de *parsing* (analitzar) quan extraiem només la informació que ens interessa en un determinat moment d'una font de dades estructurada per filtrar els elements en funció de certs criteris o per accedir a determinats descriptors (per exemple, en una llista d'alumnes, mostrar el nom de pila dels alumnes amb edats compreses entre certs valors). La idea de l'*scrapping* (literalment «rascar») se sol considerar com un últim recurs en absència de dades estructurades i consisteix bàsicament a anar a buscar les dades allà on es publiquin encara que no estiguin en el format més idoni. Reconstruir taules i llistats des de documents en format PDF, buscar en diverses botigues en línia el preu d'un mateix producte o extreure els temes més debatuts en els missatges d'un determinat fòrum són exemples comuns d'aquesta mena de pràctica per la qual se solen desenvolupar programes *ad hoc*. Google i Amazon AWS han afegit recentment als seus serveis eines de *web-scrapping* per als seus clients.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.6. API i temps real

Moltes aplicacions de visualització de dades tenen el seu punt de força, i el seu principal interès, en la possibilitat d'utilitzar informació actualitzada en temps real o, en tot cas, amb una freqüència adequada al tema tractat.

Les fonts per aquesta mena d'aplicacions poden ser molt diferents, des de repositoris de dades –en modalitat *opendata*– d'organismes oficials, a iniciatives més o menys formals que actuen com agregadors de dades procedents de diferents fonts, fins a situacions encara més dinàmiques en les quals és possible connectar-se a les dades directament en el lloc on es produeixen a través d'API (*Application Programming Interface*).

Per posar un exemple: el Centro Nacional de Epidemiología d'Espanya (CNE) publica una web per al seguiment de la pandèmia de la covid-19: [cneccovid](#) des de la qual és possible accedir a diversos indicadors, com el factor de retransmissió o la incidència acumulada en els últims 14 dies.

Tot i estar en una web, les dades publicades pel CNE no són fàcilment accessibles –de manera directa– des d'una aplicació de visualització de dades, caldria descarregar-les cada dia per actualitzar la informació publicada; potser per això ha nascut el projecte [Escovid19data](#) que s'encarrega d'agregar, de manera col·laborativa, dades oficials sobre la pandèmia.

Altres iniciatives anàlogues, com [ourworldindata](#) han afegit als seus repositoris de dades informació sobre la covid-19 a escala global, com per exemple l'evolució del [procés de vacunació](#) en diferents països i àrees geogràfiques.

Gràcies a la disponibilitat d'aquestes dades, l'ONG Mèdics del Mundo ha pogut integrar indicadors actualitzats diàriament a una [web monogràfica](#) en la qual es documenten les seves activitats des de març de 2020.

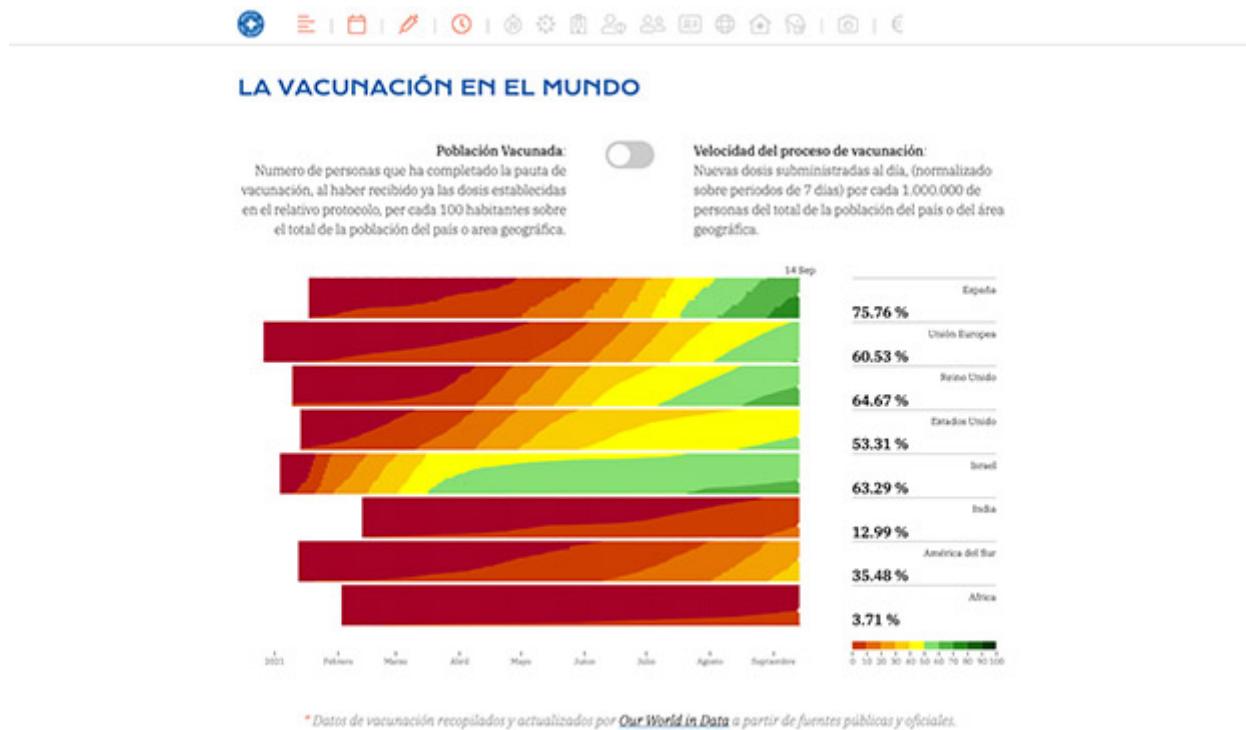


Figura 35. Mèdics del Mundo: informe covid-19

En aquest cas, la font de les dades està en cadascun dels documents en format json, allotjats en els servidors d'Escovid19data i ourworldindata, que les dues entitats s'encarreguen d'actualitzar diàriament.

Una altra modalitat comuna per accedir a dades en temps real és accedir a la seva font a través d'una API; per exemple, podríem imaginar una aplicació que llegeixi (o compti) els tuits marcats amb una determinada etiqueta en un lapse de temps concret. Per a això no seria necessari descarregar-se i filtrar tots els tuits publicats perquè l'API de Twitter ens permet (amb certes limitacions) formalitzar una consulta específica i ens proporciona només els resultats d'aquesta cerca.

Ja són molts els serveis de plataforma que ofereixen accés públic a certes dades sobre les seves transaccions, des de les fotos publicades a [Flickr](#) a [Velib](#), el servei públic de lloguer de bici de París.

The screenshot shows the Flickr API documentation homepage. At the top, there's a navigation bar with links for 'Sign In', 'Explore', 'Prints', 'Get Pro', a search bar, and a 'Sign In' button. Below the header, the title 'The App Garden' is displayed in a large blue font. Underneath the title, there are links for 'Create an App', 'API Documentation', 'Feeds', and 'What is the App Garden?'. A note states: 'The Flickr API is available for non-commercial use by outside developers. Commercial use is possible by prior arrangement.' To the left, a sidebar titled 'Read these first:' lists several developer resources. To the right, a section titled 'API Methods' lists categories like 'activity', 'auth', and 'auth.oauth' each with their respective API methods.

The App Garden

Create an App | API Documentation | Feeds | What is the App Garden?

The Flickr API is available for non-commercial use by outside developers. Commercial use is possible by prior arrangement.

Read these first:

- [Developer Guide](#)
- [Overview](#)
- [Encoding](#)
- [User Authentication](#)

- [Dates](#)
- [Tags](#)
- [URLs](#)
- [Buddyicons](#)
- [Collections](#)

API Methods

activity

- [flickr.activity.userComments](#)
- [flickr.activity.userPhotos](#)

auth

- [flickr.auth.checkToken](#)
- [flickr.auth.getFrob](#)
- [flickr.auth.getFullToken](#)
- [flickr.auth.getToken](#)

auth.oauth

- [flickr.auth.oauth.checkToken](#)

Figura 36. Les API de Flickr

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.7. Diagrames interactius

La interacció és, sens dubte, un altre dels avantatges que ofereix la visualització de dades en un entorn digital.

L'exemple del qual ens ocuparem ara ofereix una interacció molt senzilla per modificar un dels diagrames més emblemàtics de la història del disseny de la informació: el mapa del metro de Londres.

Londres va ser una ciutat pionera en el transport públic, va introduir les primeres línies del seu Metropolitan Railway el 1863, i el seu mapa de 1931 –atribuït a Henry Beck– va servir de referència per la resta de xarxes de metro. Com és prou conegut, els primers mapes de la xarxa del metro tenien una base cartogràfica: marcaven les parades en la seva ubicació exacta i traçaven les línies seguint el seu recorregut subterrani. La gran intuïció de Beck va ser adoptar les convencions amb les quals es dibuixen els circuits elèctrics, que ja havien estat utilitzades a Londres per la xarxa de clavegueram, i representar la xarxa de manera qualitativa, col·locant les parades a distàncies regulars, normalitzant la inclinació de les línies en múltiples de 45 graus, diferenciant gràficament les estacions amb intercanvi de línia i mantenint com a única referència geogràfica amb la superfície el recorregut esquematitzat del riu Tàmesi.



Figura 37. Tom Carden: *Travel Time Tube Map*

El 2005, el dissenyador Tom Carden va convertir la xarxa en un graf d'estacions, unides per les línies de metro, per crear [Travel Time Tube Map](#) una versió interactiva que permet reorganitzar el diagrama al voltant de cada estació i mostrar, a través d'isòcrones de 15 minuts, els temps necessaris per arribar a la resta d'estacions de la xarxa.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.8. Simuladors

Una dels avantatges de Processing respecte a altres entorns més específicament orientats a la visualització de dades, com per exemple d3.js, és la relativa facilitat per crear sistemes d'objectes i simuladors.

Tant en la seva versió p5 per javascript com en la seva versió original per Java, Processing és un entorn de programació orientat a objectes; com molts ja haureu pogut comprovar, el veritable potencial del que es pot obtenir amb algorismes de generació formal es revela en el moment en què comencem a ser capaços de formalitzar regles a través de funcions i variables. La possibilitat de definir objectes de certes classes –cadascun amb els seus atributs i variables– i de generar instàncies d'aquests objectes, ens permet definir sistemes en els quals el comportament de cada element està relacionat amb la resta, simulant així les dinàmiques dels sistemes complexos.

Un dels exemples clàssics d'aquesta modalitat de programació és l'algorisme de Flocking, presentat per Craig Reynolds en el SIGGRAPH de 1987 i implementat a Processing per Daniel Shiffman, que imita el comportament d'un estol d'ocells. La simulació s'obté gràcies a tres regles: alineació –els ocells giren seguint orientació dels seus veïns–, cohesió –els ocells s'acosten a la posició dels seus veïns– i separació –volen junts sense solapar-se.



Figura 38. Daniel Shiffman: *Flocking*

Els simuladors permeten recrear models de funcionament simplificats i tenen un enorme valor didàctic per explicar fenòmens complexos. Vegeu, per exemple, aquesta explicació explorable sobre la covid-19 feta per Nicky Case.

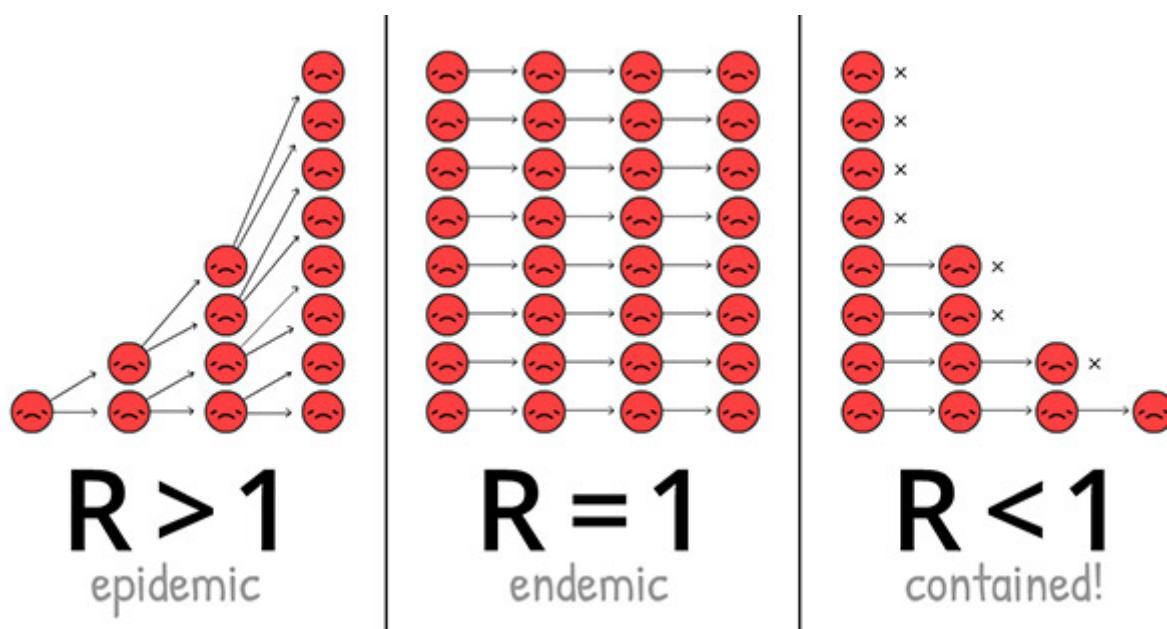


Figura 39. Nicky Case i Marcel Salathé : covid-19 Playable Simulations

Com en moltes de les seves altres Explorable Explanations, Case mostra una habilitat especial per equilibrar la narració didàctica amb altres moments de lliure exploració dels continguts. La simulació de Case, amb l'assessorament de l'epidemiòleg Marcel Salathé, ens mostra diferents comportaments de la ja nota corba en funció d'un nombre cada vegada més elevat de variables que podrien afectar l'evolució de l'epidèmia; la interacció consisteix essencialment a modificar, en temps real, algunes d'aquestes variables per comprovar-ne l'impacte hipòtic en el comportament de la corba.

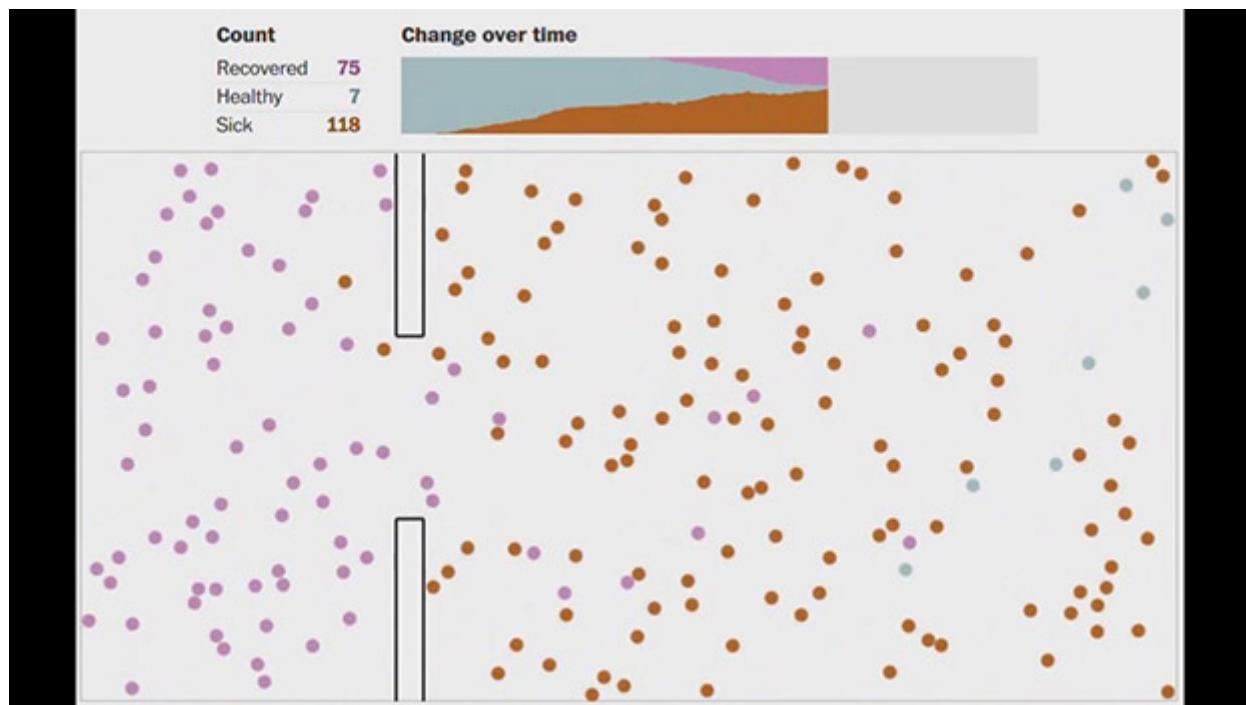


Figura 40. *The Washington Post*: simulador de coronavirus

Sobre el mateix tema, el *Washington Post* va arribar a desenvolupar un altre simulador de coronavirus interessant comparant diferents escenaris a través d'un *particle system* molt semblant en el seu funcionament, als que es poden trobar en els exemples bàsics de p5js (fàcilment transportables a Processing) – simulate particles.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.9. Visions de síntesi de dades estructurades

La programació orientada a objectes resulta de gran utilitat per crear aplicacions que ens permeten consultar de manera visual grans conjunts de dades estructurades. Trobem molts exemples d'aquest tipus en el món dels béns culturals. Els catàlegs digitals de les col·leccions de museus, arxius i biblioteques ens ofereixen diferents descriptors dels objectes catalogats; les aplicacions més tradicionals per consultar aquests conjunts de dades han privilegiat l'accés a la descripció detallada, a la fitxa de catàleg, a través d'aquests mateixos descriptors per mitjà de diferents modalitats de cerca. Amb el pas dels anys s'han anat consolidant altres aplicacions que complementen aquesta modalitat d'ús, destinada generalment a un públic expert, amb visualitzacions que ens ofereixen una visió de síntesi del conjunt d'objectes catalogats.

Un dels exemples més reeixits d'aquest enfocament el trobem a [Coins](#), una aplicació desenvolupada per l'[Urban Complexity Lab](#) a partir de les més de 500.000 monedes que formen part de la col·lecció del Münzkabinett de Berlín, un dels conjunts numismàtics més exhaustius del món.



Figura 41. Urban Complexity Lab – Potsdam: Coins

Coins ens ofereix inicialment la visió del conjunt desordenat de les monedes com si estiguessin totes amuntegades en una mateixa taula; la interefície ens permet ordenar-les en funció dels diferents descriptors: procedència, metall, dimensions, etc., i crear disposicions que recorden en molts casos els diagrames utilitzats en estadística, com ara corbes de freqüències o treemap, construïts en aquest cas a partir de cada element.

1. Visualització de dades

1.2. La visualització de dades en el nostre entorn digital

1.2.10. Poètica i estètica de les dades

Les tècniques de visualització de dades no s'utilitzen exclusivament amb finalitats pragmàtiques: ja hi ha una consolidada tradició, tant en entorns digitals com en contextos analògics, de treballs que exploren els aspectes estètics d'aquestes pràctiques amb una aproximació més poètica.

És el cas, per exemple, de l'artista i programador [Aaron Koblin](#), que en la seva sèrie [Flight patterns](#) de 2005 va utilitzar dades públiques de control aeri per crear il·lustracions i animacions *time-lapse* (càmera ràpida) capaces d'evocar i fer tangible, a través d'una aproximació estètica, el volum d'operacions diari del trànsit aeri dels EUA i la seva evolució al llarg del dia en una àrea geogràfica que inclou fins a quatre fusos horaris.



Figura 42. Aaron Koblin: *Flight patterns* – 2005

El treball de Koblin va més enllà de l'estètica i ens permet reflexionar sobre alguns temes clau del nostre entorn tecnològic. A [Bicycle Built For Two Thousand](#) més de 2.000 veus enregistades a través d'internet s'uneixen per cantar *Daisly Bell* (Bicicleta feta per a dos), un tema popular de finals del segle XIX que, a més, havia estat la primera cançó interpretada per un [ordinador](#) –un IBM 7094– programat el 1961 per John Kelly, Carol Lockbaum i Max Mathews amb tècniques de síntesi vocal precursores dels actuals vocoder.

En la proposta de Koblin, les veus van ser gravades en microfragments a través d'Amazon Mechanical Turk (MTurk), el servei d'Amazon que permet repartir, i remunerar, petites tasques digitals que requereixen una intervenció humana. En aquest cas, la tasca consistia a intentar reproduir –i enregistrar– un so d'una durada ínfima. Cap de les persones que van col·laborar a la iniciativa eren conscients que es tractava d'un tema musical ni de les finalitats del projecte.

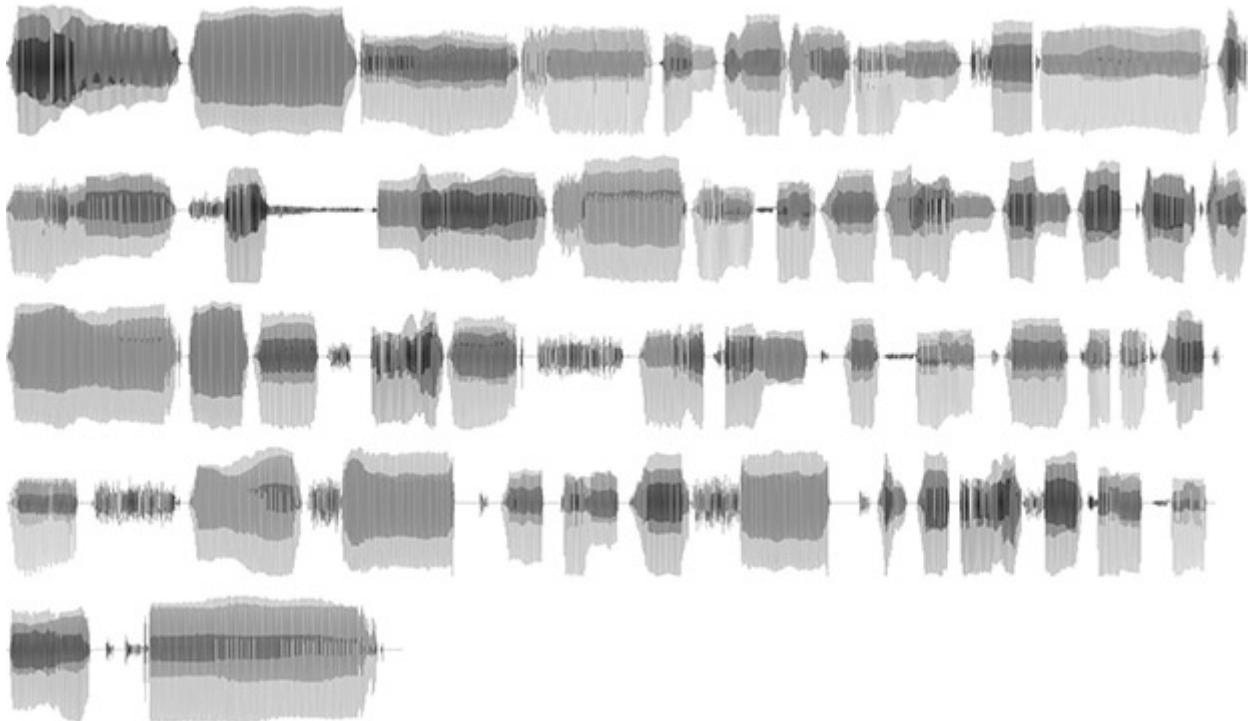


Figura 43. Aaron Koblin i Daniel Massey: Bicycle Built For Two Thousand – 2009

Bicycle Built For Two Thousand ens permet reflexionar sobre alguns temes clau en la web dels nostres dies creant una situació de paradoxa entre programes intel·ligents, web col·laborativa i serveis de plataforma.

1. Visualització de dades

Bibliografia

- Alpaydin, E. (2016). *Machine learning*. Cambridge, DT.: The MIT Press.
- Bertin, J. (1973). *Semiolegie graphique*. París: Mauton.
- Brinton, W. C. (1914). *Graphic methods for presenting facts*. Nova York: McGraw Hill.
- Bolzoni, L. (2004). *The web of images: Vernacular preaching from its origins to St Bernardino da Siena*. Ashgate.
- Brinton, W. C. (1939). *Graphic presentation*. Nova York: Brinton associates. Disponible a: archive.org
- Capra, F. (2009). *La trama de la vida: Una nueva perspectiva de los sistemas vivos*. Anagrama S.A.
- Elkins, J. (1999). *The domain of images*. Ithaca: Cornell University Press.
- Herdog, W. (1974) *Graphis Diagrams: The Graphic Visualization of Abstract Data – Die graphische Visualisierung abstrakter Gegebenheiten – la visualisation graphique de données abstraites*. Suïssa: The Graphis Press.
- Llima, M. (2014). *Book of Trees: Visualizing Branches of Knowledge*. Princeton Architectural Press.
- Marey, É. J. (1885). *L'érñthode graphique dans els sciences expérimentales...* París: Masson.
- Neurath, M., & Kinross, R. (2009). *The transformer principles of making Isotype charts*. Londres: Hyphen Press.
- Rendgen, S. & Wiedemann, J. (2019). *History of Information Graphics*. Colònia: Taschen.
- Tufte, E. (1983). *The visual display of quantitative information*. Graphics Press.

Enllaços

- Ben Fry: The Preservation of Favoured Traces <https://fathom.info/traces/>
- Ferdio. Dataviz Project <http://www.datavizproject.com>
- Frederic Brodbeck: Cinematics <http://cinematics.fredericbrodbeck.de>
- Projecto Gutemberg <https://www.gutenberg.org>
- Santiago Ortiz: Lostalgic <http://intuitionanalytics.com/other/lostalgic/>
- Tom Carden: Travel Time Tube Map http://www.tom-carden.co.uk/p5/tube_map_travel_times/applet/

2. Art generatiu

2.1. Introducció: Què és l'art generatiu?

La comunitat artística entén per art generatiu aquella pràctica que usa sistemes (màquines, codi informàtic, instruccions procedimentals o un conjunt de regles) que es posen en funcionament amb un cert grau d'autonomia i que resulten en una peça artística (Galanter, 2003).

L'estudi teòric de l'art generatiu s'estén vinculat amb la popularització de l'ordinador personal i l'expansió del medi digital. Tot i que es coneixen projectes generatius que no usen ordinadors, la majoria de projectes cauen dins l'ampli camp de l'art digital, tot aquell que usa algun tipus de tecnologia electrònica digital.

L'art generatiu s'estén també en diversos camps artístics i creatius com ara la música electrònica, la composició sonora algorítmica, el disseny industrial d'objectes o l'arquitectura paramètrica, entre d'altres. Les obres resultants són molt diverses: inclouen música, so, arts visuals, videoart, instal·lacions multimèdia, realitat virtual, escultura cinètica, robòtica, *performance* i text.

Hi ha diferents termes que s'utilitzen habitualment per indicar tot el camp d'art generatiu: art computacional, art procedural, art basat en processos o art electrònic. Tot i que tenen lleugeres diferències, sovint es tracten com a sinònims. A més, hi ha uns quants noms per a subcamps de l'art generatiu o per a projectes que poden incloure parts d'art generatiu. Hi ha l'art interactiu, art evolutiu, videoart, *new-media art* i art multimèdia, art robòtic i *net art*.

Així que endrecem una mica els termes seguint una taxonomia proposada per Boden i Edmons [Boden i Edmons 2009]. Sobretot fent la distinció inicial entre peces d'art digital que fan servir els ordinadors com a eina per a la creació (*computer assisted art*), que serien aquelles que usen programari informàtic com el Photoshop, Illustrator, After Effects, etc. I les peces d'art digital que usen el codi i els algoritmes com a mètode fonamental i substrat de creació de la peça artística (*computer art*).

El primer terme a definir és el concepte ampli d'art electrònic que cobreix qualsevol obra d'art que usa electrònica, inclosos els ordinadors, per a la producció. Inclou des d'obres amb dispositius analògics simples (sistemes electromecànics, sensors i actuadors, oscil·loscopis, etc.) fins a escultures cinètiques, robots o realitat virtual.

Vegeu les imatges del projecte de Ben Laposky fetes manipulant els tubs de rajos catòdics d'un oscil·loscopi.



Figura 44. Oscil·lacions o abstraccions electròniques de Ben Laposky (Proyecto IDIS)

L'art generatiu produeix peces mitjançant un conjunt de regles. L'artista deixa que un ordinador o un sistema prengui algunes decisions que produeixen la peça. Tot i que, per descomptat, és l'artista qui determina les regles i les decisions que deixa a l'algorisme.

Algunes obres d'art generatiu no usen ordinadors, com per exemple, una peça musical de Haydn i Mozart que usava daus per ordenar trossos de composicions ja definides. O els projectes actuals que es poden seguir a les xarxes sota #generativeart. Com per exemple un projecte del 2021 de Mike Brondbjerg on demanava per Twitter que l'audiència usés un dau i una moneda per determinar certs números que ell feia servir per dibuixar amb paper i bolígraf. La majoria de peces d'art generatiu sí que utilitzen ordinadors i, per tant, cauen dins de l'àmbit de l'art digital i el *computer art*.

