BRICKS | TEMA

Arte e coding: quando l'algoritmo impara a sbagliare

a cura di:
Andrea Maccioni

Coding creativo, glitch art, H-STEAM, intelligenza artificiale, estetica digitale, didattica laboratoriale, pensiero computazionale, databending, interdisciplinarità

Introduzione: il paradosso dell'efficienza

La didattica digitale contemporanea ha prodotto un effetto paradossale: quanto più gli strumenti disponibili si sono fatti potenti, tanto più gli artefatti che generano si sono fatti simili. L'AI generativa ha accelerato e reso visibile questa convergenza, non perché i modelli siano privi di capacità espressiva, ma perché il modo in cui vengono integrati nei percorsi didattici tende sistematicamente ad eliminare il rischio. Si chiede, si ottiene, si consegna.

La sperimentazione descritta in questo articolo ha assunto come ipotesi di partenza che questa efficienza sia, didatticamente, il problema, e non la soluzione: mettere l'errore al centro, quindi, non come fallimento da correggere, ma come materia prima espressiva. Il suo oggetto è la *glitch art*, la pratica artistica che fa dell'imperfezione tecnica un codice espressivo, dell'errore un linguaggio, della corruzione un metodo. L'ipotesi centrale è che la comprensione profonda di un sistema digitale passi non attraverso il suo uso corretto, ma attraverso la sua rottura controllata. E che questa rottura, opportunamente situata in un contesto interdisciplinare, produca un tipo di apprendimento che né la didattica trasmissiva né l'uso strumentale degli strumenti digitali sono in grado di generare.

Perché il glitch: capire un sistema rompendolo

Il *glitch* non è semplicemente un errore visivo: è un evento che rivela. Quando un sistema digitale viene forzato oltre i propri limiti, la sua struttura interna, normalmente invisibile sotto l'interfaccia, diventa percepibile. Questa proprietà non è una metafora: è letteralmente ciò che accade quando, ad esempio, si apre un file JPEG con un editor esadecimale e se ne modificano i valori. Il risultato visivo è imprevedibile, ma non casuale, ed è proprio quella non-casualità a rendere il processo didatticamente potente.

A livello tecnico, quindi, consente di capire un sistema rompendolo. I *Vernacular of File Formats* di Rosa Menkman[1] sono, da questo punto di vista, un testo tanto tecnico quanto estetico: comprimendo la stessa fotografia con *codec* diversi, l'opera mostra che ogni formato possiede una "firma estetica" specifica, che è al tempo stesso un vincolo computazionale e una possibilità espressiva. È, implicitamente, una lezione sulla struttura binaria dei file che non necessita di una spiegazione frontale: basta guardare.

A livello critico, il *glitch* aiuta ad emanciparsi dall'output dell'AI. Uno dei rischi più concreti nell'uso didattico dell'AI generativa, infatti, è la passività. Introdurre un passaggio di destrutturazione deliberata – corrompere un output e reinserirlo nel modello come nuovo prompt, per esempio – trasforma il rapporto tra autore e strumento: il risultato smette di essere un prodotto finito da accettare e diventa materia grezza da valutare, selezionare, rielaborare. Questa operazione apre una domanda che il percorso ha

[1] Menkman, R. (2010), *A Vernacular of File Formats*. Opera visiva. Acquisita nel 2016 dallo Stedelijk Museum Amsterdam e dal MOTI, Breda. Si veda anche:
https://www.wausonline.com/files/glitch/Rosa%20Menkman%20-%20A%20Vernacular%20of%20File%20Formats.pdf

lasciato volutamente irrisolta: se un modello generativo produce un artefatto visivamente interessante in seguito ad un'allucinazione, in quale misura quell'artefatto è riconducibile ad un'intenzione? È arte? E chi ne è l'autore?[2]

A livello estetico, infine, è necessario precisare che il *glitch* non è certo una pratica nata con l'AI: radica, infatti, la produzione digitale nella storia dell'arte attraverso una genealogia precisa da ripercorrere a ritroso. Il punto di partenza è Nam June Paik e la sua *Magnet TV* (1965)[3]: un'installazione in cui un potente magnete industriale, posizionato sopra un televisore in bianco e nero, interferisce con i circuiti interni e trasforma il segnale trasmesso in una forma astratta e continuamente mutevole, manipolabile spostando il magnete. È il primo atto consapevole di un'estetica dell'errore nel segnale elettronico: l'immagine non viene più ricevuta, ma deformata, usata come materia.

Nel 1982, *Koyaanisqatsi* di Godfrey Reggio[4] – un poema visivo senza dialogo né narrazione, composto interamente da immagini in *time-lapse* e *slow motion*, che esplora il rapporto tra natura, uomo e tecnologia – costruisce qualcosa di diverso ma complementare: il sublime attraverso il flusso continuo delle immagini analogiche. Denso, ininterrotto, senza parole. Il titolo – una parola della lingua hopi[5] che significa "vita in disintegrazione", "condizione che richiede un altro stile di vita" – è già una dichiarazione sul sistema in crisi. E la sequenza finale, in cui la cinepresa insegue i detriti di un razzo esploso in caduta verso terra, è il momento in cui il film smette di osservare la tecnologia e la mostra che si rompe: non *glitch* in senso tecnico, ma la stessa logica, con la struttura che cede che rivela qualcosa che il funzionamento normale occulta.

