UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA TRIENNALE IN INGEGNERIA INFORMATICA

PROGETTAZIONE E SVILUPPO SOFTWARE PER LA RIATTIVAZIONE E L'ACCESSO DELL'OPERA ARTISTICA INTERATTIVA MULTIMEDIALE "IL TEMPO CONSUMA"

Relatore: Prof. Sergio Canazza Targon

Correlatore: Dott. Alessandro Fiordelmondo

Laureando: Samuele Disarò

ANNO ACCADEMICO 2021-2022
Data di Laurea 16/11/2022

Sommario

Nel 1978 l'artista padovano Michele Sambin creò *Il tempo consuma*, un'opera ispirata da un desiderio dell'artista: essere in grado di interagire in tempo reale con un immagine di sé stesso registrata poch secondi prima. Per fare ciò inventò il *video-loop*, tecnica che consente di riprodurre di flussi audio e video in tempo reale contemporaneamente a quelli registrati precedentemente.

Con l'intento di rendere l'opera nuovamente fruibile al pubblico, nel 2021 il Centro di Sonologia Computazionale si occupò del suo restauro e riattivazione, tramite l'utilizzo di apparecchiatura digitale anzichè analogica, cosa che portò a non pochi ostacoli di natura progettuale.

Il presente lavoro si propone di rendere più accessibile il programma di gestione dell'opera, modificando l'interfaccia utente (GUI) e i moduli al suo interno. Successivamente si è deciso di aggiungere nuove funzionalità o modificarne alcune già esistenti, così da offrire più libertà di scelta e sperimentazione all'artista.

Viene trattata la storia dell'artista e dell'opera, insieme alla creazione della tecnica del *videoloop*, per poi analizzare il primo lavoro di riattivazione compiuto. Si passa poi ad esporre i problemi relativi al software di gestione del sistema, le modalità con cui sono risolti e i risultati ottenuti da queste modifiche.

Ringraziamenti

Vorrei ringraziare innanzitutto il Prof. Canazza e il Dott. Fiordelmondo del Centro di Sonologia Computazionale (CSC), grazie ai quali sono stato in grado di produrre una tesi nel breve tempo a mia disposizione.

Ringrazio i miei compagni di corso con i quali ho condiviso questi tre anni, ovvero Alex, Enrico, Marco, Daniele e Manuel; sebbene la pandemia ci abbia diviso dopo pochi mesi dalla nostra conoscenza non ci ha fermato dal continuare a conoscerci, e di questo vi sono grato.

Desidero ringraziare i miei amici, i quali mi hanno supportato dall'inizio alla fine di questo percorso; la vostra presenza è stata molto importante per me, soprattutto quella di Nadir Dalla Pozza, (finalmente) collega ingegnere il quale mi ha aiutato molto in questi tre anni.

Infine voglio ringraziare tutti i miei parenti, ma in particolare i miei genitori, poiché se sono qui in questo momento lo devo principalmente a loro. Grazie per avermi dato la possibilità di intraprendere questo percorso, per aver creduto in me in tutti questi anni, e per esserci in ogni momento, vi voglio bene.

Indice

Elenco delle figure

Capitolo 1

L'artista e la creazione dell'opera

Michele Sambin è un musicista, pittore e regista nato a Padova nel 1951. Dedicatosi fin da giovane al mondo dell'arte, utilizza la tecnologia come strumento e mezzo principale per realizzare i suoi lavori. La sperimentazione è la chiave delle opere dell'artista, che predilige l'impiego di tecnologie semplici e ben conosciute, rispetto ad altre più avanzate e nuove. Sebbene sia conosciuto in particolar modo per il teatro mulimediale con cui lavorò dal 1980, l'artista ebbe una pregressa esperienza con la sperimentazione audio-video [11]. Di particolare rilevanza è la tecnica del *videoloop*, inventata dall'artista nel 1978 e impiegata per realizzare l'opera *Il tempo consuma* (Figura 1.1 [12]), oggetto di questo lavoro.

1.1 Le origini de "Il tempo consuma"

Tutto nasce da un'idea che Sambin ebbe alla fine degli anni '70: "In quel periodo cercavo con determinazione un modo per ridurre a zero l'intervallo tra registrazione e possibilità di rivedere il registrato"[12]. Il miglior risultato raggiunto al tempo fu con l'opera *In Looking*, nella quale riuscì a suonare contemporaneamente con un sé stesso registrato, dovendo però riavvolgere il nastro manualmente. Fu in occasione della Galleria del Cavallino a Venezia, nell'autunno del 1978, che l'artista concepì l'idea di un nuovo sistema, tale da permettergli di interagire in tempo reale con un immagine registrata. Nel preparare le strumentazioni per la Galleria insieme agli altri artisti, Michele notò qualcosa aprendo un bobina Sony vergine: un pezzo di nastro adesivo argentato, predisposto dal produttore per effettuare giunture in caso di rotture [12]. Era solito effettuare operazioni di giuntura tra nastri audio (cosa abbastanza comune al tempo), ma mai gli capitò di farlo con dei nastri video; eccitato al solo pensiero ne parlò con due suoi colleghi

Figura 1.1: *L'opera originale "Il tempo consuma".*

artisti, i quali catturati dalla nuova idea decisero di aiutarlo a realizzare il sistema. Sambin non perse tempo e il giorno stesso disegnò uno schema raffigurante la strumentazione necessaria e la sua disposizione. Il giorno successivo i tre artisti si diedero appuntamento alla Galleria, dove cominciarono a predisporre i vari dispositivi. Terminata la fase di montaggio del sistema, occorreva testarne il funzionamento; il tasto *play* dei due videoregistratori venne premuto contemporaneamente e il nastro iniziò a scorrere; non sapendo bene in che modo verificare il sistema, Sambin decise di mettersi davanti alla telecamera, parlando e muovendosi. Dopo pochi secondi l'artista potè vedersi sullo schermo posto alle sue spalle, ripetendo le azioni compiute attimi prima. Più il tempo passava e più le immagini a schermo e i suoni riprodotti si sovrapponevano fra loro, generando un caos sempre maggiore: il sistema del *videoloop* funzionava!