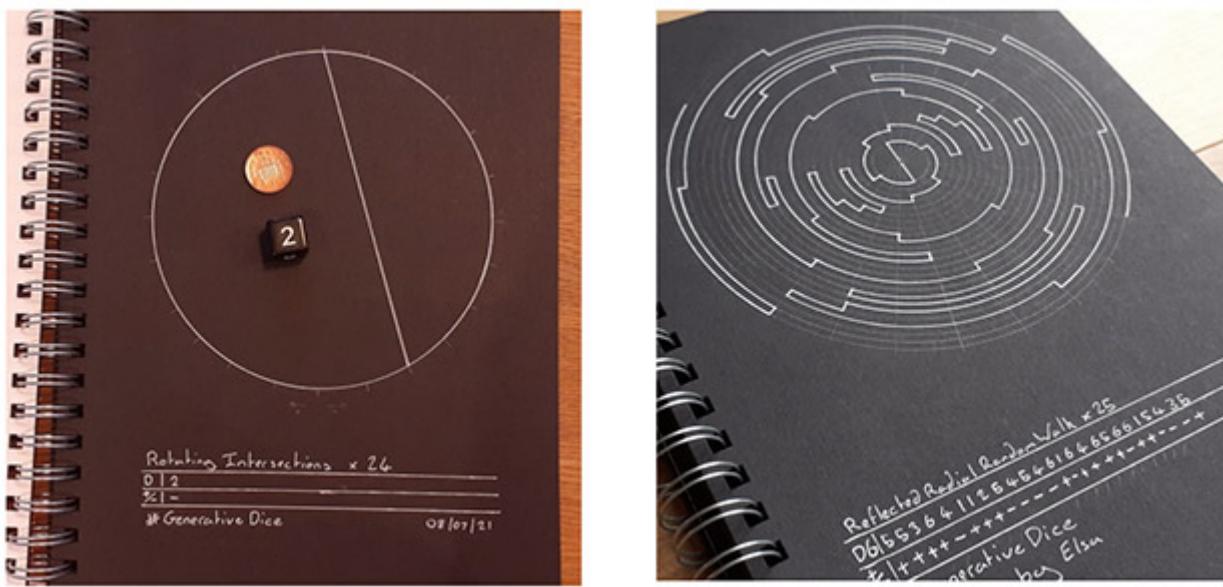


Figura 45. Dos projectes d'art generatiu no digital de Mike Brondbjerg

Diversos algorismes s'utilitzen per crear les peces d'art generatiu que presenten més o menys complexitat en el resultat i que fan emergir patrons a vegades inesperats. Les peces d'art evolutiu, en concret, fan servir algorismes de certa complexitat (algorismes genètics) que muten i evolucionen amb el temps mitjançant processos de variació aleatòria i reproducció selectiva que afecten la peça d'art generada o el programa en sí.

Quan l'obra construeix robots amb finalitats artístiques, o en l'obra hi ha robots o màquines autònomes, parlem d'art robòtic. N'és un exemple l'artista Monica Rikić [Monica] que treballa amb robots i comunitats de robots per especular sobre el nostre futur i conceptes com societat, espiritualitat o frustració.

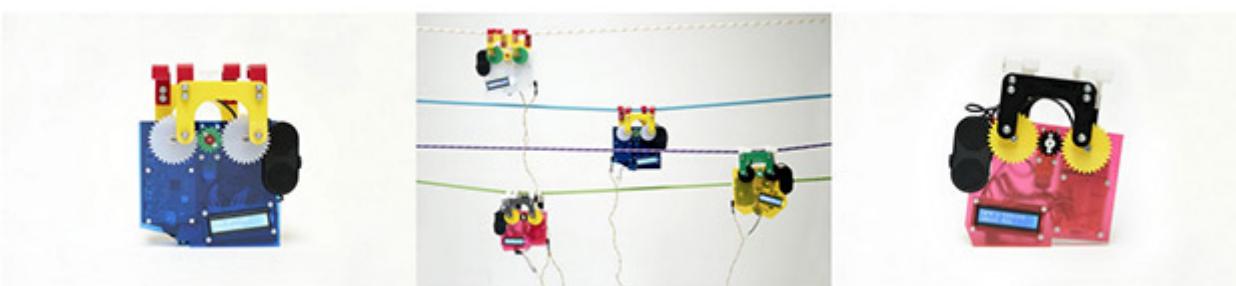


Figura 46. Projecte «Data gossiping robots» de Monica Rikić

L'art interactiu és aquell que es veu afectat substancialment per les accions del públic. I és generatiu ja que el resultat de la interacció es crea, es genera, en temps real a cada instant segons les manipulacions del públic. Dins d'aquest apartat d'art interactiu es fa èmfasi a les peces que es veuen modificades per condicions externes (fins i tot per dades) i es deixa el terme art generatiu per les peces comentades anteriorment.

I finalment detallem les característiques de l'art fet amb realitat virtual, que sovint inclou interacció de l'espectador, on es genera la peça a temps real. En aquest cas, la interacció i la creació dels gràfics busca conduir a la illusió d'un món virtual on l'espectador es troba a dins i que pot experimentar-lo perquè respon com si fos real.

2. Art generatiu

2.2. Art digital

2.2.1. Introducció

Des dels anys noranta fins a principis del segle XXI, el mitjà digital ha estat objecte de desenvolupaments tecnològics de velocitat sense precedents, passant de la revolució digital a l'era dels mitjans i xarxes socials. Amb l'aparició d'ordinadors de sobretaula i de diversos llenguatges de programació, l'ús d'algoritmes ha esdevingut més habitual. Des de l'art programat dels anys seixanta, la multiplicació de llenguatges de programació i l'accés a la xarxa global d'Internet, els mons computacionals han obert el camí a múltiples experiments artístics. Al principi, anys cinquanta, els fruits de la feina i recerca en el mitjà digital es van exhibir principalment en conferències, festivals i simposis dedicats a la tecnologia o als mitjans electrònics i van ser considerats perifèrics al món de l'art principal. Les primeres exposicions dedicades a peces d'art digital engeguen als anys seixanta. I a finals de segle XX i inicis del XXI l'art digital és un terme establert, i museus i galeries fan exposicions i retrospectives de treballs digitals.

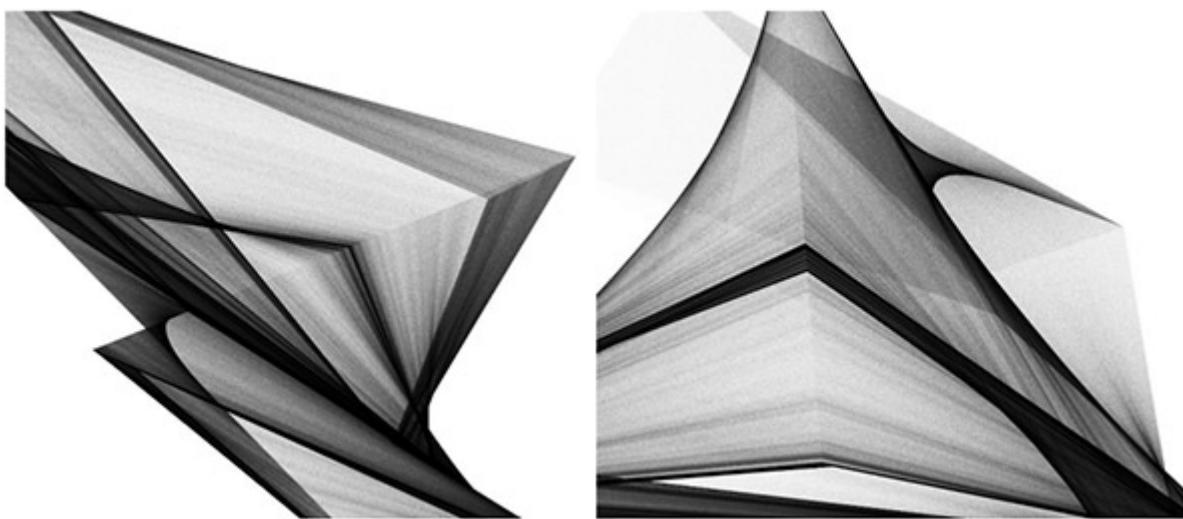


Figura 47. Dues obres gràfiques digitals generatives sense títol d'Anders Hoff (també coneugut com *Inconvergent*)

La terminologia de les formes d'art tecnològic sempre ha estat canviant i el que ara es coneix com a art digital ha patit diversos canvis de nom des que va sorgir per primera vegada. Inicialment es va anomenar art per ordinador (*computer art*), també cibernètica i art multimèdia (l'art digital dels anys 60 i el dels 90 respectivament) per acabar sota el paraigües del *new media art*. Però què té aquest art de nou, de «new»? Que la tecnologia digital ha arribat a una etapa de desenvolupament que ofereix possibilitats completament noves per a la creació i experimentació de l'art. El terme art digital ha esdevingut paraigua per a una gamma molt àmplia d'obres artístiques i no engloba, ni pot encabir, a la pràctica, un conjunt d'estètiques unificades. En aquesta secció fem un repàs històric de l'evolució de l'art digital, els seus orígens, alguns artistes i algunes de les seves característiques.

Però una de les distincions bàsiques i crucials que cal fer és entre l'art que utilitza les tecnologies digitals com a eina per crear objectes d'art tradicionals, com ara la fotografia, el vídeo, la impressió, l'escultura o la música, i l'art digital de fonament, és que es crea, emmagatzema i es distribueix a través de tecnologies digitals; l'art digital que empra les característiques pròpies del mitjà.

Aquestes dues grans categories separades d'art digital són clarament diferents en les seves manifestacions i l'estètica que proposen. S'entenen com una simplificació en dos grans grups de l'amalgama de projectes presents en l'art digital. Ens centrarem en l'art digital que utilitza els trets propis del mitjà.



Figura 48. Quatre generacions de Color Wander de Matt Deslauriers

2. Art generatiu

2.2. Art digital

2.2.2. Precedents de l'art digital

La història de l'art digital s'ha conformat tant per les influències històriques artístiques com per la història de la ciència i la tecnologia. Aquesta darrera va lligada a la computació, a l'ordinador, els llenguatges de programació i les xarxes. En els primers anys inevitablement vinculat al desenvolupament militar, industrial i als centres de recerca. Els ordinadors neixen essencialment en un àmbit acadèmic i entorn d'investigació, i encara avui els centres de recerca tenen un paper important en la producció d'algunes formes d'art digital, com ara l'Ars Electronica Futurelab [Ars Electronica Futurelab] i el Massachusetts Institute of Technology [MIT], entre molts d'altres.

L'art digital té fortes connexions amb moviments artístics anteriors, entre ells les avantguardes del s. xx i l'art conceptual. Moviments com el surrealisme, el constructivisme, l'op art o fluxus. La importància d'aquests moviments per a l'art digital resideix en l'èmfasi que posen en el concepte i les instruccions formals, en l'esdeveniment o en el públic i la seva participació.

Les primeres avantguardes del s. xx fan néixer i créixer l'abstracció on la concepció de la peça d'art va lligada amb l'estruatura. Afegeixen al procés de creació idees com l'atzar o les regles (que són un procés per crear art).

Això té una clara connexió amb l'aleatorietat i els algorismes que formen la base de qualsevol art generatiu. La primera idea, l'aleatorietat, es pot entreveure en algunes de les propostes d'art trobat: obres que usen un objecte trobat o escollit a l'atzar o bé un objecte quotidià (*objet trouvé*) com a element d'elaboració d'una peça artística. Una peça de Picasso (1912), de les primeres obres en introduir aquesta idea.



Figura 49. *Bodegón con asiento de rejilla* de Pablo Picasso

Les regles i la fragmentació, la segona idea, també estan al cor dels cadàvers exquisits dels surrealistes (1925). Joc que consisteix en crear una peça a partir de frases, o dissenys, que van afegint varíes persones, sense que puguin conèixer i tenir en compte les aportacions prèvies.

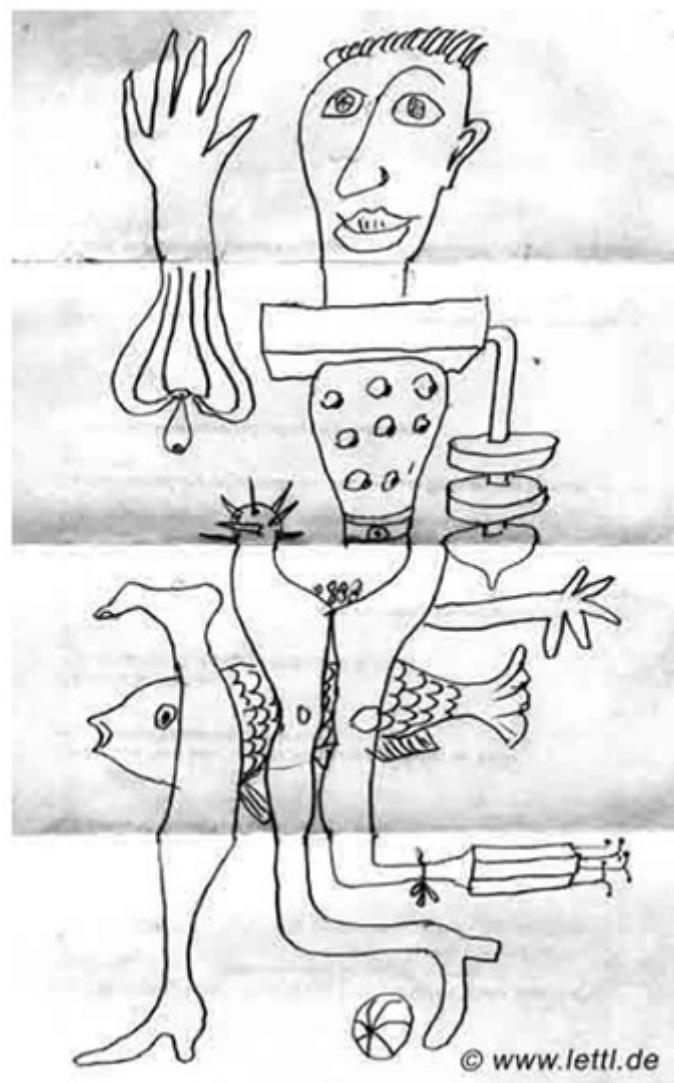


Figura 50. *Cadavre exquis* dels surrealistes



Figura 51. *Cadavre exquis* d'Yves Tanguy, André Breton, Jacqueline Lamba

La poesia dadaista converteix en peça la construcció de poemes de variacions aleatòries de paraules i línies, utilitzant instruccions formals per crear el resultat, barreja d'aleatorietat i control. Nocións com la generació s'exploren ja a les *Rotary glass plates* de Duchamp, creades el 1920 amb Man Ray, que consisteixen en una màquina òptica que convida els espectadors a encendre l'aparell i mantenir-se a una certa distància per veure l'efecte que es genera.

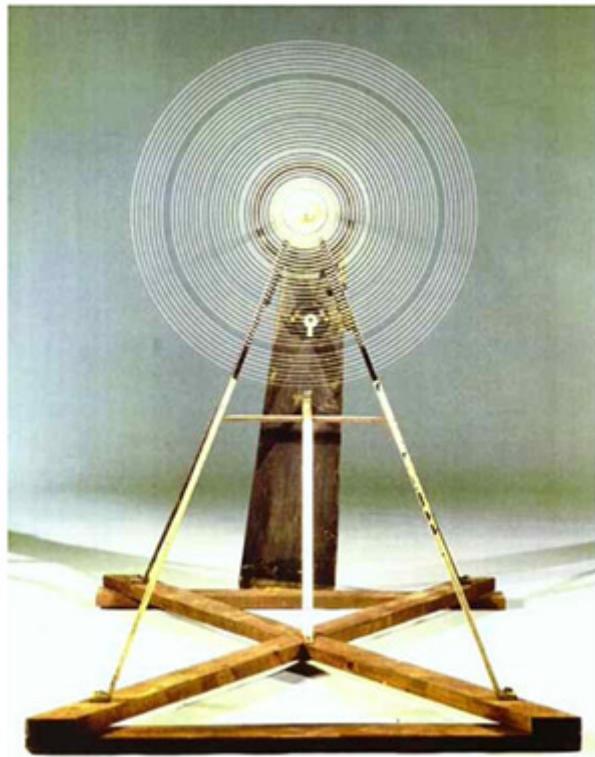


Figura 52. *Rotary glass plates* de Duchamp

De l'interès pel moviment i la geometria de corrents avantguardistes, l'abstracció deriva cap a l'art cinètic i l'op art. Aquestes corrents i exploracions interessades en l'objectivitat d'allò geomètric neixen propostes de peces que estructuren de manera interna el camí o el circuit analògic del que esdevindrà l'obra. Exploracions detallades de paletes de colors i elements que en combinar-se de certa manera esdevenen l'obra final de l'artista. Exploracions que comparteix l'art generatiu en l'ús de probabilitats, aleatorietat, i del formalisme de regles i algorismes per a la creació final de la peça.

L'artista conceptual Sol LeWitt influeix i es considera precursor de l'art generatiu, sobretot pels seus *wall drawings*, vegeu les imatges murals a continuació. Aquests murals són un conjunt d'instruccions (una descripció de text acompañada opcionalment per un diagrama) que descriuen una estructura visual que s'ha d'executar en un mur. L'obra visual no l'executa el mateix LeWitt sinó que és executada, diverses vegades en diferents llocs, per diferents dibuixants que segueixen les seves instruccions.

Cada vegada que es reproduceix un dibuix a la paret és diferent segons el lloc i el dibuixant. Aquest té certa llibertat per adaptar-se a l'espai i prendre decisions quan aquestes no són precises i deixen certa vaguetat.

Hi ha una separació total entre el concepte de l'obra i la seva execució, el seu resultat i la seva manifestació perceptiva. La relació entre LeWitt i el seu dibuixant sovint es compara amb la relació entre un compositor i un intèpret. I, en el nostre cas, podem fer la comparació entre un programador i l'entitat d'execució. LeWitt escriu instruccions (algorismes, programes) per tal que altres persones els interpretin i executin en lloc de per ser executades per màquines o ordinadors.

Detalls sobre la inspiració, el concepte i el procés de dibuix del mural 47, juntament amb informació sobre Sol LeWitt i l'art conceptual poden explorar-se al web del Museu Reina Sofia.



Figura 53. Quatre murals (*wall drawings*) de Sol LeWitt. Per ordre el 340, el 681c amb un tros, a la foto, del mural 414, el mural 792 i el mural 915. Vegeu-ne més a <https://massmoca.org/sol-lewitt> (visitada el 27 de setembre de 2021)

Inspirat per Sol LeWitt, l'artista conceptual d'art generatiu Casey Reas, vegeu la figura 54, va dur a terme el 2004 un projecte per explorar els vincles entre les instruccions i l'estètica generativa del programari, comissariat pel Whitney Museum of American Art. Podeu llegir tot el procés i veure els resultats del projecte {Software} a la pàgina del museu [Whitney 1 Projecte] i [Whitney 2 Text sobre el projecte].

Structure •003

A surface filled with one hundred medium to small circles. Each circle has a different size and direction, but moves at the same slow rate. Display:

- A. The instantaneous intersections of the circles
- B. The aggregate intersections of the circles



Interpretation

A. Tarbell



B. Tarbell



A. Hodgin



B. Hodgin



A. Ngan



B. Ngan



Material

A. FlashMX



B. FlashMX



A. C++

B. C++

Process

01



02



03



04



05



06



07



08



09



10



Figura 54. Captura de pantalla de {Software} Structures, de Casey Reas amb els artistes Jared Tarbell, Robert Hodgin i William Ngan. Versió restaurada d'agost de 2016.

La dècada de 1960 va ser una època important en la història de l'art digital, ja que va ser quan els artistes van començar a experimentar amb ordinadors. En aquest moment, John Whitney va crear el curtmetratge animat *Catalog* (1961) (vegeu la figura 55) generat per un ordinador analògic creat per ell mateix. Va utilitzar funcions matemàtiques per transformar les imatges visuals de la seva peça. També van resorgir els processos combinatoris basats en regles de la poesia dadaista en les obres d'OULIPO (Ouvroir de Littérature Potentielle), l'associació de literatura artística francesa fundada el 1960 per Raymond Queneau i François Le Lionnais, que van argumentar que tota inspiració creativa ha de ser objecte de càlcul i convertir-se en un joc intel·lectual.

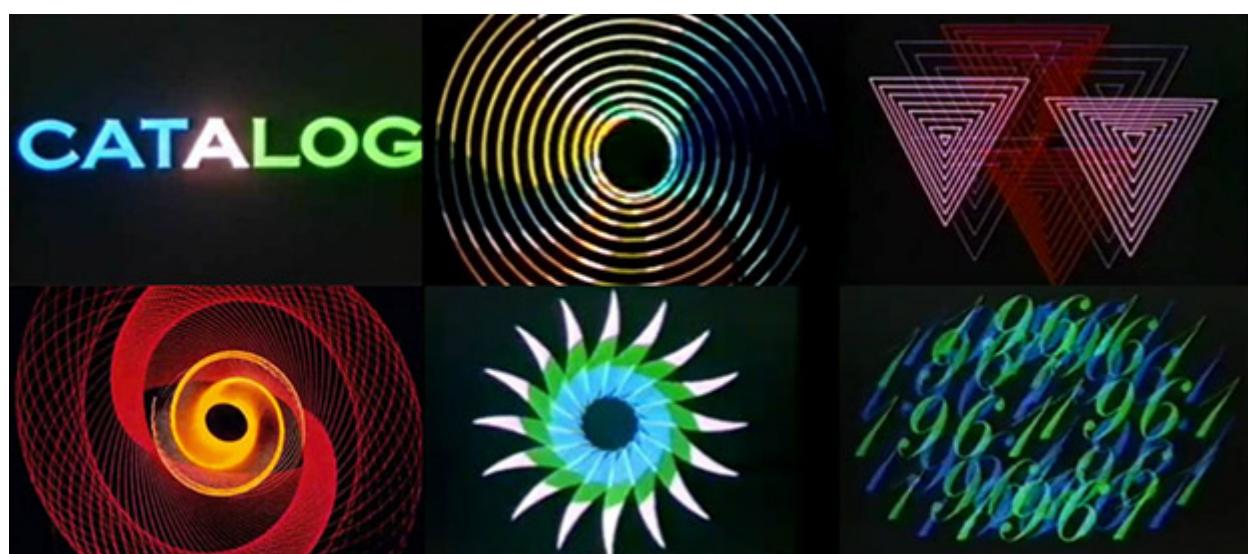


Figura 55. Sis fotogrames de *Catalog* de John Whitney.

<https://www.youtube.com/watch?v=TbV7loKp69s> (visitada el 27 de setembre de 2021)

Els *happenings* del grup d'artistes, músics i performers internacionals fluxus es basaven sovint en l'execució d'instruccions precises. La seva noció de la participació de l'audiència i del moment com a unitat bàsica per construir un esdeveniment va anticipar la naturalesa interactiva d'algunes obres i la idea de basar peces d'ordinador en esdeveniments i instruccions.

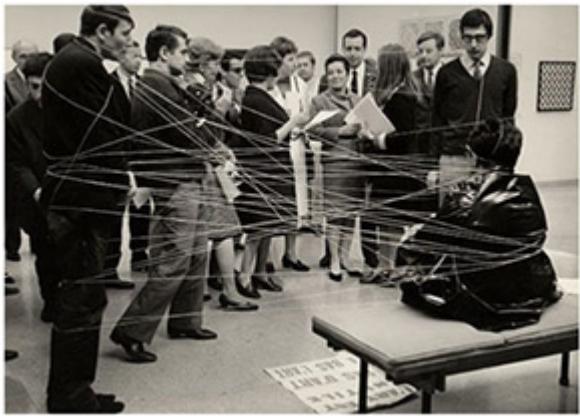


Figura 56. Performances de fluxus

Els conceptes d'element i instruccions en els quals intervé l'atzar també van ser la base de composicions musicals de l'avantguarda, com les del compositor nord-americà John Cage, que va arrencar treballant aquesta idea als anys 50 i 60. Cage va descriure l'estrucció de la música com «its divisibility into successive parts» i sovint omplia les parts predefinides de les seves composicions amb sons preexistents trobats.

2. Art generatiu

2.3. Espais artístics generatius

2.3.1 Literatura

Després de les dues guerres mundials i influenciat per les avantguardes apareixen escriptors, artistes i investigadors que exploren noves narratives i qüestionen el llenguatge natural i l'artificial, experimenten amb lògiques i gramàtiques formals, aleatorietat i combinatòria per produir obres textuais i noves ficcions. L'ordinador personal, la pantalla i Internet permeten noves formes d'entendre i presentar el text: la literatura generativa, els poemes animats o l'hipertext revisen el rol d'autor i lector, [Baillehache, 2013] [Migayrou, 2018].

Els usos contemporanis de l'aleatorietat en la literatura digital són més propers a l'ús dadaista de les operacions d'atzar mecànic (com treure paraules retallades d'una bossa per crear poemes, Tristan Tzara (1920)) que a l'escriptura automàtica surrealista. El 1960 es presenta el poema *I am that I am* de Brion Gysin que experimenta amb les permutacions de paraules d'una sola frase de llenguatge natural per crear la primera proposta literària generada per un algorisme. El 1961, Nanni Balestrini utilitzà un ordinador per recombinar poemes d'altres autors de manera nova i original, generant una font infinita de versos poètics. A partir d'un algorisme inventat es recombinen frases i versos i l'ordinador en va generar una llarga còpia impresa. El poeta selecciona alguns versos que li resulten especialment interessants i que recull a *Tape Mark I*.

El 1960 l'escriptor Raymond Queneau i el matemàtic François Le Lionnais creen OuLiPo (acrònim d'Ouvroir de Littérature Potentielle) un grup d'experimentació literària que basa les seves propostes en regles quasi matemàtiques i restriccions creatives, fugint de l'atzar i d'allò aleatori. Les restriccions oulipianes poden ser un element lingüístic (lletra, paraula, fonema) o bé una equació o un algorisme. Per exemple, la tècnica coneguda com $n + 7$ substitueix tots els substantius d'un text pel substantiu que es troba set entrades més endavant al diccionari. Les propostes d'OuLiPo donen lloc el 1981 al naixement d'ALAMO, Atelier de Littérature Assistée par la Mathématique et les Ordinateurs, que segueix, fins a avui, la mateixa línia de recerca i experimentació literària [ALAMO].

Als Estats Units, a finals dels anys seixanta i principis dels setanta, sorgeix L=A=N=G=U=A=G=E, un moviment de poesia avantguardista que respon a la poesia tradicional nord-americana. Pren el nom del magazín editat per Charles Bernstein i Bruce Andrews i creix a partir de diverses comunitats de poetes a les dues costes dels EUA. Exploraven idees provinents de la Black Mountain School, de l'escola de Nova York o dels Beat Poets de San Francisco. La poesia L=A=N=G=U=A=G=E emfatitza el rol del lector que pot aportar significat al text. Usen recursos literaris per crear textos de diferents textures, amb repeticions i jocs estructurals. Usen recursos lingüístics per inventar noves paraules o estructures gramaticals. Els resultats són difícils d'entendre amb una única lectura. La intenció de L=A=N=G=U=A=G=E és precisament aquesta, la participació activa del lector per crear i donar significat al text i al poema.

Actualment, la creació literària usant intel·ligència artificial (IA) i els bots a Internet són les propostes digitals més importants de la poesia contemporània. Llegiu diversos exemples de bots a [Some Strategies of Bot Poetics] i escolteu recitar pels seus autors varíes propostes de literatura generada amb IA al capítol 4 de la sèrie documental Artificio [Artificio]. Els bots combinen tècniques conceptuales avantguardistes, intervenció eticopolítica i alt potencial expressiu. A més, ho fan en un espai social popular, les xarxes socials, com és el cas dels bots de Twitter.

Bots com @everyword, que tuiteja, una per una, totes les paraules de l'anglès exhaustivament: el contingut d'aquest poema no és només el diccionari sinó també el nombre de *likes* i retuits de cada tuit rebut, donant un significat preferent a determinades paraules; fins a bots com @poem_exe que escriu haikus japonesos [Some Strategies of Bot Poetics]. Els algorismes d'aprenentatge d'intel·ligència artificial permeten a artistes com Nick Montfort, Allison Parrish, David (Jhave), Johnston o Pablo Gervás alimentar amb quantitats enormes de text literari els algorismes i extreure'n propostes generades diverses. Parrish, per exemple, alimenta amb dos textos els seus algorismes per generar obres que barregen un cert percentatge de les característiques literàries de l'una i l'altra creant una peça literària nova. Montfort, al contrari, usa un corpus literari enorme per ensenyar als seus algorismes de *machine learning* (concretament els *generative adversarial network* (GAN)) a construir noves paraules i frases, jugant amb la morfologia i la fonètica. Tot tècniques d'intel·ligència artificial emprades per a crear llibres de poemes.

2. Art generatiu

2.3. Espais artístics generatius

2.3.2. Dansa

A la dècada de 1880, aproximadament cinquanta anys després de la invenció de la fotografia química, les millores en les tecnologies de la càmera fotogràfica (temps d'exposició més curts i temporitzadors elèctrics) van permetre els primers estudis del moviment humà i animal. Les investigacions d'Étienne-Jules Marey i Eadweard Muybridge donen lloc a la cronofoografia. Aquesta tècnica permet una millor comprensió del cos i del moviment i contribueix al naixement del cinema. Alguns cronofoògrafs usaven múltiples exposicions superposades en una sola planxa fotogràfica (cronofotografia plana), mentre que altres utilitzaven diverses càmeres per produir imatges separades (sèrie fotogràfica).