Con l'avvento del digitale, l'estetica dell'errore si trasferisce sul codice: Aphex Twin e il regista Chris Cunningham, con il video di *Windowlicker* (1999)[6], producono una delle prime opere di cultura popolare in cui la distorsione digitale del volto umano cessa di essere effetto speciale e diventa linguaggio. Menkman, nei primi anni 2000, ne ha fatto teoria: nel *Glitch Moment(um)* (2011)[7] argomenta che il *glitch* non è né puramente un artefatto né puramente un processo, ma oscilla tra le due condizioni, ed è precisamente questa oscillazione a renderlo produttivo esteticamente. Nel *Vernacular of File Formats*, opera visiva dell'anno precedente, aveva già dimostrato la tesi concretamente: comprimendo la stessa fotografia con *codec* diversi, infatti, ogni formato produceva artefatti visivi derivati dalla propria struttura matematica

[2] Sul rapporto tra AI generativa e autorialità in ambito artistico si veda anche: Hertzmann, A. (2018), Can Computers Create Art?, *Arts*, 7(2), 18. MDPI. https://doi.org/10.3390/arts7020018

[3] Paik, N.J. (1965), *Magnet TV*. Installazione. Riferimento museale: Whitney Museum of American Art, New York; Smithsonian American Art Museum.

[4] Reggio, G. (regia), Glass, P. (musica) (1982), *Koyaanisqatsi*. Film sperimentale. Conservato nel National Film Registry della Library of Congress (2000).

[5] Britannica Editors (2024, September 5). Hopi language. Encyclopedia Britannica. https://www.britannica.com/topic/Hopi-language

[6] Cunningham, C. (regia), Aphex Twin (musica) (1999), *Windowlicker*. Videoclip. Warp Records.

[7] Menkman, R. (2011), *The Glitch Moment(um)*, Network Notebook #04, Institute of Network Cultures, Amsterdam. Si veda anche: https://networkcultures.org/_uploads/NN%234_RosaMenkman.pdf

interna. Non sono difetti, quindi, ma schemi. L'opera precede la teoria e la dimostra: un'inversione che è essa stessa un gesto critico.

Il percorso da Paik a Menkman, passando per l'analogico di *Koyaanisqatsi* e il digitale di *Windowlicker*, è anche la storia della materialità dell'immagine: un filo che appartiene a pieno titolo alla storia dell'arte e che questo percorso ha usato come struttura portante.

La convergenza disciplinare come metodo

La *glitch art* non appartiene naturalmente a nessuna disciplina scolastica e questa è precisamente la sua utilità didattica. Si colloca nell'intersezione tra materie che raramente si incontrano: la storia dell'arte, l'informatica, la produzione grafica. Ciascuna di queste discipline, presa singolarmente, coglie solo una dimensione del fenomeno, ma è la loro convergenza che ne restituisce la complessità.

La storia dell'arte fornisce la genealogia e il quadro interpretativo: senza di essa, il *glitch* rimane un effetto tecnico privo di storia e di senso critico. L'informatica fornisce gli strumenti analitici e operativi: senza di essa, la produzione rimane gesto estetico non fondato sulla comprensione dei sistemi che lo rendono possibile. La grafica fornisce il terreno progettuale: senza di essa, l'esperienza tecnica non si traduce in linguaggio visivo consapevole. Questo non è un rapporto di semplice complementarietà, ma di reciprocità produttiva, in cui ciascuna disciplina modifica il modo in cui le altre vengono praticate.

Vale la pena notare che questo tipo di convergenza non è replicabile attraverso la sola didattica trasmissiva. Richiede un formato laboratoriale in cui il sapere si costruisce per manipolazione diretta, per errore e correzione, per confronto tra prodotti e intenzioni: quello che Resnick e Silverman hanno descritto come *tinkering*[8]. La *glitch art* è, in questo senso, un contenuto ideale per un approccio H-STEAM: non perché coinvolga più discipline, ma perché le costringe a lavorare davvero insieme.

Architettura della sperimentazione

Il percorso si è articolato intorno alla collaborazione tra diversi ambiti disciplinari: Discipline Grafiche, Informatica e Storia dell'Arte, operanti su piani distinti ma interconnessi. In Discipline Grafiche, l'analisi dei *glitch* di Menkman e del videoclip di Aphex Twin ha offerto un repertorio immediato di *texture*, composizioni stratificate e distorsioni cromatiche, tradotto in esercizi progettuali: riprodurre manualmente un'estetica del *glitch*, capire come la "firma" di un formato diventi elemento visivo consapevole. In Informatica, il lavoro ha riguardato la struttura binaria dei file, il *databending*[9], la compressione, trattando Menkman come testo

[8] Questo approccio di apprendimento per manipolazione diretta è affine al concetto di *tinkering* sviluppato da Mitchel Resnick e Brian Silverman nel contesto dei lavori del MIT Media Lab. Si veda: Resnick, M. & Silverman, B. (2005), Some reflections on designing construction kits for kids, *Proceedings of the 2005 conference on Interaction Design and Children* (IDC '05). ACM Press.

