Frederik la Cour [201707873]
Digital Lyd
04-01-2021
Antal tegn: 30.416

Digital Lyd Eksamen

I denne opgave har jeg valgt at arbejde med et uddrag af den litauenske komponist og maler M. K. Čiurlionis billede-sonater. Idet Čiurlionis billedsonater beror sig i et synæstetisk forhold. Den visuelle dialogiske relation mellem klassiske kompositions teknikker og malerkunsten [1]. Dette spænd undersøges og reflekteres gennem en konstruktiv praksis med programmeringssproget max/MSP. Den konstruktive praksis med Čiurlionis billedsonater kan siges at være i overensstemmelse med kravet om, at billeder skal indgå i et enten konceptuelt eller direkte forhold i udfærdigelsen af en algoritmisk komposition.

Synæstesi er et generelt begreb der dækker over mange former for sensoriske krydsninger hvori stimuleringen af en af de fem sanser produceres i respons til en af de andre sanser [2, p. 42]. Det synæstetiske fænomen farve-høring er en specifik variation af synæstesi der ifølge Jonathan W. Bernard har været undersøgt videnskabeligt siden 1900-tallet. I undersøgelserne af farve-høring fandt man frem til, at fænomenets manifestation individuel. Der er med andre ord ikke nogen specifik relation i farve-hørings fænomenet fra person til person [2, p. 42].

Andre komponister med synæstesi der også arbejder med grænsefladen mellem kunst og musik, er komponisten Alexander Scriabin og hans farve associationer til toner i kvintcirklen. Hans synæstesi kan siges at forene farver med tone-højder hvori de enkelt toner tillægges betydninger med afsæt i Rudolf Steiners farve teori [3].

Dog har jeg valgt at arbejde med Čiurlionis fremfor Scriabin, idet Čiurlionis synæstesi ikke har med direkte korrespondance mellem tone-højde og farver at gøre, i stedet beror hans synæstesi i krydsning mellem malerkunsten og komponistens metoder; rytmer i linje og plan, fleksible bølgeformer og overlappende landskaber der skaber rummelighed [1]. Denne metodiske krydsning finder jeg mere interessant end forsøget i at konstruerer en arbitrær relation mellem tone-højde og farver.

Figur 1 M. K. Čiurlionis - Sonata of the Sun

I Čiurlionis billedsonater blandes forskellige motiver fra virkeligheden indenfor forskellige rummeligheder og tidsligheder, såvel som kontrasterende symbolske billeder, der gennem sonatens dynamiske bestanddele struktureres [1]. Čiurlionis billedsonater består af tre til fire elementer; *Allegro, Andante, Scherzo & (Finale)*. Et eksempel kunne være Čiurlionis *Sonata of the Sun* (Figur 1) hvor rækkefølgen korrespondere til sonatens bestanddele fra venstre mod højre.

Med mit begrænsede kendskab til sonatens kompositionelle træk så spilles *Allegroen* hurtigt og dramatisk, *Andanten* spilles i langsomt gående tempo, *Scherzoen* kan siges at have en lettere og mere dæmonisk karakter. Endelig spilles *Finale* livligt med et motiv der gentages igen og igen. Čiurlionis har i sine billedsonater arbejdet med denne sonate-struktur som fundament.

Sonate-strukturen har også været grundlaget for udviklingen af min produktion. Arbejdet sonate-strukturen i en algoritmisk komposition har givet anledning til spørgsmål: Hvordan kan et hierarkisk kompositions system som sonaten forenes med den algoritmiske kompositions iboende organiske og tilfældige egenskaber[4]? Hvilken musisk diskurs vil en sådan kombineret algoritmisk sonate system udmunde i [5]? hvilken grad af handlekraft tildeles brugeren i interaktion med systemet? Hovedfokusset i produktionen har været konstruktionen af de algoritmisk/generative principper rodfæstet i databehandlingen af Čiurlionis billede-sonater.

Fremgangsmåde

Udgangspunktet for udvælgelsen af billedsonater var erfaringer fra tidligere projekter med billede-behandling i max/MSP, kombineret med min egen smag, i forhold til hvilke billeder jeg syntes bedst om. I forhold til databehandlingen var vigtigt at sonaterne havde en vis grad af diversitet i forhold til farve, tekstur og struktur. Formålet med diversiteten indenfor de førnævnte aspekter var at opnå så forskelligartede udtryk både visuelt, auditivt og

Figur 2 M. K. Čiurlionis - Sonata of the Pyramids, Sonata of the Sun & Sonata of the Spring

kompositionsmæssigt som muligt. Det endelige udvalg bestod af *Sonata of the Pyramids, Sonata of the Spring og Sonata of the Sun* (Figur 2).