La prima performace ad utilizzare il *videoloop* fu *VTR&I* (1978) [12], seguita da *Sax* (1979), *Anche le mani invecchiano* (1979) in Figura 1.2 [13], *Autointervista* (1980) e molte altre. Vi è un'opera che però spicca su tutte, sulla quale ovviamente si basa questo lavoro di restauro e riattivazione: *Il tempo consuma* del 1978. Purtroppo, tutta la produzione del *videoloop* venne trascurata nei primi anni della sua creazione, complice un minor interesse generale nei confronti delle performace dal vivo. Successivamente, a partire dal 1990 circa, le performance dal vivo e la video-arte riacquistarono valore, e di conseguenza gli studiosi e gli esperti ini-

Figura 1.2: *Immagine proveniente dalla performace "Anche le mani invecchiano".*

ziarono a mostrare un nutrito interesse per le opere di Michele Sambin. Oggi, la tecnica del *videoloop* e le performance di video-arte di Sambin sono considerate una serie significativa di opere nella storia della video-arte italiana, come si può leggere in molte fonti (libro monografico [7]; molti articoli sull'arte multimediale di Sambin [4] [5] [6] tra gli altri; esibizione monografica [8]; infine l'esibizione di video-arte nazionale [15]).

1.2 Il funzionamento del *videoloop*

Il sistema del *videoloop* teorizzato da Michele Sambin è composto dalla seguente strumentazione, come in Figura 1.3 [9]:

- **Un nastro video**.
- **Una telecamera**.
- **Due videoregistratori** (uno impiegato per registrare, l'altro per riprodurre).
- **Due monitor** (quello inferiore riproduce l'immagine nel nastro, quello superiore l'immagine proveniente dalla telecamera).

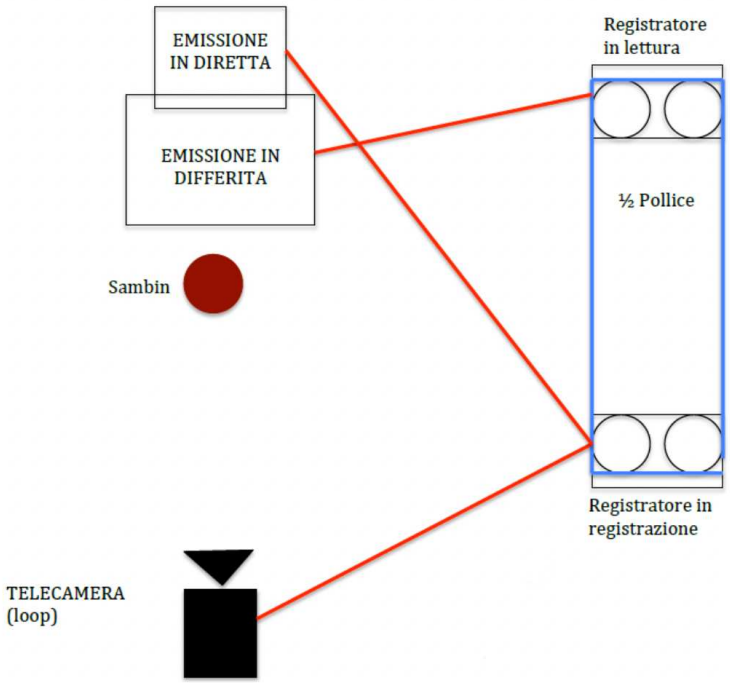

Figura 1.3: *Schema del sistema di videoloop originale.*

La telecamera è posizionata di fronte al monitor inferiore e collegata al videoregistratore con la funzione di registrazione, mentre l'altro videoregistratore (quello con la funzionalità di riproduzione) viene collegato al monitor inferiore, così da riprodurre il materiale registrato [10]. L'artista, posizionandosi tra la telecamera e il monitor, è quindi in grado di registrare la sua immagine contemporaneamente a quella riprodotta alle sue spalle (e registrata pochi secondi prima). Una caratteristica fondamentale è il tempo di *delay* (ritardo) che intercorre tra l'immagine registrata e quella in tempo reale; questa quantità è modificabile aumentando o diminuendo la distanza presente fra i due videoregistratori, poichè una maggiore distanza equivale ad un maggior tempo prima che il nastro passi da un videoregistratore all'altro, e viceversa. Grazie a questo sistema è possibile accumulare sul nastro, materiale audio e video in maniera pressochè infinita, con la conseguenza che, dopo un sufficiente numero di cicli, l'effetto di degradazione corromperà il materiale registrato, rendendo irriconoscibili suoni e immagini [10].

Capitolo 2

Primo restauro e riattivazione de ”Il tempo consuma”

Con il passare degli anni non fu più possibile replicare la performace de *Il tempo consuma* poichè Sambin non possedeva gli strumenti utilizzati per far funzionare il sistema. Dall'altro lato era presente una vasta documentazione riguardante il *videoloop*: appunti, schemi, registrazioni audio-video e immagini. Con queste premesse, nel 2021 su richiesta personale di Michele Sambin, Alessandro Fiordelmondo, Sergio Canazza, Niccolò Pretto e Paolo Ostan si occuparono della riattivazione digitale del *videoloop* all'interno del Centro di Sonologia Computazionale (CSC).

2.1 Dall'analogico al digitale

Il primo inconveniente affrontato riguardò l'attrezzatura utilizzata al tempo, poiché non venendo più prodotta divenne di difficile reperibilità; fu quindi presa la decisione di convertire il sistema analogico con uno digitale. Questo cambiamento introdusse aspetti positivi soprattutto a livello di praticità, essendo che in origine predisporre e gestire il sistema del *videoloop* non era un processo molto semplice: se prima i dispositivi erano ingombranti e pesanti, ora sono più pratici e di dimensioni ridotte; l'impiego del computer semplifica la gestione della performance, richiedendo l'azione di una singola persona (per l'opera originale erano necessarie due persone per far funzionare il sistema del *videoloop*).

