En què consisteix el disseny generatiu

PID_00267102

David Casacuberta

Temps mínim de dedicació recomanat: 2 hores

Universitat Oberta de Catalunya

David Casacuberta

Com a professor de Filosofia de la ciència a la Universitat Autònoma de Barcelona (UAB), la seva línia de recerca actual són els impactes socials i cognitius de les TIC, tema sobre el qual ha publicat diversos llibres i articles.

Actualment és membre del Grup de Treball d'Ètica, Seguretat i Regulació de Bioinformàtica Barcelona i investigador del Grup d'Estudis Humanístics en Ciència i Tecnologia (GEHUCT). També és codirector del màster de Disseny i Direcció de Projectes per a Internet d'Elisava i participa com a professor en diversos postgraus de gestió cultural, teoria de l'art contemporani i disseny de tecnologies digitals.

Ha rebut el premi Eusebi Colomer de la Fundació Epson al millor assaig sobre els aspectes socials, antropològics, filosòfics o ètics relacionats amb la nova societat tecnològica amb el llibre *Creació col·lectiva*. També ha guanyat el premi Enginyo 400, organitzat pel Ministeri de Cultura i la Societat Estatal de Commemoracions Culturals, al millor projecte de net.art amb l'obra *X-Reloaded* (en col·laboració amb Marco Bellonzi).

L'encàrrec i la creació d'aquest recurs d'aprenentatge UOC han estat coordinats pel professor: Quelic Berga (2019)

Primera edició: setembre 2019
Autoria: David Casacuberta

Av. Tibidabo, 39-43, 08035 Barcelona
Realització editorial: FUOC

Índex

1. Què és el disseny generatiu

El disseny gràfic va iniciar una transformació profunda amb l'aparició de l'ordinador personal en els anys vuitanta –especialment el Macintosh, amb programes específics per al dibuix o l'edició de text, la interfície gràfica i la inclusió de tipografies. Mentre els professionals del disseny treballaven dur per reconvertir la seva creativitat i habilitats en el nou entorn digital, altres professionals, amb sòlids fonaments en enginyeria i programació, imaginaven una altra manera d'entendre l'art i el disseny, que consistia a crear processos automatitzats de disseny basats en matemàtiques. Naixia, així, el disseny generatiu.

Com podem definir el disseny generatiu? La Viquipèdia connecta el disseny generatiu amb eines associades a l'arquitectura i des d'un model concret de com automatitzar els dissenys, de manera que no és gaire útil. Una cerca per internet us conduirà per camins similars en els quals cada definició s'organitza des d'una disciplina concreta i a partir d'una tècnica específica. La meva favorita és la de Lars Hesellgren, que va afirmar que «el disseny generatiu no busca crear un edifici. Vol dissenyar el sistema que construeix un edifici». Encara que una vegada més està basat en l'arquitectura, crec que apunta a la distinció més rellevant per entendre les particularitats del disseny generatiu. En síntesi, ens està dient que l'objectiu final del dissenyador és un altre: ja no volem fer un cartell, una torradora o un hospital, volem crear un sistema que serveixi per generar diferents cartells, torradores o hospitals per poder decidir quin ens sembla més apropiat o, simplement, oferir-los tots i que el client esculli. Els diversos sistemes que creiem que seran els diferents models de disseny generatiu amb els quals volem treballar.

1.1. Disseny generatiu analògic

És important no confondre disseny amb ordinador amb disseny generatiu. Bona part del disseny gràfic que es fa actualment es fa amb ordinador. Fins i tot la dissenyadora més hàbil i analògica del món probablement acabarà introduint l'ordinador en algun moment del procés, encara que només sigui per escanejar el dibuix que ha fet i enviar-lo a la impremta. L'ordinador facilita certament el procés d'automatitzar sistemes de disseny, però no és obligatori.

Considereu per un moment un dels llibres més antics de la història: l'*Yijing*. Aquest llibre oracular consta de 64 hexagrames, totes les combinacions possibles de sis elements d'una línia sencera o partida. Cadascuna d'aquestes combinacions té un nom. Així, el primer hexagrama, format per sis línies contínues, es diu «El poder»; el segon, amb sis línies partides, representa la «Recepció», etc. Cada hexagrama va acompanyat d'un text que ofereix consells sobre com actuar. No us ho imagineu com un horòscop que prediu que «et trobaràs per

sorpresa una vella amistat que feia anys que no veies». Són més aviat propostes genèriques, de múltiple interpretació, com «El moviment del Cel és poderós. De la mateixa manera, el noble es va enfortint sense descans».

Figura 1. Alguns hexagrames de l'Yijing

—— —— —— —— —— ——	— — — — — — — — — — — —	— — — — — — —— —— ——
El poder	Recepció	Pau

Per rebre consell de l'*Yijing* el lector primer formula a la seva ment la pregunta a la qual vol obtenir resposta. Per exemple es qüestiona: «Com haig d'afrontar els exercicis pràctics d'aquest mòdul de disseny generatiu?». Tira tres monedes sis vegades i apunta les cares i creus que apareguin. En funció de quantes cares i creus sorgeixin, el lector apunta una línia sencera o partida. Això li generarà un hexagrama que serà la resposta a la seva pregunta. D'una manera molt senzilla hem creat així un sistema de lectura no convencional. En lloc de llegir el llibre de principi a fi, hem ideat un sistema que ens selecciona un hexagrama a l'atzar. De fet, el sistema d'endivinació és una mica més complex que el que hem explicat aquí, però com que això no és un text de filosofia oriental, ho podem deixar així.