[9] Per un'introduzione pratica al *databending* con editor esadecimale, la comunità *Glitch Artists Collective* raccoglie tutorial, esempi e risorse aggiornate.

tecnico, oltre che artistico. In Storia dell'Arte, tutto è stato collocato su una linea temporale: dal sublime analogico di *Koyaanisqatsi* al *glitch* come genere critico, con una riflessione sul sublime tecnologico che va dal Romanticismo all'era della post-produzione. Questa struttura non è stata semplicemente parallela, ma ricorsiva: i materiali emersi in ambito informatico sono stati riportati nel contesto della grafica e della storia dell'arte, e viceversa. Il dialogo tra i tre piani si è dimostrato operativo, non nominale.

Dalla teoria alla pratica: le tre fasi del percorso

Fase 1: Il glitch come problema storico e percettivo (Storia dell'Arte e Discipline Grafiche)

La prima fase ha posto una domanda preliminare: esiste una storia dell'imperfezione tecnica come scelta estetica consapevole, distinta dall'errore involontario? E come si riconosce la differenza?

Per rispondere, sequenze da *Koyaanisqatsi* sono state analizzate nei loro elementi costitutivi – tempo, paesaggio, accelerazione, assenza di narrazione verbale – e messe a confronto con artefatti di *rendering* contemporanei e con opere di Menkman. Il percorso visivo rendeva tangibile il passaggio dal segnale continuo al dato discreto, introducendo il concetto di *codec* non come strumento neutro ma come scelta estetica, con una propria logica interna di perdita e trasformazione dell'informazione: ogni formato di compressione produce infatti una firma visiva specifica, riconoscibile e potenzialmente progettuale. La domanda guida ("Questo errore è intenzionale? Come fai a dirlo? Cambia qualcosa se lo è?") ha attraversato tutta la fase, aprendo una riflessione sull'intenzione artistica, sull'autorialità e, inevitabilmente, sull'AI[10].

Questa fase non ha prodotto una risposta, ma una disposizione analitica: la capacità di guardare un'immagine digitale chiedendosi cosa ci sia dentro, prima ancora di chiedersi cosa rappresenti.

Fase 2: La struttura interna dell'immagine come materiale (Informatica e Discipline Grafiche)

La seconda fase ha spostato il lavoro dal piano interpretativo a quello operativo, introducendo il coding non come scrittura di istruzioni ma come manipolazione diretta di strutture dati. La domanda sottostante era: cosa rivela un sistema digitale quando viene forzato oltre i propri limiti?

Lo strumento principale è stato il *databending*: file immagine (JPEG, GIF, RAW) aperti con un editor esadecimale (HxD[11] o Hex Fiend[12]) e modificati manualmente, direttamente nei dati dell'immagine, intervenendo quindi sui valori binari che li compongono[13]. Il risultato sono immagini alterate in modi

[10] Hertzmann, A., *op. cit.*

[11] Su Windows: https://mh-nexus.de/en/hxd/

[12] Su MacOs: https://hexfiend.com/

[13] Evitando l'*header*, che altrimenti avrebbe causato un errore irrecuperabile. L'*header*, infatti, è la porzione iniziale del file che contiene le informazioni fondamentali per la sua interpretazione: il formato, le dimensioni dell'immagine, la profondità di colore e altri metadati strutturali. È, in sostanza, ciò che permette al sistema operativo e alle applicazioni di riconoscere il file come immagine e di aprirlo correttamente. Modificarlo compromette questa informazione di base, rendendo il file illeggibile per qualsiasi software. I dati visivi veri e propri, quelli che codificano i pixel e che il *databending* vuole alterare, si trovano invece nella parte successiva del file: è lì che l'intervento manuale produce effetti visivi senza distruggere la struttura portante.

imprevedibili ma non casuali, in cui la fotografia originale è stata filtrata attraverso *pattern* di corruzione specifici del formato (Figura 1). Un esempio emblematico di quanto emerso: la comparsa di striature orizzontali anziché verticali in un'immagine corrotta ha reso immediatamente evidente – senza alcuna spiegazione teorica – che il file codificava l'informazione per righe. Quella scoperta, nata da un errore nell'errore, ha condensato in un'osservazione diretta un concetto che una lezione frontale avrebbe richiesto ben più tempo a trasmettere.