Efter udvælgelsen af billeder udvikles først "*imageProcessing.maxpat*" hvori billederne analyseres. Dernæst udvikles "*mainpatcher.maxpat*" der udgør samlingen af systemet. Det er her *bpatchere* og *subpatchere* anvendes i samspil med forskelligt farvede baggrunde og kabler med henblik på at organiserer og skabe overblik over de forskellige moduler i det samlede system. I "*mainpatcher.maxpat*" udvikledes et modul der har med omsætning af billeddata til sekventiel midi data at gøre. Efterfølgende er der arbejdet særligt med hvordan indsamlingen og struktureringen af billeddata behandles. Ligeledes hvordan den indsamlede data kan opsplittes i fire kanaler, der repræsenterer de fire bestanddele af sonaten. Dernæst er et mindre tilfældighedssystem med afsæt i den indsamlede data konstrueret, og endelig er der tilføjet lydgenerende enheder i form af en klassisk digital synthesizer med filter og envelope, samt en sampler fra min portefølje og et *noteout* max objekt der muliggør sending af midi data til en ekstern synthesizer.

For at anvende systemet vælges først et billede fra en af Čiurlionis billedsonater, hvorefter der klikkes på en farve. "*Threshold"* parameteret indstilles i forhold til hvor meget af den valgte farve der skal videre behandles. Dernæst aktiveres "*Render"* knappen hvorefter tempo, skalering og step-længde kan ændres. Brugeren står herefter overfor valget mellem at arbejde videre med billedet. Dette kan ved at aktiverer "*New iteration"* knappen hvorefter en variation af databehandling af samme billede kan foregå. Alternativt kan brugeren skifte til et nyt sonate-billede ved at angive billedets navn i ImageProcessing sektionen af "Mainpatcher.maxpat".

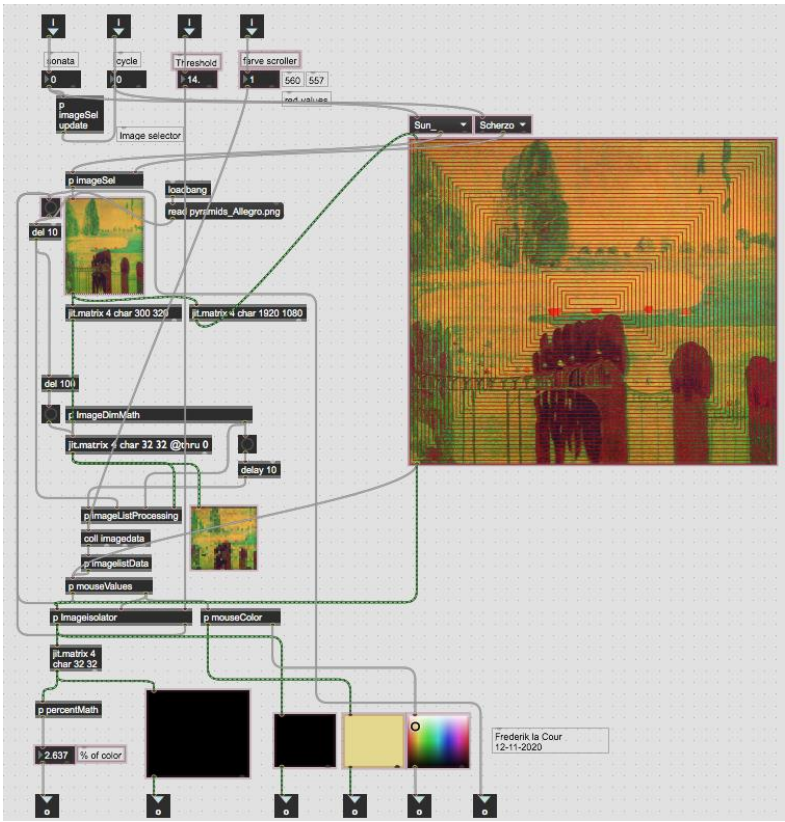
Når bruger har analyseret x antal billeder kan databehandlingen gemmes ved at eksporterer det som en "*.json*" fil. Her kan tidligere databehandlings data også loades i "Mainpatcher.maxpat". Dernæst har bruger mulighed for at aktiverer "*Sonata splitter"* knappen for at opsplitte den indsamlede databehandling i hver sin sonate kanal. efterfølgende kan disse kanaler aktiveres eller automatiseres ved at aktiverer "*turn on sonata cycle"* knappen. Brugeren kan således skifte mellem de forskelligt generede frekvenser, ændrer på parametre i forhold til notationen. Derudover kan bruger også vælge hvordan notationen skal opføres i forhold til om det skal ske gennem hhv. digital synthesizer, sampler eller en ekstern hardware synthesizer.