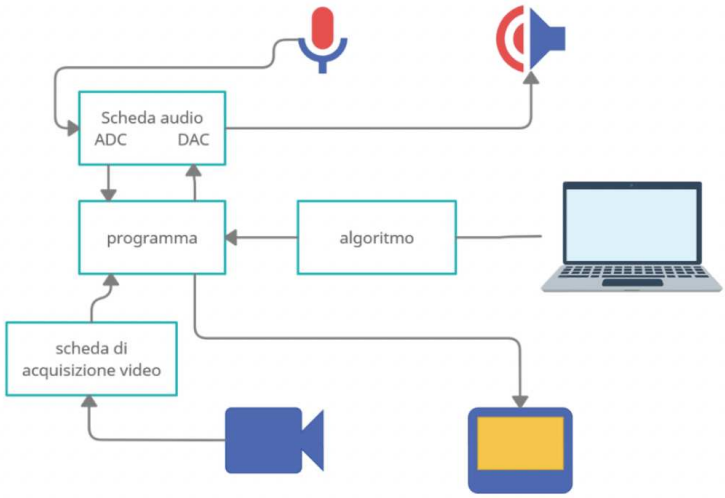

Figura 2.1: *Schema dei componenti hardware utilizzati nel restauro.*

2.2 I componenti hardware utilizzati

Di seguito viene presentata e descritta la lista dei componenti hardware digitali utilizzati per replicare il sistema analogico del *videoloop*, come in Figura 2.1 [9]:

- **Dispositivo di acquisizione video**: Telecamera accoppiata ad una scheda di acquisizione video.
- **Dispositivo di acquisizione audio**: Convertitore analogico-digitale con microfono a condensatore supercardioide.
- **Computer**: Unità di elaborazione principale; riceve il flusso video della telecamera e attraverso un algoritmo lo elabora per replicare l'effetto del *videoloop*.
- **Dispositivo di riproduzione video**: Schermo con risoluzione 16:9 col quale riprodurre l'opera.
- **Dispositivo di riproduzione audio**: Convertitore digitale-analogico con impianto di diffusione audio (speaker stereo).

2.3 La scelta del software

Il software scelto per realizzare la riattivazione del *videoloop* è *Max/MSP*, un ambiente di sviluppo grafico per la musica e la multimedialità. Questo software fu creato tra il 1980 e il 1990 da Miller Puckette, presso l'IRCAM di Parigi (Institute de Recherche et Coordination Acoustique/Musique), per permettere ai compositori l'accesso ad un sistema per la composizione musicale; attualmente viene distribuito dalla ditta *Cycling '74* di David Zicarelli, software designer che nel 1990 sviluppò ed estese *Max* (sotto il nome di *Max/Opcode*) per la *Opcode Systems Inc*. Gli impieghi di *Max/MSP* sono molteplici, dalla realizzazione di

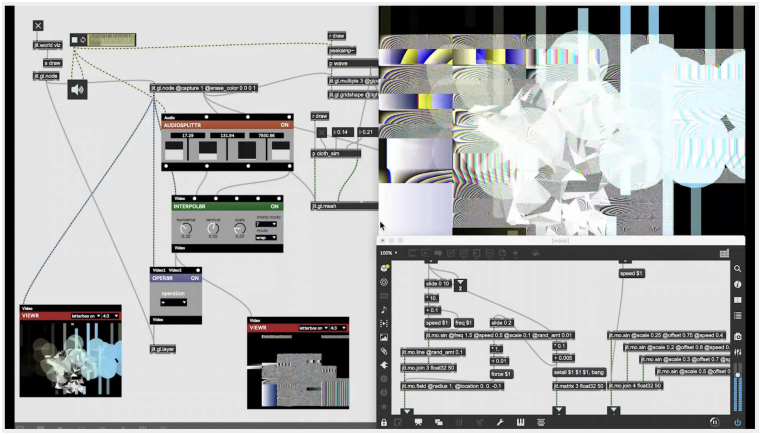

Figura 2.2: *Esempio di una performace realizzata con Max/MSP.*

performace video-musicali in tempo reale come in Figura 2.2 [2], all'elaborazione di *plug-in audio* per case di produzione di sistemi *hi-fi*, fino alla creazione di file inseribili in applicazioni *standalone*. Alla base di questo ambiente vi è l'utilizzo di oggetti, ognuno con funzioni e caratteristiche proprie; essi vengono collegati insieme tramite dei cavi, permettendo il passaggio di segnali audio/video e dati [1]. *Max/MSP* si divide in tre componenti:

- **Max**: Componente del linguaggio dedicata alla generazione e alla gestione grafica di flussi dati. Utilizza anche segnali (ad esempio il *MIDI*) per il controllo di unità esterne dedicate alla sintesi audio [16].
- **MSP**: Implementato successivamente alla componente originale Max, permette di catturare, sintetizzare e manipolare l'audio attraverso l'utilizzo di algoritmi; vi è la possibilità di collegare hardware addizionale per interfacciarsi col mondo fisico [16].

- **Jitter**: Estensione del linguaggio dedicata alla generazione ed elaborazione grafica; i suoi oggetti consentono la manipolazioni di matrici di pixel, come l'acquisizione e la manipolazione di immagini e video.

2.4 Lo sviluppo dell'algoritmo

Sebbene si sia passati da un ambiente analogico ad uno digitale, il processo di creazione dell'effetto del *videoloop* è rimasto identico. Ciò che è cambiato è il modo con cui si ottiene il tempo di ritardo tra l'immagine in tempo reale e quella registrata: per replicare il tempo necessario a percorrere la distanza tra un videoregistratore e l'altro, è stato sviluppato un algoritmo per generare un ritardo nel flusso audio/video. Esso è diviso in due parti, una per ciascun flusso:

- **Video**: il sistema acquisisce il flusso video tramite l'oggetto `jit.grab` (Figura 2.3 [9]); dopo averne controllato caratteristiche e parametri (ad esempio i *frame per seconds - FPS*) lo invia alla sezione di delay (Figura 2.4) [9] che genera una sua copia ritardata nel tempo. In particolare la matrice dei colori viene divisa nelle sue componenti tramite l'oggetto `jit.matrixset` [3], e in seguito viene elaborata solo una delle matrici risultanti (questa operazione viene effettuata per rendere più leggera l'elaborazione dei dati). I singoli frame vengono poi riprodotti secondo un indice ascendente fornito dall'oggetto `counter`.

- **Audio**: il flusso audio viene acquisito tramite l'oggetto *adc* ed equalizzato tramite una serie di tre filtri, più precisamente da un filtro passa-alto (frequenza di taglio intorno ai 100 Hz) e due filtri a campana, come in Figura 2.5 [9]; tali filtri vengono generati dall'oggetto `biquad` [3]. Il flusso audio equalizzato viene trasmesso alla sezione di delay, che genera una sua copia ritardata nel tempo. Per memorizzare e trasmettere i segnali si utilizza l'oggetto `MSP buffer` . Rimane la sincronizzazione del flusso audio con quello video; per questa operazione si utilizzano tre buffer sincronizzati tra loro, più precisamente **registrazione** (acquisizione del segnale audio), **riproduzione** e **cancellazione** (pulizia del segnale).