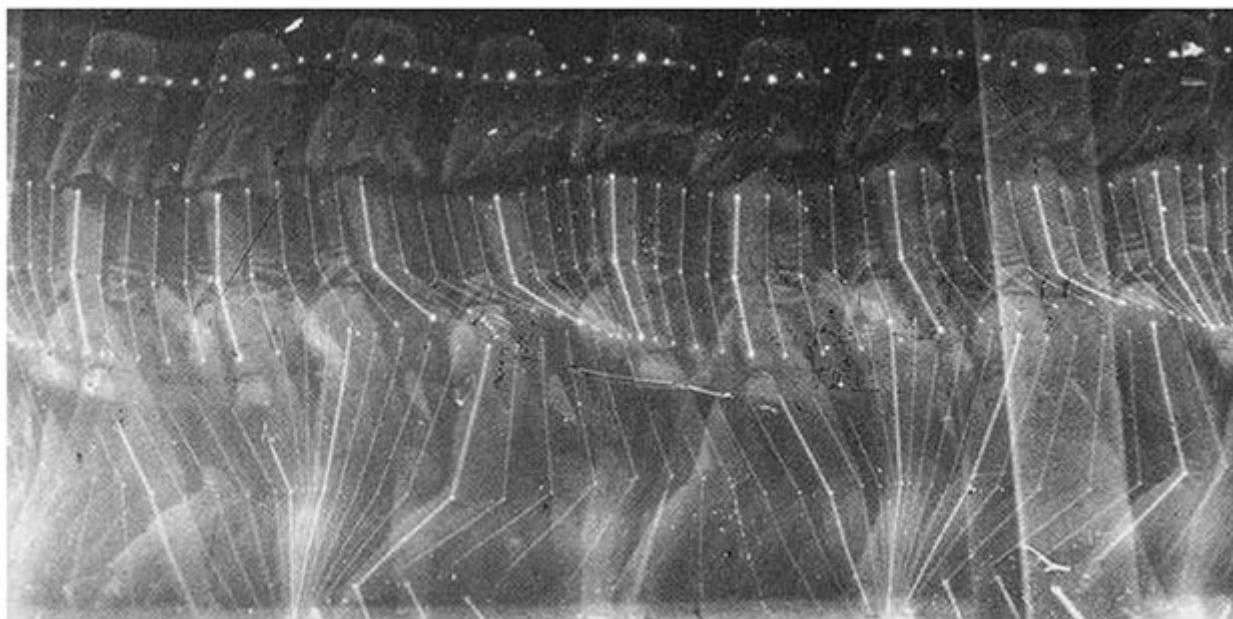


Figura 57. Cronofotografia d'un home caminant d'Étienne Marey (1883)

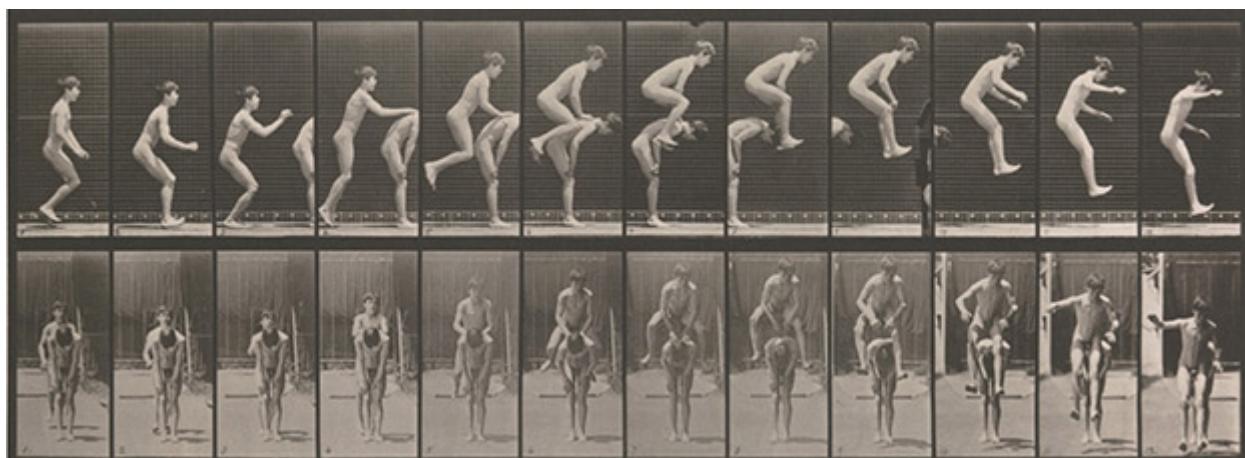


Figura 58. Cronofotografia de nois jugant a saltar el potro d'Eadweard Muybridge (1883-1886)

Cap a 1915, Lillian i Frank Gilbreth desenvolupen una tècnica semblant per dur a terme estudis de moviment, amb l'objectiu de reduir els cicles de treball repetitius dels operaris tot buscant la seqüència de gestos més curta i eficient. El seu cronociclògraf és essencialment una càmera de temps d'exposició llarg que registrallums connectats als cossos humans en moviment.



Figura 59. Cronociclògraf de la mà esquerra d'un operari de premsa de perforació (posicionament després del transport) de Lillian i Frank Gilbreth (1915)

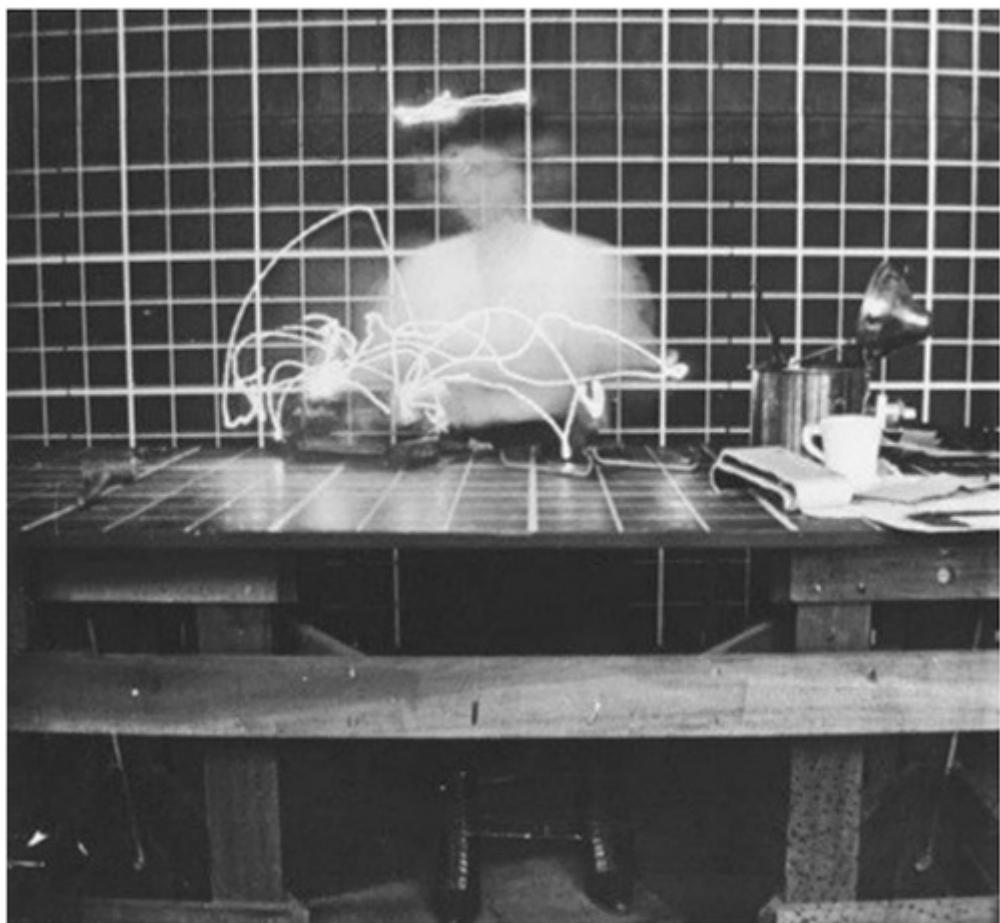


Figura 60. Cronociclògraf per a un estudi d'eficiència de Lillian i Frank Gilbreth (1914)

El 1931, Harold Edgerton inventa la llum estroboscòpica que permet exposicions fotogràfiques breus, de l'ordre dels microsegons, i representa un altre progrés tècnic significatiu en la captura del moviment.



Figura 61. Fotografia usant llum estroboscòpica d'una noia saltant a corda de Harold Edgerton (1952)

La cronofoografia influeix en les idees i llenguatges visuals dels primers moviments artístics modernistes. Els cubistes s'interessen en com la pintura podia desmuntar i representar subjectes dinàmics. Els futuristes s'interessen en com l'art pot mostrar i valorar la velocitat i la cinètica del segle xx. Els artistes i dissenyadors de la Bauhaus alemanya volen comprendre els principis fonamentals dels elements estructurals de la forma, el color i el moviment.

Aquestes tècniques que permeten descompondre el moviment humà en seqüències influencien el teòric de la dansa Rudolf Laban. El 1910, es planteja les relacions harmòniques entre moviment humà i espai i imagina un volum que delimita l'espai del ballarí, la kinesfera, on traça les direccions possibles de moviment. Aquesta recerca dona lloc el 1928 a la *labanotation*, la notació Laban, sistema de notació de dansa i altres disciplines que impliquen moviments corporals.

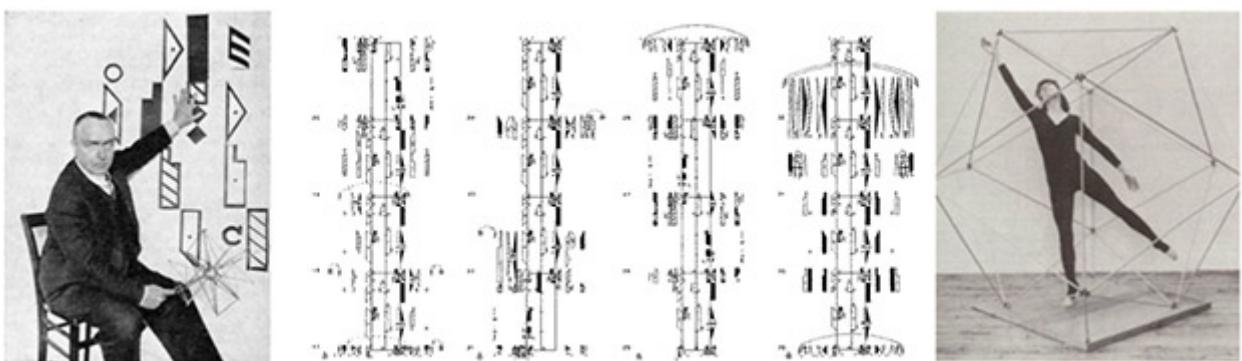


Figura 62. Rudolf Laban mostrant la seva notació; exemple de partitura; i kinesfera

De forma semblant l'escultor, dissenyador i coreògraf Oskar Schlemmer, de la Bauhaus, transforma els ballarins en formes geomètriques com si fossin escultures, vegeu-ho en la seva *performance Stelzenläufer* (Slat Dance) que crea el 1927.

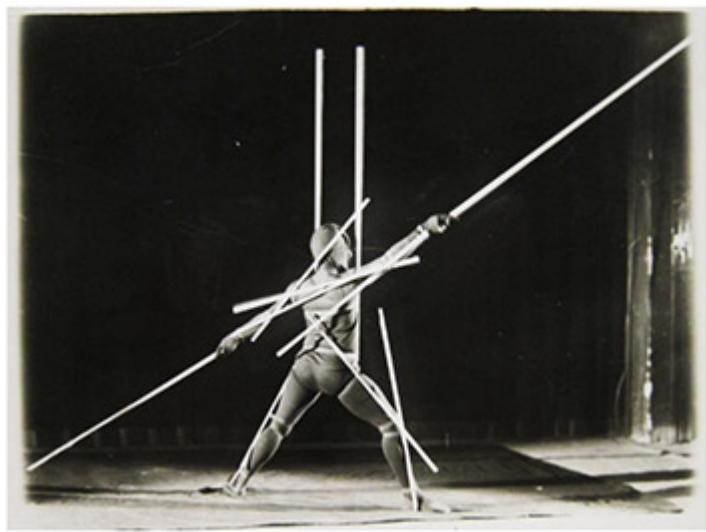


Figura 63. *Stelzenläufer* d'Oskar Schlemmer (1927)

La introducció de la tecnologia i la cibernètica en escena apareix als anys cinquanta. Nicolas Schöffer crea el 1956 la primera escultura cinètica, anomenada *CYSP 1* que fa servir un cervell electrònic connectat a sensors que permeten a l'escultura respondre a canvis del so, la intensitat de la llum, el color i el moviment, inclòs el del públic.



Figura 64. Escultura cinètica *CYSP 1* de Nicolas Schöffer (1956)

El coreògraf Maurice Béjart crea un espectacle amb l'escultura interactuant amb els ballarins en escena.

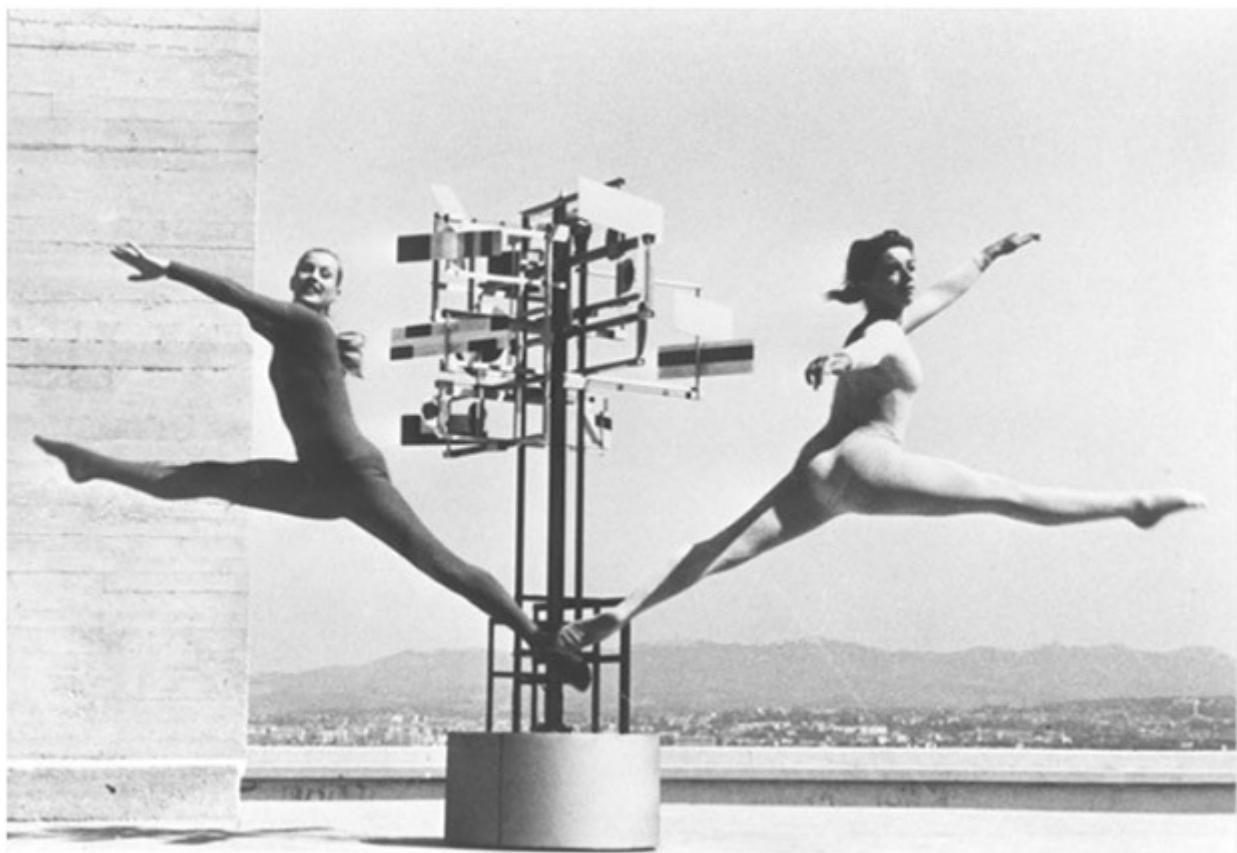


Figura 65. Ballet Béjart actuant al Festival d'art d'avantguarda de Marsella (9 d'agost de 1956)

El mateix any, Merce Cunningham crea la performance *Variations V* on els moviments dels ballarins a l'escenari s'envien a dotze antenes construïdes per Robert Moog i a un conjunt de fotocèl·lules dissenyades pel científic de Bell Labs Billy Klüver. La transmissió de so i d'informació s'envia a un mesclador de 50 canals, on John Cage i David Tudor mesclen a temps real el so rebut amb sons de ràdios, i crea el directe que se sent per sis altaveus al voltant de la sala. La posada en escena es complementa per un *collage* de pel·lícules de Stan VanDerBeek que inclou imatges de televisió processades per Nam June Paik i imatges dels ballarins de VanDerBeek durant els assajos.



Figura 66. Performance *Variations V* de Merce Cunningham (1956)

Per altra banda, l'ordinador ofereix possibilitats noves en la notació coreogràfica i la visualització de les coreografies. El 1964 la ballarina Jeanne H. Beaman, conjuntament amb Paul Le Vasseur i Dale Isner del Computer and Data Processing Center de la Universitat de Pittsburgh, desenvolupen el programa *Random Dances*, capaç de compondre seqüències de dansa per ser interpretades. El programa tria aleatoriament entre llistes de moviments, variacions rítmiques i direccions de l'espai; i crea fins a 70 composicions de dansa diferents que s'imprimeixen per ser interpretades per ballarins. Michael Noll, el 1965 crea *Computer-Generated Ballet* un film d'animació de tres minuts que mostra una coreografia generada aleatoriament per ordinador amb sis ballarins representats esquemàticament.

Entre el 1973 i el 1976 la ballarina, coreògrafa i videoartista Analívia Cordeiro presenta a televisió les seves coreografies per ordinador, les primeres de les quals titula *M 3x3*. L'ordinador genera instruccions per a la càmera i seqüències de passos aleatoris de ball per als ballarins.



Figura 67. M 3x3 vídeo en blanc i negre d'Analívia Cordeiro (1973)

John Lansdown, pioner dels gràfics per ordinador, reflexiona parallelament sobre els ordinadors i la dansa. Analitza l'art generatiu contemporani, les propostes probabilístiques (que incorporen aleatorietat a les decisions creatives) i les deterministes (que es basen en regles de tipus gramatical). Relacionant-ho amb la notació Benesh, una alternativa a la notació Laban, el 1978 proposa una simplificació visual de la figura humana per crear imatges clau de posicionament. La coreografia s'articula a partir d'aquests keyframes i els ballarins tenen la llibertat de crear els moviments intermedis.

El 1986, amb la generalització i més fàcil accés als ordinadors, apareix LifeForms un programa d'animació 3D que permet crear coreografies i veure-les des de 3 punts de vista diferents. Merce Cunningham crea diverses peces usant aquest programari que li permet coreografiar gestos que no veia i creia impossibles.

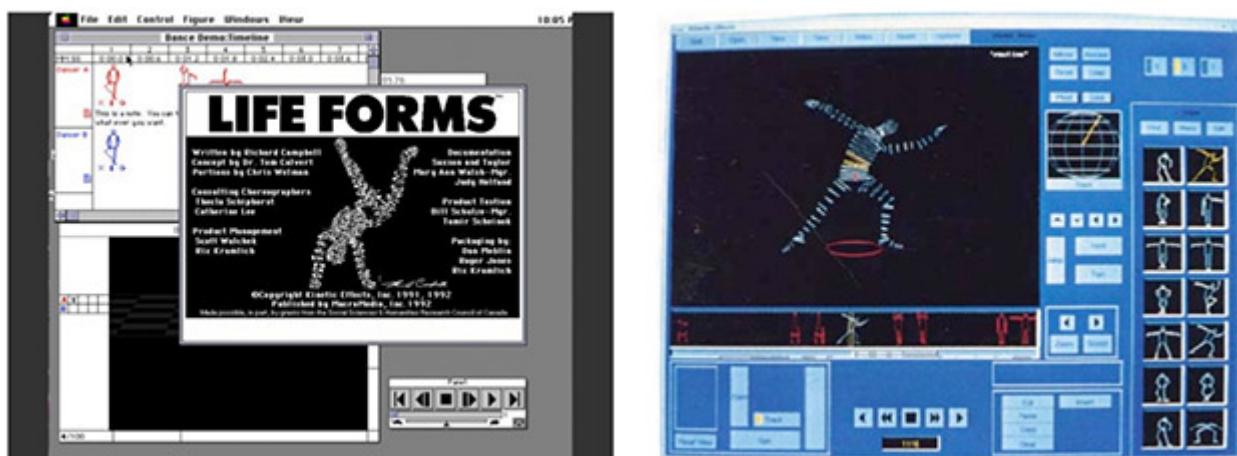


Figura 68. Programari LifeForms

Als anys 90, el coreògraf nord-americà William Forsythe proposa desconstruir el vocabulari clàssic de la dansa per renovar-lo. Usa mètodes d'improvisació coreogràfica i processos generatius barrejant computació i vídeo. El 1999 publica el CD-ROM *Improvisation Technologies: A Tool for the Analytical Dance Eye* on detalla la seva visió de la coreografia com a pràctica organitzativa. Mostra vídeos curts que expliquen moviments fonamentals del llenguatge de la dansa. Els vídeos són il·lustrats amb línies blanques en 2D que s'han rotoscopiat a mà sobre vídeo.



Figura 69. «Point Point Line» i «Dropping Curves» de William Forsythe dins *Improvisation Technologies*

La dansa ha emprat la tecnologia per entendre i analitzar el moviment, amb diferents tècniques de *motion capture*. Alguns projectes fan servir algorismes, aleatorietat i regles per cocrear coreografies. La tecnologia, fins i tot pot augmentar els ballarins en escena, barrejant-se amb l'audiovisual i l'àudio a temps real, o mesclar els ballarins amb avatars virtuals o remots, com fan

artistes contemporanis com Daito Manabe. Algunes idees, estratègies i projectes que usen algorismes vinculen aquestes pràctiques de dansa amb les idees de l'art generatiu i han influenciat l'estat contemporani d'aquesta pràctica.

2. Art generatiu

2.3. Espais artístics generatius

2.3.3. Gràfic: Pioners de l'art digital generatiu

Cap als anys seixanta, artistes, enginyers, matemàtics i curiosos de totes les disciplines van ser els pioners en experimentar les possibilitats de la computació digital en el camp artístic, vegeu [Media Art Net] i el [Digital Art Museum].

Els primers ordinadors de centres de càlcul i universitats es van utilitzar per a la creació d'obres artístiques. Moltes d'aquestes màquines eren de difícil accés i estaven en laboratoris. Un dels grups pioners neix als laboratoris Bell de Nova Jersey, on Béla Julesz i Michael Noll van usar alguns dels primers equips informàtics per generar imatges que es van exhibir el 1965, del 6 al 24 d'abril, com a part de l'exposició *Computer-Generated Pictures* a la galeria Howard Wise de Nova York (vegeu les obres de Julesz com a exemples del treball dels artistes).

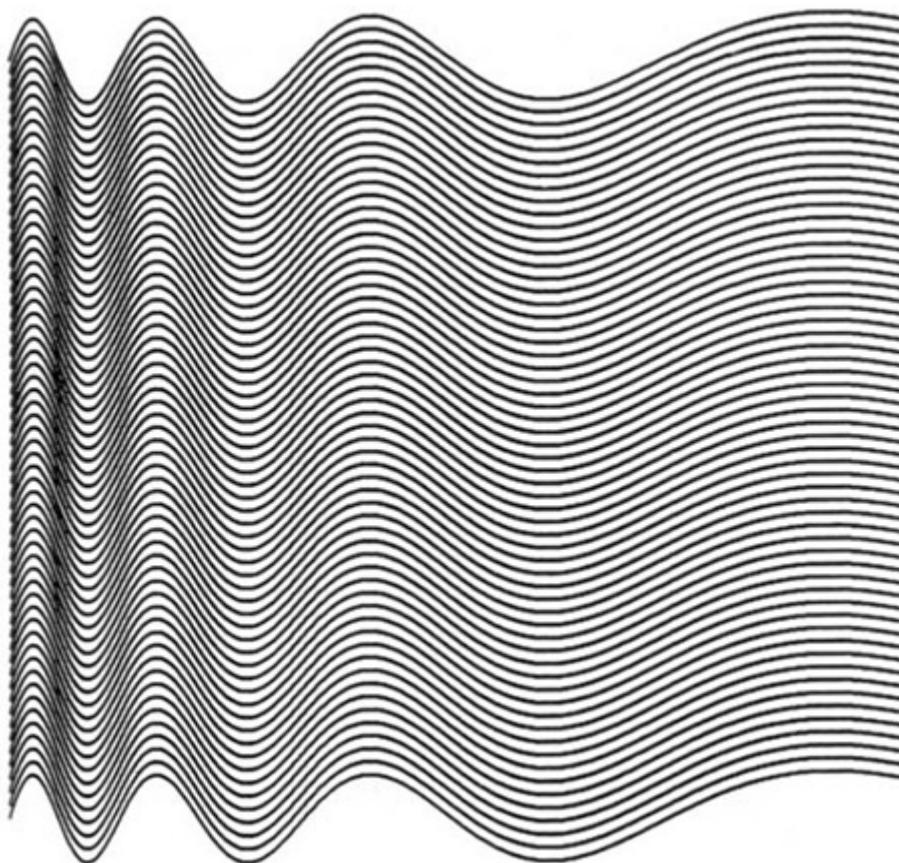


Figura 70. *Ninety Parallel Sinusoids With Linearly Increasing Period* de Michael Noll (1960)

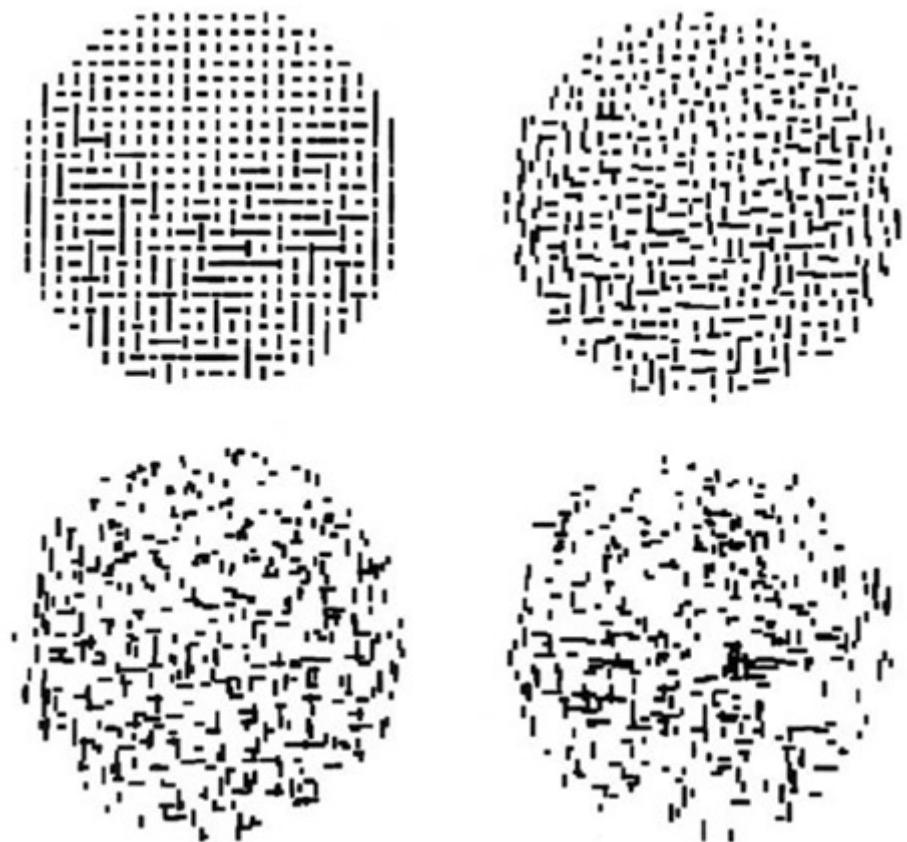


Figura 71. Four computer-generated random patterns based on the composition criteria of Mondrian's *Composition With Lines* de Michael Noll (1965)