Així mateix, no hi ha res més fàcil que inventar-se sistemes analògics per dissenyar generativament. Podem parametritzar aleatòriament el disseny d'un logo, fent que el color de fons es decideixi pel llançament d'un dau, el tipus de tipografia per un altre, o associar-lo a altres esdeveniments atzarosos, com el temps que fa avui, fer una ràpida ullada a un rellotge digital i utilitzar els segons que marca, etc. Uns paràgrafs més avall teniu un exemple de disseny generatiu analògic i a la bibliografia algunes referències curioses per explorar el disseny generatiu analògic.

De totes maneres, hem de veure aquests exemples com el que són, curiositats, ja que traurem molt més partit del nostre sistema de crear dissenys gràfics si els implementem en un programari. Tanmateix, ens ha semblat important insistir en aquest tema, atès que apunta a una qüestió central sobre què és el disseny generatiu. Com dèiem en la definició anterior, no es tracta de fer un edifici, sinó de crear un sistema que s'encarregui de fer edificis. En un context de disseny gràfic, volem un sistema generador de cartells que ens generi tot tipus de cartells. Per fer-ho, en lloc de crear nosaltres el cartell amb un programari com Photoshop o InDesign (o amb paper i retoladors), el que fem és crear un

seguit d'instruccions que defineixin el procés pel qual es crearà el cartell. En el llenguatge de les ciències de la computació aquesta col·lecció d'instruccions s'anomena «algorisme».

1.2. Què és un algorisme?

Un algorisme no ha de ser una cosa misteriosa i matemàticament inescrutable.

> Un algorisme són simplement una sèrie d'instruccions que donem a un agent (humà o digital) perquè dugui a terme una tasca.

Un programari per jugar als escacs contra un ordinador està generat a partir d'un algorisme, però una recepta en un llibre de receptes és també un algorisme.

> L'únic requisit realment vital a l'hora de crear un algorisme és que l'agent sigui capaç d'interpretar cada instrucció de l'algorisme de manera unívoca, sense crear confusió.

Així, si en una recepta posem: «Poseu 2 l d'aigua a bullir i afegiu-hi un polsim de sal», el xef que seguirà la recepta sabrà què ha de fer. En canvi, si diem: «Poseu ous, llet, farina, iogurt i ratlladura de llimona en un bol i barrregeu-ho fins que quedi la massa del pastís, fiqueu-la al forn i traieu-la'n quan estigui feta», aquesta recepta és bàsicament inútil, ja que no especifica la quantitat d'ous, llet i farina que cal posar-hi, o el temps que ha de ser al forn, per la qual cosa no podrem executar les instruccions amb seguretat.

Els humans tolerem l'ambigüitat molt millor que els ordinadors, per la qual cosa un algorisme que especifiqui un sistema de disseny generatiu haurà de ser molt més precís que una recepta de cuina. Aquesta serà una de les habilitats clau que aprendreu al llarg d'aquest curs: especificar un seguit d'instruccions, objectes i paràmetres perquè el vostre ordinador dissenyi per vosaltres.

Independentment de com s'executa un sistema de disseny generatiu, si és un algorisme analògic que interpreta un humà o un llenguatge de programació perquè l'executi un ordinador, podem caracteritzar-ho formalment de la manera següent:

Un sistema de disseny gràfic generatiu està format pels components següents:

1) Una llista dels diferents objectes gràfics que poden aparèixer al nostre disseny.

2) Una llista de les diverses propietats que aquests objectes poden tenir.

A 1 i 2 els anomenarem «ontologia» ja que expliciten el tipus d'objectes que apareixeran al nostre disseny i les propietats que tenen.

3) Un seguit de regles que indiquen, en funció del que ja ha aparegut en el disseny gràfic, quin(s) nou(s) element(s) cal incloure, i quines propietats han de mostrar. Aquestes regles inclouran també instruccions de quan el disseny es considera acabat i s'ha d'aturar l'execució del procés.

4) Un agent que executi les regles i creï el disseny generatiu.

5) Uns objectes (físics o virtuals) que l'agent usi per seguir les regles i desenvolupar el disseny.

Vegem un exemple senzill per dissenyar el logo generatiu de la UOC.

Ontologia:

- El nom «UOC», que pot aparèixer:
  - En sis tipografies diferents: Times New Roman, Helvetica, Futura, Optima, Courier o Baskerville.
  - Dues grandàries diferenciades: petit o gran.

- Un contenidor, que pot ser:
  - Cercle, dos cercles concèntrics, triangle isòsceles, triangle equilàter, triangle rectangle, quadrat, rectangle, pentàgon, hexàgon, heptàgon, octògon.
  - Aquest contenidor pot ser: vermell, blau, verd, groc, taronja o blanc.
  - Si és blanc, llavors la línia que el defineix pot ser vermella, blava, verda, groga, taronja o violeta.