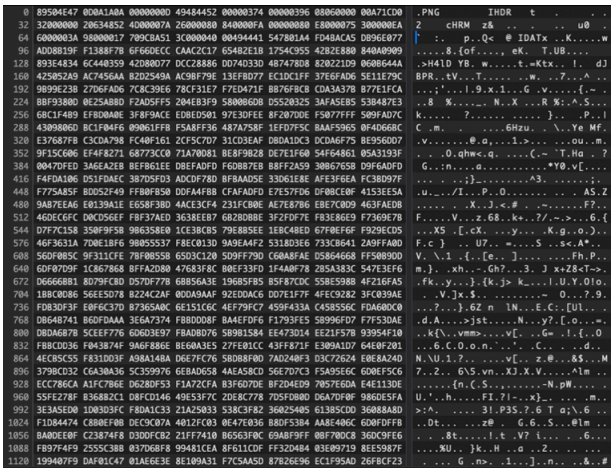


Fig. 1 – *Visualizzazione esadecimale di un file PNG aperto con Hex Fiend: struttura dati, header e chunk del formato immagine*

A quel punto, è stato possibile sviluppare una tavola progettuale basata sull'utilizzo consapevole e voluto di errori e deformazioni generate dai formati digitali, trasformandoli in *texture*, *pattern* o elementi compositivi di un unico progetto grafico (Figura 2). Questo passaggio ha trasformato l'esplorazione tecnica in decisione progettuale, spostando il *databending* dall'ambito dell'esperimento a quello del linguaggio dell'errore.

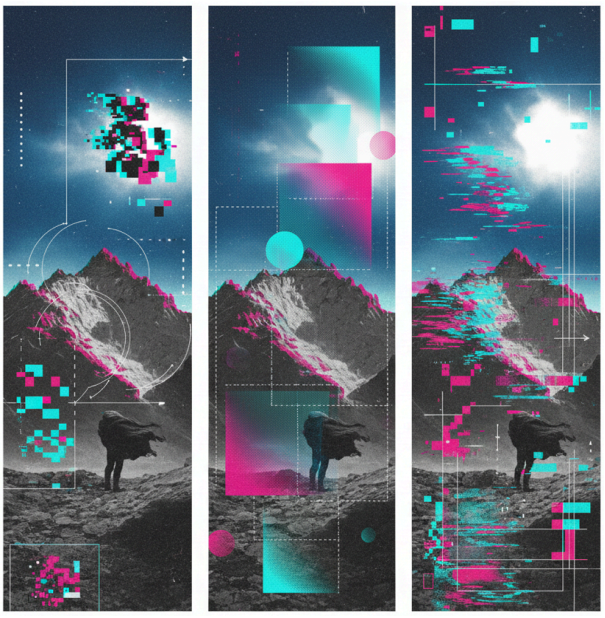


Fig. 2 – *Tavole progettuali: glitch come elemento compositivo (JPEG, GIF, RAW a confronto). Immagini originali generate con Nano Banana in tre diversi formati, successivamente elaborate con HxD*

Fase 3: Il glitch come sistema espressivo (Discipline Grafiche e Informatica)

La terza fase ha introdotto *Scratch*[14] come ambiente di coding visuale, con una specifica ipotesi metodologica: che l'imposizione di un vincolo preciso (*l'errore deve essere voluto, ripetibile e comunicare qualcosa*) costringa ad uscire dalla casualità pura e a progettare sistemi. La differenza, concettualmente, è quella tra rumore e linguaggio.

Questa ipotesi si è dimostrata produttiva. Il vincolo ha generato un passaggio osservabile nella logica progettuale: dall'esplorazione non orientata – in cui qualsiasi corruzione era accettabile – ad una progettazione in cui ogni scelta tecnica doveva essere difendibile in termini di senso. In tutti i casi, il comportamento del sistema è stato costruito, non trovato.

In questa fase è stato anche introdotto il ciclo dell'AI: un'immagine generata con strumenti generativi (DALL·E, Adobe Firefly) è stata poi elaborata con *databending* e in alcuni casi reinserita come nuovo prompt visivo (*image-to-image*). Ripetuto per tre o quattro volte, il processo produce una serie che documenta la progressiva "erosione" dell'immagine originale, rendendo visibile (Figura 3), in modo diretto e fisico, la logica interna di un modello generativo, invece di accettarlo semplicemente come scatola nera[15].



Fig. 3 – *Serie di quattro iterazioni: da output AI originale ad immagine corrotta e reimmessa nel modello. Immagine originale generata con DALL·E, elaborata con Hex Fiend e reimmessa come prompt per Adobe Firefly*

[14] *Scratch* è un ambiente di programmazione visuale sviluppato dal MIT Media Lab, disponibile gratuitamente su https://scratch.mit.edu/

[15] Pasquale, F. (2015), *The Black Box Society: The Secret Algorithms That Control Money and Information*. Harvard University Press.

I progetti realizzati: una grammatica della corruzione

Nel corso del percorso sono emersi tre artefatti che si prestano ad una lettura comparata. Condividono la superficie – *glitch*, degradazione progressiva, residuo figurativo – ma divergono profondamente nella logica interna, nel tipo di sistema che incarnano e nel rapporto che stabiliscono tra l'azione del soggetto e la perdita dell'immagine. Vale la pena descriverli in sequenza, perché le differenze strutturali tra loro sono, in sé, un risultato della sperimentazione.