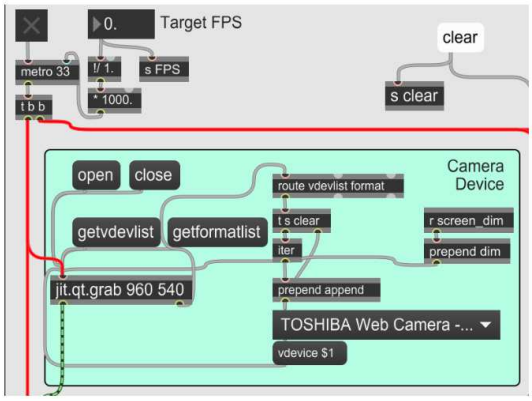

Figura 2.3: *Blocco dell'acquisizione video dell'algoritmo.*

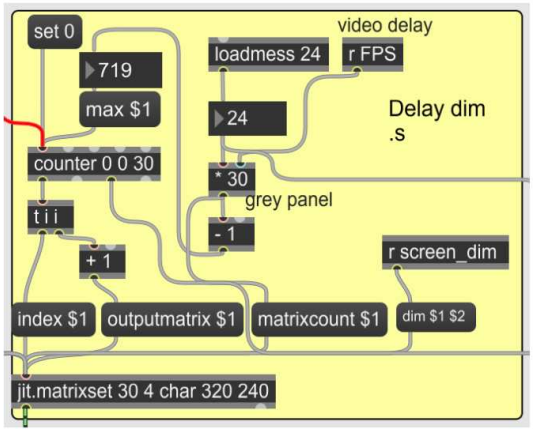

Figura 2.4: *Parte dell'algoritmo responsabile del delay video.*

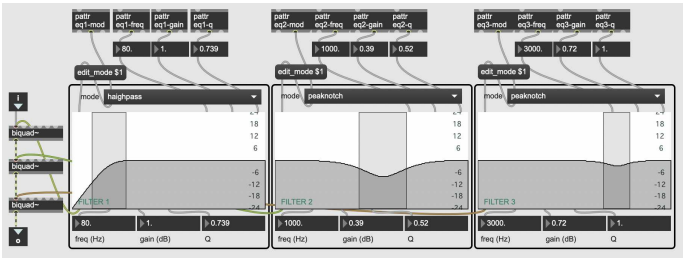

Figura 2.5: *Sistema di equalizzazione dell'audio, formato da tre filtri.*

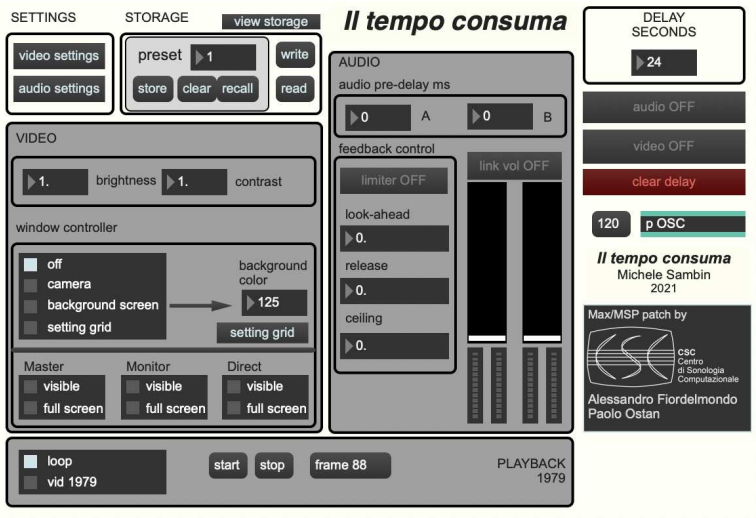

Figura 2.6: *Interfaccia grafica del programma.*

2.5 L'interfaccia grafica

Come possiamo notare dalla Figura 2.6, l'interfaccia grafica è divisa in sezioni, costituite come segue:

- **Settings**: qui vengono selezionate le sorgenti audio e video da utilizzare, insieme ai rispettivi parametri di input (dimensione pixel, frequenza audio, ecc.).
- **Storage**: memorizza le impostazioni dei vari parametri del software.
- **Video**: si ha la possibilità di regolare *luminosità* e *contrasto* dell'immagine; vi è inoltre un oggetto che permette di spegnere la camera, oppure impostarla in modalità ripresa, sfondo o griglia di settaggio.
- **Audio**: contiene il sistema di equalizzazione, il controllo del feedback audio e gli indicatori dei livelli sonori.
- **Delay seconds**: necessario per impostare il tempo di *delay* desiderato, ovvero il numero di secondi di ritardo dalla registrazione alla riproduzione. I due bottoni grigi sottostanti hanno la funzione di attivare/disattivare il flusso audio e video istantaneamente, mentre quello rosso rimuove i dati presenti nel buffer del delay.

2.6 Comparazione tra la versione analogica e digitale

Grazie a questo processo di riattivazione e restauro è stato riportato in vita *Il tempo consuma*, permettendo ancora una volta al pubblico di assistere e interagire con l'opera creata più di quarant'anni fa. Mettendo però a confronto la perfomance analogica con quella digitale, come in Figura 2.7 [12] [14], non possiamo non notare delle differenze, soprattutto a livello visivo. Questo è dovuto all'impiego di strumentazione digitale anziché analogica, portando all'elaborazione dei flussi audio e video in maniera differente, per certi aspetti, rispetto alla performance del 1978. La domanda che sorge spontanea è la seguente: si è mantenuta l'originalità dell'opera? Una cosa è certa, il processo per ottenere la degradazione è il medesimo dell'opera originale, ovvero sfruttare lo spazio presente tra la camera, il performer, e il monitor alle sue spalle. Questo, unito alla presenza dell'artista durante le opere di riattivazione, sono sufficienti a poter garantire l'originalità di questo lavoro di restauro.

Figura 2.7: *Nella figura (a) possiamo vedere le immagini della performance originale, mentre nella figura (b) quella odierna.*

2.7 Il tempo consuma 2.0

In seguito all'esecuzione di varie performace è nata la necessità di apportare modifiche al programma: ci sono funzioni da migliorare, alcune che possono essere rimosse, mentre altre richiedono di essere implementate. Per questo motivo si è deciso di procedere ad un ulteriore lavoro di riattivazione dell'opera, con lo scopo di migliorare le funzionalità del programma e la sua accessibilità. Nei prossimi capitoli verrà esposto il lavoro effettuato, dividendolo in tre parti: Audio, Video e Interfaccia grafica.