Regles:

R.1 Llançament d'un dau per al nom:
1 = Times New Roman
2 = Helvetica
3 = Futura
4 = Optima
5 = Courier
6 = Baskerville

R.2 Llançament d'una moneda per a grandària del nom:
Cara = Petit
Creu = Gran

R.3 Llançament de dos daus per al contenidor:
2 = Cercle
3 = Dos cercles concèntrics
4 = Triangle isòsceles
etc.

R.4 Llançament d'un dau per al color del contenidor:
1 = vermell
2 = blau
3 = verd
4 = groc
5 = taronja
6 = blanc

R.5 Si el llançament de daus en R.4 = 6 (color blanc) cal anar a R.7.

Si no:

R.6 Aturar l'algorisme.

R.7 Llançament d'un dau per decidir el color de la línia del contenidor:
1 = vermell
2 = blau
3 = verd
4 = groc
5 = taronja
6 = violeta

R.8 Aturar l'algorisme.

Executor: Humà

Objectes per a l'executor:

- Dos daus de sis cares
- Una moneda d'un euro
- InDesign

Vegem com funcionaria el sistema:

Llanço un dau i trec un 3. Per tant, el nom «UOC» anirà en Futura. Seguidament llanço una moneda a l'aire. Surt una cara, així que la grandària del nom «UOC» serà gran.

Ara llanço dos daus i surt un 2. El contenidor serà un cercle.

Llanço un dau i surt un 6. Això vol dir que el cercle serà blanc, amb la qual cosa haig d'aplicar la regla R.7. Tiro una vegada més el dau i obtinc un 3, de manera que el color de la línia serà verda.

Així doncs, un cop executat l'algorisme, obtindria aquest logo:

Figura 2. Logo UOC

Sí, estem totalment d'acord. No guanyaré cap premi nacional de disseny amb aquest logo, però ens ha servit per exemplificar què és un disseny generatiu i comprovar que no és res misteriós ni cal tenir un doctorat en intel·ligència artificial per desenvolupar dissenys generatius.

2. Història i evolució del disseny generatiu

En aquest apartat descriurem de manera breu com ha evolucionat i s'ha transformat el disseny generatiu. Això ens servirà també per oferir un principi de classificació dels diferents tipus de disseny generatiu que un es pot trobar actualment i les característiques de cadascun. No es tracta de fer un exercici acadèmic historicista, ni tampoc d'oferir una taxonomia sistemàtica. És simplement una primera visió de les diverses formes en les quals el disseny generatiu es pot expressar.

2.1. Enter Maeda

Encara que hem parlat a la introducció de l'ordinador personal com el punt d'inflexió de la transformació del disseny, la veritat és que el disseny generatiu és, de fet, anterior a l'ordinador personal. Enginyers i programadors que van començar a experimentar amb pantalles i ordinadors als anys seixanta ja van albirar les possibilitats que els entorns digitals oferien a l'art i el disseny, i, sent de formació matemàtica, els seus projectes i propostes es basaven sobretot a parametritzar formes geomètriques connectades, a fi de crear poderoses imatges de polígons enfilats, estructures fractals, etc. Aquestes pràctiques són precedents interessants que marquen el camí que caldrà seguir als futurs dissenyadors.

Tanmateix, hi ha una figura clau, un dissenyador que marca un abans i un després del disseny generatiu. Es tracta de John Maeda. John Maeda és un dissenyador amb formació en ciències de la computació i que als anys noranta es va adonar d'una sèrie de problemes en el desenvolupament de la professió de dissenyador gràfic. Maeda va observar, d'una banda, la inexistència d'un llenguatge comú entre dissenyadors i desenvolupadors de programari. Les converses típiques en aquella època entre un dissenyador i un programador que col·laboraven en el desenvolupament d'un *website* eren de l'estil de: «Vull una estructura de continguts similar a aquesta pàgina de diari, amb la foto aquí, un encapçalat allà, etc.», a la qual cosa el programador responia: «Això és tècnicament impossible». A causa d'aquesta incapacitat de comunicar-se, els dissenyadors estaven limitats pel que el programari els deixava fer. El cartell d'una pel·lícula fet amb Photoshop o la *homepage* d'un *website* desenvolupat amb Dreamweaver no era exactament com el grafista esperava que fos, sinó com el programador havia concebut el programari. Aquest fenomen, Maeda el va batejar com l'«autocràcia del PostScript». És a dir, l'acte de dissenyar quedava limitat per les característiques que els desenvolupadors de programari donaven als seus programes.

Per a Maeda, l'única manera d'evitar aquesta «autocràcia del PostScript» era que els dissenyadors es construïssin les seves pròpies eines. És a dir, que aprenguessin a programar. Però, a causa de la falta de llenguatge comú entre desenvolupadors de programari i dissenyadors gràfics, els llenguatges de programació que hi havia als noranta convertien els processos de creació gràfica en una cosa molt desmanegada i complexa, associada a fórmules matemàtiques difícils. Per evitar-ho, Maeda va desenvolupar el seu propi llenguatge de programació, que va batejar com a Design by Numbers.

2.2. De Design by Numbers a Processing

Design by Numbers era un llenguatge pensat per ensenyar conceptes de programació a dissenyadors, i així facilitar la conversa amb desenvolupadors de programari oferint un llenguatge comú.