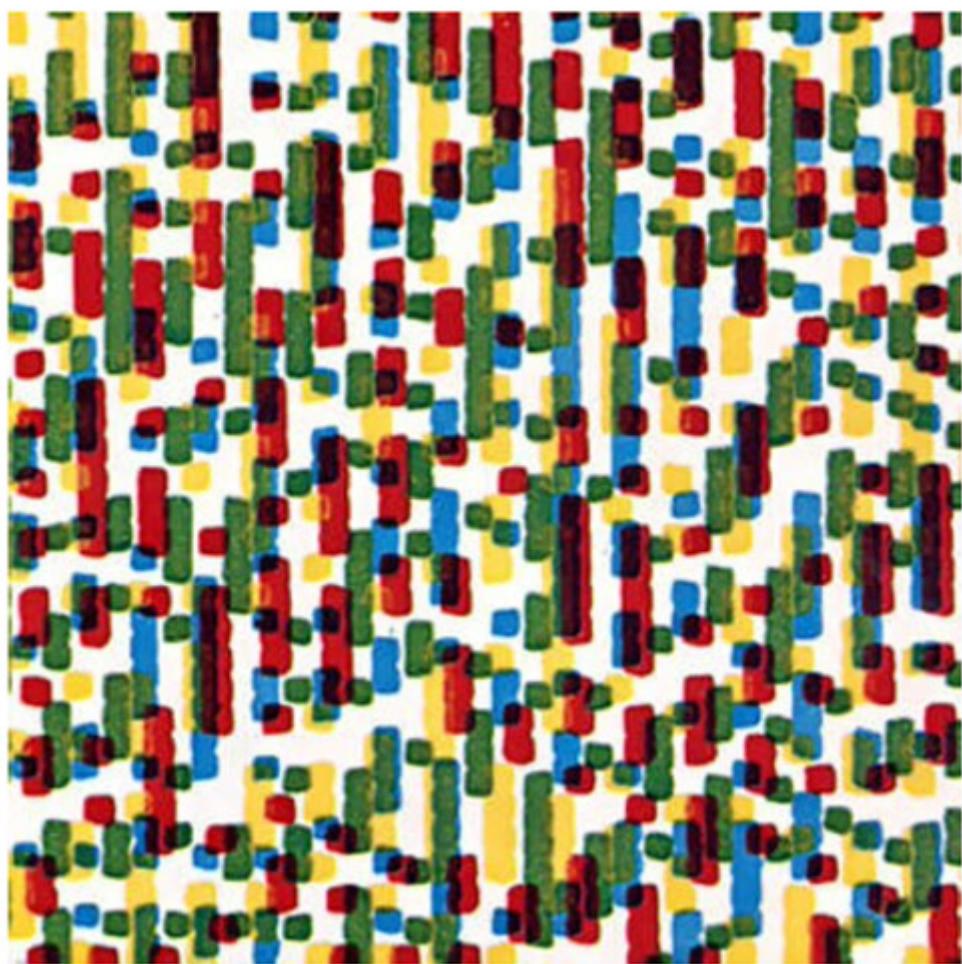


Figura 72. 3D julesz de Béla Julesz (1960)

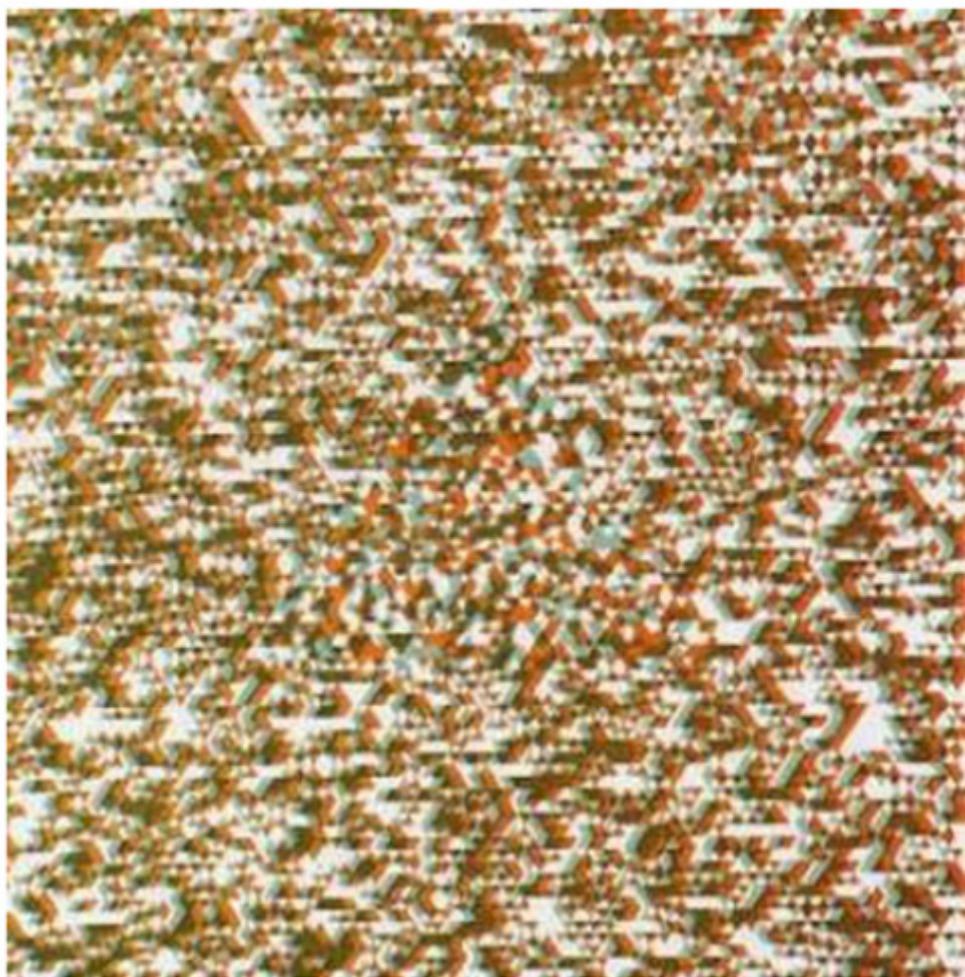


Figura 73. *Anaglyph* de Béla Julesz (1960)

Les obres de John Whitney, Charles Csuri, Vera Molnár i Manfred Mohr, realitzades als anys 60 també continuen sent influents per les seves investigacions sobre les transformacions generades per ordinador a través de funcions matemàtiques (vegeu les imatges de Molnár i de Mohr). Whitney utilitzava equips informàtics militars analògics antics i Csuri va crear les seves primeres imatges digitals el 1964 amb un ordinador IBM 7094.

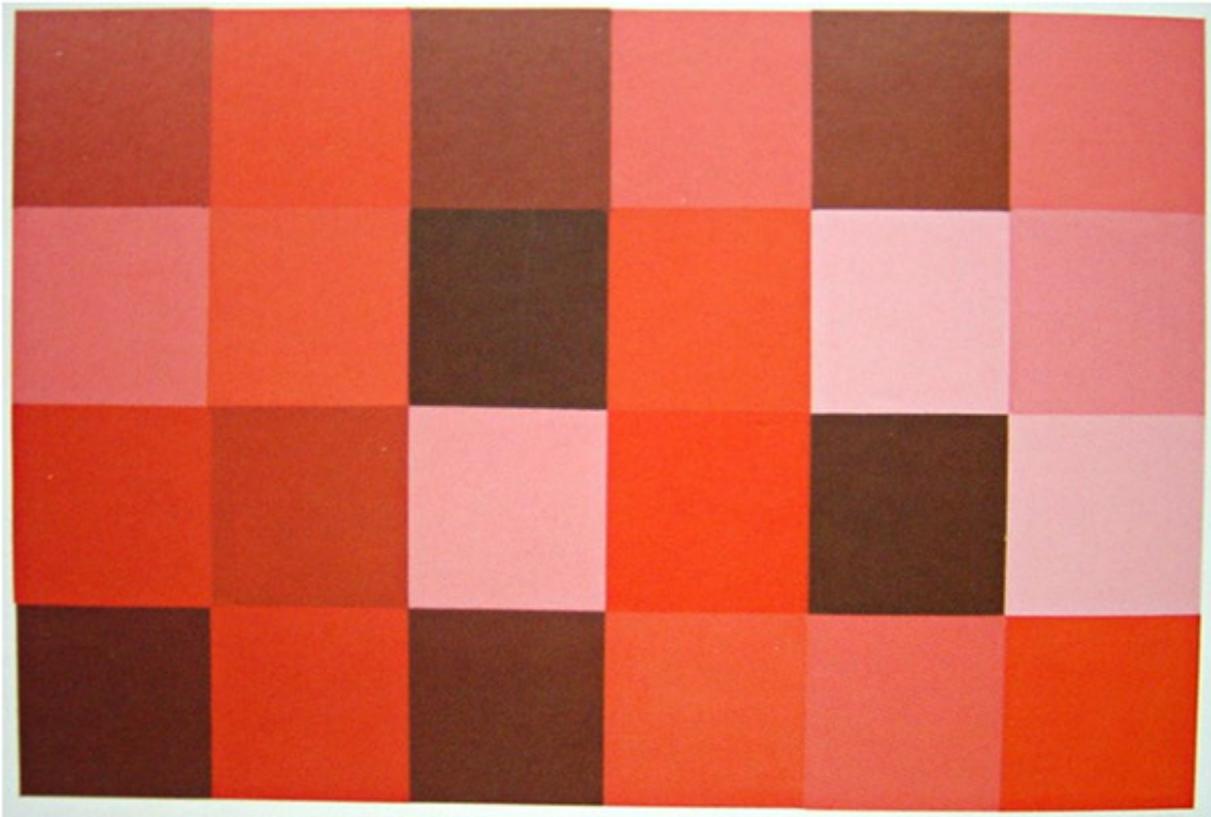


Figura 74. *Rouge au hasard* de Vera Molnár (1961)



Figura 75. *Structure de quadrilatères* de Vera Molnár (1988)

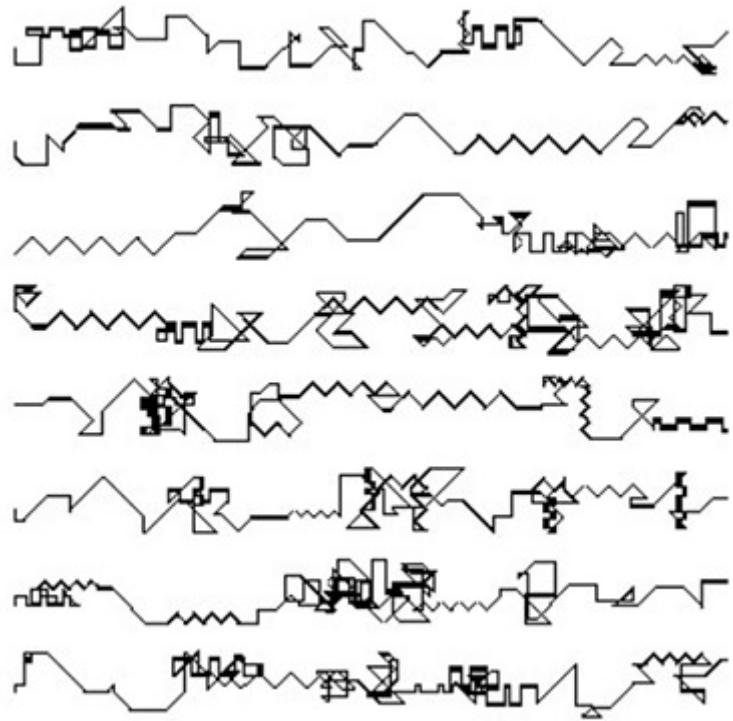


Figura 76. P-021/A + B band-structure de Manfred Mohr (1969)

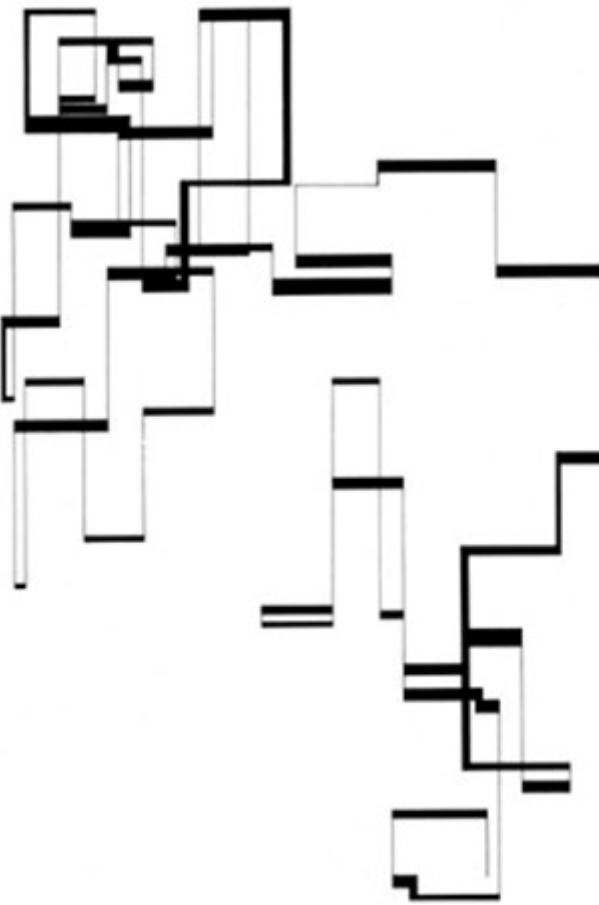


Figura 77. P-10 random walk de Manfred Mohr (1969)

A Europa, als anys 60, diversos artistes exploren el potencial plàstic de la informàtica. Per iniciativa de Max Bense (en les idees de qui molts s'inspiren [compArt]), del 5 al 19 de febrer de 1965, es presenta a la Hochschule für Technik d'Stuttgart l'exposició *Generative Computer Graphics* amb peces de Georg Nees. Aquest artista, que es troba entre els primers del mitjà, també exhibirà del 5 al 26 de desembre a la Galerie Niedlich d'Stuttgart, juntament amb Frieder Nake, dins l'exposició *Computer Graphics*. Les obres d'ambos semblen dibuixos abstractes i formes replicades, molt familiars als mitjans tradicionals, però capten l'estètica

essencial del suport digital i esbossen les funcions matemàtiques bàsiques que impulsen qualsevol procés de dibuix digital (vegeu les obres de Nake i de Nees).

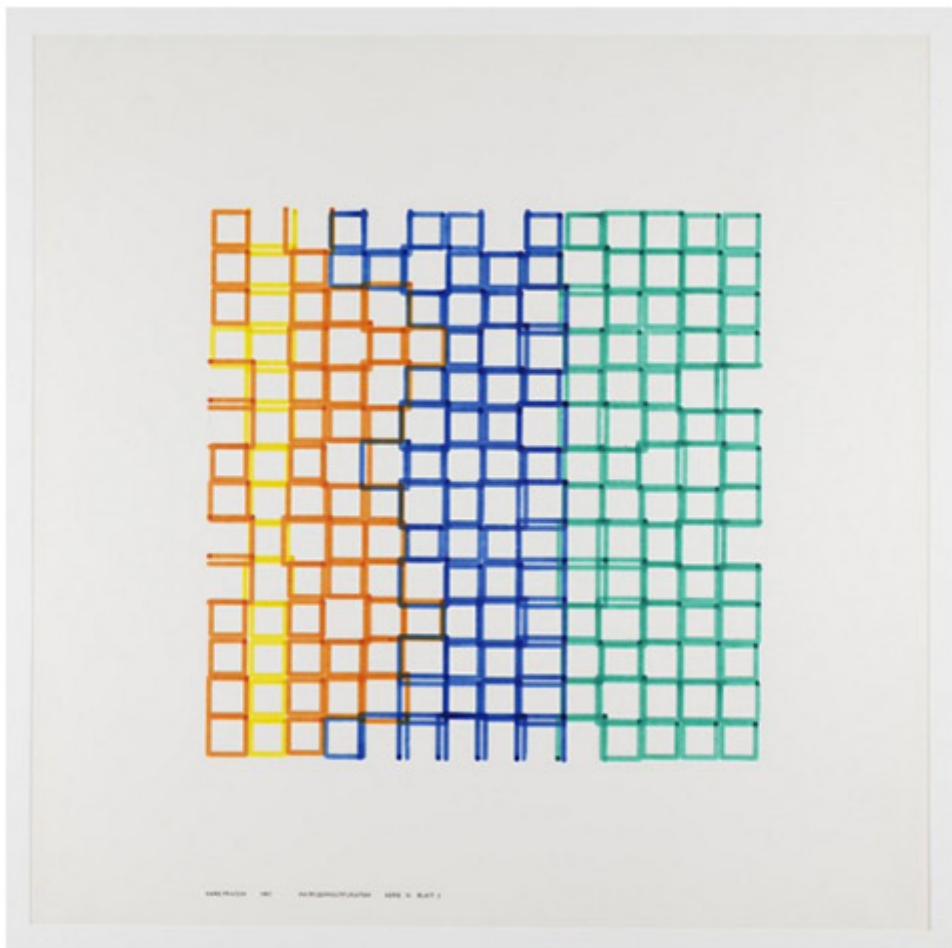


Figura 78. Obra sense títol de Frieder Nake (1967)



Figura 79. Detall de *Polygonzüge Matrizenmultiplikation Serie 31* de Frieder Nake (1965)

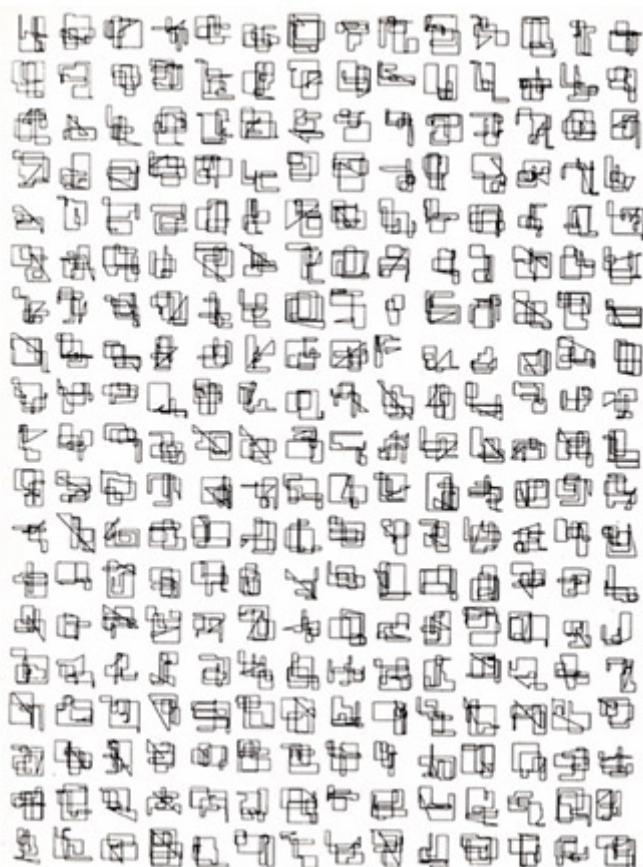


Figura 80. Obra sense títol (gràfics basats en 23 punts) de Georg Nees (1965)



Figura 81. *Schotter* de Georg Nees (1960)

Les arts dels mitjans electrònics no es van considerar un fenomen aïllat, sinó que es van incorporar a la història i al discurs de les belles arts i arts performatives. A Zagreb, del 1961 al 1973, la sèrie de cinc exposicions que cristal·litzen a l'entorn de Nove Tendències busca acompanyar la transició de l'ordinador com a mitjà de creació artística. Es qüestiona l'estètica realista socialista i els organitzadors estableixen treballs generats per ordinador en relació al constructivisme, l'art cinètic (1968/69) i l'art conceptual (1973).

Als Estats Units, l'exposició *The Responsive Eye* organitzada al MoMA el 1965 [Responsive Eye al MoMA] amb la col·laboració de la galeria Denise René legitima l'op art i proposa una alternativa al pop art.

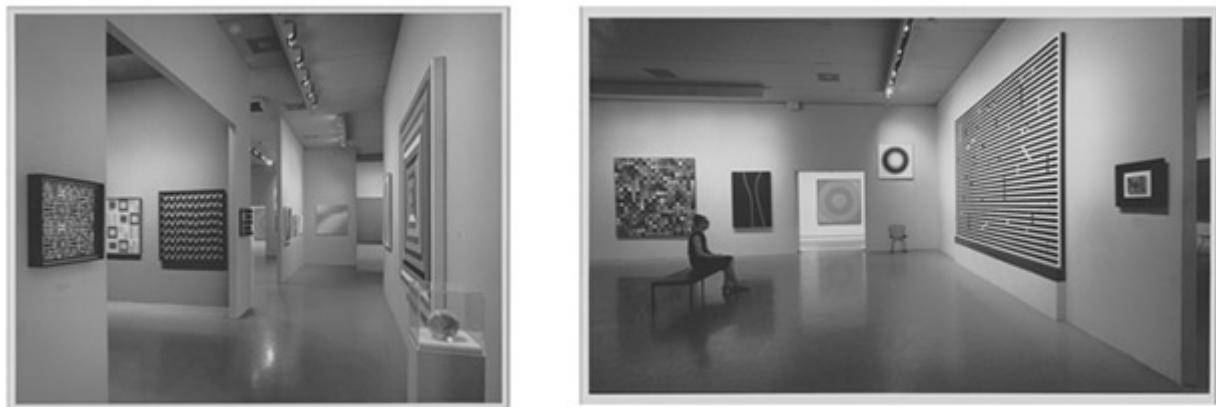


Figura 82. Exposició *The Responsive Eye*, MoMA de Nova York (1965)

La noció d'art programat, creix també amb l'exposició *Arte programmata* concebuda el 1962 per Bruno Munari i Giorgio Soavi. Umberto Eco fa el prefaci del catàleg on explica l'aposta per la creació digital, un ordre nou de creació estètica que neix amb els algorítmes, l'expressió lògica d'una idea estètica.

El 1968 l'exposició *Cybernetic Serendipity* comissariada per Jasia Reichardt a l'Institut d'Arts Contemporànies de Londres, del 2 d'agost al 20 d'octubre, va presentar treballs gràfics traçats amb plòter, entorns de llum i so amb detecció o robots. Peces que van anticipar moltes de les característiques importants del mitjà digital actual. Algunes obres es van centrar en l'estètica de màquines i transformacions, màquines automatitzades per pintar o generadors de patrons i de poesia. Altres propostes eren dinàmiques i orientades als processos, explorant les possibilitats de la interacció i de sistemes oberts com a objecte artístic [Paul 2015].

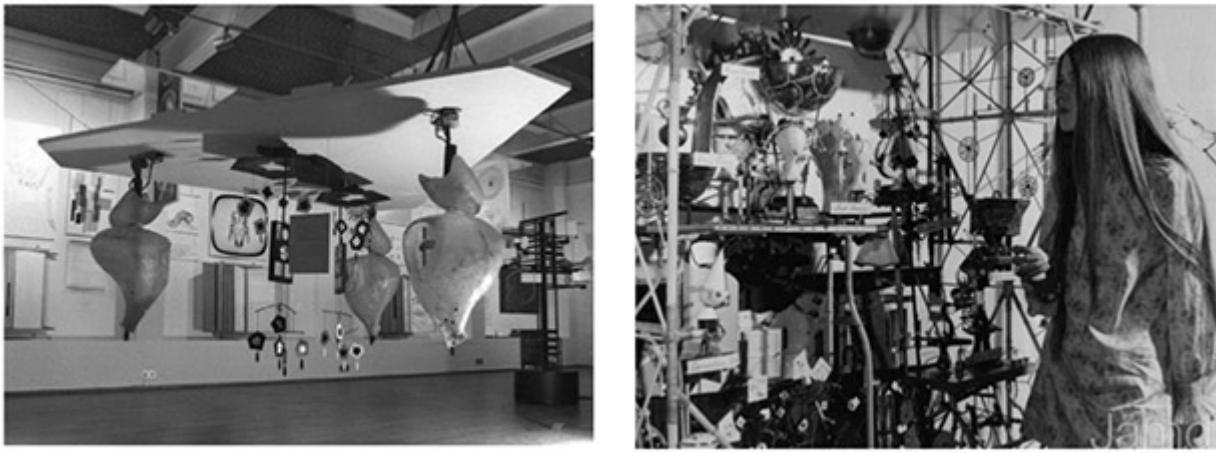


Figura 83. Exposició *Cybernetic Serendipity*, Institut d'Arts Contemporànies de Londres (1968)

El 1966 es crea el Centro de Cálculo de la Universidad de Madrid (CCUM) com a resultat d'un acord entre la Universitat i IBM [Castaños, 2000]. El centre crea beques de recerca, cursos de formació en programació i ofereix l'equipament a tot investigador que vulgui emprar les computadores per a finalitats no habituals d'enginyeria o matemàtica. El Centro de Cálculo crea el 1968 el Seminario de Generación Automática de Formas Plásticas destinat a qüestions purament plàstiques centrat en la recerca de les bases matemàtiques de l'art. Impulsen i formen part del seminari destacats artistes com Manuel Barbadillo i Eusebio Sempere, entre molts d'altres.

The Recode Project és un arxiu de treballs publicats entre 1976 i 1978 a la revista *Computer Graphics and Art*. El web [The Recode Project] recull els 11 volums d'aquesta revista que conté el pseudocodi dels algoritmes juntament amb imatges de diverses peces d'art. El web recull també el nou codi que algun membre de la comunitat creativa ha desenvolupat o adaptat per imitar la peça original.

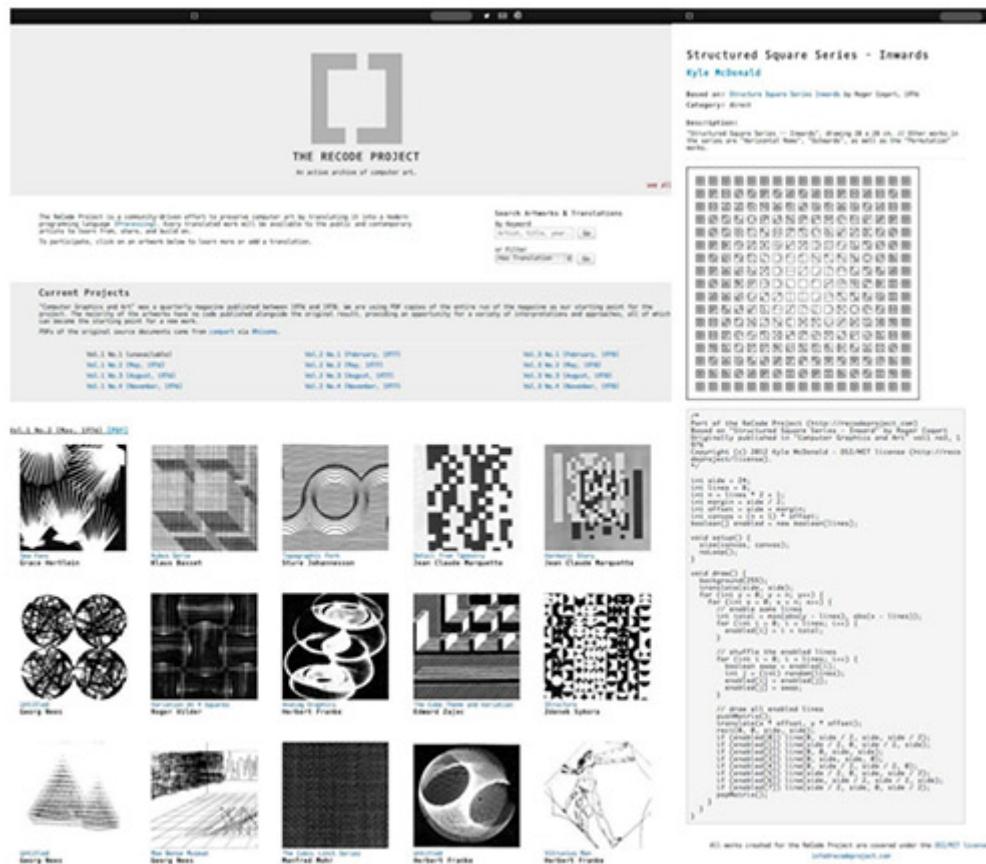


Figura 84. Web del projecte The Recode Project

“ En paraules del propi projecte «The ReCode Project is a community-driven effort to preserve computer art by translating it into a modern programming language Processing».

El projecte recull 171 peces dels pioners de l'art generatiu, de les quals 70 han estat reescrites a codi actual.

2. Art generatiu

2.4. Artistes generatius destacats

Finalment, us proposem una llista d'artistes d'art digital per perdre-us explorant les seves obres i codi.

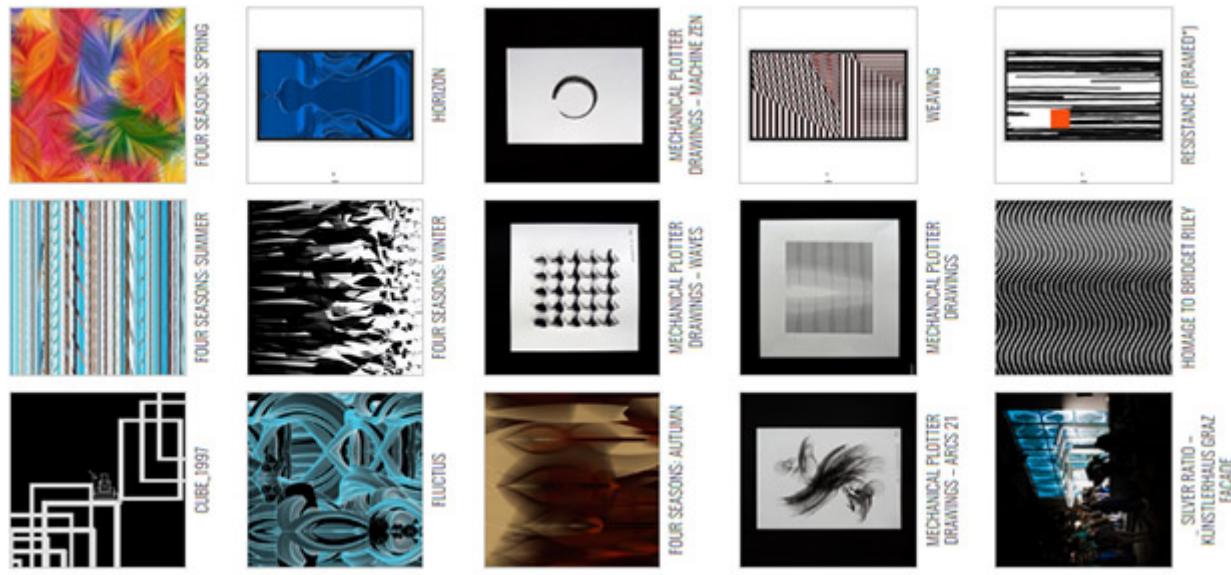


Figura 85. Web del projecte The Recode Project

Manolo ide: <http://manoloide.com/>

The dot is black: <https://thedotisblack.com/>

Ravenkwok: <http://ravenkwok.com/>

Okazz: <https://openprocessing.org/user/128718>

Hamoid: <https://hamoid.com/code/>

Casey Reas: <https://reas.com/>

Jared Tarbell: <http://www.complexification.net/gallery/>

Lia: <https://www.liaworks.com/category/theprojects/>

wks: <https://openprocessing.org/user/193247>

Marius Watz: <http://mariuswatz.com/>

Alba G. Corral: <https://blog.albagcorral.com/>

Sergio Albiac: <https://www.sergioalbiac.com/>

Generate Me: <https://generateme.tumblr.com/>

2. Art generatiu

Bibliografia

Baillehache, J. (2013) «Chance Operations and Randomizers in Avantgarde and Electronic Poetry: Tying Media to Language». *Textual Cultures*, vol. 8, nº 1, pp. 38-56 Indiana University Press.