Capitolo 3

Espansione delle opzioni Video

Il sistema di gestione del flusso video non ha bisogno di modifiche a livello di algoritmo; sono state aggiunte funzionalità non presenti nel primo lavoro di riattivazione, con lo scopo di offire più libertà all'artista nell'esecuzione della performance.

3.1 Aggiunta dei colori

Originariamente, *Il tempo consuma* era una performace realizzata esclusivamente in bianco e nero, caratteristica dovuta all'attrezzatura esistente nel 1978. Si è deciso di aggiungere la possibilità di visualizzare l'immagine video sia come l'opera originale, sia a colori, per permettere all'artista nuove sperimentazioni rispetto al passato. Le due opzioni selezionabili sono:

- **Monochrome**: scala di grigi.
- **Color**: scala dei colori, RGBA (*Red, Green, Blue, Alpha* - Rosso, Verde, Blu, Alpha).

Per implementare questa opzione sono stati aggiunti degli oggetti per permettere di elaborare in maniera differente il flusso video acquisito da `jit.qt.grab`, come in Figura 3.1. Esso viene collegato a `jit.unpack`, il quale scompone il flusso dati nelle quattro matrici dei colori, rispettivamente *Alpha*, *Red*, *Green* e *Blue* [3]. Le matrici vengono poi riunite e il flusso video viene trasmesso ad un oggetto `switch`, il quale riceve sia quello monocromatico che quello a colori. Il passaggio di uno o dell'altro viene permesso dallo *switch*, relativamente all'opzione selezionata [3].

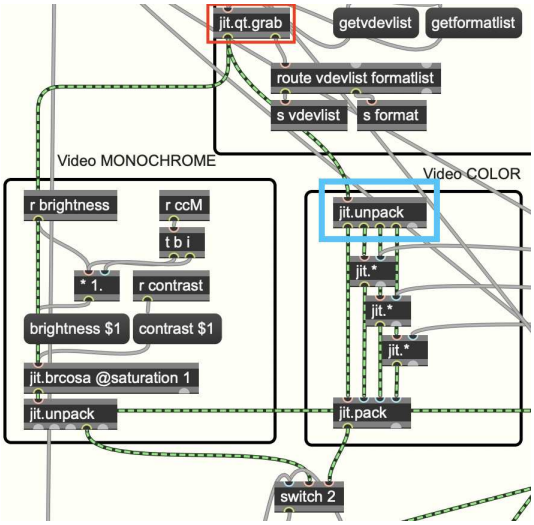

Figura 3.1: *Acquisizione dell'immagine a colori.*

3.2 Problema nella gestione dei parametri

In seguito alla modifica sopra citata si è presentato un problema nella gestione del flusso video: selezionando l'opzione **Color** si ottiene come output un immagine prevalentemente di colore giallo. Questo errore è dovuto ad un valore errato di uno dei parametri di `jit.matrixset`, più precisamente quello che indica il numero di matrici colore da elaborare. Nel caso dell'immagine monocromatica l'algoritmo elabora solo una delle quattro matrici acquisite (le altre non sono necessarie); al contrario, selezionando l'opzione **Color** l'algoritmo deve obbligatoriamente elaborare tutte e quattro le matrici per restiuire un'immagine corretta. Per ovviare questo è stata creata una condizione, come in Figura 3.2, la quale in base all'opzione selezionata restituisce in output il numero corretto di pianicolore da elaborare. Il parametro viene poi inviato a `jit.matrixset` tramite il messaggio `planecount`, che ha la funzione di impostare il numero di matrici [3].

3.3 Controllo delle matrici RGB

Oltre all'aggiunta dell'opzione **Color**, si è deciso di implementare un controllo di intensità per ciascuna delle singola matrici. Vengono utilizzati tre oggetti `jit.*` come in Figura 3.1, che hanno la funzione di impostare il valore d'intensità del colore [3]; gli oggetti sono connessi in entrata a `jit.unpack`, e in uscita a `jit.pack`,

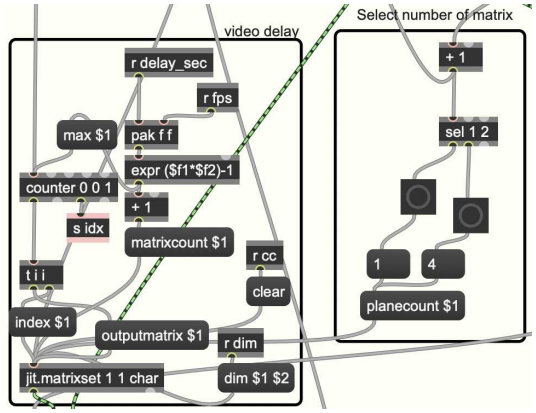

Figura 3.2: *Ciclo di selezione del numero di matrici di colore.*

ed ognuno di essi è collegato alla rispettiva matrice: rossa, verde e blu, ad esclusione della alfa. In questo modo l'artista potrà impostare i valori delle matrici in base alla di palette cromatica desiderata. Unica limitazione sta nel range di valori impostabile: il valore minimo è settato a zero, mentre il massimo ad uno; in caso contrario l'immagine risulterebbe troppo snaturata o addirittura assente. (con il valore zero viene prodotta un'immagine completamente nera). Per quanto riguarda l'impostazione effettiva dei valori, questa verrà discussa nel capitolo rigurdante l'interfaccia grafica, alla sezione 5.3.

Capitolo 4

Modifica dell'equalizzazione

Per quanto riguarda la parte audio del programma, l'idea è quella di realizzare un controllo del feedback audio attraverso i parametri di filtraggio comuni in un mixer analogico. Si ricorre sempre all'utilizzo di oggetti `biquad`, i quali generano dei filtri impostabili [3]. Viene aggiunto un ulteriore filtro, portando il loro numero a quattro, come in Figura 4.1; successivamente vengono modificati i loro parametri, in base al tipo di filtro necessario.

4.1 Sostituzione del filtro *passa-alto*

Il filtro *passa-alto* viene sostituito da un *low-shelf*. La differenza sostanziale tra questi due filtri (come quella tra un *passa-basso* ed un *high-shelf*) sta nel modo con cui modificano il flusso audio. Laddove il primo elimina le frequenze al di sotto della soglia impostata, il secondo le attenua di un fattore dipendente dal guadagno (*gain*) impostato. Il compito di questi filtri è quindi quello di attenuare certe frequenze anziché rimuoverle completamente, con l'effetto di produrre un suono più naturale e lineare. Per permettere di filtrare anche le frequenze alte è stato aggiunto un filtro *high-shelf*, non presente nel primo equalizzatore. Le frequenze di lavoro alle quali vengono impostati i due filtri sono le seguenti:

- **Low-shelf**: 250 Hz
- **High-shelf**: 4000 Hz

La frequenza alla quale vengono impostati è fissa, pertanto l'unico parametro che può essere modificato è il guadagno (anche detto *gain*).