No estava realment concebut per poder desenvolupar creacions funcionals que un professional del disseny pogués oferir a un client. Però Design by Numbers oferia una novetat desconeguda fins llavors en el món dels llenguatges de programació: a Design by Numbers, per traçar una línia, un cercle o un polígon i donar-los color, no calia desenvolupar complexes funcions algorítmiques. Totes aquestes accions estaven recollides en instruccions primitives. Així, una instrucció com ara «circle (56, 46, 55)» dibuixava un cercle amb el centre a les coordenades x=56,i=46 amb un radi de 55 píxels.

Dit d'una altra manera, Design by Numbers era el primer llenguatge de programació en el qual les instruccions primitives eren conceptes que els dissenyadors coneixien a la perfecció i, per tant, era la manera perfecta d'introduir un professional del disseny gràfic al món de la programació, i així poder plantejar-se aprendre a programar les seves pròpies eines.

Dos estudiants de Maeda, Ben Fry i Casey Reves, fascinats per Design by Numbers, van decidir fer un pas més i desenvolupar un llenguatge de programació que fos realment funcional i que permetés a un professional del disseny gràfic crear els seus propis dissenys en dues dimensions, així com tot tipus de projectes de disseny d'interacció.

Va néixer així Processing, un llenguatge de programació molt més fàcil d'aprendre i aplicar en el disseny gràfic i que permet a un dissenyador crear les seves pròpies eines i no dependre de l'autocràcia del Post-Script, no estar limitat pel que un programari comercial li deixa fer o no.

Originalment, Processing era molt bàsic, era complex fins i tot crear PDF professionals del disseny que un acabava de fer, però en ser un sistema de codi obert, lliure i gratuït, molta més gent es va unir a col·laborar amb el projecte

per crear nous algorismes, funcions i rutines, amb la qual cosa ara a Processing podem generar cartells professionals, maquetar llibres, desenvolupar complexes instal·lacions interactives, processar dades de sensors i càmeres per crear videojocs, etc. Si el Processing original es basava en el llenguatge de programació Java, ara també hi ha versions per a Javascript que faciliten el desenvolupament d'aplicacions interactives a la web, o la versió a Python, un altre llenguatge de programació que molts desenvolupadors de programari utilitzen.

Processing serà també el llenguatge de programació que utilitzarem en aquest curs, ja que és, amb diferència, el codi més ben estructurat per fer disseny generatiu, i on resulta més fàcil fer-ho.

2.3. Disseny computacional

Maeda va anomenar el seu estil de dissenyar «disseny computacional», per distingir-lo del disseny clàssic i del *Design Thinking*. Al *Design Thinking* qui dissenya usa l'empatia per establir una connexió amb el client i treballar junts per aconseguir un disseny que realment resolgui els problemes que se li plantegen al client. En el disseny clàssic, el professional del disseny és l'expert que analitza el problema i ofereix la solució que li sembla més adequada, considerant el problema que cal analitzar i el context concret que el defineix. Aquest professional del disseny probablement treballarà amb ordinador la majoria de les fases del procés.

Però el disseny computacional no és simplement disseny fet amb ordinador. És disseny que aprofita les capacitats de l'ordinador per generar variacions infinites d'un tema i construir eines específiques que permetin tractar els problemes d'una manera individualitzada i seguidament automatitzar aquestes solucions.

> En el disseny computacional, el grafista deixa de ser un creador d'objectes visuals (cartells, portades, *homepages*, etc.) per ser el creador d'instruccions que generin aquests objectes gràfics de manera automàtica.

L'analogia amb la música ens pot ser útil. En música clàssica distingim entre l'intèrpret i el compositor. El compositor agafa una partitura i omple el pentagrama de símbols (compassos, claus, notes musicals, etc.). L'intèrpret agafa aquesta partitura i l'executa, fent possible al públic poder escoltar la composició de l'autor. Així, Johann Sebastian Bach compon en la seva ment una sonata només per a violí. La transcriu a paper pautat i així queda convertida en instruccions. Un temps després –fins i tot segles– un intèrpret agafa aquesta partitura i l'executa, fent així audible la música de Bach per a nosaltres un cop més.

El disseny computacional funciona de la mateixa manera. El grafista ja no genera ell mateix un cartell, clicant botons i arrossegant objectes al Photoshop. Escriu un programa en el qual es descriu un algorisme per al tipus de cartell que volem que es generi. Aquesta «partitura gràfica» la introduïm a l'ordinador, que fa de «intèrpret» i fa visible el cartell per a nosaltres.

2.4. Un cop de daus

La metàfora musical, encara que sens dubte és útil, pot confondre'ns i limitar excessivament les possibilitats del disseny generatiu. Quan el violinista interpreta la sonata de Bach, la seva execució està molt ben delimitada. Quan interpreta per al públic, no es pot inventar notes, no pot repetir un passatge que li agradi especialment tret que la partitura ho indiqui expressament, no pot accelerar una part que consideri especialment emocionant i, en general, ha de respectar totes les indicacions que Bach va deixar. Si penséssim que el disseny generatiu és exactament equivalent a compondre una simfonia, el disseny generatiu no seria gaire diferent del disseny clàssic: hi ha una solució concreta al problema que el client planteja. L'única diferència és que, en comptes del fet que la solució la crea el grafista, directament la fa l'algorisme.

Afortunadament, el disseny generatiu s'assembla més al jazz: l'executor de la peça pot improvisar, inventar-se notes, accelerar, frenar, repetir… Hi ha un esquema que cal seguir, però hi ha també llibertat creativa.