Boden, M. i Edmonds, E. (2009). «What is generative art?». *Digital Creativity*, 20 (vol. 1-2) pp 21-46. ISSN 14626268.

Castaños, E. (2000) *Los orígenes del arte cibernetico en España: el seminario de Generación Automática de Formas Plásticas del Centro de Cálculo de la Universidad de Madrid: (1968-1973)*. Biblioteca Virtual Miguel de Cervantes. ISBN: 84-688-4599-X.

Galanter, P. (2003). *What is Generative Art? Complexity theory as a context for art theory. Proceedings of the International Conference on Generative Art*. Milà, Itàlia.

Migayrou F. (2018). *Coder le Monde*. Editions HYX. ISBN 978-2-37382-011-9.

Paul C. (2015) *Digital Art*. Ed Thames and Hudson Ltd, UK. ISBN 0500204233.

Enllaços

ALAMO. Atelier de Littérature Assistée par la Mathématique et les Ordinateurs www.alamo.free.fr (visitada el 27 de setembre del 2021).

Ars Electronica FutureLab. <https://ars.electronica.art/futurelab/en> (visitada el 27 de setembre del 2021).

Artificio. <https://vimeo.com/369541372> (visitada el 27 de setembre del 2021).

CompArt Projekte generativer Ästhetik. (The projects of generative aesthetics) <http://dada.compart-bremen.de/item/publication/339> (visitada el 27 de setembre del 2021).

Digital Art Museum. <https://dam.org/museum> (visitada el 27 de setembre del 2021).

Giles H. Some Strategies of Bot Poetics <https://harrygiles.org/2016/04/06/some-strategies-of-bot-poetics/amp> (visitada el 27 de setembre del 2021).

Massachusetts Institute of Technology. www.mit.edu (visitada el 27 de setembre del 2021).

Media Art Net. www.medienkunstnetz.de/mediaartnet (visitada el 27 de setembre del 2021).

Monica Rikić. <https://monicarikic.com> (visitada el 27 de setembre del 2021).

Museo Reina Sofia. <https://www.museoreinasofia.es/en/featured-artwork/sol-lewitt> (visitada el 27 de setembre del 2021).

Proyecto IDIS. Ben Laposky. <https://proyectoidis.org/ben-laposky> (visitada el 27 de setembre del 2021).

The Recode Project. <http://recodeproject.com> (visitada el 27 de setembre del 2021).

The Responsive Eye. The Museum of Modern Art. February 23 – April 25, 1965 www.moma.org/calendar/exhibitions/2914 (visitada el 27 de setembre del 2021).

Whitney Museum of American Art. Projecte {Software Structures} <https://whitney.org/exhibitions/software-structures> (visitada el 27 de setembre del 2021).

Whitney Museum of American Art. Text sobre el projecte {Software Structures} <https://artport.whitney.org/commissions/softwarestructures2016/text.html> (visitada el 27 de setembre del 2021).

3. Joc i videojoc

3.1. Introducció

La relació entre el videojoc i la programació creativa, i amb l'art digital en general, ha estat sempre molt estreta. D'entrada, perquè com que comparteixen eines creatives –tant codi com motors de joc, entorns, etc.– els encontres es produueixen de manera natural. Però la relació es mostra també tant a nivell formatiu, on el joc sempre és present en algun punt quan s'aprèn a programar, com a nivell expressiu, en moltes obres que s'inspiren en el joc, en beuen o el disseccionen.

La bibliografia al voltant del joc i del videojoc és immensa, i no és pas l'objectiu aquí cobrir el tema en tota la seva amplitud, sinó anar directament a tractar alguns dels aspectes centrals del joc i el videojoc, i la seva relació amb la programació creativa i l'art digital en general. A partir d'aquí, podeu seguir explorant tant la bibliografia com altres recursos per ampliar els vostres coneixements.

Tanmateix, si voleu una molt bona i àmplia introducció als videojocs, podeu consultar el recurs d'aprenentatge UOC «Introducció als videojocs», on trobareu una explicació molt detallada de tots els aspectes rellevants amb relació als videojocs, des de la seva història a un recorregut per diferents gèneres, estratègies de disseny, etc.



Figura 86. Un fotograma de l'aventura gràfica *The Day of The Tentacle*, LucasArts, 1993.
Font: https://en.wikipedia.org/wiki/File:Day_of_the_Tentacle_Founding_Fathers.jpg

3. Joc i videojoc

3.2. Joc i videojoc: aspectes bàsics

3.2.1. Què és un joc?

Hi ha idees tan transversals i complexes, com la del joc (i per extensió el videojoc), que fan inevitable començar qualsevol discurs sense intentar definir i acotar mínimament de què s'està parlant exactament. Sense pretensió de trobar la definició perfecta, és molt útil revisar-ne algunes de rellevants, per així començar a endreçar les idees i posar el focus on pertoca.

En el seu clàssic *Homo Ludens*, publicat originalment el 1938, Johan Huizinga comença afirmant, ni més ni menys, que el joc és més antic que la cultura mateixa. La raó és que la cultura pressuposa sempre una societat humana, mentre que el joc pertany també al regne animal.

Més endavant, encara a l'inici d'un llibre que aprofundeix en diversos aspectes del joc i la seva relació amb la cultura, n'ofereix la definició següent:

«El joc és una activitat voluntària, que es desenvolupa dins uns límits prefixats d'espai i de temps, segons unes regles totalment vinculants però acceptades lliurement, que té l'objectiu en si mateix i que va acompanyat d'un sentiment d'emoció i alegria, així com de la consciència de d'estar en quelcom diferent respecte a la vida ordinària.»

Huizinga, 1938

Text original: «*Spiel ist eine freiwillige Handlung oder Beschäftigung, die innerhalb gewisser festgesetzter Grenzen von Zeit und Raum nach freiwillig angenommenen, aber unbedingt bindenden Regeln verrichtet wird, ihr Ziel in sich selber hat und begleitet wird von einem Gefühl der Spannung und Freude und einem Bewusstsein des ‚Andersseins‘ als das, gewöhnliche Leben‘.*»

Huizinga, 1956



Figura 87. El tres en ratlla és un dels jocs més conegut arreu del món, més senzill d'entendre, i més flexible quant a la versatilitat dels materials amb què es pot jugar.

Font: https://en.wikipedia.org/wiki/File:Tic_tac_toe.svg

Per tant, el joc és fonamentalment una activitat que es fa de manera voluntària, on hom es dona unes normes i es deixa portar per la sensació d'alteritat. Però fins i tot en una definició en una obra tan central com la de Huizinga s'hi pot deixar de mencionar algun aspecte, per la qual cosa val la pena revisar més definicions. Per exemple:

“ «Un joc és un sistema en què els participants entren en un conflicte artificial, definit per normes, i amb un resultat quantificable.»
Katie Salen i Eric Zimmerman

Text original: «*A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome.*»
Katie Salen i Eric Zimmerman

«Un joc és un tipus d'activitat lúdica, desenvolupada en el context d'una realitat simulada, en què els participants intenten assolir almenys un objectiu arbitrari i no trivial, d'acord amb una sèrie de regles.»
Ernest Adams

Text original: «*A game is a type of play activity, conducted in the context of a pretended reality, in which the participants try to achieve at least one arbitrary, nontrivial goal by acting in accordance with rules.*»
Ernest Adams

«Un joc és un sistema basat en regles amb un resultat variable i quantificable al qual s'assigna un valor, en què els jugadors s'esforcen per influir-hi, el jugador se sent emocionalment lligat al resultat, i les conseqüències de l'activitat són negociables.»
Jesper Juul

Text original: «*A game is a rule-based system with a variable and quantifiable outcome, where different outcomes are assigned different values, the player exerts effort in order to influence the outcome, the player feels emotionally attached to the outcome, and the consequences of the activity are negotiable.*»
Jesper Juul

«[Un joc és] un sistema tancat i formal en què els jugadors entren en un conflicte estructurat que resol la seva incertesa en una resultat desigual.»
Tracy Fullerton

Text original: «*A closed, formal system that engages players in structured conflict and resolves its uncertainty in an unequal outcome.*»
Tracy Fullerton

Més enllà de les dificultats de traducció, hi ha certs conceptes que apareixen a moltes de les definicions, com ara **sistema formal, conflicte, resultat, quantificació...** I només un que apareix a tot arreu: **les regles**.

3. Joc i videojoc

3.2. Joc i videojoc: aspectes bàsics

3.2.2. Regles i mecàniques

Dos dels aspectes clau a tenir en compte quan volem dissenyar un joc, i per extensió un videojoc, són les regles i les mecàniques. Les regles són allò que ens diu què es pot fer i què no al joc. Les mecàniques inclouen les regles, però engloben també, a més del què, el com es pot fer.

Huizinga, explica la importància de les regles d'aquesta manera:

«Cada joc té les seves pròpies regles. Determinen allò que és vàlid dins el món temporal que ha creat el joc. Les regles són absolutament vinculants i no admeten cap dubte. Paul Valéry ho va dir de passada, en una idea molt important: no hi ha escepticisme possible respecte a les regles del joc. Al cap i a la fi, la base que el determina està donada de manera inamovible. Tan bon punt es violen les regles, el joc s'enfonsa. I llavors el joc s'ha acabat. El xiulet de l'àrbitre desfà l'encaixeri i restableix el món ordinari immediatament.»

Text original: «*Jedes Spiel hat einen eigenen Regeln. Sie bestimmen, was innerhalb der zeitweiligen Welt, die es herausgetrennt hat, gelten soll. Die Regeln eines Spiels sind unbedingt binden und dulden keinen Zweifel. Paul Valéry hat es einmal beiläufig gesagt, und es ist ein Gedanke von ungemeiner Tragweite: Gegenüber den Regeln eines Spiels ist kein Skeptizismus möglich. Ist doch die Grundlage, die sie bestimmt, unerschütterlich gegeben. Sobald die Regeln übertreten werden, stürzt die Spielwelt zusammen. Dann ist es aus mit dem Spiel. Die Pfeife des Schiedsrichters hebt den Bann auf und setzt die 'gewöhnliche Welt' für einen Augenblick wieder in Gang.*»

Huizinga, 1956



Figura 88. Els escacs són un altre joc d'abast global, amb molts anys d'antiguitat.

Font: <https://ca.wikipedia.org/wiki/Fitxer:Checkmate.jpg>

Les regles, doncs, delimiten el joc. Estableixen què s'hi admet i què no. Qui i on pot tocar la pilota amb les mans, què passa si la meva fitxa va a parar a la casella on n'hi havia una altra, etc. Les regles han de ser clares i inequívoces, i conegeudes per tots els jugadors. El joc només és possible si tothom les segueix.

A més, a banda d'aquestes regles explícites, també hi ha les implícites, que caurien dins l'àmbit social on es desenvolupa el joc, i que no necessàriament tenen una delimitació tan clara. Així, al joc d'escacs el respecte a l'adversari és sempre molt present, i en un joc de taula s'espera que tothom es comporti amb educació i estigui pendent del joc des de l'inici fins a la fi. Però en altres àmbits, especialment en esports competitius com el futbol o el tennis, la línia entre què s'accepta i què no respecte a les regles implícites –per exemple, el comportament dels jugadors durant els partits– pot ser molt més fina.

La mecànica del joc en defineix el desenvolupament. Engloba les regles, les dades i els processos, i marca les condicions que determinen quan es guanya o es perd. Més enllà de les regles, doncs, la mecànica del joc és el que defineix com es juga. Què cal fer per avançar, i què és possible fer i què no en cada moment. Si un joc es juga sobre un tauler o sobre una superfície física concreta, si és per torns o és continu, si s'avança segons el resultat de tirar els daus, per exemple, són exemples de mecàniques de joc tradicionals. En el cas de videojocs, un entorn 2D o 3D, un joc de plataformes o de món obert, si el moviment és lliure en tots els eixos o s'avança només cap a la dreita, quins moviments pot fer i quins no l'avatar, etc.

3. Joc i videojoc

3.2. Joc i videojoc: aspectes bàsics

3.2.3. Altres aspectes rellevants

Així doncs, les regles i la mecànica del joc són dos dels aspectes centrals del disseny de qualsevol joc, i cal tenir-los presents des del primer moment. Tanmateix, hi ha molts altres aspectes que el defineixen. Aquí en llistem alguns, i us convidem a pensar en alguns exemples, tant de joc com de videojoc, per tal de veure com s'hi presenta cadascun. De nou, no és una llista exhaustiva ni tots els seus elements són necessàriament rellevants per a tots els jocs, però sí que pretén posar sobre la taula elements importants a tenir en compte de cara al disseny i el desenvolupament de qualsevol joc, juntament amb les regles i mecàniques.

Els definirem en tres nivells, com cercles concèntrics de més a prop a més allunyats del nucli del joc, si més no des del punt de vista del dissenyador, que és el que ens interessa aquí. Val la pena remarcar, també, que són aspectes molt interrelacionats els uns amb els altres, tant dins dels nivells definits aquí com entre els de diferents nivells.



Figura 89. Jugadors amb el sistema PlayStation 4 VR.

Font: [https://en.wikipedia.org/wiki/File:Gamescom_Playstation_VR_Playseat_\(36454815300\).jpg](https://en.wikipedia.org/wiki/File:Gamescom_Playstation_VR_Playseat_(36454815300).jpg)

Primer nivell:

- **Regles.** Com hem explicat, delimiten què es pot fer i què no.
- **Mecànica.** Com s'ha dit, engloben les regles i delimiten el com i amb què es pot avançar.
- **Base lògica.** Tant si és un tauler de 8 x 8 caselles, un camp rectangular de 105 x 68 metres, o un nivell de plataformes en 2D per a una pantalla de mòbil, o un joc de cartes, la base lògica del joc és l'espai, físic o virtual, on aquest es desenvolupa. Aquesta base també inclou si s'hi juga amb peces, amb una pilota, amb un avatar i enemics, etc., més enllà dels materials o l'aspecte que tinguin aquests.
- **Objectiu.** Defineix què cal fer per guanyar.
- **Jugadors.** Per a quants jugadors és, i què fan aquests jugadors.

Segon nivell:

- **Materials.** Amb què es juga? Quins materials, quina mena de camp, o de taulell. En el cas digital, podríem substituir aquesta consideració pel suport: PC, mòbil, consola, ulleres de RV...
- **Narrativa.** Quin és el relat del joc? Té una història? Si és així, quina és? Des de la història gairebé cinematogràfica de *The Last of Us* fins al context de batalla dels escacs, la narrativa sol ser un element definidor del joc.
- **Estètica / Game art.** Quin aspecte tenen els elements del joc? Aquest és un altre element definidor, no necessàriament del joc mateix des del punt de vista lògic, però sí acompanyant la narrativa i marcant-ne el to i fins i tot el públic objectiu. També en aquest aspecte, no és el mateix dissenyar un joc infantil –colors, dibuixos, etc.– que un joc de terror per a adults.
- **Context.** On es juga? Tot i que mai ho podem anticipar per tots els casos, sí que és rellevant pensar el context on es desenvoluparà el joc. Entorn domèstic o entorn social, joc de taula o d'equips, en xarxa, etc. definiran com anticipem que s'hi jugarà.



Figura 90. Jugadors d'escacs al port de Kotka, Finlàndia, 1958.

Font:

[https://en.wikipedia.org/wiki/File:Kulutusosuuskuntien_Keskusliiton_kokoelma_D1974_11194AxxH_\(30776867142\).jpg](https://en.wikipedia.org/wiki/File:Kulutusosuuskuntien_Keskusliiton_kokoelma_D1974_11194AxxH_(30776867142).jpg)

Tercer nivell:

- **Habilitat/destresa.** Quines habilitats haurà de tenir o de desenvolupar el jugador per tal de poder guanyar? No en tots els jocs és rellevant.
- **Estratègia.** Quines accions i maneres de jugar m'ajudaran a aconseguir l'objectiu per guanyar el joc? Des de coses molt senzilles com decidir quina fitxa del parxís moure, a anticipar diversos moviments d'escacs, aquest acostuma a ser un dels elements clau en el desenvolupament de qualsevol joc.
- **Metajoc.** El metajoc té a veure amb les dinàmiques que es creen al voltant del joc. Un joc en xarxa en què es competeix per equips crearà una dinàmica social al seu voltant molt diferent que un joc en què competeixen dos jugadors, un contra l'altre, o encara més d'un joc per a un únic jugador. De la mateixa manera, en alguns jocs de taula cada jugador fa la seva, però en altres es poden crear complicitats per perjudicar el contrincant o ajudar-se mútuament.
- **Context cultural.** El context cultural on es desenvoluparà el joc és també un aspecte rellevant. Hi ha jocs que podem argumentar que són multiculturals, tant conceptualment com pels materials i espais que requereixen per desenvolupar-se, però d'altres poden estar més circumscrits a un context concret, ja sigui per requeriments tecnològics, per temàtica o per l'estètica emprada en el joc.



Figura 91. El comandament de la NES va marcar l'inici d'una línia de comandaments de consola que encara segueix viva avui dia.

Font: <https://en.wikipedia.org/wiki/File:Nintendo-Entertainment-System-NES-Controller-FL.jpg>

En el cas específic del **videojoc**, també cal tenir en compte:

- **Control.** Quins mecanismes de control donem al jugador? Un joc on controlarem un cotxe de carreres oferirà experiències molt diferents si el control es fa girant el mòbil, amb el teclat de l'ordinador, o bé amb el comandament DualShock de la Playstation. Un bon exemple d'això és el temps que van tardar els videojocs a adaptar-se als *smartphones*, en el sentit de trobar formes específiques de joc amb la pantalla tàctil, més enllà dels primers exemples que bàsicament imitaven els botons d'un *joystick*.
- **Avatar.** Quina és i què fa la representació que tindrà el jugador a dins del joc? És una representació en primera o en tercera persona? Cerquem la identificació de l'usuari, o és un avatar abstracte? Aquestes i moltes altres qüestions són molt rellevants per la seva influència en l'experiència com a jugadors.
- **Agents autònoms.** Els agents autònoms són normalment els *enemics* del joc. Tenen les seves mecàniques internes pròpies, i una importància cabdal en el disseny del joc.



Figura 92. Els videojocs de carreres són un dels gèneres més populars, amb múltiples versions per a tots els suports.
Font: https://ca.wikipedia.org/wiki/Fitxer:Supertuxkart_0.7.png

3. Joc i videojoc

3.3. Disseny iteratiu

Tots aquests aspectes del joc, que com hem dit estan fortament interrelacionats entre ells, s'han de tenir en compte a l'hora de dissenyar qualsevol joc o videojoc. Òbviament, segons la proposta donarem més pes a uns o altres, ja que en uns jocs la narrativa i l'estètica poden ser l'element central, mentre en d'altres ho és el tauler de joc i els materials amb què es juga.

Sigui com sigui, el procés creatiu de creació d'un joc rarament és lineal, sinó que el més habitual és seguir un procés de disseny iteratiu. És a dir, partint d'una idea inicial, es fa un primer prototip del joc, es posa a prova, i d'aquesta prova s'extreuen conclusions de millora que s'aplicaran al prototip següent. Aquest procés, que busca anar introduint millores i anar refinant el joc –regles, mecàniques i qualsevol dels elements comentats anteriorment– fins a obtenir el producte final.

El cicle, doncs, comença amb la planificació, el disseny i la creació del prototip, que després passa per una fase de proves que seran successivament avaluades per decidir quines modificacions s'implementen al prototip següent.

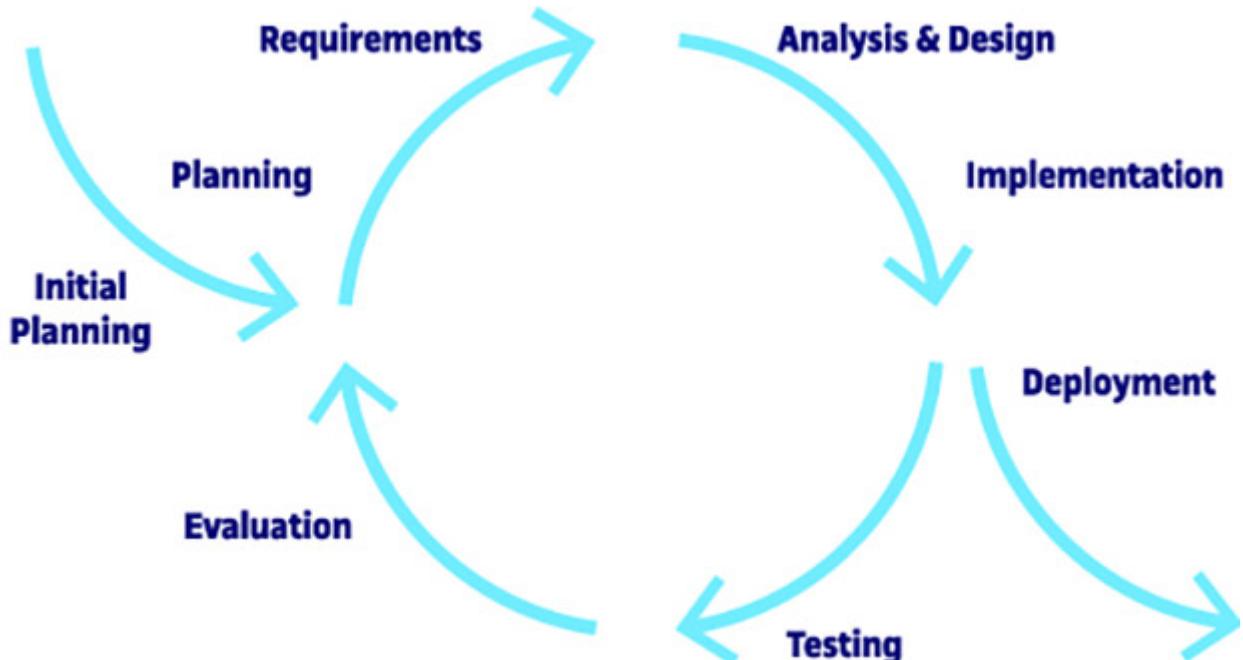


Figura 93. El disseny iteratiu consisteix en la repetició del cicle de disseny, prova ivaluació, fins que s'obté el producte final.

Font: Design Toolkit. Iteratiu. <http://design-toolkit.uoc.edu/iteratiu/>

Recordeu que la idea del prototip és la de crear una versió del vostre projecte –un videojoc en aquest cas– que té menys funcionalitats i/o menys contingut que el que serà el projecte final.

Per saber-ne més

Vegeu les fitxes de disseny iteratiu i de prototipat al Design Toolkit de la UOC per a més informació sobre aquestes dues idees.

Per tant, fins i tot per a un videojoc, es pot començar fent prototips en suport analògic. Per exemple, amb paper i cartró, i fer una primera partida o simulació de joc per tal de veure si la base sobre la que esteu començant a construir el projecte s'entén i funciona.

També és natural que aspectes com el metajoc, l'estrategia o les habilitats i destreses necessàries es vagin definint al llarg del procés iteratiu de prototipatge i testatge del joc, ja que no necessàriament es poden anticipar del tot abans de començar. De la mateixa manera, durant el procés es poden anar refinant les regles i les mecàniques, es pot optar per una estètica i no una altra, etc.

Val molt la pena, doncs, que en qualsevol projecte d'aquest tipus planifiqueu aquestes sessions d'avaluació de prototips i que per fer-les busqueu jugadors que coincideixin plenament amb el vostre públic objectiu i altres jugadors ben diferents. En qualsevol cas, han de ser jugadors que no estiguin familiaritzats prèviament amb el vostre projecte per tal que pugueu extreure'n el màxim d'informació de cara a refinjar i millorar el joc que esteu creant.

3. Joc i videojoc

3.4. Joc obert i joc tancat

Acabarem el repàs dels conceptes clau del disseny de videojocs amb una idea molt útil que va desenvolupar Jasper Juul i que han adoptat altres teòrics del videojocs: la diferenciació entre el joc obert (que Juul anomena emergent) i el joc tancat (joc progressiu). Els seus *games of emergence* i *games of progression* es refereixen a dues estructures que estan, segons ell, als dos extrems del disseny de jocs.

En el primer cas, el joc emergent o obert es basa en un nombre petit de normes senzilles que poden produir resultats molt variats. La majoria de jocs tradicionals (escacs, joc de cartes, tres en ratlla...) cauen dins d'aquesta categoria, però també alguns videojocs, com per exemple el Tetris.

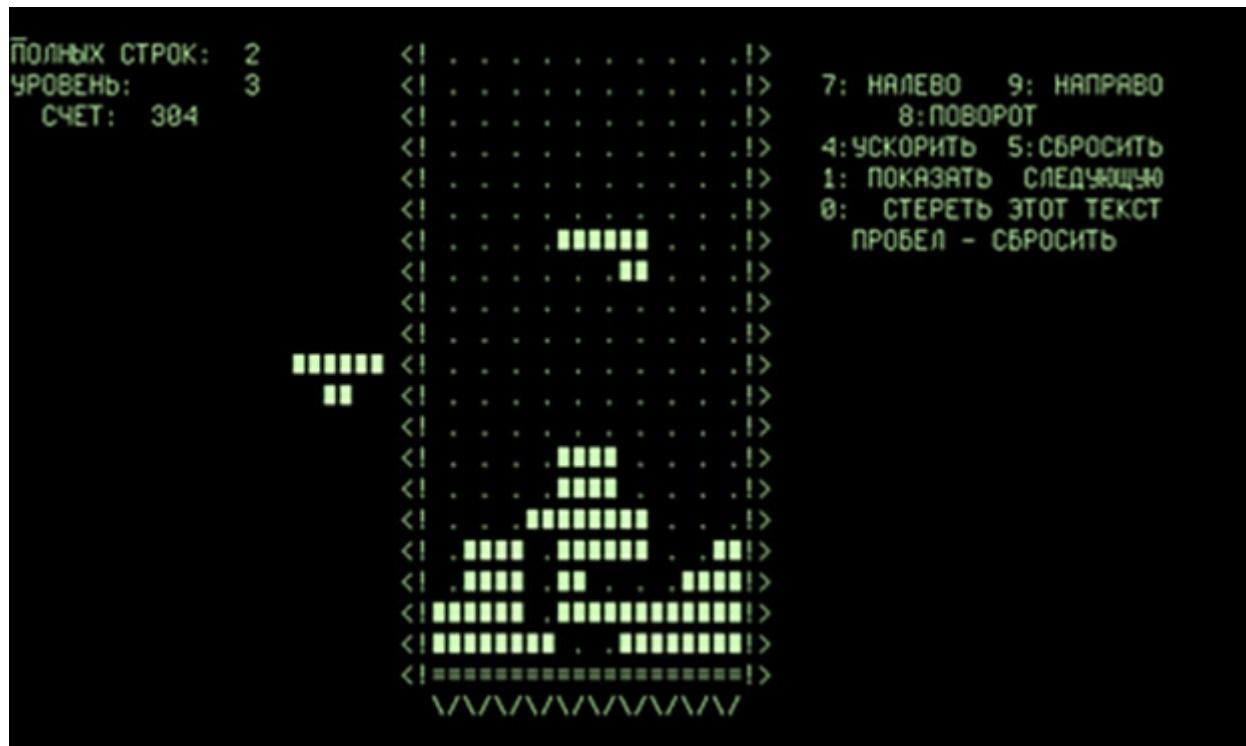


Figura 94. El Tetris és un dels videojocs més coneguts i fàcilment recognoscibles.

Font: <https://en.wikipedia.org/wiki/File:Tetris-VeryFirstVersion.png>

Els jocs de progrés o tancats són més nous històricament, i molt més arrelats als videojocs. Es tracta de jocs basats en petits reptes que són presents de manera seqüencial. Les aventures gràfiques de les primeres èpoques dels videojocs, o títols actuals com *Grand Theft Auto* o *The Last of Us*, o també *Super Mario Bros*, estarien dins d'aquesta categoria.