Dette afsnit har været et kort overblik over fremgangsmåden i forhold til udfærdigelsen og anvendelsen af systemet. Videre i denne opgave vil disse bestanddele af systemet blive diskuteret i forhold til bl.a. Umberto Eco's begreb om 'open work'[5, p. 3], iboende magtstrukturer, og videre diskuterer brugerens rolle i sin interaktion med systemet.

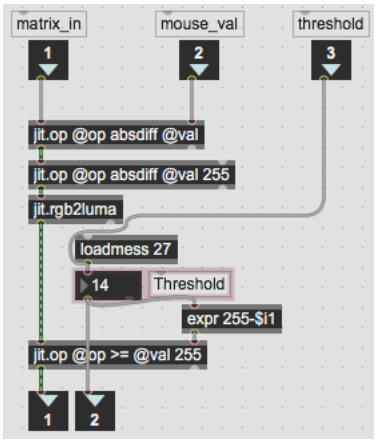
Frederik la Cour [201707873]
Digital Lyd
04-01-2021
Antal tegn: 30.416

imageProcessing.maxpat

De udvalgte billedsonater uploades til *"imageProcessing.maxpat"* (Figur 3) som en selvstændig patch der så efterfølgende kaldes i "*Mainpatcher.maxpat"* gennem et bpatcher objekt.



Figur 3 ImageProcessing.maxpat



Figur 4 p Imageisolator

I bpatcheren *"imageProcessing.maxpat"* vælges først hvilken sonate der ønskes analyseret. Herefter vælges hvilken af de fire bestanddele af sonaten. Dette er demonstreret i (Figur 3) hvor *sonata of the sun's scherzo* er behandles. Billedet er overlagt med max-objektet *suckah* der ses repræsenteret af de røde linjer der ligger over det store billede i (Figur 3). *Suckah* objektet gør det muligt at interagerer et vilkårligt sted på billedet med musen og udtage dets RGB-værdier.

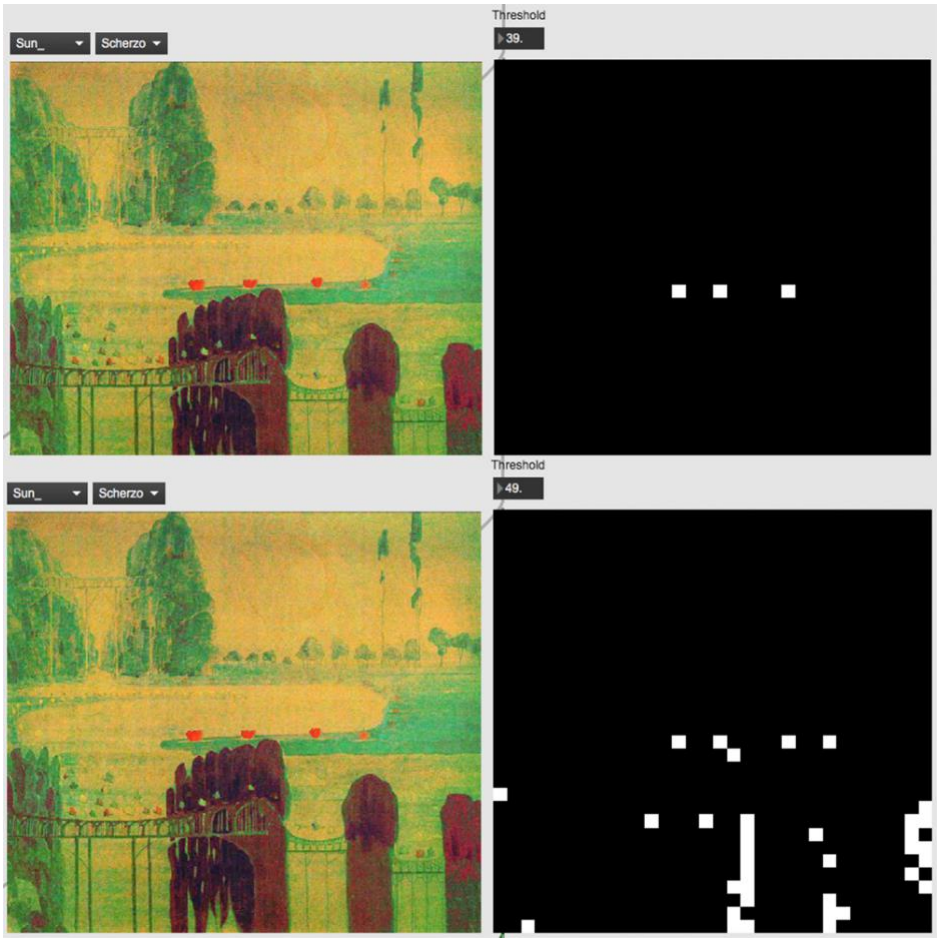
I subpatcheren "*p Imageisolator"* (Figur 4) isoleres den valgte farve fra billedet. Farve isoleringens sensitivitet i forhold til den valgte farve kan indstilles med "*Threshold"* parameteret.