Els ordinadors no tenen llibertat creativa *strictu sensu*, és clar, però podem aconseguir una cosa similar afegint atzar als algorismes que especifiquen com generar la nostra proposta gràfica. En l'exemple descrit a l'apartat 1.2. d'aquest mòdul aconseguíem donar a l'algorisme certa llibertat creativa en fer que l'elecció de la tipografia, la forma geomètrica del contenidor i el color depenguessin del llançament de diversos daus.

La majoria de llenguatges de programació inclouen instruccions per generar aquests números aleatoris i disposar-los a la franja que vulguem.

Generació de números aleatoris

Un ordinador segueix les instruccions de manera determinista, així que en realitat no podem comptar-hi per generar un autèntic atzar, però hi ha fórmules matemàtiques complexes que ens permeten crear números pseudoaleatoris: números que segueixen un procés de generació tan complex que per als efectes pràctics del disseny generatiu són tan bons com si fossin generats per un procés realment aleatori, com el llançament de daus.

Imaginem que tenim 100 fotografies de gats que usarem en un cartell, i volem que sigui l'atzar el que decideixi quina foto seleccionarem. A Processing tenim instruccions molt senzilles perquè l'ordinador associï un número de l'1 al 100 a cada foto de gats, generi un número aleatori de l'1 al 100 (com si llancéssim un dau de 100 cares) i posi la foto seleccionada a l'atzar. El procés

podria continuar, i podríem decidir aleatòriament en quines coordenades hi ha d'haver la foto del gat, i un altre número aleatori decidiria la grandària final de la foto, etc.

2.5. Disseny evolutiu

Recordem com funciona la selecció natural. Fa milions d'anys, a Àfrica, els parents propers de les actuals girafes tenien un coll curt. D'entre els milions d'avantpassats de les girafes que hi havia llavors, alguns tenien el coll més llarg, de manera que podien menjar fulles de branques d'arbres on els seus congèneres no arribaven. Aquesta habilitat extra en va facilitar la supervivència i va ajudar al fet que tinguessin més cries. Les cries s'assemblaven a ells, i bona part tenien també el coll més llarg. Algunes d'aquestes cries van tenir cries que tenien el coll una mica més llarg encara, amb la qual cosa es podien alimentar encara millor, i així tenien més descendència. A cada generació, el tret de tenir el coll més llarg es va anar seleccionant en ser útil en el context concret de la sabana, i així, milions d'anys després, tenim les nostres girafes de colls llarguíssims.

Els algorismes genètics són una manera alternativa de programar en la qual el desenvolupador del programari imita aquest procés d'evolució per selecció i crea diferents programes a l'atzar, centenars, perquè processin un problema. Al principi els programes ho faran malament, i la solució que oferiran estarà a anys llum de la resposta correcta. Tanmateix, en treballar amb centenars de programes, sempre n'hi haurà algun que ho farà una mica millor que els altres. Aquests programes queden seleccionats i la resta eliminats. Es creuen llavors els programes supervivents entre si, imitant el procés de reproducció sexual, i així es crea una nova generació de programes. D'aquests programes, n'hi haurà uns quants que ho seguiran fent malament, però s'aproparan una mica més a la solució correcta. Seran els programes que sobreviuran i es creuaran. Al cap d'unes hores –afortunadament aquí no cal esperar milions d'anys com en biologia– tindrem programes que solucionaran de manera òptima el problema que estàvem analitzant.

El disseny evolutiu utilitza aquesta tècnica per generar sistemes de creació gràfica. Imaginem que volem un fons multicolor de punts aleatoris per a un cartell que estem dissenyant. Volem que el fons mostri una certa estructura –no volem aleatorietat simplement–, però tampoc volem res rígid: ens venen de gust zones de colors marcades, però amb transicions sorprenents. En lloc de fer-ho a mà, desenvoluparem un petit programa de disseny evolutiu. Aquest programa generarà 100 imatges aleatòries de píxels multicolors i ens les presentarà en pantalla. Al principi, totes estaran molt lluny de la imatge que tenim en ment, però segur que algunes s'assemblaran més que unes altres. Després d'examinar-les, seleccionarem les quatre que ens han semblat més interessants i el programa ens crearà una nova generació d'imatges, resultat de combinar les quatre imatges que hem seleccionat de manera aleatòria. D'aquesta iteració sorgeixen 100 noves imatges. Tornem a seleccionar les quatre més prometedo-

res i generem una nova llista de 100 imatges. Al cap d'unes quantes iteracions tindrem una imatge amb exactament la càrrega d'estructura i aleatorietat que buscàvem i sense haver de fer ni un sol dibuix!

També podem quedar-nos en una posició intermèdia, per exemple crear nosaltres 25 imatges de sortida bàsica i llavors utilitzar disseny evolutiu que combini a l'atzar aquestes creacions nostres fins a aconseguir una cosa nova, una cosa que recordi els nostres dissenys originals però que ha estat transformat i ha evolucionat gràcies a l'algorisme i les nostres seleccions.

La idea d'utilitzar els algorismes genètics per fer disseny i art evolutiu la devem sobretot a William Lathan i Karl Sims, que van desenvolupar impactants projectes visuals interactius que van ajudar a estendre aquesta manera d'entendre el disseny generatiu en la professió i també en l'art contemporani. En la bibliografia trobareu referències a obres seves.