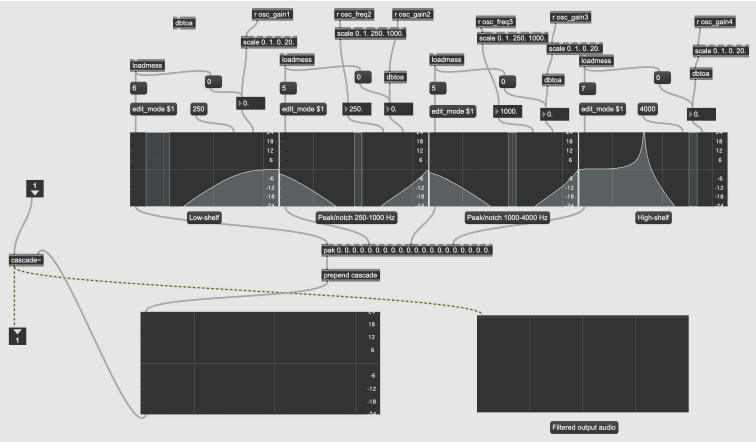

Figura 4.1: *Sistema di equalizzazione formato dalla cascata di quattro filtri.*

4.2 Modifica dei filtri *peak/notch*

Si è deciso di mantenere i due filtri *peak/notch* (o filtri a campana), poichè utili per eliminare frequenze risonanti che potrebbero compromettere il segnale e generare feedback indesiderati [9]. Si agisce però sulle loro frequenze di lavoro, impostando i range di valori come segue:

- **1° peak/notch**: 250-1000 Hz
- **2° peak/notch**: 1000-4000 Hz

In questo caso i parametri modificabili per questi due filtri sono due: il guadagno, come nel caso dei due filtri *shelving*, e la frequenza centrale alla quale operano; questa dovrà essere modificata in base all'ambiente in cui si svolgerà la performace, unitamente alla strumentazione utilizzata.

4.3 Output dell'audio filtrato

Ognuno dei quattro filtri descritti in precedenza produce come output una serie di cinque coefficienti, i cui valori dipendono dai parametri impostati precedentemente. Unendo i coeffcienti dei filtri nel rispettivo ordine possiamo realizzare un unico filtro tramite l'oggetto `cascade` , il quale genera un filtro a cascata sulla base dei coefficienti ricevuti [3]. È stato aggiunto anche un oggetto `spectroscope` , il quale genera uno spettrogramma che, collegato all'output della serie di filtri, permette di visualizzare la forma d'onda equalizzata [3].

Capitolo 5

La nuova interfaccia grafica

La prima versione del programma presentava un interfaccia grafica funzionale, ma era chiaro che fosse necessario apportarvi delle modifiche. In primo luogo bisognava rendere la *GUI* più accessibile e di semplice utilizzo per l'artista; successivamente dovevano essere integrate le nuove funzionalità nell'interfaccia. Anche l'interfaccia grafica necessitava di una modifica, poichè nella prima versione la disposizione delle varie sezioni era abbastanza confusionaria.

5.1 Settings

In questa sezione troviamo due bottoni interagibili, *video settings* e *audio settings*, come in Figura 5.1; ognuno di essi permette di aprire la finestra di configurazione della relativa risorsa. Le finestre contengono le seguenti impostazioni:

- **video settings**: risoluzione dell'immagine con un rapporto fisso a 16:9, *camera device*, formato dei colori e frame per secondo (*fps*).
- **audio settings**: driver audio, dispositivo di input e output, tasso di campionamento (*Sampling rate*) e altre impostazioni riguardanti i vettori audio da utilizzare.

5.2 Storage

Qui troviamo i vari comandi per gestire il salvataggio, la scrittura e la modifica dei parametri del programma, come in Figura 5.1. Gli oggetti interagibili sono i seguenti:

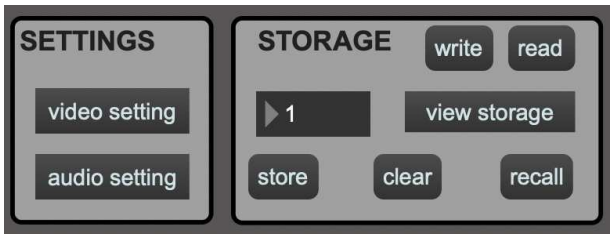

Figura 5.1: *Setting (a sinistra) e Storage (a destra).*

- **view storage**: apre una finestra contenente l'elenco dei vari oggetti del programma, insieme al loro valore in quel momento.
- **write**: permette di salvare in un file *.json* i parametri del programma.
- **read**: permette di leggere i parametri del programma da un file *.json* nel quale sono stati salvati.
- **store/clear/recall**: comandi per salvare/pulire/richiamare la memoria dei parametri.

5.3 Video

Per quanto riguarda la sezione video, luminosità (*brightness*) e contrasto (*contrast*) erano già presenti nella prima interfaccia. È stata aggiunta la possibilità di scegliere la palette cromatica dell'immagine, come spiegato alla sezione 3.1 e mostrato in Figura 5.2. Per poter selezionare l'opzione desiderata si è utilizzato l'oggetto `radiobutton`, il quale fornisce un interfaccia personalizzabile per la selezione delle opzioni [3]; nel nostro caso abbiamo un box con due opzioni di selezione. Vi è anche la possibilità di modificare le tre matrici dei colori *RGB*, come anticipato precedentemente alla sezione 3.3. Il controllo avviene tramite un oggetto chiamato `multislider` formato da tre cursori, ognuno dei quali permette di modificare il valore della rispettiva matrice [3]. I parametri vengono acquisiti tramite gli oggetti `jit.*` e sono collegati direttamente al multislider.