Memoria ROM: la perdita come interazione

Memoria ROM è un'animazione *Scratch* il cui *sprite* iniziale è il ritratto fotografico di un soggetto anziano (Figura 4). Dal punto di vista implementativo, lo *sprite* è stato strutturato attraverso una sequenza di costumi progressivamente degradati, ottenuti mediante manipolazioni successive dell'immagine – compressione, riduzione della definizione, introduzione di artefatti visivi e alterazioni cromatiche – oppure tramite l'applicazione iterativa di effetti grafici nativi del motore. L'interazione è modellata secondo un paradigma *event-driven*: ogni click incrementa una variabile di stato che controlla la transizione tra livelli successivi di corruzione, conducendo iterativamente alla dissoluzione completa dell'immagine. La fase finale produce uno stato visivo residuale privo di informazione riconoscibile: non più un soggetto, ma un residuo informazionale indistinto.

Dal punto di vista strutturale, *Memoria ROM* è un sistema dinamico a stati finiti, deterministico e chiuso: tutto avviene all'interno di un unico ambiente, il deterioramento è continuo, tecnicamente irreversibile, causato direttamente dall'uso. La memoria dell'immagine non è conservata ma progressivamente erosa attraverso trasformazioni irreversibili: il codice non agisce come strumento di conservazione ma come meccanismo entropico di trasformazione. Il punto non è l'immagine in sé, ma il comportamento del sistema: un dispositivo che dimentica mentre viene interrogato.

In questa prospettiva, l'artefatto dialoga con la riflessione di Bernard Stiegler sulla memoria tecnica come forma di "ritenzione terziaria"[16]: una esternalizzazione che al contempo conserva e altera, rendendo la memoria dipendente dai supporti materiali e algoritmici. L'interazione dell'utente diventa parte integrante del processo di oblio – l'atto del ricordare si traduce in un'operazione progressiva di cancellazione – in un'eco della concezione borgesiana del linguaggio come dispositivo ambiguo di conservazione e perdita. Parallelamente, Vilém Flusser offre un ulteriore quadro[17]: le immagini tecniche sono superfici programmate in cui il gesto umano non produce direttamente il reale, ma interagisce con apparati di codifica già determinati. Il click non crea: accelera una dissoluzione già scritta nel sistema.

[16] Stiegler, B. (1994), *La Technique et le Temps, 1: La faute d'Épiméthée*. Galilée, Parigi. Trad. it.: *La tecnica e il tempo*. Melangolo, Genova.
[17] Flusser, V. (1983), *Für eine Philosophie der Fotografie*. European Photography, Berlino. Trad. it.: *Per una filosofia della fotografia*. Bruno Mondadori, Milano.



Fig. 4 – *Memoria ROM: tre stati dell'animazione (iniziale, intermedio, finale). Immagine originale generata con Nano Banana e successivamente elaborata con Scratch*

Self-portrait (damaged): la perdita come traduzione

Self-portrait (damaged) è una serie di cinque immagini in cui un autoritratto fotografico viene progressivamente sottoposto a processi di corruzione digitale attraverso tecniche di *databending* e successiva reimmissione del materiale visivo in DALL·E come prompt iterativo (Figura 5). In questo processo il file fotografico non è trattato come unità stabile, ma come struttura informazionale aperta, soggetta ad interferenze, alterazioni e reinterpretazioni algoritmiche: ogni passaggio introduce un ulteriore livello di instabilità semantica e perdita di controllo rappresentazionale. Il risultato finale non è più leggibile come ritratto, ma porta ancora tracce della forma iniziale: una tensione tra riconoscibilità e dissoluzione che produce una carica emotiva non pianificata, in cui la persistenza parziale del volto genera un attrito tra memoria visiva e cancellazione algoritmica.

Dal punto di vista strutturale, questo artefatto è l'opposto di *Memoria ROM*: un sistema aperto, iterativo, a feedback umano e algoritmico. La trasformazione avviene per salti discreti: ogni passaggio esce dallo strumento e rientra in un altro, dal file fotografico all'editor esadecimale, dall'editor al modello generativo, dal modello ad un nuovo file da ricorrompere. La reiterata reimmissione dell'immagine nel sistema generativo introduce una dinamica di retroazione algoritmica in cui il risultato diventa progressivamente meno derivato dall'originale e sempre più dal sistema stesso: il modello, ad ogni iterazione, risponde non alla fotografia ma alla sua versione già corrotta, amplificando le derive invece di correggerle. Il soggetto non viene semplicemente rappresentato o distorto, ma progressivamente redistribuito tra sistemi di corruzione dati e generazione sintetica, fino a diventare una traccia instabile della propria origine. Il deterioramento non è causato dall'uso ma dal passaggio tra sistemi incompatibili: la perdita è traduttiva, non performativa.