2.6. Intel·ligència artificial

Si observem les diverses fases del disseny generatiu que hem estat veient fins ara, observarem que la tendència general és desenvolupar sistemes cada vegada més autònoms en els quals l'individu que dissenya té cada vegada menys capacitat de decisió. En el disseny computacional, l'ordinador és el que fa el disseny, però aquest disseny ha estat pensat del tot per l'autor, que ha decidit les propietats de cadascun dels objectes gràfics que formen la seva proposta. A través de l'atzar deixem que el programa improvisi, i aquesta improvisació es converteix en el mecanisme central de creació en el disseny evolutiu.

Seguint aquesta tendència, el següent pas lògic és deixar les decisions a mans del sistema, de manera que l'autor passa a ser un mer constructor d'un algorisme generatiu. Aquest és el pas en el qual ens trobem ara, en què dissenyadors i artistes creen programes d'intel·ligència artificial que s'ocupen de manera bàsicament autònoma de generar les obres.

Un exemple clàssic és el programa Aaron de Harold Cohen. Aaron és un sistema artificial que controla un traçador modificat que ha desenvolupat el seu propi estil de pintura i que és capaç de pintar del natural, desenvolupant bodegons, retrats, etc. A diferència dels casos anteriors, Cohen –l'autor humà– no intervé directament en el procés creatiu. En el seu moment va desenvolupar unes regles de com el sistema aprendria a dibuixar a partir de la informació oferta per una càmera, i ara el programa Aaron s'encarrega de plasmar aquesta imatge en un llenç. I Aaron no és simplement una curiositat de l'enginyeria. Els quadres que en sorgeixen estan prou ben considerats per ser exposats en museus i galeries d'art, i estan molt cotitzats entre els col·leccionistes.

Actualment hi ha una sèrie de projectes d'art i disseny generatiu fascinants que utilitzen xarxes neuronals. En aquests projectes s'ofereix al programa un seguit d'exemples que es volen desenvolupar, per exemple, rostres de celebritats

humanes o quadres famosos de la història de l'art, i el sistema, a través d'un complex procés d'aprenentatge en el qual es localitzen patrons rellevants a les imatges, intenta generar els seus propis exemples de quadres o rostres famosos.

Xarxa neuronal

Entenem per xarxa neuronal una forma de produir algorismes inspirada en com treballen les neurones. Bàsicament, es codifiquen una sèrie d'*inputs* que volem processar en una primera capa de neurones que representen aquestes entrades de manera numèrica. Aquesta primera capa de neurones està connectada a una segona capa que processa aquestes entrades. Cada neurona de la primera capa està connectada amb les neurones de la segona, imitant les sinapsis que connecten les neurones entre si. Cada connexió entre neurones té associat un «pes», és a dir, un número que amplifica o disminueix el senyal que genera una neurona. Després hi pot haver més capes o no de neurones intermèdies, fins que arribem a una capa final que ofereix l'*output*, la resposta associada a l'*input*.

Les xarxes neuronals s'«entrenen» a través d'un seguit d'algorismes que comproven la distància que hi ha entre la resposta correcta associada a un *input* i la que la xarxa genera. Aquests algorismes modifiquen els pesos de manera progressiva fins que la xarxa és capaç de generalitzar el problema prou perquè hi hagi una bona correspondència entre l'*input* i l'*output*.

Per exemple, per ensenyar a una xarxa neuronal a reconèixer lletres manuscrites donem com a *input* gifs de lletres escanejades i com a *output*, el codi ASCII d'aquesta lletra. Després d'una sèrie de processos d'entrenament tindrem una xarxa amb uns pesos de connexió optimitzats que permeten reconèixer diverses «a» manuscrites com una lletra «a» i així ser capaç de transcriure un text escrit a mà en un document de text.

El tipus d'algorisme més utilitzat en aquests processos rep el nom de GAN, sigles de Generative Adversarial Networks. La idea és treballar amb dues xarxes neuronals: una d'elles rep l'*input* d'una imatge i intenta determinar si pertany al conjunt d'imatges d'entrenament –diferents cares de famosos, per exemple. L'altra xarxa neuronal genera imatges intentant passar el filtre de la primera. El procés de creació i avaluació es va repetint, millorant el funcionament de les dues xarxes neuronals, fins que el sistema està convençut que ha creat una cosa prou bona per passar per un exemple de les imatges seleccionades.

De fet, és una mica com el disseny evolutiu que hem explicat a l'apartat anterior, però en el qual l'humà ja no és necessari atès que la fase d'avaluació la duu a terme una altra xarxa neuronal.

En aquesta última iteració del disseny generatiu, la persona encarregada del projecte només té una funció: decidir quin tipus d'imatge vol generar: es tracta de generar rostres humans creïbles per a un projecte de creació de bots en xarxes socials? O potser volem un sistema que creï retrats a l'estil de Cezanne? L'única tasca creativa aquí és fer la selecció del tipus d'imatges que ens agradaria veure replicades. L'artista/dissenyador no necessita ni tan sols crear l'algorisme, ja que hi ha algorismes en el domini públic plenament funcionals i que treballen de forma molt genèrica, de manera que només són necessàries nocions de programació per saber on posar les imatges, de quina manera han d'anar organitzades i quins paràmetres cal tocar per maximitzar el resultat final.