Figura 95. *The Last of Us* és un exemple contemporani de videojoc amb una alta càrrega narrativa, que troba un balanç molt adequat entre història i jugabilitat.

Font: https://en.wikipedia.org/wiki/File:The_Last_of_Us_Listen_Mode.jpg

Per tant, a l'hora de dissenyar qualsevol videojoc, és molt útil pensar en aquestes dues categories per tal de decidir cap a quina banda volem que graviti el projecte en el qual estem treballant. Si bé no és pas una classificació totalment binària, sí que és molt útil pensar si el que estem fent és un joc obert, que ofereix moltes possibilitats de resolució a partir d'una base relativament simple, o bé un joc progressiu on el jugador va superant reptes, fases, nivells, etc. fins a arribar a un objectiu final.

Aquest segon enfocament, més propi del món dels videojocs, ofereix tal com reconeix Juul més control al dissenyador, i és especialment adequat als jocs amb ambicions més narratives o fins i tot cinemàtiques. Tanmateix, el seu principal problema potencial és que hom acabi jugant-hi mecànicament, és a dir, simplement fent els moviments correctes, predefinitos, per tal d'avancar amb el joc. A més, típicament, un joc obert és atractiu de repetir, mentre que un joc de progrés ja no ho és tant, un cop s'ha completat una vegada. Òbviament, com més tancat sigui, més certa serà aquesta darrera afirmació. No és el mateix repetir una partida a *Maniac Mansion* que una de *Super Mario Bros*.

3. Joc i videojoc

3.5. Estudi de cas: art digital i videojoc

3.5.1. Introducció

Acabarem repassant algunes aproximacions al món del joc des de l'art digital i la programació creativa, amb dues categories. D'una banda, des de la idea del *hacking*: la modificació del joc original amb finalitats artístiques. De l'altra, la creació d'un joc com a tal amb finalitats artístiques. En tots dos casos hi ha un diàleg interessant entre obres molt properes a l'activisme i altres que es queden molt més en l'estètica i el formalisme sense aparentment anar més enllà.

3. Joc i videojoc

3.5. Estudi de cas: art digital i videojoc

3.5.2. Hacking

Malgrat certes connotacions socials i cinematogràfiques, la idea de *hack* quan es refereix a la programació informàtica normalment es refereix simplement a *truc*. És a dir, a trucar, a arreglar o modificar de manera ràpida i no necessàriament el més eficient possible algun programa o dispositiu electrònic. Que això es faci amb finalitats artístiques, simplement reparatories, o perseguint un altra mena d'objectius, ja és un altre tema.

Per part dels artistes, hi ha hagut molts exemples d'aplicació del *hacking* als videojocs o jocs amb elements electrònics, igual que hi ha hagut exemples de modificació de jocs tradicionals, o adopció d'estrategies del joc a les obres d'art. N'hem triat quatre exemples que van des de l'activisme fins al formalisme estètic. Els comentem per ordre cronològic.

The Barbie Liberation Organization, RTmark, 1993

Tot i que aquest exemple no té a veure amb els videojocs sinó amb joguines amb complements electrònics, és molt il·lustratiu d'una certa tendència que gravita entre l'activisme i l'humor.

L'any 1993, el collectiu d'artistes The Barbie Liberation Organization (BLO), amb l'objectiu de denunciar els marcadíssims rols de gènere a algunes de les joguines més populars del moment, va decidir hackear les Barbies i els J.I. Joes que havien sortit al mercat amb funcions de parla mitjançant microxips incorporats a les nines. Així, mentre unes deien coses com «m'encanta anar a comprar» o «les mates són difícils», els altres feien crits de guerra mentre sonaven les metralletes.

L'acció va consistir en dues fases: la quirúrgica i la mediàtica. Primer, el collectiu va comprar un número indeterminat de Barbies i J.I. Joes –que va d'una dotzena a alguns milers segons les versions existents– i, amb molta cura, va extreure'n els xips, els va intercanviar. Així, ara era el J.I. Joe que amb una veu femenina anunciava que no se'n sortia amb les matemàtiques, mentre la Barbie feia crits de guerra.



Figura 96. Un dels cartells creats per la Barbie Liberation Army.

Font: <https://www.jotdown.es/2016/11/viva-la-barbie-liberation-organization/>

Fins aquí no tindria més, però la gràcia és que els van tornar a la caixa i, d'amagat, els van tornar als prestatges de les botigues en el que van definir com l'acte invers a robar. I, a partir d'aquí, comença la campanya mediàtica. Els artistes van enviar cintes de vídeo als mitjans de comunicació explicant la seva acció. Amb això, van aconseguir el ressò mediàtic esperat, arribant a sortir a diversos programes de notícies, que entrevistaven pares i mares indignats.

Per a saber-ne més

Més informació a:

https://en.wikipedia.org/wiki/Barbie_Liberation_Organization

<https://sniggle.net/barbie.php>

<https://www.youtube.com/watch?v=DzTWF1jVwH4>

<https://www.youtube.com/watch?v=OVT4T70R3iQ>

Super Mario Cloud i I Shot Andy Warhol, Cory Arcangel, 2002

L'artista Cory Arcangelo, l'any 2002, va hackear dos jocs de la consola Nintendo Entertainment System (NES): el *Super Mario Bros* i *Hogans Alley*. D'aquí en van sortir les dues peces, presentades en forma d'instal·lació anomenades *Super Mario Clouds* i *I Shot Andy Warhol*.

En el primer, va hackear el cartutx del joc original per tal d'esborrar tots els elements excepte els núvols i el cel blau. El resultat és ben simple visualment: els núvols dels gràfics en 8 bits que es van movent lentament sempre cap a l'esquerra. Cap més dels elements del joc és present al pla digital. Al físic sí, ja que la consola i els comandaments formen part de la peça quan aquesta s'exhibeix.

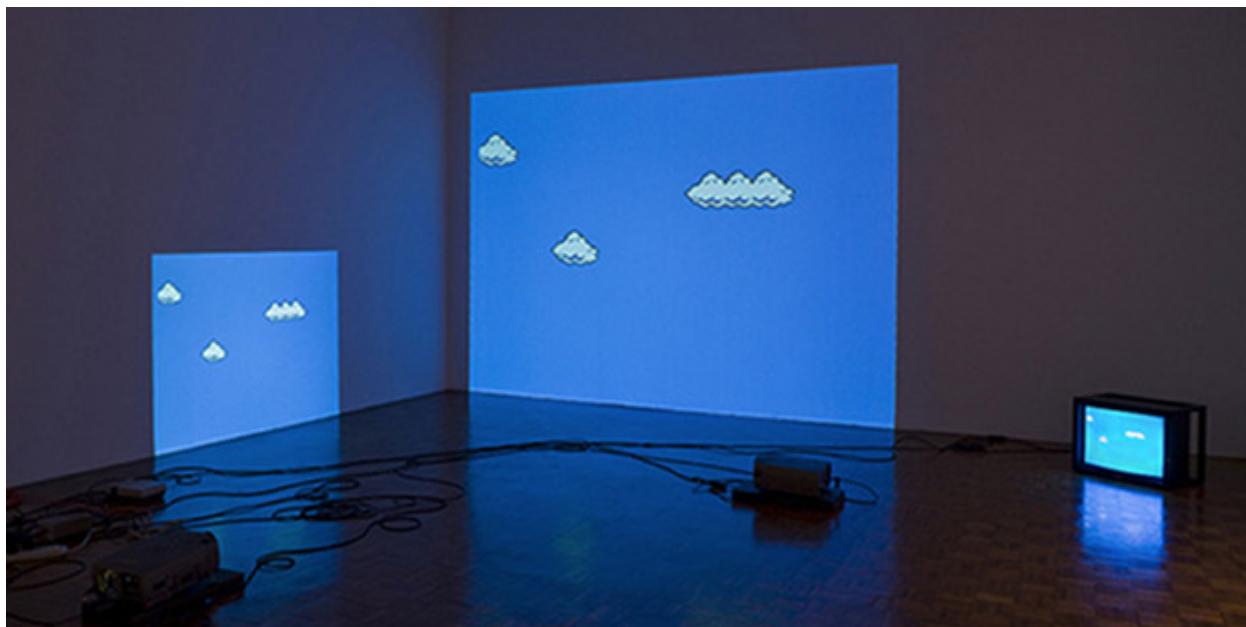


Figura 97. *Super Mario Clouds* al Whitney Museum of American Art, Nova York, 2009.

Font: <https://coryarcangel.com/shows/synthetic>

El segon joc hackejat és *I Shot Andy Warhol*, que el que fa és canviar els gràfics originals de *Hogans Alley*. En aquest joc, on s'interactua amb la llavors molt popular pistola d'infrarrojos que s'accionava apuntant cap al televisor, el jugador havia de disparar uns «dummies» (ninots) que representaven dolents (lladres i delinqüents), i evitar disparar els bons (al joc, policies i gent del carrer).

Arcangel el que fa és canviar aquests ninots per Andy Warhol en representació dels dolents, i el papa de Roma, el cantant de rap Flavor Flav i Col Sanders, el coronel famós per haver fundat i ser la imatge de la cadena de menjar ràpid Kentucky Fried Chicken.



Figura 98. Un fotograma d'*I Shot Andy Warhol*.

Font: <https://coryarcangel.com/things-i-made/2002-002-i-shot-andy-warhol>

A diferència de *Super Mario Clouds*, aquí el hacking no consisteix a desmuntar el joc i deixar tan sols un dels elements formals a pantalla, sinó de mantenir tot el joc com a tal i canviar simplement el camp semàntic, mitjançant la modificació de part dels elements gràfics. Ja no estem disparant contra lladres i evitant disparar policies i vianants i, per tant posar-nos a la pell del policia nosaltres mateixos, sinó que el nostre paper com a jugadors és molt més ambigu perquè ens hem de dedicar a eliminar tots els Andy Warhols que vagin apareixent i deixar estar la resta de personatges.

Per a saber-ne més

Més informació a:

<https://coryarcangel.com/things-i-made/2002-001-super-mario-clouds>

<https://coryarcangel.com/things-i-made/2002-002-i-shot-andy-warhol>

<https://www.youtube.com/watch?v=fCmAD0TwGcQ>

<https://www.youtube.com/watch?v=Nbej4iuUN40>

Max Payne Cheats Only, JODI, 2004

JODI és un collectiu format per dos artistes que formen part del grup selecte de creadors que van fer néixer la disciplina del net.art als anys 90 del segle passat. Inicialment molt gelosos de la seva intimitat i treballant sempre des de l'anonimat, amb el temps van anar normalitzant la seva presència. Són l'holandesa Joan Heemskerk i el belga Dirk Paesman.

Lligant amb altres obres anteriors dels mateixos artistes, que havien treballat a partir de *shooters* com *Wolfenstein Castle*, amb *Max Payne Cheats Only*, el hacking consistia en modificar un dels jocs de moda del moment, *Max Payne*, un típic *shooter* en tercera persona on el jugador és un agent que va disparant més o menys tothom que li passa per davant.

Tot i que actualment els estàndards de realisme als jocs 3D han avançat moltíssim respecte el 2004, en aquell moment *Max Payne* estava a l'avantguarda en aquest sentit. D'aquí que tingués sentit una obra que amb el que juga, precisament, és amb mostrar l'artifici darrere la simulació del món real que persegueix el joc. Moviments repetitius, angles morts, càmeres 3D dins l'avatar per

mostrar que és buit per dins... Tots aquests artificis formen part d'una peça de documentació videogràfica que es mostrava en format d'instal·lació.

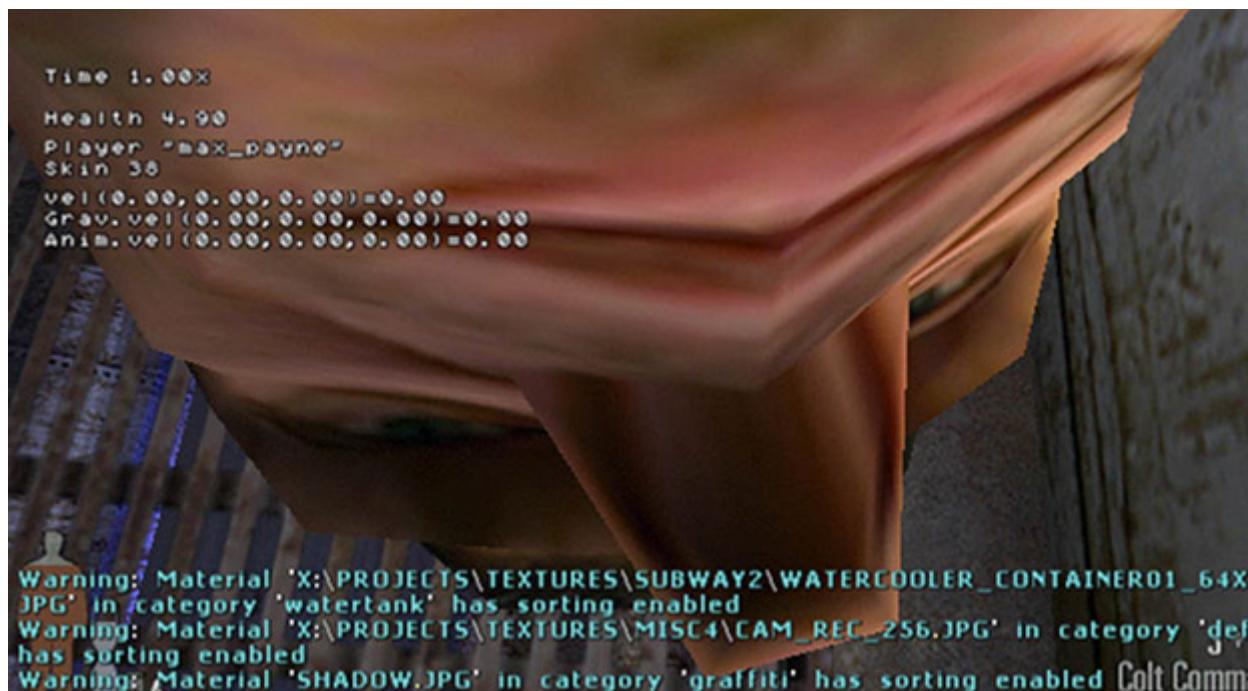


Figura 99. Un fotograma de *Max Payne Cheats Only*.

Font: <https://maxpaynecheatsonly.jodi.org/f12/012.html>

Al text del festival Transmediale 2006 definien *May Payne Cheats Only* d'aquesta manera:

«JODI ha intervenint l'estrucció del programa de tal manera que les perspectives absurdes i els efectes alteren el que eren gràfics realistes: veiem el gran heroi repetint moviments estúpids; posa el cap angular dins una matriu virtual; el cos se'n mostra semitransparent. El reprocessament es pot entendre com una interpretació lliure de l'efecte de *bullet-time* [temps alentit, de quan es dispara una bala] que distingeix Max Payne d'altres jocs i, amb aquesta càmera lenta, ens ofereix una nova perspectiva de l'espai i el temps.»

Text original: «*Jodi have intervened in the programme structure in such a way that absurd perspectives and effects alter the game's otherwise realistic graphics: we see the massive hero repeating idiotic movements; he dips his angular head into a virtual matrix; his body appears semitransparent. The re-processing can be read as a free interpretation of the bullet-time effect that distinguishes Max Payne from other games, by which the slow motion enables a new perception of space and time*. <https://www.eai.org/titles/max-payne-cheats-only-1>

Per a saber-ne més

Més informació a:

<https://www.eai.org/titles/max-payne-cheats-only-1>

[https://en.wikipedia.org/wiki/Jodi_\(art_collective\)](https://en.wikipedia.org/wiki/Jodi_(art_collective)).

<https://maxpaynecheatsonly.jodi.org/f12/012.html>

<https://www.youtube.com/watch?v=M7uWZFusIIQ>

<https://vimeo.com/10646976>

3. Joc i videojoc

3.5. Estudi de cas: art digital i videojoc

3.5.3. El joc com a forma artística

Més enllà del *hacking* i la modificació de jocs, hi ha artistes que han adoptat el joc com a forma artística. És a dir, que han fet servir el joc o videojoc com a mitjà d'expressió, no fent modificacions sinó creant les experiències lúdiques des de zero. N'analitzem tres exemples.

The Intruder, Natalie Bookchin, 1999

Natalie Bookchin és una altra de les referents del net.art, igual que JODI, un moviment nascut alhora de la web, que té el mèrit de ser un dels primers subàmbits de l'art digital que es va establir com a tal. *The Intruder* és un exemple de la varietat d'obres que es feien en aquest col·lectiu.

Es tracta d'una obra de narrativa interactiva, basada en el conte de Jorge Luís Borges *La intrusa*. El conte explica la història dels germans Nilsen, que comparten l'enamorament i les relacions amb la mateixa noia, la Juliana, fins que l'acaben assassinant. Una història tan crua com ben narrada, com és sempre el cas amb aquest autor. La versió que en fa Bookchin és una narració del conte, traduït a l'anglès, que es va desplegant a mesura que hom avança al llarg dels deu videojocs que conformen la peça. Es tracta de deu jocs que enllacen, de manera més o menys directa, en cada una de les fases del conte –per exemple, el Pong inicial es una metàfora prou clara de la trama del conte. Superant-los, la narració avança fins que s'arriba al desenllaç final.

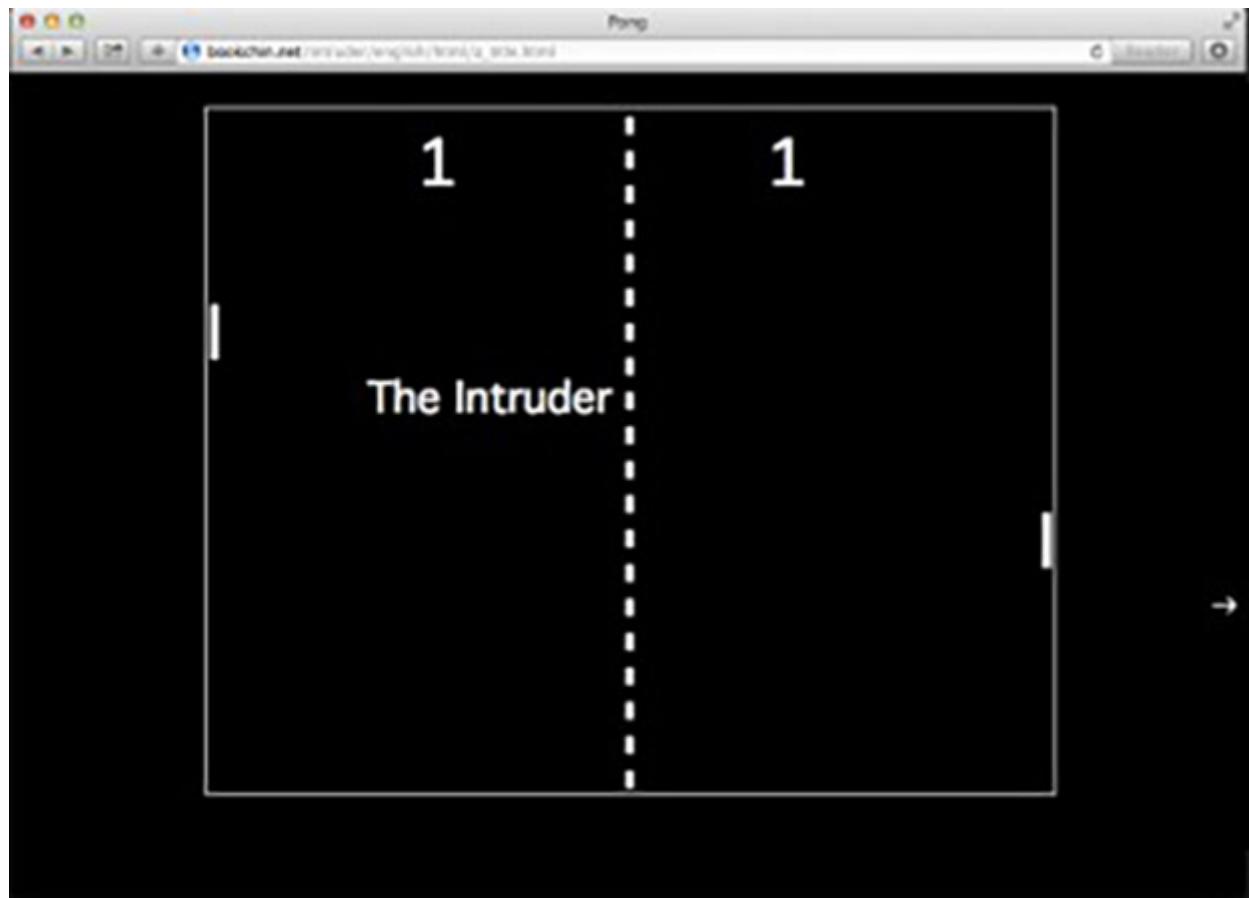


Figura 100. *The Intruder*, tal i com es visualitzava al navegador.

Font: <https://bookchin.net/projects/the-intruder/>

No es tracta, doncs, d'un joc ni complex ni difícil. Aquí, la successió de jocs no és més que la forma que troba l'autora de reinterpretar l'obra, i oferir-ne el control al lector perquè sigui aquest qui en determina l'avanç.

Com a obra de net.art, es tractava d'una experiència pensada per a accedir-hi via el navegador web, i això ens porta a un tema sovint obviat però importantíssim tant pels creadors que treballen amb la programació creativa, com pels creadors de videojoc en general: la obsolescència.

The Intruder es va fer amb Macromedia Director (més tard Adobe Director), l'opció lògica l'any 1999 per crear una experiència interactiva per ser vista mitjançant un navegador web. En aquell moment, era fàcil descarregar-se un *plug-in* per veure contingut

creat amb aquesta plataforma, i relativament habitual tenir-lo instal·lat, especialment entre l'audiència a qui anava adreçada *The Intruder*. Fins i tot hi havia ordinadors que el duien instal·lat de fàbrica.

De mica en mica, però, i a mesura que l'accés a internet d'alta velocitat s'anava estenent, Director va anar deixant lloc a Flash com a eina preferent de creació *online*. I com si fos una premonició del que li passaria precisament a Flash uns anys més tard, Director va anar perdent pes progressivament, fins al punt que cada cop era menys habitual que els navegadors fossin compatibles amb uns *plug-ins* que s'anaven quedant enrere. Tot i que oficialment el *plug-in* no es va discontinuar del tot fins el 2019, ben bé una dècada abans obres com la de Bookchin ja estaven condemnades a desaparèixer si no es reprogramaven des de zero.

Afortunadament, existeix documentació videogràfica del *game-play* de *The Intruder*, que també va ser pioner en això: quedar-se fora de joc, tecnològicament parlant, com ha passat recentment a tants i tants grans projectes web fets amb Flash, i com havia passat també a l'onada creativa de projectes desenvolupats amb Processing que van trobar el seu lloc a la web mitjançant el *plug-in* de Java.

Per a saber-ne més

Més informació a:

<https://bookchin.net/projects/the-intruder/>

<https://vimeo.com/30022802>

<https://www.literatura.us/borges/laintrusa.html>

Dys4ria, Anna Anthropy, 2012

Fet tretze anys més tard, però també ja inaccessible a la web degut a la obsolescència del *plug-in* de Flash, però també documentat videogràficament, Dys4ria és un joc creat per l'artista Anna Anthropy. Es tracta d'un projecte autobiogràfic en forma de joc d'estètica retro, on l'autora documenta, amb un ton seriós però no sense tocs d'humor, un període de sis mesos en què es va sotmetre a un tractament hormonal, com a part de la seva transició de gènere.

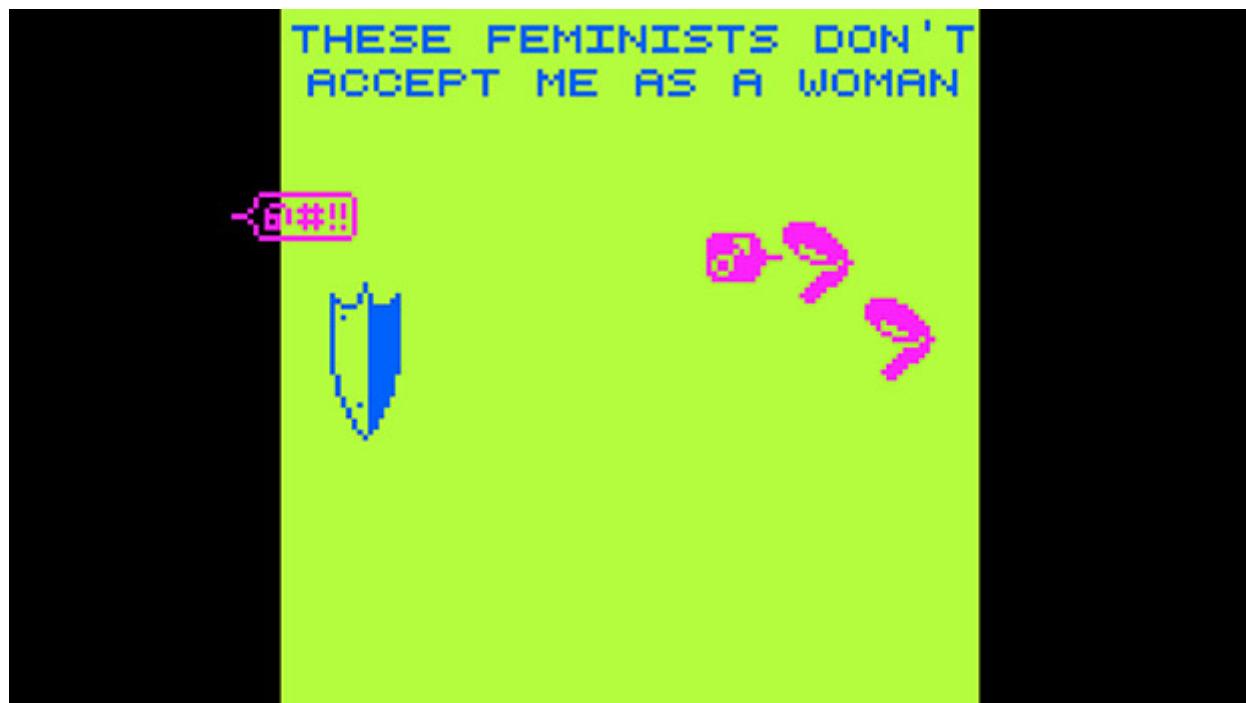


Figura 101. Un fotograma de Dys4ria.

Font: <https://zkm.de/en/dys4ia>

La web del ZKM (Center for Art and Media Karlsruhe), un dels centres de referència en art digital a Europa, defineix el projecte d'aquesta manera:

«En quatre seccions, els jugadors experiencien com l'artista decideix fer teràpia hormonal i l'acompanyen amb la seva transició del seu gènere de naixement al seu nou gènere. El joc documenta aquest període de sis mesos a través d'una sèrie de mini-jocs que reflecteixen polítiques de gènere, identitat, responsabilitat personal, el privilegi de ser blanc, i desenvolupament personal.

No hi ha puntuacions màximes; els jugadors no poden guanyar ni perdre. Els mecanismes del joc no són més que artefactes narratius.»

Text Original: «*In four sections, the players experience how the artist decides to have hormone therapy and they accompany her through the transition from her birth gender to her new gender. The game documents a period of six months through a series of mini-games that reflect gender politics, identity, personal responsibility, white privileges, and personal development.*

There are no high scores; the players can neither win nor lose. The game mechanisms serve only as narrative devices. <https://zkm.de/en/dys4ia>

Per a saber-ne més

Més informació a:

<http://www.editmedia.org/teaching-material/dy4ia-video-gameanna-anthropy-2012/>

<https://zkm.de/en/dys4ia>

Papers Please, Lucas Pope (3909 LLC), 2013

Ja dins el circuit comercial, tot i que clarament emmarcat dins els *indie games*, *Paper, Please* és una aventura gràfica per a iPad, PC o Mac. Es tracta d'un videojoc en format d'aventura gràfica, dividit en 31 dies i estructurat com a narrativa no lineal, amb fins a 20 possibles finals diferents.

A *Papers, Please*, el jugador assumeix el rol d'un oficial d'immigració a un punt de control d'immigració, al país fictici d'Arstotzka, que està en conflicte amb tots els seus veïns. El joc consisteix en acceptar o denegar l'entrada d'immigrants, i fer-ho correctament segons unes normes cada cop més complicades. El jugador ha de prendre les decisions correctes –no segons la seva ètica personal sinó aquestes normes– ja que fer-ho o no afectarà les seves finances personals, que és l'altre element que cal controlar.



Figura 102. Una escena de Papers, Please.

Font: [https://en.wikipedia.org/wiki/File:Papers,_Please_\(video_game_screenshot\).jpg](https://en.wikipedia.org/wiki/File:Papers,_Please_(video_game_screenshot).jpg)

Papers, Please és considerat un exemple de joc d'empatia (*empathy games*), una categoria en què el jugador es posa a la pell d'algú que no pertany al seu grup social (sigui per origen, gènere, estatus...) i adquireix així una perspectiva nova. En aquest cas, però, el rol assumit no és el més típic, que és el vulnerable, sinó un personatge més complex que, si bé té un poder important, també està subjecte ell mateix a unes normes i uns interessos propis –guanyar prou diners per alimentar la seva família. Per tant, el jugador es trobarà a vegades havent de decidir entre deixar passar algú que realment ho necessita, però que per algun motiu tècnic podria deixar fora, i que això el perjudici a ell personalment, o bé deixar l'ètica de banda i assegurar-se de mantenir ben plena la seva butxaca.