Lev Manovich descrive questa condizione come caratteristica della logica computazionale contemporanea, in cui le immagini non sono più entità statiche ma variabili di un processo, continuamente rielaborate da sistemi di automazione estetica[18]. La progressiva perdita di riconoscibilità non rappresenta una degradazione, ma l'emergere di un'autonomia del processo rispetto alla sua origine. Corrisponde a ciò che Menkman ha teorizzato come condizione produttiva del *glitch*: l'errore non come malfunzionamento, ma come generatore di nuove forme di significazione.



Fig. 5 – *Self-portrait (damaged): la serie completa dei cinque stati, dal file originale all'immagine residuale. Immagine originale generata con DALL·E, elaborata con HxD e reimmessa come prompt per DALL·E*

Memoria ROM e *Self-portrait (damaged)*, pur condividendo la superficie del *glitch*, divergono su un punto strutturalmente decisivo: nel primo, la memoria si dissolve mentre la interroghi; nel secondo, si dissolve mentre la traduci. Entrambi i lavori possono essere letti come due varianti dello stesso dispositivo operativo e concettuale: una grammatica della corruzione dell'immagine in cui la memoria visiva non viene semplicemente rappresentata, ma sottoposta a procedure di degradazione controllata. Nel primo caso, la progressiva dissoluzione è guidata dall'interazione dell'utente all'interno di un sistema a stati discreti, in cui il click funziona come agente di oblio programmato; nel secondo, la stessa logica viene traslata su un piano più instabile e retroattivo, dove la trasformazione emerge dal ciclo iterativo tra *databending* e reimmissione algoritmica. In entrambi i casi, ciò che si osserva non è la perdita accidentale dell'immagine, ma l'emergere di una struttura processuale in cui il ricordo viene continuamente riscritto, fino a rendere indistinguibile il confine tra archivio, errore e generazione.

Il suono che si rompe: la perdita come risposta

Il terzo artefatto introduce una variabile assente nei precedenti: il tempo reale e l'input sensoriale esterno. *Il suono che si rompe* è un'animazione *Scratch* in cui la deformazione visiva degli *sprite* è controllata dal livello sonoro catturato dal microfono: al superamento di una soglia, le figure si corrompono; al calare della voce, tendono a ricomporsi. Il sistema non è deterministico né iterativo: risponde in modo continuo ad un flusso di dati ambientali, producendo una corruzione reattiva.

Se *Memoria ROM* è un sistema chiuso che dimentica attraverso l'interazione, e *Self-portrait* è un sistema aperto che si perde attraverso la traduzione, *Il suono che si rompe* è un sistema reattivo che si degrada in

[18] Manovich, L. (2001), *The Language of New Media*. MIT Press, Cambridge. Trad. it.: *Il linguaggio dei nuovi media*. Olivares, Milano.

risposta all'ambiente. La corruzione non è causata dall'utente né dall'attraversamento di macchine diverse: è indotta da una pressione esterna che il sistema traduce in instabilità visiva. La descrizione che ha accompagnato l'artefatto – "una metafora dell'ansia" – coglie qualcosa di preciso nella struttura stessa del codice: un sistema che perde coerenza sotto pressione e tende a riorganizzarsi quando la pressione cessa, secondo una logica che assomiglia meno ad un archivio che ad un organismo.

Competenze sviluppate

La comprensione della struttura dei file digitali non è arrivata da spiegazioni frontali, ma dal momento in cui è diventato evidente – attraverso un errore nell'errore – perché un'immagine mostrasse, ad esempio, striature orizzontali e non verticali: la sequenza di *byte* modificata codificava le righe del file, non le colonne. Quella scoperta ha spiegato in pochi secondi quello che avrebbe richiesto un'intera lezione teorica. Parallelamente, il vincolo di rendere il *glitch* intenzionale e comunicativo ha prodotto un passaggio visibile nei progetti finali: dalla logica di produzione ("fare qualcosa di bello") ad una logica di autorialità ("costruire un sistema che dice qualcosa"). Questa distinzione – tra fare e progettare – è tra le più difficili da insegnare in un contesto di grafica digitale, e qui è emersa dalla pratica, non da una lezione.
Sul versante critico, anche il confronto con i limiti dei modelli generativi è avvenuto attraverso l'esperienza diretta: usare l'AI come materiale da corrompere mette in evidenza cosa c'è dentro l'*output*, che altrimenti si accetterebbe come dato. *Pattern* stereotipati, ripetizioni, allucinazioni diventano riconoscibili, non perché qualcuno ne abbia parlato, ma perché il prodotto finito è stato smontato. Infine, la connessione tra le tre discipline si è dimostrata operativa: analizzare Menkman ha cambiato il modo di guardare un file e capire un file ha cambiato il modo di guardare Menkman. La circolarità era reale.