3. Què ens aporta el disseny generatiu?

Superficialment, podríem dir que fer projectes de disseny generatiu ens fa semblar moderns. És un avantatge competitiu que tindrem davant d'altres professionals més convencionals. Podem parlar d'intel·ligència artificial, disseny evolutiu i altres termes que sonen *cool*, i ens donarà una aurèola de creador innovador que està a anys llum de la competència.

Afortunadament, hi ha altres avantatges molt més clars i funcionals en l'ús del disseny generatiu, que aporten veritable valor als nostres dissenys, més enllà de com de moderns ens trobin.

D'una banda, la possibilitat de parametritzar un disseny ens permet dur a terme una sèrie d'accions que en el disseny convencional són molt complexes. La més clara i impactant és la capacitat de crear obres úniques. El disseny és un producte derivat de la Revolució Industrial, i pensa en les seves creacions com a objectes repetits, fets en sèrie, amb les facilitats i els problemes que això comporta, tal com va analitzar Walter Benjamin en el seu famós text *L'art en l'època de la reproductibilitat tècnica*. Si el dissenyador convencional està lligat a la forma de reproducció industrial i a plantejar-se crear sèries de cartells, portades de discos i llibres, etc., un grafista generatiu, basant-se en les tècniques de creació i distribució digitals, pot crear 1.000 cartells o 1.000 portades de llibres i que totes siguin diferents entre si. En lloc de deixar escollir a un client entre tres opcions de logos, pot presentar-se un sistema iteratiu i jugar amb paràmetres i crear 300 logos diferents per a aquest client en una hora, fins que tots dos estiguin satisfets. I res obliga al fet que aquest logo es quedi fix. Li podem vendre directament un sistema generador de logos i que el logo es transformi cada setmana en un objecte gràfic diferent.

El disseny generatiu també pot ajudar-nos en moments de bloqueig creatiu. Quan cap idea ens funciona, en lloc de buscar directament la solució, podem desenvolupar un sistema generatiu –per exemple, un disseny evolutiu– i mirar diferents configuracions fins que una ens permeti desbloquejar-nos i tenir una proposta realment interessant. Les ments humanes es configuren des de preferències i disgustos, molts resultat de la nostra inclusió en una cultura concreta. En ser un mecanisme molt més bàsic de creació, un algoritme és paradoxalment lliure d'aquests condicionants culturals, i pot generar un disseny que mai se'ns hauria ocorregut directament.

Una altra raó central és com el disseny generatiu ens permet crear objectes gràfics que demanarien setmanes d'esforç si es fessin a mà o que directament serien impossibles. Si John Maeda triomfa com a dissenyador als noranta és en gran part perquè mostra com amb programació poden fer-se coses que resulten impossibles de fer a mà o amb el programari comercial d'edició gràfica.

Intentar replicar el cartell Absolut Maeda, on aquest dissenyador crea la silueta de la coneguda marca de vodka amb milers de línies corbes, és una empresa impossible sense programació.

Com hem vist en parlar del disseny evolutiu, un altre factor disruptiu del disseny generatiu és la capacitat que té de rebre *feedback* en temps real i modificar el disseny a partir de les decisions que un humà prengui en directe. Quan parlàvem de disseny evolutiu a l'apartat 2.5., parlàvem més del dissenyador que seleccionava la millor imatge i així l'algorisme creava noves seqüències. Però res no impedeix que el públic agafi la posició del dissenyador i cada persona individualment sigui la que esculli com ha de ser el seu logo, la seva targeta de visita o el cartell que anuncia la seva exposició. Alguns dels projectes de Karl Sims van en aquesta línia.

Però la raó més important per la qual el disseny generatiu importa és la qüestió del control. En una època en què les eines del professional del disseny les desenvolupen enginyers informàtics, és molt important que entenguem com funcionen aquestes eines, quins són les seves limitacions, i puguem crear les nostres pròpies eines per solucionar aquelles qüestions per les quals les eines de programari comercials són clarament insuficients. En una època en què un pot predir determinades tendències a partir dels nous filtres que apareixen a Photoshop, disposar d'una eina pròpia, adaptada perfectament als nostres interessos, gustos i possibilitats, és la millor manera de garantir la nostra originalitat i valor com a professionals del disseny.

4. Qui és l'autor del disseny generatiu?

Es podria replicar a l'apartat anterior que tot això és molt bonic, però que finalment en utilitzar aquest tipus d'eines matem l'autor: és el programa el que crea el grafisme, no la persona. La meva resposta davant d'aquesta objecció és que s'està partint d'una visió caduca d'autor, que no té cap sentit al món digital, i que en realitat el disseny generatiu revalora el concepte d'autor. Quan la força d'un cartell ve donada en bona part pels efectes i les transformacions que aconseguim crear en una fotografia a través d'uns filtres de programari, qui és més autor? Qui aplica els quatre filtres comercials que aquesta temporada estan de moda? O la persona que entén com funcionen els filtres i ha creat els seus de propis? És clar que és la segona persona, la persona que té el control del disseny generatiu.

> En realitat, qualsevol disseny digital és un disseny generatiu: apliquem algorismes per transformar els nostres objectes gràfics. L'única diferència és que el grafista que sap programar té el control de les seves pròpies eines.