Fra *"imageProcessing.maxpat"* udledes først hvor mange procent af den valgte farve der udgør det samlede antal pixels i billedet. En lav og høj opløsning af farvens placering i det valgte billede med den givne *"Threshold"*, renderes som en *jitter* matrix, sammen med en del andet data der kan udledes fra billedets aspekter. Den mest centrale af de ovenstående elementer der udledes fra *"imageProcessing.maxpat"* er den lavt opløste farveplacering i det valgte billede, da det er databehandlingen af denne hvorfra billedernes notationen konstrueres.

Frederik la Cour [201707873]
Digital Lyd
04-01-2021
Antal tegn: 30.416

Image-to-MIDI

Farvens position i billedet angives i en kontrasterende sort/hvid og nedskaleret version af det pågældende billede. Dette sendes ud af *"imageProcessing.maxpat"* som "*Low-Res ImageData*". Det hvide indhold i denne rendering vil repræsentere det pågældende område hvori den valgte farve rød fra *suckah* objektet er placeret. På (Figur 5) er demonstreret hvordan en højere "*Threshold"* værdi vil resultere i en større tolerance og dermed flere hvide elementer i de sort/hvide renderinger af den røde farve.

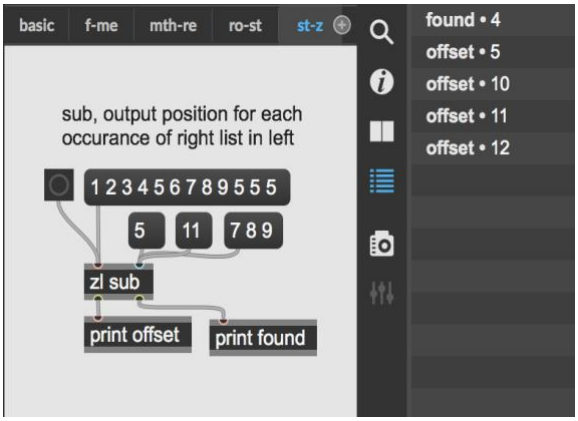


Figur 5 Threshold værdi demonstration

Herfra overgår billeddatabehandlingen fra *"imageProcessing.maxpat" til "Mainpatcher.maxpat"* hvori *jitter* matrixen omformes til en array gennem *jit.iter* objektet. En hvid firkant i en *jitter* matrix er betegnet som *255, 255, 255, 255* i en RGBA-skala, denne må skaleres til 127 i midi skalaen. Først afgøres placeringen af de tre hvide firkanter fra (Figur 5) med en "*Threshold"* værdi på 39.

Frederik la Cour [201707873]
Digital Lyd
04-01-2021
Antal tegn: 30.416

Tonehøjden der tildeles hver af de hvide firkanter afgøres ikke af den valgte farve, idet Čiurlionis synæstetik ikke arbejder med en direkte korrespondance mellem pitch og farve som i Scriabins synæstetik. De hvide firkanters pitch genereres i stedet som funktion af de hvide firkanters placering i sekvensen, gennem *zl sub* objektet.