5.4 Audio

Dopo aver modificato il sistema di filtri per l'equalizzazione audio, si è ritenuto necessario avere la possibilità di intervenire sugli stessi anche nel corso della

performance. Per permettere ciò vengono utilizzati sei oggetti `dial`, quattro per regolare il guadagno di ogni filtro, più due per impostare la frequenza di lavoro dei due filtri a campana, come in Figura 5.3. Questi oggetti restituiscono in output un valore corrispondente al loro grado di rotazione, permettendo di regolare a piacimento i parametri sopra indicati [3]. Per poter funzionare correttamente i *dial* sono collegati al sistema di equalizzazione tramite gli oggetti *reciever* e *sender*. Il primo si occupa di mandare il segnale provienete dal *dial*, il secondo trasmette il valore al filtro corrispondente cambiando quindi l'equalizzazione. L'opzione *feedback control* è stata rimossa poichè rivelatasi superflua dopo un periodo di utilizzo del software.

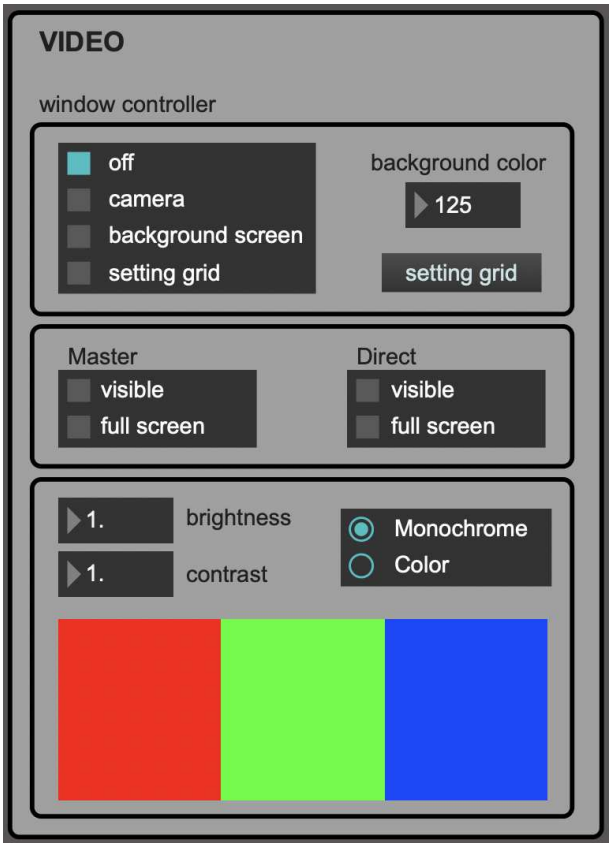

Figura 5.2: *Sezione Video del programma.*

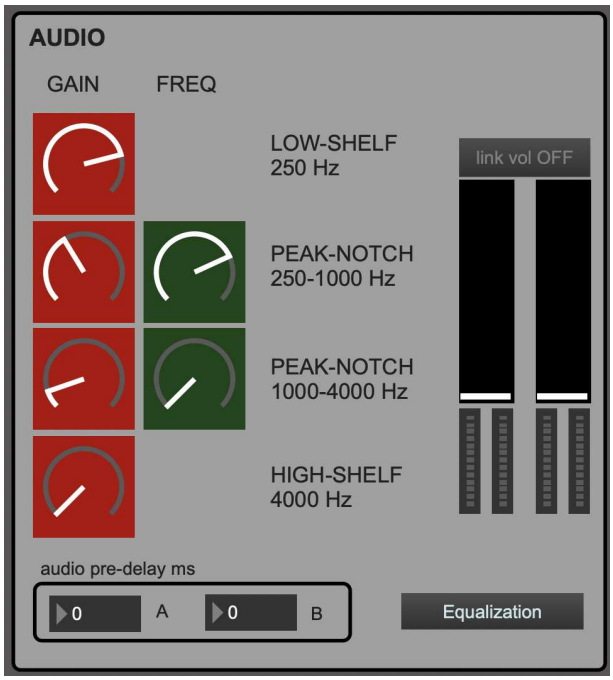

Figura 5.3: *Sezione Audio del programma.*

5.5 Window Controller

Qui è presente l'oggetto *radiogroup* per la selezione della camera da visualizzare, come in Figura 5.2. Le opzioni sono:

- **off**: disattiva la visualizzazione della camera.
- **camera**: attiva la visualizzazione della camera.
- **background screen**: genera uno sfondo statico e monocromatico, utilizzato per i settaggi iniziali della performance.
- **setting grid**: apre il sistema di settaggio della griglia, necessario per allineare la camera al monitor.

È stata lasciata solamente la possibilità di visualizzare il monitor diretto (con il quale viene visualizzato l'input della camera in tempo reale) ed il monitor loop (con il quale viene visualizzato l'input della camera ritardato).

5.6 Initialize/Play/Stop button

Si è deciso di aggiungere due bottoni interagibili per conferire più immediatezza al controllo in tempo reale della performance, come in Figura 5.4. Il primo bottone è formato da una serie di azioni, composta come segue:

- **INITIALIZE**: avvia il processo di inizializzazione della performance (*background screen*).
- **WAIT**: presente fino a quando il tempo di delay non sarà terminato, non può essre clickato.
- **PLAY**: dà il via alla performace.
- **playing...**: indica che la performace è attualmente in riproduzione, non può essere clickato.

Il secondo invece svolge un'unica azione:

- **STOP**: interrompe l'esecuzione della performance istantaneamente.

Gli oggetti utilizzati sono dei `textbutton`, semplici bottoni con testo capaci di inviare segnali. Il bottone *PLAY* è stato collegato ad un oggetto contatore `counter` con in serie un oggetto selettore `sel`. Il contatore cresce ad ogni click del primo *textbutton*, ed in base al suo valore il selettore permetterà lo svolgersi di una delle azioni sopra citate. Il bottone *STOP* è stato collegato in modo da poter unicamente arrestare il flusso video.

5.7 L'interfaccia completa

In Figura 5.5 possiamo osservare il risultato delle modifiche apportate alla interfaccia precedente. Oltre alle sezioni sopra descritte è presente quella riguardante il controllo del *delay*, rimasta invariata poichè non necessita di modifiche. I tasti **AUDIO ON/OFF** e **VIDEO ON/OFF** sono mantenuti dalla vecchia interfaccia, poichè utilizzati per attivare/disattivare le sorgenti in maniera immediata.