I, d'altra banda, la nostra concepció d'autor està en transformació contínua. El *Design Thinking* i la cocreació són dos exemples de tendències molt fortes en el disseny actual en les quals el professional del disseny interactua contínuament amb el públic al qual va dirigit el seu projecte, amb la qual cosa la creació queda distribuïda entre el grafista i el públic participant en aquestes sessions de cocreació. Si originalment el disseny d'un llibre era tasca bàsicament d'una persona, ara tenim tot tipus de professionals que interactuen: dissenyador, maquetador, il·lustrador, impressor, etc., i no tenim cap problema a assignar a cada professional la seva parcel·la creativa i la seva influència en el disseny final. Quan escoltem Glenn Gould interpretant Bach, a ningú se li ocorre dir que tot el mèrit és de Gould i que Bach no hi pinta res. No tenim cap problema a reconèixer que hi ha dos processos creatius en paral·lel i que, finalment, les variacions Goldberg les devem a Johann Sebastian Bach i ell és el creador d'aquestes increïbles peces per a piano.

La nostra apreciació de l'ofici del disseny ha d'anar així, lliscant la idea de creador que genera un objecte gràfic amb les seves mans a un creador que idea un sistema que generarà objectes digitals únics, parametritzats i adaptats al públic que el rep i el context en el qual es troba. I en aquest nou context d'apreciació tindran cada vegada més pes aquelles persones que, en lloc d'utilitzar sistemes predissenyats, són capaces de desenvolupar les seves pròpies eines per així no posar traves a la seva originalitat creativa.

Bibliografia

General

Ceccato, C.; Hesselgren, L.; Pauly, M., Pottmann, H.; Wallner, J. (eds.). (2016). *Advances in Architectural Geometry 2010*. Birkhäuser.

Generative Design Blog <https://generativedesign.wordpress.com/>

Pérez, Eduardo (2018). «Diseño generativo: inteligencia artificial y diseño. *Revista Codi*. <https://revistacodigo.com/diseno/diseno-generativo/>

Roncoroni Osio, Umberto (2017). *Manual de diseño generativo*. Fondo Editorial Universidad de Lima.

Soddu, Celestino. *Generative Design Futuring Past* <http://www.generativeart.com/ga2015_web/futuringpast_soddu.pdf>

Disseny generatiu analògic

Munari, Bruno (2015). *Dibujar un árbol*. Bilbao: Anti.

Pacheco, Joshua. *Paths: An exploration of analogue algorithms*. <https://interface.fh-potsdam.de/gestalten-in-code/projects/analogue-paths/>

On Analogue Generative Design. <https://www.youtube.com/watch?v=xvjf4ufv8ng>

Yi Jing (2012). *El libro de los cambios*. Editorial Atalanta.

Diseño computacional

Flake, Gary William (1998). *The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation*. MIT Press.

Maeda, John (2000). *Maeda @ Media*. Thames & Hudson.

Maeda, John (2001). *Design by Numbers*. MIT Press.

Maeda, John (2010). *Las leyes de la simplicidad*. Gedisa.

Processing <www.processing.org>

Reves, Casey; Fry, Ben (2014). *Processing: A Programming Handbook for Visual Designers and Artists*. <https://processing.org/handbook/>

Website oficial de Design by Numbers <https://dbn.media.mit.edu/>

Disseny evolutiu

Banzhaf, Wolfgang; Nordin, Peter; Keller, Robert; Francone, Frank (1998). *Genetic Programming - An Introduction*. San Francisco, CA: Morgan Kaufmann.

Climent, María (2018). «El Divino Diseño generativo. La maquina piensa ya como la naturaleza». Innovadores. *El Mundo*. <https://www.elmundo.es/economia/2018/01/08/5a5340be268e3e8a648b4610.html>

Latham, William. Homepage <https://www.doc.gold.ac.uk/mes01whl/>

Martín, Santiago. *Diseño generativo y naturaleza*. Gijón: TEDX. <https://www.youtube.com/watch?v=c7uypqjbp7o>

Núñez, Laura (2019). «Qué son los algorismos genéticos?» *El País*. <https://elpais.com/elpais/2019/01/31/ciencia/1548933080_909466.html>

Sims, Karl. Homepage <https://www.karlsims.com/>

Disseny i intel·ligència artificial

Cohen Harold. Homepage <http://www.aaronshome.com/aaron/index.html>

Knight, Will (2018). «Las caras falsas imaginadas por una nueva IA son cada vez más reales». *MIT Technology Review*. <https://www.technologyreview.es/s/10818/las-caras-falsas-imaginadas-por-una-nueva-ia-son-cada-vez-mas-reales>

Elgammal, Ahmed *et al.* (2017). *CA: Creative Adversarial Networks Generating «Art» by Learning About Styles and Deviating from Style Norms*. <https://arxiv.org/pdf/1706.07068.pdf>

Col·lectiu Estampa. *El mal alumne*. <https://tallerestampa.com/estampa/el-mal-alumne/>

Greene, Tristan (2018). «Someone paid $432K for art generated by an open-source neural network». *The Next Web*. <http://thenextweb.com/artificial-intelligence/2018/10/25/someone-paid-432k-for-art-generated-by-an-open-source-neural-network/amp/>