Competenza sviluppata	Strumento / attività	Com'è stata osservata
Comprensione della struttura interna dei file digitali	*Databending* con editor esadecimale (HxD, HexFiend)	Non attraverso spiegazione frontale, ma nel momento in cui un effetto inatteso – striature orizzontali invece di verticali – ha reso visibile la logica di codifica del file. La comprensione è emersa dall'errore, non preceduta da esso.
Coding come progettazione di sistemi espressivi	Scratch con trigger sonori e variabili di stato	Osservabile nel passaggio dalla corruzione casuale alla corruzione intenzionale: il vincolo – l'errore deve comunicare qualcosa – ha prodotto una distinzione tra rumore e linguaggio verificabile nel comportamento del programma rispetto all'intenzione dichiarata.
Lettura critica degli output dell'AI generativa	Ciclo corruzione → *databending* → reimmissione come prompt	Emergente dalla pratica diretta: pattern stereotipati, derive algoritmiche e allucinazioni sono diventati riconoscibili nel momento in cui l'output è stato smontato e reimmesso nel sistema, non attraverso analisi teorica preliminare.

Integrazione storico-artistica e genealogia del *glitch*	Analisi comparata di Menkman, Paik (*Magnet TV*), *Koyaanisqatsi*, *Windowlicker*	Verificabile nella fase di restituzione finale: capacità di collocare le scelte tecniche all'interno di una linea storica e di argomentare la relazione tra errore scelto e tradizione critica di riferimento.
Autorialità progettuale: dal fare al progettare	Tavola progettuale (Fase 2), artefatti finali (Fase 3)	Osservabile nella coerenza tra dichiarazione d'intento e risultato: la relazione tecnico-poetica ha reso verificabile se una scelta fosse difendibile in termini di senso, non solo eseguita correttamente.
Lettura teorica dei media digitali	Analisi degli artefatti prodotti attraverso Stiegler, Flusser, Manovich	Emergente nella contestualizzazione degli artefatti: capacità di leggere un sistema computazionale non solo come strumento ma come dispositivo culturale, con una propria logica di conservazione, perdita e trasformazione dell'informazione.

Competenze sviluppate, strumenti e modalità di osservazione

Replicabilità e criticità

Perché un percorso di questo tipo funzioni davvero, è importante che la componente tecnica non venga separata dalla dimensione critica e progettuale. Non è una questione di metodo didattico in senso stretto, ma una condizione strutturale: il *glitch* acquista senso solo quando chi lo produce è in grado di situarlo: storicamente, esteticamente, tecnicamente. Quando uno di questi piani manca, il risultato è o un esercizio tecnico privo di linguaggio o un gesto estetico privo di fondamento. La collaborazione reale tra discipline diverse non è quindi un arricchimento opzionale del percorso: ne è una condizione di fattibilità.

Sul piano delle condizioni favorevoli, il formato laboratoriale con gruppi ridotti e tempi elastici si è rivelato determinante. La possibilità di sperimentare senza la pressione della prestazione immediata – di "sporcarsi le mani", fare errori, tornare indietro – è difficilmente ottenibile in lezioni esclusivamente frontali o in orari curricolari compressi. Allo stesso modo, il dialogo tra un docente di area grafica e uno di area storico-artistica ha permesso un movimento ricorsivo tra i piani disciplinari che un singolo docente avrebbe faticato a sostenere: gli stimoli visivi emersi nel lavoro tecnico sono stati rilanciati nel contesto storico-artistico, e viceversa.

Il percorso trova inoltre terreno particolarmente fertile in contesti in cui i destinatari siano già immersi in un ecosistema visivo fatto di immagini degradate, compressioni aggressive, filtri che simulano la bassa risoluzione delle VHS, meme a bassa fedeltà, *glitch* estetizzati. Tra i 17 e i 18 anni questo immaginario è spesso già presente, seppur in modo passivo. Partire da quella familiarità consente di trasformarla in consapevolezza critica: mostrare che la bassa fedeltà può essere una scelta espressiva e non soltanto un difetto tecnico, produce un effetto di riconoscimento che predispone all'esplorazione in modo più immediato rispetto a qualsiasi introduzione teorica. In questo senso il percorso non introduce un

immaginario estraneo, ma nomina e problematizza qualcosa che è già lì, contribuendo anche a ridimensionare l'ansia della perfezione digitale e a restituire ai partecipanti un ruolo critico e autoriale rispetto agli strumenti che usano ogni giorno – editor grafici, piattaforme social, modelli generativi – smontandone l'illusione di trasparenza.

Allo stesso tempo, si evidenziano alcune criticità che richiedono mediazione didattica. Il *databending* presuppone una minima alfabetizzazione tecnica: aprire un file con un editor esadecimale può risultare disorientante senza un'introduzione preliminare di concetti come *header*, struttura del file e compressione, e senza aver salvato una copia di backup del file originale. In assenza di questa preparazione si rischia di corrompere il file in modo irrecuperabile, oppure, più semplicemente, di non comprendere ciò che si sta osservando. In quel caso il disorientamento non produce apprendimento ma frustrazione. È necessario prevedere uno spazio di orientamento prima di iniziare a sperimentare, con esempi visivi che mostrino concretamente la relazione tra struttura del file e risultato visivo della corruzione.