Figura 5.4: *Pulsanti aggiunti per il controllo della performance.*

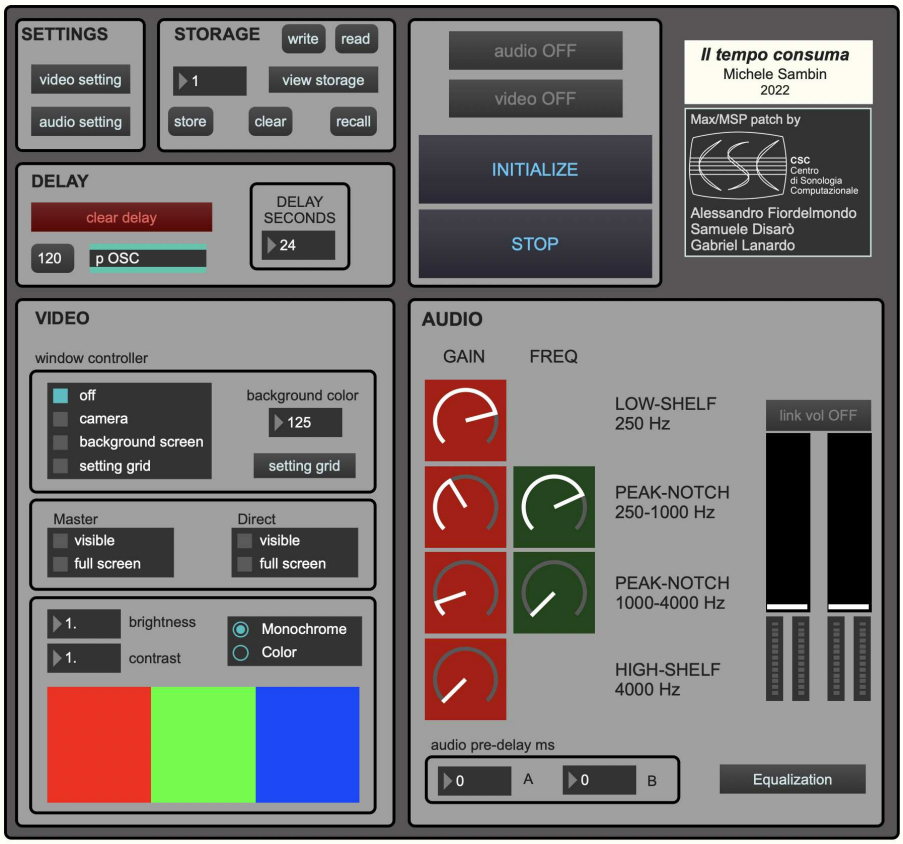

Figura 5.5: *Interfaccia del programma con tutte le sezioni.*

Capitolo 6

Conclusioni

Il lavoro descritto in questa tesi è incentrato sulla modifica e il miglioramento del software per la gestione della performance *Il tempo consuma*, opera del 1978 dell'artista padovano Michele Sambin, riattivata nel 2021 tramite un processo di digitalizzazione. Viene ripercorsa la storia dell'artista e le origini dell'opera, esponendo le motivazioni e i processi che hanno portato alla creazione del sistema del *videoloop*, passando poi all'analisi del funzionamento della tecnica originale e dei dispositivi che la componevano. Successivamente viene presentato il primo lavoro di riattivazione effettuato nel 2021 dal Centro di Sonologia Computazionale (CSC); qui è stato descritto il processo di creazione del nuovo sistema del *videoloop*, le parti che lo costituiscono e il loro funzionamento. Inoltre sono stati riportati i problemi riscontrati nella fase di conversione del sistema da analogico a digitale, e la loro soluzione. In seguito è stato esposto il nuovo lavoro di modifica al software di gestione del *videoloop*, volto a migliorare il funzionamento e l'accessibilità del sistema: sono state discusse le funzioni implementate (poichè mancanti) e quelle modificate, spiegando i motivi che hanno portato a queste scelte e le azioni effettuate. Infine sono state trattate le modifiche all'interfaccia grafica, attuate sia per facilitarne il suo utilizzo, che per migliorarne l'aspetto grafico.

Sia per la natura artistica dell'opera sulla quale è basata questa tesi, sia per la tecnologia utilizzata, in futuro potranno essere apportate ulteriori modifiche al software, sebbene la sua nuova versione abbia già implementato aspetti mancanti e risolto molti problemi. Un aspetto che potrebbe essere trattato in un futuro lavoro è la gestione del sistema tramite *smartphone*, realizzabile creando un'applicazione su misura, oppure utilizzando *app* già esistenti per il controllo di sistemi (ad esempio *OSC Pilot*).

Bibliografia

[1] A. CIPRIANI AND M. GIRI, *Musica Elettronica e Sound Design - Teoria e Pratica con Max 8*, ConTempoNet, 2009.

[2] CYCLING'74, *Cycling '74*. https://cycling74.com/products/max.

[3] ——, *Max reference*. https://docs.cycling74.com/max8/vignettes/docrefpages.

[4] F. D. D'AMICO, *Le invenzioni tecnologiche di michele sambin nello spazio teatrale*, Mimesis Journal. Scritture della performance 10, 50 (2021), pp. 91–108.

[5] F. GALLO, *Intimacy and emotions at the dawn of performance art in italy*, Modern Italy, 3 (2021), pp. 275–289.

[6] L. LEUZZI, *Re-enacting early video art as a research tool for media art histories*, In Digital art through the looking glass: new strategies for archiving, collecting and preserving in digital humanities, Donau-Universität Krems (2021), pp. 161–178.

[7] L. LISCHI, SANDRA E PAROLO, *Michele Sambin: performance tra pittura, musica e video*, 2014.

[8] B. D. MARINO, *Archè/téchne*. Michele Sambin, 2002.

[9] P. OSTAN, *Un modello per il trasferimento delle performance multimediali nel dominio digitale. un studio di caso: "il tempo consuma" di michele sambin*, Master's thesis, Università di Padova, 2021.

[10] L. PAROLO, *Il linguaggio artistico di michele sambin dal film alla video-performance musicale (1968-1982). ipotesi per la conservazione, il restauro e la riproposta attuale di looking for listening (1977)*, Master's thesis, Università di Padova, 2013.

[11] R. RIVI, *Più della vita*, (2019).

[12] M. SAMBIN, *Il tempo consuma*. https://michelesambin.com/projects/il-tempo-consuma, 1978.

[13] ——, *Anche le mani invecchiano*. https://michelesambin.com/projects/anche-le-mani-invecchiano, 1979.

[14] ——, *Il tempo consuma live*. https://michelesambin.com/projects/il-tempo-consuma-live-2022, 2022.

[15] C. VALENTINI, VALENTINA E SABA, *Il video rende felici*, 2022.

[16] G. WANG, *A history of programming and musica*, (2012).