Per a saber-ne més

Més informació a:

<https://papersplea.se/>

https://en.wikipedia.org/wiki/Papers,_Please

https://en.wikipedia.org/wiki/Video_games_as_an_art_form

3. Joc i videojoc

Bibliografia

- Adams, E. (2010). *Fundamentals of Game Design. Second Edition*. Berkeley, CA: New Riders.
- Adams, E., & Dormans, J. (2012). *Game Mechanics. Advanced Game Design*. Berkeley: New Riders.
- Bourdin, P., Duch, J., & Tejedor, H. (2020). *Introducció als videojocs*. UOC.
- Fullerton, T. (2008). *Game Design Workshop. A Playcentric Approach To Creating Innovative Games*. Burlington, MA: Elsevier.
- Huizinga, J. (1956). *Homo Ludens. Von Ursprung der Kultur im Spiel*. Hamburg: Rowohls Enzyklopädie.
- Huizinga, J. (1972). *Homo Ludens*. Madrid: Alianza Editorial.
- Juul, J. (2005). *Half-Real. Video Games between Real Rules and Fictional Worlds*. Cambridge, Massachusetts: MIT Press.
- Juul, J. (2002). «The Open and the Closed: Games of Emergence and Games of Progression». In: *Computer Games and Digital Cultures Conference Proceedings* (pp. 323–329). Tampere.
- Salen, K., & Zimmerman, E. (2004). *Rules of Play – Game Design Fundamentals*. Cambridge, Massachusetts: MIT Press.
- Schell, J. (2008). *The Art of Game Design: A book of lenses*. Boca Raton, FL: CRC Press.
- Soler-Adillon, J. (2019). «The Open, the Closed and the Emergent: Theorizing Emergence for Videogame Studies». *Game Studies*, 19(2). Recuperat de <http://gamestudies.org/1902/articles/soleradillon>.

Enllaços

Game Studies: <http://gamestudies.org/>

4. Més Processing

4.1. Introducció

Aquesta part del quadern d'assignatura es planteja com a continuació del recurs *Programació en Processing: activitat pràctica*. Per tant, d'una banda dona per suposats els continguts que s'hi presenten i, de l'altra, el que fa és plantejar altres estructures, estratègies, i exemples del treball en Processing per a la programació creativa. Es tracta també d'una aproximació incremental. Així, si bé obviament podeu saltar a la part que més us interessi a cada moment, hi podran aparèixer sempre estructures que ja s'hagin presentat a les seccions anteriors.

4. Més Processing

4.2. Vectors i forces

En matemàtiques o física, un vector és la representació de la distància entre dos punts. Una informació que serveix per definir posició, velocitat, acceleració... i de la qual es poden extreure paràmetres com la magnitud o la direcció.

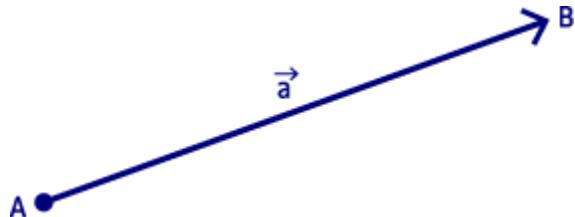


Figura 103. Vector

Font: adaptació a partir d'[https://en.wikipedia.org/wiki/Vector_\(mathematics_and_physics\)#/media/File:Vector_from_A_to_B.svg](https://en.wikipedia.org/wiki/Vector_(mathematics_and_physics)#/media/File:Vector_from_A_to_B.svg)

L'estructura `PVector` serveix essencialment per a dues coses. D'una banda, com és obvi pel nom que té, defineix un vector i permet, per tant, fer servir matemàtica vectorial. Però a efectes més pràctics i mundans des del punt de vista de la programació creativa, ens permet fer-los servir per guardar posicions dels elements que tenim en pantalla i ens facilita calcular-ne moviments, acceleracions, distàncies, etc.

De fet, no és res que no poguéssim fer sense `PVector`, ja que aquesta essencialment no és més que una estructura que guarda o bé dos o bé tres valors (per tant coordenades a l'espai 2D o 3D). Però ja que aquesta estructura existeix i ens facilita la feina, val la pena fer-la servir.

Per a saber-ne més

Com sempre, a la referència de Processing hi trobareu amb detall la informació de com fer-lo servir i els mètodes associats: <https://processing.org/reference/PVector.html>

Per començar, farem servir `PVector` simplement per guardar les coordenades `x` i `y` d'un objecte que mostrarem a la finestra de Processing:

```
PVector pos = new PVector(100,50);

void setup(){
  size(400,200);
}

void draw(){
  background(192);
  circle(pos.x,pos.y,25);
}
```

De moment això no fa res. Però ens serveix per explicar la base de `PVector`.

Primer, el declarem, li posem nom (aquí 'pos', per posició), i li donem un valor inicial.

```
PVector pos = new PVector(300,200);
```

I un cop el volem fer servir, podem separar-ne els components amb `pos.x` i `pos.y`, que podrem modificar o fer servir directament com a paràmetres com qualsevol altra variable:

```
circle(pos.x,pos.y,25);
```

Fins aquí cap secret, i tampoc cap guany respecte a haver fet servir simplement dues variables per guardar la `x` i la `y`. Però si més no, `PVector` ens serveix per guardar la posició en una sola estructura enlloc de dues.

Ara bé, si podem fer-ho amb la posició, també ho podem fer amb la velocitat. Podem crear un altre `PVector`, i fer servir `add()` per fer l'addició d'un vector a l'altre (el de velocitat al de posició) amb una sola operació. Hi afegirem un condicional, també, per mantenir l'element dins de la pantalla:

```
PVector pos = new PVector(100,50);
PVector vel = new PVector(1,0);

void setup(){
    size(400,200);
}

void draw(){
    background(255);
    pos.add(vel);
    circle(pos.x,pos.y,25);
    if(pos.x > width){
        pos.x = 0;
    }
}
```

I ben fàcilment, aquesta manera de treballar ens permet afegir un tercer element, l'acceleració:

```
PVector pos = new PVector(150,100);
PVector vel = new PVector(1,0);
PVector acc = new PVector(0.01,0);
void setup(){
    size(400,200);
}

void draw(){
    background(192);
    vel.add(acc);
    pos.add(vel);
    circle(pos.x,pos.y,25);
    if(pos.x > width){
        pos.x = 0;
    }
}
```

A partir d'aquí, tant podem fer-ho servir per a creacions més imaginatives, com per fer simulacions, més o menys acurades, de fenòmens de la realitat. Per exemple, podem simular una pilota que cau i rebota a terra per l'efecte de la gravetat.

Abans de seguir endavant, atureu-vos un moment i mireu de pensar què caldria canviar respecte a l'exemple vist fins ara per tal d'aconseguir-ho. Si us hi veieu amb cor, proveu d'implementar-ho.

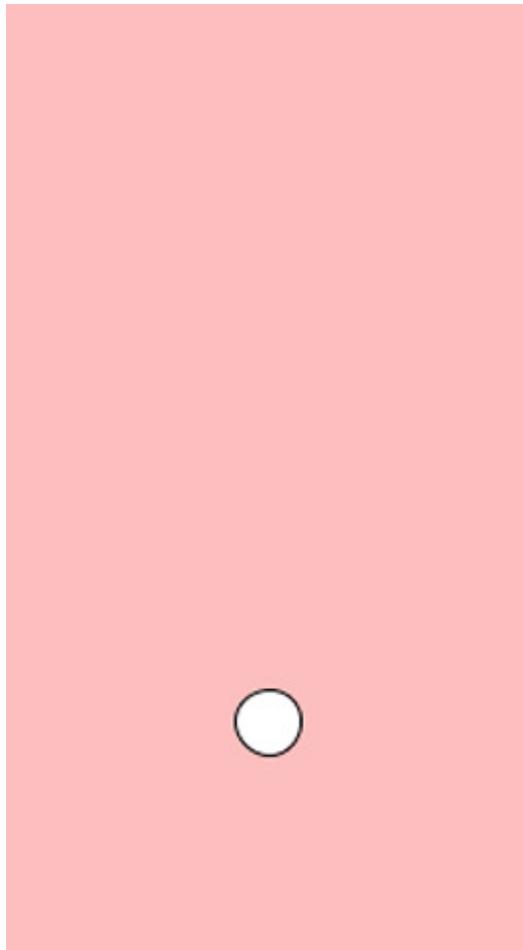


Figura 104. Podeu fer una finestra vertical per remarcar l'efecte de caiguda.

Els passos per passar del que tenim a la simulació simple de la llei de la gravetat són els següents:

Primer, ens caldrà modificar els valors inicials de posició, velocitat i acceleració. Començarem amb una posició a la part central i alta de la finestra, i amb velocitat zero. La simulació, doncs, es basarà en el valor de l'acceleració, que serà de 0,98 cap avall. Podeu provar altres valors i veureu en quins casos hi ha canvis visualment significatius.

```
PVector pos = new PVector(100,50);
PVector vel = new PVector(0,0);
PVector acc = new PVector(0,0.98);
```

Això farà que la bola comenci estàtica i vagi cap a baix cada cop més de pressa, però ens calen un parell de canvis més perquè la simulació ens funcioni. Primer, volem que reboti a terra (és a dir, a *height*), i que ho faci no pel centre sinó per la part inferior de la bola. Per fer-ho, ens caldrà una variable que defineixi la mida de la bola, i un condicional que li digui a Processing que, si la bola toca la part inferior de la finestra, inverteixi la velocitat. Tal com amb una variable la podem multiplicar per -1 per invertir-ne el valor, amb un PVector podem fer exactament el mateix i invertirem el vector sencer, amb `mult()`.

```
float ballSize = 25;
...
...
```

```

...
if(pos.y + ballSize/2 > height){
    vel.mult(-1);
    pos.y = height - ballSize/2 ;
}

```

Això ens crearà un robot més o menys aconseguit, però perquè sembli un efecte gravitacional ens cal una última cosa: la fricció que vagi frenant la bola cada cop que toca a terra. Un valor que definirem com a variable i que aplicarem a dins del condicional que s'activa quan la bola toca la part de sota, multiplicant la velocitat per aquest valor per tal que es vagi frenant cada cop que rebota.

```

float friction = 0.9;
...
...
if(pos.y + ballSize/2 > height){
    vel.mult(-1);
    vel.mult(friction);
    pos.y = height - ballSize/2 ;
}

```

Com és obvi, això no crea una simulació perfecta de l'efecte gravitacional, però és prou efectiu per donar-nos molt de joc i un punt més de realisme als nostres *sketch* de Processing. El codi complert queda així:

```

PVector pos = new PVector(100,50);
PVector vel = new PVector(0,0);
PVector acc = new PVector(0,0.98);
float friction = 0.9;
float ballSize = 25;

void setup(){
    size(200,400);
}

void draw(){
    background(#FCBFBF);
    vel.add(acc);
    pos.add(vel);
    if(pos.y + ballSize/2 > height){
        vel.mult(-1);
        vel.mult(friction);
        pos.y = height - ballSize/2 ;
    }
    circle(pos.x,pos.y,25);
}

```

A partir d'aquí, podem afegir altres forces que afectin els elements que tenim a la pantalla, d'una manera relativament senzilla a partir de l'ús de PVector. Per exemple, podem modificar l'exemple anterior per tal que si cliquem generem un efecte de vent que faci moure la pilota cap a una banda o altra, o afecti també la velocitat a la que cau. Podeu provar-ho i jugar amb el codi d'aquest exemple per veure-ho:

```

PVector pos;
PVector vel;
PVector acc;
float friction = 0.9;
float ballSize = 25;
float windStrenght = 0.3;

void setup() {
    size(400, 400);
    pos = new PVector(width/2, 50);
    vel = new PVector(0, 0);
    acc = new PVector(0, 0.98);
}

void draw() {
    background(#FCBFBF);
    vel.add(acc);
    pos.add(vel);

    if (mousePressed) {
        line(width/2, height/2, mouseX, mouseY);
        PVector wind = new PVector(mouseX - width/2, mouseY - height/2);
        wind.normalize();
        wind.mult(windStrenght);
        vel.add(wind);
    }

    if (pos.y + ballSize/2 > height) {
        vel.y *= -1;
        vel.mult(friction);
        pos.y = height - ballSize/2 ;
        if (vel.mag() < 0.5) {
            reset();
        }
    }

    if (pos.x + ballSize/2 < 0) { vel.mult(friction); vel.x *= -1; pos.x = ballSize/2 ; } if (po
        vel.mult(friction);
        vel.x *= -1;
        pos.x = width - ballSize/2 ;
    }
    circle(pos.x, pos.y, 25);
}

void reset() {
    pos = new PVector(width/2, 50);
    vel = new PVector(0, 0);
}

```

La clau aquí és que tenim un element, la bola, que té una posició i una velocitat. La velocitat s'afegeix sempre a la posició, però alhora, just abans, afegim una sèrie de forces a la velocitat perquè afectin el moviment. En aquest cas, un vector sempre cap avall que simula la gravetat, i un vector definit segons la posició del ratolí que simula el vent. Les dues forces sumades afecten la velocitat que donarem a la bola a cada fotograma i, per tant, el resultat visual de l'element en moviment.

Per a saber-ne més

Per veure l'aplicació amb tot detall de forces mitjançant PVector, us recomanem llegir els dos primers capítols del llibre *The Nature of Code* de Dan Shiffman que estan dedicats exactament a això: vectors (mitjançant el PVector implementat a Processing precisament de la mà de Shiffman) i forces.

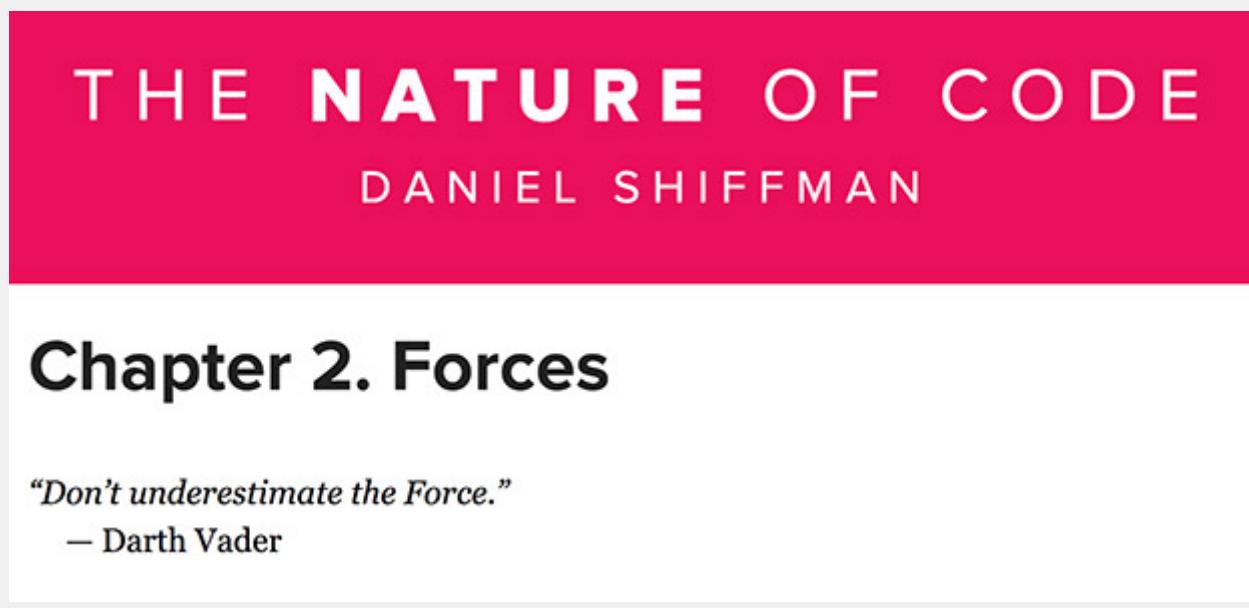


Figura 105. Podeu fer una finestra vertical per remarcar l'efecte de caiguda.

4. Més Processing

4.3. Objectes

Processing, com que està basat en el llenguatge de programació Java, és un dels molts llenguatges que està pensat per a treballar amb programació orientada a objectes (POO) (*).

Si heu treballat amb altres llenguatges de programació, de ben segur que hi estareu familiaritzats. I, en tot cas, en qualsevol manual de programació trobareu explicacions ben detallades de com funciona i com cal emprar el codi de la millor manera possible per treballar amb aquest paradigma.

Pels que no ho tingueu tan controlat, en fem un resum pràctic aquí per poder seguir, des d'aquest punt endavant, treballant amb programació orientada a objectes. I ho farem a partir de l'exemple que hem treballat fins ara, per després de poder-lo complicar amb més elements, varietat, etc.

La idea bàsica és que cada objecte ens servirà per definir els diferents elements, o almenys els més importants, que fem servir a cada programa. Per exemple, en un joc podem crear una classe per a l'avatar del jugador, i una altra per a cada tipus d'enemic o obstacle. O bé si aquests són similars, es poden fer classes que derivin d'altres, etc. El grau de complexitat depèn del projecte i sempre l'acaba decidint el programador.

En programació creativa no ens preocupem tant de què el codi sigui el màxim d'eficient com en altres contextos, però evidentment és important tenir tot això en compte. Al final, hi ha sempre diverses maneres d'arribar a un mateix resultat mitjançant el codi, i es tracta de trobar la que ens funcioni millor en cada moment. A vegades això suposa tornar a començar de zero, o fer passos una mica pesats a l'iniciar un projecte, però en poques ocasions no compensa començar des del principi amb la programació orientada a objectes al cap.

Tornem doncs al nostre exemple: primer de tot crearem la classe `bola` que contindrà la informació i els mètodes necessaris per generar el que ens interessa. L'estruccura bàsica és la següent:

```
class bola{  
  
    bola(){  
    }  
  
}
```

L'estruccura gran és la classe, i entre el claudàtor que segueix el nom i el que la tanca hi posarem tot allò que necessitem per a cada objecte que instanciem. I, de tot això, l'únic que hi hem de posar sí o sí és el constructor, que seria l'equivalent dins la classe al que a un programa de Processing és el `setup()`: una funció que s'executa abans de fer res més. En aquest cas, en el moment que instanciem l'objecte. El constructor ha de tenir el mateix nom exacte que la classe i pot rebre paràmetres com qualsevol altra funció.

A partir d'aquí, afegirem a la classe les variables i mètodes que ens interessin. El que farem és traslladar cap aquí tot allò que fa referència en el nostre cas a la bola (posició, velocitat, color, etc.). I també els mètodes per actualitzar, dibuixar, etc. Per exemple, la classe completa podria quedar així:

```
class bola {  
  
    PVector pos, vel, acc;  
    float friction, mySize;  
  
    bola() {  
        pos = new PVector(100, 50);  
        vel = new PVector(0, 0);  
        acc = new PVector(0, 0.98);  
        friction = 0.9;  
    }  
}
```

```

mySize = 25;
}

void run() {
    update();
    display();
}

void update() {
    vel.add(acc);
    pos.add(vel);
    if (pos.y + mySize/2 > height) {
        vel.mult(-1);
        vel.mult(friction);
        pos.y = height - mySize/2 ;
    }
}

void display() {
    circle(pos.x, pos.y, 25);
}
}

```

L'estructura és molt paràllela a la que teníem a l'exemple anterior. L'únic canvi una mica significatiu és la funció `run()` que engloba tot allò que farà l'objecte. Seria l'equivalent al `draw()` de Processing. A diferència del constructor, `run()` no és més que una convenció, que podeu adoptar o no. Aquí serveix per fer les coses més fàcils. `run()` crida `update()`, que modifica la posició, i `display()` que fa el dibuix.

A partir d'aquí, el nostre programa de Processing, classe a banda, és molt concís:

```

bola b = new bola();

void setup() {
    size(200, 400);
}

void draw() {
    background(#FCBFBF);
    b.run();
}

```

Primer, declarem i inicialitzem l'objecte `b`, que és una instància de la classe `bola`. Com que el constructor no demana cap paràmetre, no posem res dins els parèntesis. I llavors, dins el `draw` cridem la funció `run()` de dins l'objecte `b` amb aquesta notació del punt que és típica de la programació orientada a objectes: instància, punt, mètode: `b.run()`.

A partir d'aquesta base, el programa pot créixer ja sigui sofisticant la classe `bola` o afegint altres objectes, que a més es poden comunicar entre ells (a l'exemple que segueix, detectant collisions).

Abans de seguir, però, dos apunts: primer, una bona pràctica a Processing, un cop ens posem a treballar amb programació orientada a objectes, és intentar mantenir el `setup()` i el `draw()` el més nets possible. Sense exagerar-ho, i entenent que és una idea simplement que ens ha d'ajudar a organitzar-nos, la idea és treure'n tot allò que pugui anar en altres funcions o bé en classes, de manera que el codi ens quedi el més endreçat i modular possible.

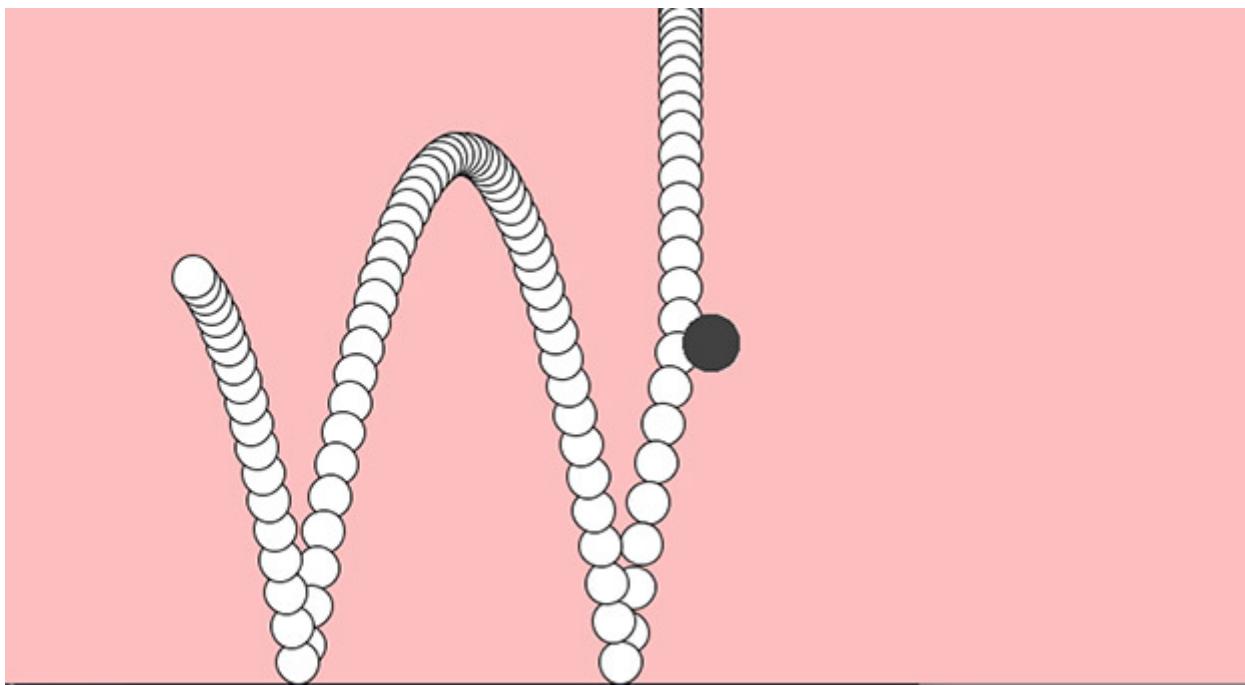


Figura 106. Evolució de l'exemple de la bola que cau per la gravetat

L'ampliació de l'exemple consisteix a introduir una segona classe, de la qual instanciem un objecte de posició aleatòria (però centrada) i mida aleatòria, que modifica la posició de la bola quan aquesta hi col·lideix. Quan cliquem, es reinicen la posició de la pilota i de l'obstacle.

```
bola ball;
obstacle obs;

void setup() {
  size(800, 400);
  ball = new bola(width/2, 0);
  obs = new obstacle(width/2+random(-20,20),200, random(10,40));
}

void draw() {
  background(#FCBFBF);
  ball.run();
  obs.run();
}
```

Pel que fa a la classe `bola`, la mantenim però hi afegim aquesta comunicació amb l'obstacle, quan comprova si hi ha col·lisió. Per fer-ho, activem la funció `checkCollision()` de la instància de l'obstacle, i hi passem com a paràmetres la posició i la mida de la bola. Si hi ha col·lisió, crearem un `PVector` i l'aplicarem a la velocitat, per tal que quedí afectada la trajectòria de la bola.

```
float obstacleCollision = obs.checkCollision(pos, mySize);
if (obstacleCollision != 0) {
  PVector movement = new PVector(obstacleCollision, 0);
  movement.mult(0.25);
  vel.add(movement);
}
```

La classe sencera, amb algun petit canvi més que podeu explorar, queda així:

```
class bola {  
  
    PVector myPos, vel, acc;  
    float friction, mySize;  
  
    bola(float x, float y) {  
        myPos = new PVector(x, y);  
        vel = new PVector(0, 0);  
        acc = new PVector(0, 0.98);  
        friction = 0.9;  
        mySize = 25;  
    }  
  
    void run() {  
        update();  
        display();  
    }  
  
    void update() {  
        float obstacleCollision = obs.checkCollision(myPos, mySize);  
        if (obstacleCollision != 0) {  
            PVector movement = new PVector(obstacleCollision, 0);  
            movement.mult(0.25);  
            vel.add(movement);  
        }  
        vel.add(acc);  
        myPos.add(vel);  
        if (myPos.y + mySize/2 > height) {  
            vel.y *= -1;  
            vel.mult(friction);  
            myPos.y = height - mySize/2 ;  
        }  
    }  
  
    void display() {  
        fill(255);  
        stroke(0);  
        circle(myPos.x, myPos.y, 25);  
    }  
}
```

Finalment, la classe `obstacle`, que conté aquesta funció una mica rocambolesca però clau per a l'efecte que buscàvem. El que fa és rebre la posició i mida de la bola, comprovar si la bola i el mateix obstacle s'estan tocant (si la distància és menor a la suma dels dos radis), i si és el cas, en funció de si la bola és més a la dreta o més a l'esquerra de l'obstacle, generar un valor que serà més gran com més gran sigui l'obstacle, i més a prop estigui la bola del seu centre en el moment de la colisió.

Per tant, si la bola toca d'esquitllada un obstacle petit, aquest valor –que al nostre programa afectarà la variació de la seva trajectòria– serà petit. Si la bola toca un obstacle gran més aviat al centre, aquest valor serà més gran.

```
float checkCollision(PVector ballPos, float ballSize) {  
    float movementToApply = 0.0;
```

```

    float movementToApply = 0;
    float dist = dist(ballPos.x, ballPos.y, myPos.x, myPos.y);
    if (dist < ballSize/2 + mySize/2) { //there's a collision! println(ballPos.x - myPos.x); i
        movementToApply = map(dist, 0, ballSize/2 + mySize/2, mySize, 0);
    } else {
        movementToApply = -map(dist, 0, ballSize/2 + mySize/2, mySize, 0);
    }
}
return movementToApply;
}

```

La classe sencera queda així:

```

class obstacle {

    PVector myPos;
    float mySize;

    obstacle(float x, float y, float sz) {
        myPos = new PVector(x, y);
        mySize = sz;
    }

    void run() {
        display();
    }

    void display() {
        fill(64,127);
        noStroke();
        circle(myPos.x, myPos.y, mySize);
    }

    float checkCollision(PVector ballPos, float ballSize) {
        float movementToApply = 0;
        float dist = dist(ballPos.x, ballPos.y, myPos.x, myPos.y);
        if (dist < ballSize/2 + mySize/2) { //there's a collision! if (ballPos.x > myPos.x) {
            movementToApply = map(dist, 0, ballSize/2 + mySize/2, mySize, 0);
        } else {
            movementToApply = -map(dist, 0, ballSize/2 + mySize/2, mySize, 0);
        }
    }
    return movementToApply;
}
}

```

Poseu tot el codi a un mateix programa de Processing i comproveu-ne el resultat. Com sempre, us recomanem jugar amb els valors, colors, etc. per introduir-hi el vostre toc i alhora explorar el que hi està passant.

Quan treballieu amb classes a Processing, podeu posar tot el codi seguit de dalt a baix, però també podeu utilitzar les pestanyes (la fletxeta cap avall a la dreta al costat del nom del vostre sketch, i separar cada classe en una de diferent. Això en realitat no fa cap

separació al programa, ja que Processing ho seguirà interpretant tot com si estigués al mateix lloc, però és molt útil per organitzar el codi.

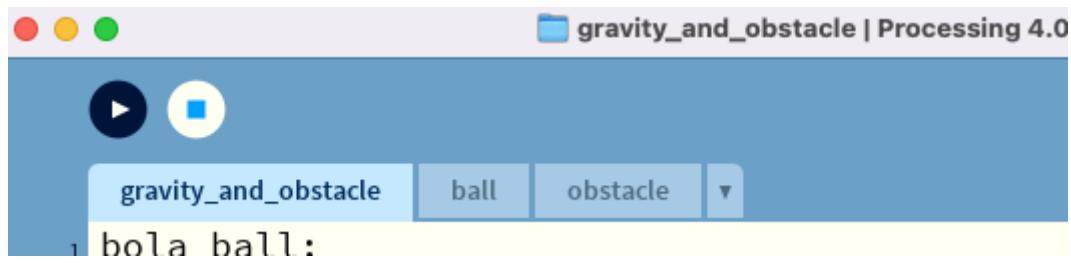


Figura 107. Organització de codi

4. Més Processing

4.4. ArrayLists

Els `ArrayList` són un tipus de llistes de dades, d'*arrays*, que poden ser molt útils en tot tipus de projectes. Bàsicament, perquè ofereixen una flexibilitat molt gran que no donen els *arrays* normals. Com que en programació creativa l'eficiència dels nostres programes acostuma a ser un element secundari, val molt la penaaprofitar aquest tipus d'estructures que ens permeten assolir més ràpidament i de manera més senzilla els nostres objectius.