La seconda criticità riguarda la gestione dell'errore stesso. Senza vincoli progettuali espliciti, la tendenza è quella di spingere la corruzione fino alla saturazione totale dell'immagine, perdendo ogni traccia dell'immagine originale. È emersa quindi la necessità di chiedere sempre una motivazione esplicita delle scelte tecniche: quale formato è stato alterato, perché, quale atmosfera si voleva ottenere, quale relazione esiste tra l'errore scelto e il risultato visivo finale, nell'ottica di un'auto-verifica critica nel distinguere un errore significativo da un semplice rumore visivo.

Anche l'utilizzo dell'AI generativa come materiale di partenza ha evidenziato una disparità iniziale di competenze: chi era già abituato a lavorare con modelli generativi tendeva ad avere maggiore controllo sul processo, fin dalle prime fasi, specialmente nel ciclo corruzione–reinserimento dell'output. La formazione di gruppi di lavoro misti ha permesso di redistribuire le competenze, evitando che alcuni partecipanti rimanessero semplici spettatori.

Dal punto di vista operativo, il laboratorio è facilmente replicabile anche senza infrastrutture complesse o hardware specializzato: sono sufficienti software gratuiti per l'editing esadecimale, strumenti base di coding visuale e qualsiasi piattaforma di AI generativa che consenta l'esportazione di immagini. Più importante dell'attrezzatura è invece la costruzione del contesto culturale: presentare almeno un'opera significativa della *glitch art* prima di qualsiasi attività tecnica cambia radicalmente la disposizione verso l'errore. Comprendere che il *glitch* possiede una genealogia estetica e teorica, e non è semplicemente un file "corrotto", rende più chiaro il senso dell'intero percorso. È una premessa che vale indipendentemente dal contesto e dall'età dei destinatari, e che nessuna spiegazione tecnica, per quanto accurata, può sostituire.

Conclusione: rompere per capire

Comprendere un sistema digitale – dalla struttura dei file al comportamento dei modelli generativi – richiede, in fondo, di attraversarlo: aprirlo, forzarlo, osservare cosa cede e cosa resiste. Un file immagine corrotto racconta la struttura dell'immagine. Un'animazione che si inceppa rivela la logica del programma.

Un output di AI reimmesso nel modello dopo la corruzione mostra qualcosa di reale su come il sistema interpreta il mondo visivo. Il *glitch* non è l'opposto dell'estetica digitale contemporanea, ma una sua forma di smascheramento. In un'epoca in cui la produzione digitale tende sistematicamente ad eliminare attrito, errore e imperfezione, imparare a vedere e costruire il difetto come scelta diventa una competenza critica, non soltanto estetica. Chi impara a rompere con metodo impara, in ultima istanza, a non accettare passivamente i sistemi che abita.

Bibliografia

Cunningham, C. (1999), *Windowlicker*. Warp Records.

Flusser, V. (1983), *Für eine Philosophie der Fotografie*. European Photography, Berlino. Trad. it.: Per una filosofia della fotografia. Bruno Mondadori, Milano.

Hertzmann, A. (2018), Can Computers Create Art?, *Arts*, 7(2), 18. MDPI.

Manovich, L. (2001), *The Language of New Media*. MIT Press, Cambridge. Trad. it.: Il linguaggio dei nuovi media. Olivares, Milano.

Menkman, R. (2010), *A Vernacular of File Formats*. Stedelijk Museum, Amsterdam.

Menkman, R. (2011), *The Glitch Moment(um)*, Network Notebook #04, Institute of Network Cultures, Amsterdam.

Paik, N.J. (1965), *Magnet TV*. Whitney Museum of American Art, New York.

Pasquale, F. (2015), *The Black Box Society: The Secret Algorithms That Control Money and Information*. Harvard University Press.

Reggio, G. (1982), *Koyaanisqatsi*. National Film Registry, Library of Congress, Washington.

Resnick, M. & Silverman, B. (2005), Some reflections on designing construction kits for kids, *Proceedings of the 2005 conference on Interaction Design and Children* (IDC '05). ACM Press.

Stiegler, B. (1994), *La Technique et le Temps, 1: La faute d'Épiméthée*. Galilée, Parigi. Trad. it.: La tecnica e il tempo. Melangolo, Genova.



Andrea Maccioni

andrea.maccioni@sacrocuoretdm.it

Informatico, analista e docente. Lavora come Responsabile Informatico presso l'Istituto Sacro Cuore Trinità dei Monti di Roma. Membro dell'AIP-ITCS (Associazione Informatici Professionisti), dal 2022 è Formatore per l'Associazione Italiana per l'Informatica e il Calcolo Automatico (AICA) e Supervisore dal 2024. Accanto all'attività sistemistica e gestionale, svolge attività di analisi, progettazione software/web e sviluppo applicativo, oltre ad un ruolo consulenziale nell'ambito dell'innovazione tecnologica. Da anni approfondisce il rapporto tra tecnologia e discipline umanistiche, con un interesse specifico per l'Intelligenza Artificiale, l'analisi dei dati e l'e-learning applicati ai contesti educativi, con particolare attenzione alle pratiche creative che integrano coding e produzione digitale.