El que ofereixen els `ArrayList` és la possibilitat de declarar-les sense haver d'especificar quants elements contindrà i, per tant, en podem posar i treure quan faci falta, i alhora ens permet no especificar quin tipus d'objectes o de dades hi emmagatzemarem. Si bé és molt poc comú trobar un context en què ens interessi guardar a una mateixa `ArrayList` diferents tipus de dades o objectes, la flexibilitat d'una estructura dinàmica on no calgui predeterminar el número d'elements, i on en podem anar posant i traient, sí que és molt útil.

Ho veurem amb dos exemples en què ens centrarem a explicar l'ús d' `ArrayLists` i deixarem la resta de coses perquè les exploreu tot jugant amb el codi. Com sempre, us animem a modificar i fer més complexos aquests exemples per anar descobrint les funcionalitats que hi ha i altres que hi pugueu aportar vosaltres. Primer, un sistema de partícules i, després, un exemple del que seria la fase inicial d'un videojoc.



Figura 108. Sistema de partícules

El nostre sistema de partícules es basa en un `ArrayList` que conté un seguit d'objectes, les partícules, i un algorisme que fa el següent: a cada iteració del bucle, crea 10 partícules noves i les actualitza totes. Pel que fa a les partícules, actualitzen la seva posició segons la posició inicial i la simulació de la gravetat, i si baixen més enllà de la finestra de Processing, surten de l' `ArrayList`, cosa que fa que siguin esborrades.

A la primera línia de l'exemple s'hi declara l' `ArrayList`. Fixeu-vos que podem fer-ho sense dir quin tipus de dades contindrà, i tampoc indiquem quants ítems hi hauran. Per tant, serà el motor de Processing mateix (per tant el llenguatge Java) qui s'encarregarà dinàmicament de fer espai a la memòria RAM per guardar la informació que hi anem posant. Per la nostra banda no cal que ens preocupem de res.

```
ArrayList particleList = new ArrayList();
```

L'únic que cal fer una mica diferent que amb la resta d'*arrays* ho trobarem al bucle que la recorre. Aquí, farem servir `ArrayList.size()` per saber quants elements té en cada moment i, per tant, quantes iteracions fer. I no tenim accés directe a les dades sinó que les hem d'extreure, en aquest cas creant un objecte `particle`, assignant-li el resultat

d' `ArrayList.get(index)` , però amb una peculiaritat: aquí sí que li hem d'indicar a Processing quin tipus de dades estem extraient de l' `ArrayList` . Per això escrivim `particle` entre parèntesis:

```
for (int i = 0; i < particleList.size(); i++) {  
    particle p = (particle) particleList.get(i);  
    p.run();  
}
```

Hi ha una manera d'estalviar-nos aquest pas i que trobarem a molts exemples de Processing, ja que com diem és molt estrany trobar-se en el context on realment es vulgui aprofitar que a un `ArrayList` hi puguem guardar diferents tipus de dades. Per tant, podem dir-li a Processing quina mena de dades hi haurà, i ens estalviarem d'anar-ho especificant cada cop que fem servir el `get()` . El codi quedaría així:

```
ArrayList<particle> particleList= new ArrayList();  
  
...  
...  
for (int i = 0; i < particleList.size(); i++) {  
    particle p = particleList.get(i);  
    p.run();  
}
```

Podeu fer servir la notació amb la qual us sentiu més còmodes. Aquí en farem servir una de diferent a cada exemple. Per tant, el del sistema de partícules, pestanya general i classe `particle`, queda així:

```
ArrayList particleList = new ArrayList();  
  
void setup() {  
    size(800, 450);  
    cursor(CROSS);  
}  
  
void draw() {  
    background(0);  
  
    for (int i = 0; i < 10; i++) {  
        particle p = new particle(mouseX,mouseY);  
        particleList.add(p);  
    }  
  
    for (int i = 0; i < particleList.size(); i++) {  
        particle p = (particle) particleList.get(i);  
        p.run();  
    }  
}
```

```
class particle {
```

```

PVector myPos, myVel, myAcc;
float mySize;

particle(float x, float y) {
    myPos = new PVector(x, y);
    mySize = 5;
    myVel = new PVector(random(-10, 10), random(-10, 10));
    myAcc = new PVector(0, 0.9);
}

void run() {
    update();
    display();
}

void update() {
    myVel.add(myAcc);
    myPos.add(myVel);
    if (myPos.y > height) {
        particleList.remove(this);
    }
}

void display() {
    noStroke();
    fill(255, 192);
    circle(myPos.x, myPos.y, mySize);
}

```

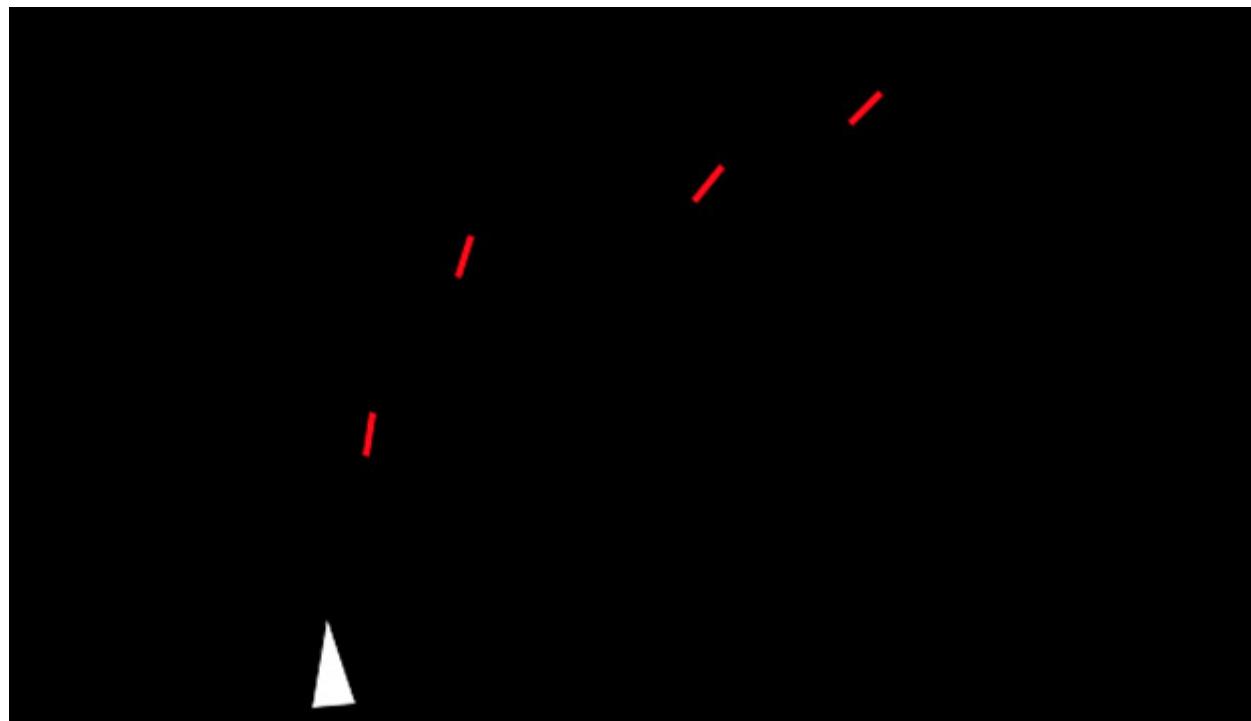


Figura 109. Fase inicial d'un videojoc

El segon exemple és una mena de nau espacial (representada aquí com a triangle), que movem amb les fletxes del teclat i que dispara al premer la tecla espai. A la pestanya principal hi instanciem la nau, l' `ArrayList` per les bales, i executem ambdues coses al `draw()`. Més avall, el `keyPressed()` ens serveix per establir el moviment de la nau d'una manera que evita els

problemes que sol causar aquesta mena d'interacció amb Processing i programes similars per com Java llegeix el primer les tecles. Potser us hi heu trobat, amb moviments que primer arrenquen i s'aturen, després arrenquen, però que a més no deixen fer dues accions alhora (com avançar i girar, per exemple). Exploreu com es fa aquí i a dins la classe `spaceShip` i veureu com podem solucionar aquest problema.

També veureu que, si premem espai, creem bales noves fent servir la posició de la nau, accedint directament a les variables que conté aquest objecte amb la notació `objecte.variable`.

```
spaceShip s;
ArrayList<bullet> bulletList= new ArrayList();

void setup() {
    size(600, 450);
    s = new spaceShip(width/4, height/2, 0);
}

void draw() {
    background(0);
    for (int i = 0; i < bulletList.size(); i++) {
        bullet b = bulletList.get(i);
        b.run();
    }
    s.run();
}

void keyPressed() {
    if (key==' ') {
        bullet b = new bullet(s.p.x, s.p.y, s.ang);
        bulletList.add(b);
        println(s.ang);
    }
    if (keyCode==RIGHT)s.turningRight = true;
    if (keyCode==LEFT) s.turningLeft = true;
    if (keyCode==UP) s.movingForward = true;
    if (keyCode==DOWN) s.movingBackwards = true;
}

void keyReleased() {
    if (keyCode==RIGHT)s.turningRight = false;
    if (keyCode==LEFT) s.turningLeft = false;
    if (keyCode==UP) s.movingForward = false;
    if (keyCode==DOWN) s.movingBackwards = false;
}
```

La classe `spaceShip` introduceix un parell de conceptes importants: la translació i rotació del dibuix, i l'ús de coordenades polars i cartesianes. Ambdues les tractarem a la secció següent. Aquí, el que ens permeten és fer que la nau giri sobre si mateixa, i que les bales surtin amb la mateixa direcció a què apunta la nau, tot de manera molt ben afinada. La resta és gestionar el moviment segons les variables que s'activen i desactiven amb `keyPressed()` i `keyReleased()`.

```
class spaceShip {
    //spaceShip variables
    float ang; //initial rotation
```

```

PVector p;//initial position
float rotateVel = 0.05; // rotation speed
float radius = 2; //movement speed
int sz = 15;
//moving vars
boolean movingForward = false;
boolean movingBackwards = false;
boolean turningRight = false;
boolean turningLeft = false;

spaceShip(float _x, float _y, float _ang) {
    p = new PVector(_x, _y);
    ang = _ang;
}

void run() {
    checkMove();
    display();
}

void checkMove() {
    //check movement of spaceShip
    float moveX = radius * cos(ang);
    float moveY = radius * sin(ang);

    if (movingForward) {
        p.x += moveX;
        p.y += moveY;
    }
    if (movingBackwards) {
        p.x -= moveX;
        p.y -= moveY;
    }
    if (turningRight) {
        ang += rotateVel;
    }
    if (turningLeft) {
        ang -= rotateVel;
    }
    //keep spaceship inside (wrap around)
    if (p.x<-sz)p.x=width+sz; if (p.x>width+sz)p.x=-sz;
    if (p.y<-sz)p.y=height+sz; if (p.y>height+sz)p.y=-sz;
}

void display() {
    //draw spaceship
    fill(255);
    pushMatrix();
    translate(p.x, p.y);
    rotate(ang);
    triangle(sz, 0, -sz, -sz/2, -sz, sz/2);
    popMatrix();
}
}

```

Finalment, la classe `bullet` ens serveix per gestionar les bales. El comportament és simple. Una bala es crea i anirà tirant endavant –fent servir la conversió de coordenades polars a cartesianes– fins que surti de la finestra de Processing, moment en el qual sortirà de l' `ArrayList` i, per tant, es destruirà. És ben obvi que el proper pas d'aquest joc seria introduir un element que fes d'enemic, o obstacle, i que les bales comprovessin si hi col·lideixin.

```
class bullet {  
    //bullet variables (off-screen)  
    PVector p;//position  
    float ang;//initial angle  
    float radius = 5; //movement speed for bullet  
    float myLength = 15;  
    float myWidth = 4;  
  
    bullet(float _x, float _y, float _ang) {  
        p = new PVector(_x, _y);  
        ang = _ang;  
    }  
  
    void run() {  
        move();  
        display();  
        checkOutOfScreen();  
    }  
  
    void move() {  
        float moveX = radius * cos(ang);  
        float moveY = radius * sin(ang);  
        p.x += moveX;  
        p.y += moveY;  
    }  
  
    void display() {  
        //draw bullet  
        pushMatrix();  
        //move it first  
        translate(p.x, p.y);  
        rotate(ang);  
        fill(255, 0, 0);  
        rectMode(CENTER);  
        rect(0,0, myLength, myWidth);  
        popMatrix();  
    }  
  
    void checkOutOfScreen() {  
        if (p.x < 0 || p.x > width || p.y < 0 || p.y > height) {  
            bulletList.remove(this);  
        }  
    }  
}
```

4. Més Processing

4.5. Coordenades polars i cartesianes

Processing treballa amb coordenades cartesianes. Tenim un eix `x` i un eix `y`, i cada píxel té la seva posició. Quan dibuixem, fem referència a aquestes coordenades, i les mides que especificuem són en píxels. Una línia va d'un punt a l'altre, per un cercle definim el centre i el diàmetre, etc.

Però què passa si volem pensar una posició en graus? Per definir un gir, com a l'exemple anterior de la nau, definir una línia a partir d'un punt i una inclinació, o bé la direcció d'un objecte en moviment?

Doncs per a això ens serveix la conversió de coordenades polars a cartesianes. La idea és que les primeres les definim a partir d'un punt, un radi, i un angle, i les segones són les que ja coneixem i hem fet servir fins ara.

Comencem amb un exemple senzill: tenim un cercle que volem fer orbitar al voltant del centre de la finestra de Processing. Per fer-ho, definim un angle, inicialment zero, i un radi, que és la distància a la qual orbitarà. Al `draw()`, el que farem serà anar variant poc a poc el valor de l'angle i calcular la posició a partir d'aquest i del radi. Les funcions trigonomètriques de sinus i cosinus ens donaran la fórmula màgica: el radi multiplicat pel cosinus de l'angle, la posició `x`, i el radi multiplicat pel sinus de l'angle, la posició `y`. Això ho sumem a `width/2` i `height/2` que és el nostre punt de referència, i ja tenim la `x` i la `y` que necessitem per dibuixar l'esfera.

Com sempre, us convidem a jugar amb els valors. Com podem accelerar la rotació? I invertir-la? Com acostem o allunyem el cercle del centre? I com definiríeu una trajectòria en forma d'espiral? Tot això són petits reptes que us ajudaran molt a entendre bé què està passant.

```
PVector p = new PVector(0, 0);
int sz = 20;
float angle = 0;
float radius = 50;

void setup() {
    size(500, 200);
    background(50);
}

void draw() {

    angle +=.02;

    p.x = width/2 + (radius * cos(angle));
    p.y = height/2 + (radius * sin(angle));

    fill(255);
    ellipse(p.x, p.y, sz, sz);
}
```

A partir d'aquí, l'exemple de fer un rellotge és força obvi. Fent servir les funcions `second()`, `minute()` i `hour()` que ens diuen l'hora segons el rellotge de l'ordinador, podeu explorar aquest programa senzill que instancia tres objectes de la classe `clockThing`, o els envia el centre, radi, i si el valor que rebran té un màxim de 60 o de 24.

```
clockThing sec, min, hour;

void setup() {
    size(500, 500);
```

```

background(50);
sec = new clockThing(width/2, height/2, 200, 60);
min = new clockThing(width/2, height/2, 120, 60);
hour = new clockThing(width/2, height/2, 60, 24);
}

void draw() {
background(64);
sec.run(second());
min.run(minute());
hour.run(hour());
}

```

La classe fa poc més que l'exemple anterior, amb la diferència important que ara la posició depèn del valor temporal que rep, i és mapejada a una escala de 0 a 60, o de 0 a 12. Com que l'angle 0 a Processing ens donaria una posició al cantó dret que correspon al quart d'hora, o a les 3, restem `HALF_PI` (un quart de circumferència) al valor obtingut:

```

class clockThing {

PVector myCenter;
PVector thingPosition = new PVector(0, 0);
int thingSz = 20;
float myRadius;
int myScale;

clockThing(float centerX, float centerY, float radius, int scale) {
    myCenter = new PVector(centerX, centerY);
    myRadius = radius;
    myScale = scale;
}

void run(int value) {
    if (myScale == 60) {
        float angle = map(value, 0, myScale, 0, TWO_PI)-HALF_PI;
        display(angle);
    } else if (myScale == 24) {
        if (value > 12) {
            value -= 12;
        }
        float angle = map(value, 0, myScale/2, 0, TWO_PI)-HALF_PI;
        display(angle);
    }
}

void display(float ang) {
    stroke(255);
    noFill();
    circle(myCenter.x, myCenter.y, myRadius*2);
    thingPosition.x = myCenter.x + (myRadius * cos(ang));
    thingPosition.y = myCenter.y + (myRadius * sin(ang));
    stroke(0);
    fill(255);
}

```

```
    ellipse(thingPosition.x, thingPosition.y, thingSz, thingSz);  
}  
}
```

Proveu de canviar la representació del rellotge i fer, per exemple, unes busques més convencionals. També podeu afegir-hi color i fer que canviï segons l' hora, o pensar com representarieu un rellotge no de 12, sinó de 24 hores.

4. Més Processing

4.6 Translació i rotació

Quan comencem a treballar amb anglès a Processing, de seguida que sortim d'exemples senzills com els del subapartat anterior ens caldrà fer servir els mètodes `translate()` i `rotate()`. La causa és que, perquè Processing entengui sempre el que volem fer, caldrà no només que li diguem amb quin angle ha de fer la rotació, sinó a partir de quin punt. I com que el punt és sempre la coordenada 0,0, el que haurem de fer és moure la coordenada 0, més ben dit, el punt de referència que Processing considera que és el 0,0.

Per entendre bé la translació, mireu aquest exemple:

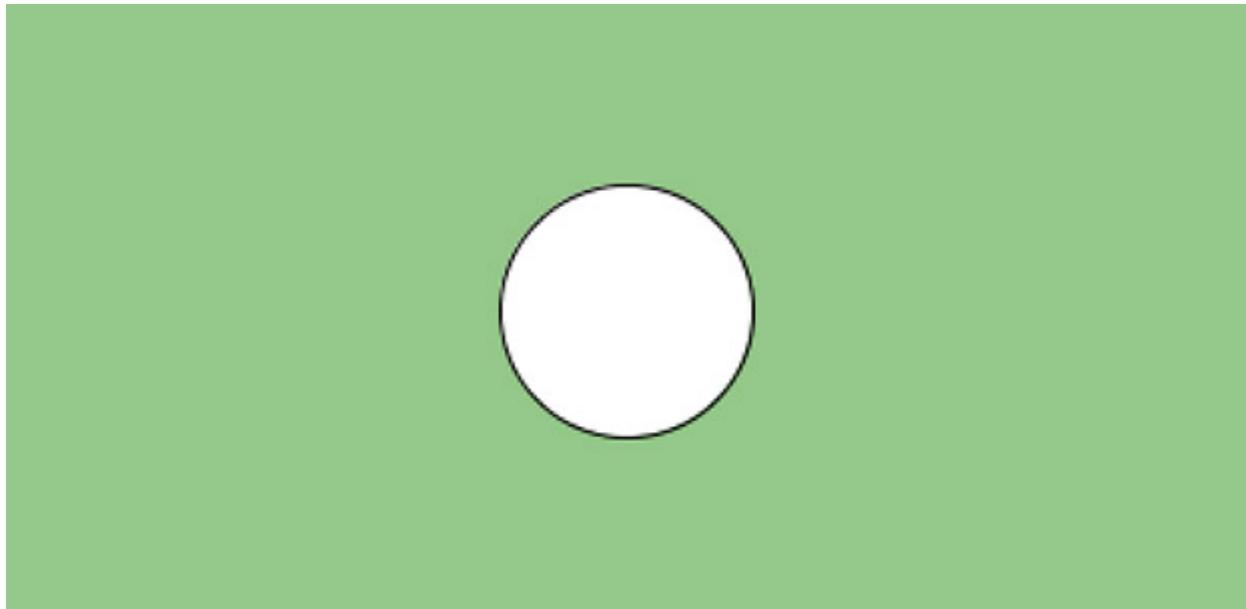


Figura 110. Exemple de translació

```
void setup(){
    size(400,200);
}

void draw(){
    background(148,200,137);
    translate(width/2,height/2);
    circle(0,0,80);
}
```

Fixeu-vos que les coordenades per dibuixar el cercle són 0,0. Això és perquè el `translate()` ha fet anar el punt de referència del dibuix (ens podem imaginar el bolígraf movent-se pel paper) al centre de la finestra. Per tant, fins que no diguem el contrari, el 0,0 ara és allà.

El pas següent és la rotació. Amb un rectangle, farem la mateixa translació i aplicarem també la rotació, en aquest cas mitjançant la funció `map()` que en realitat el que fa és una regla de tres, escalant un valor d'entre un màxim i un mínim a un altre. En aquest cas, jugarem amb la posició del ratolí a l'eix X, `mouseX`, i l'escalarem perquè enlloc de ser un valor de 0 a `width`, sigui un valor de menys 90 graus a 90 graus positius, que, com que Processing treballa en radians, els expressarem mitjançant la variable de sistema `HALF_PI` (podeu canviar `HALF_PI` per `radians(90)` i obtindreu el mateix resultat).

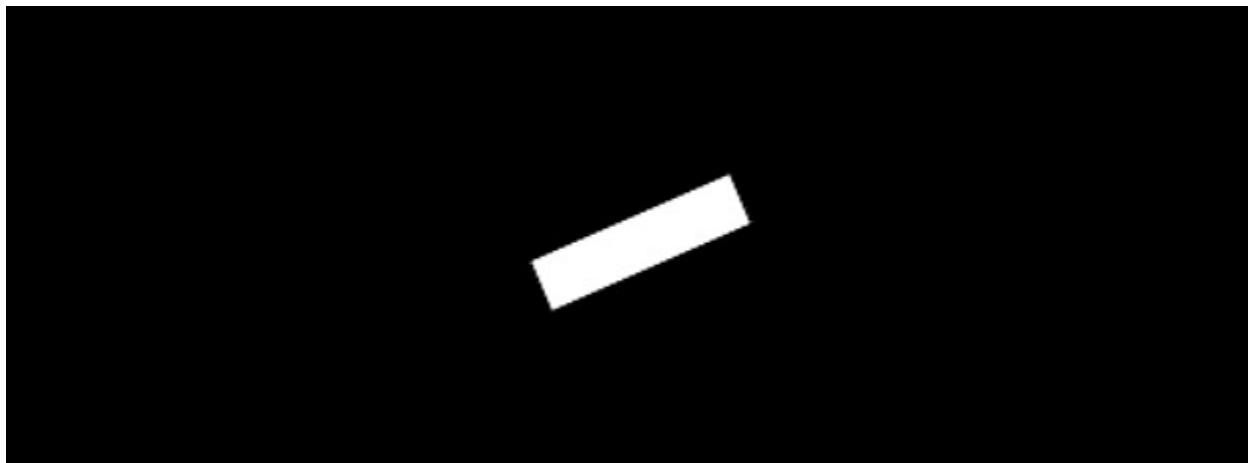


Figura 111. Exemple de rotació

```
void setup() {
    size(500, 500);
    rectMode(CENTER);
}

void draw() {
    background(0);
    translate(width/2, height/2);
    rotate(map(mouseX, 0, width, -HALF_PI, HALF_PI));
    rect(0, 0, 20, 80);
}
```

Aquests dos darrers exemples funcionen bé, però perquè els hem mantingut extremadament senzills. Hi ha una sola acció de translació i rotació dins el `draw()`, i amb això en tenim prou perquè Processing fa la feina per nosaltres: quan torna a començar el `draw()`, el punt de referència del dibuix torna a ser l'inicial.

En canvi, si fem més d'una translació o rotació, aquestes s'acumulen. És a dir, el desplaçament es fa a partir del punt on s'havia deixat, no pas del 0,0. Això pot ser molt útil per algunes coses, com els típics arbres que es fan mitjançant estructures recursives que imiten els fractals, però en canvi ens crearà problemes si volem fer translacions i rotacions de múltiples elements a cada iteració del `draw()`.

Per evitar aquests problemes, podem fer servir `pushMatrix()` i `popMatrix()`, que podem entendre com «guarda el punt de referència actual» i «retorna l'últim punt de referència guardat» respectivament. Així, si fem servir `pushMatrix()` just abans del desplaçament que fa `translate()` i `rotate()`, i `popMatrix()` just després, tornarem al punt anterior (normalment el 0,0) i podrem fer la translació i rotació tantes vegades com vulguem.

Per exemplificar-ho, tornem al sistema de partícules. Ara, enlloc de dibuixar cercles, farem servir triangles. Abans de dibuixar-ne cada un, `pushMatrix()` ens permet guardar la referència. Fem la translació al lloc que el volem dibuixar, rotació perquè giri com volem, dibuixem el triangle, i `popMatrix()` per recuperar el punt i rotació d'abans de començar.

```
pushMatrix();
translate(myPos.x, myPos.y);
rotate(ang);
triangle(mySize, 0, -mySize, -mySize/2, -mySize, mySize/2);
popMatrix();
```

Pel que fa a l'angle, farem servir el mètode `atan2()`, que ens dona l'angle d'un punt respecte un altre, i que es pot fer servir per fer apuntar, o mirar, un objecte respecte a un altre. En aquest cas, farem que cada triangle miri cap al ratolí en tot moment:

```
float ang = atan2(mouseY-myPos.y, mouseX-myPos.x);
```

Executeu el codi de sota i fixeu-vos com, si movem el ratolí d'una banda a una altra, no només canvia el punt d'origen de les partícules com abans, sinó que a més cada una es va girant per estar sempre mirant en la direcció on tenim el ratolí.



Figura 112. Sistema de partícules

```
ArrayList particleList = new ArrayList();

void setup() {
    size(800, 450);
    cursor(CROSS);
}

void draw() {
    background(0);

    for (int i = 0; i < 10; i++) {
        particle p = new particle(mouseX, mouseY);
        particleList.add(p);
    }

    for (int i = 0; i < particleList.size(); i++) {
        particle p = (particle) particleList.get(i);
        p.run();
    }
}
```

```

class particle {

    PVector myPos, myVel, myAcc;
    float mySize;

    particle(float x, float y) {
        myPos = new PVector(x, y);
        mySize = 5;
        myVel = new PVector(random(-1, 1), random(-1, 1));
        myAcc = new PVector(0, 0.01);
    }

    void run() {
        update();
        display();
    }

    void update() {
        myVel.add(myAcc);
        myPos.add(myVel);
        if (myPos.y > height) {
            particleList.remove(this);
        }
    }

    void display() {
        float ang = atan2(mouseY-myPos.y, mouseX-myPos.x);
        noStroke();
        fill(255, 192);
        pushMatrix();
        translate(myPos.x, myPos.y);
        rotate(ang);
        triangle(mySize, 0, -mySize, -mySize/2, -mySize, mySize/2);
        popMatrix();
    }
}

```

A partir d'aquí, com sempre, us convidem a explorar aquest exemple i d'altres. A anar a la referència de Processing i provar funcions que no havíeu provat encara, a buscar referents, tutorials i exemples de codi amb els quals anar aprenent i inspirant-vos. La programació creativa no és més, ni menys, que tot això.

4. Més Processing

Bibliografia

- Bohnacker, H., Gross, B., Laub, J., & Lazzeroni, C. (2012). *Generative Design*. Princeton Architectural Press.
- Maeda, J. (2004). *Creative Code: Aesthetics and Computation*. Nova York: Thames & Hudson.
- Reas, C., & Fry, B. (2014). *Processing: A Programming Handbook for Visual Designers and Artists*. Cambridge, Massachusetts: MIT Press.
- Reas, C., & Fry, B. (2015). *Make: Getting Started with Processing, Second Edition*. O'Reilly.
- Runberg, D. (2015). *The SparkFun Guide to Processing*. San Francisco, CA: No Starch Press.
- Rushkoff, D. (2010). *Program or Be Programmed. Ten Commands for a Digital Age*. Berkeley: Soft Skull Press.
- Shiffman, D. (2012). *The Nature of Code: Simulating Natural Systems with Processing*. The Nature of Code.
- Shiffman, D. (2008). *Learning Processing. A Beginner's Guide to Programming Images, Animation, and Interaction*. Burlington, MA: Morgan Kaufmann.
- Zhang, Y., & Funk, Yu Zhang, M. F. (2021). *Coding Art: The Four Steps to Creative Programming with the Processing Language*. Apress.

Entorns, repositoris i galeries de projectes

CreativeApplications.Net: <https://www.creativeapplications.net/>

Open Processing: <https://openprocessing.org/>

p5js: <https://p5js.org/>

Processing: <https://processing.org/>

The Processing Foundation: <https://processingfoundation.org/projects>

(*) Contingut disponible només en web.