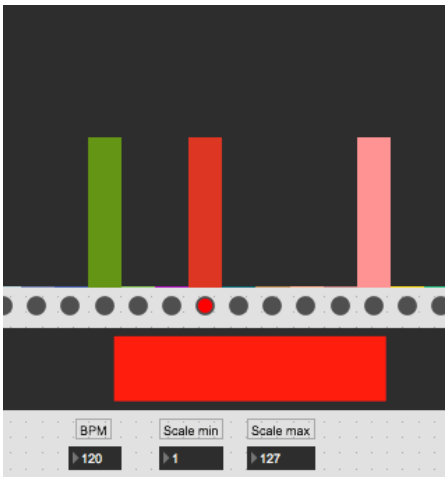
Figur 6 zl sub objektet

I (Figur 6) demonstreres *zl subs* funktionalitet, når man ønsker at finde ud af hvor mange 5-taller der findes i det array man benytter i venstre input. Det højre output angiver hvor mange 5-taller der er fundet i arrayet, mens det venstre output angiver deres indeksering i arrayet.

Generelt så er tonehøjden determineret af indekseringsværdien af hver hvide firkant fra jitter matrixen. Dog er der yderligere faktorer der også er med til at definerer tonehøjde og den sekventielle placering, særligt i forhold til max/MSP iboende tidslighed i forhold til kodens eksekvering og hvilke billeddata der måtte være behandlet forud for en ny billeddata behandling. Dette har jeg formodninger om, delvist skyldes hvordan de mange forskellige *zl* objekter interagerer med hinanden i systemet.

Figur 7 MatrixMIDI.maxpat – multislider sequencer med led step-indikatorer

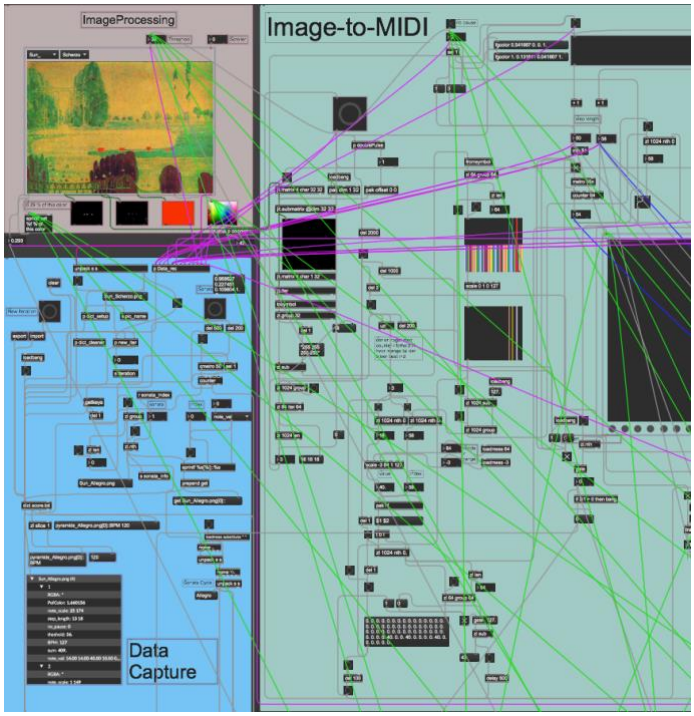
Hvis man i *"Mainpatcher.maxpat",* efter at have indstillet farve og "*Threshold"* værdien, trykker på "Render" knappen i øverste venstre hjørne, da vil billedets data renderes som midi notation i en Bpatcher *"MatrixMIDI.maxpat"*. Denne indeholder en multislider med 64 sliders samt en led step-indikatorer (Figur 7). Hvis man da igen trykker "Render", da kan placering af notationen blive forskudt og/eller der kan forekomme yderligere tilføjelser af toner eller større mellemrum mellem tonerne. Man kunne muligvis undgå disse ændringer, men det er netop disse iboende uforudsigelige forandringer i hvordan billedet fortolkes til midi data der gør notationen mere flydende og mindre steril.



Figur 8 efterbehandling af Midi data; Tempo, sekvens længde slider, scalering af midi data

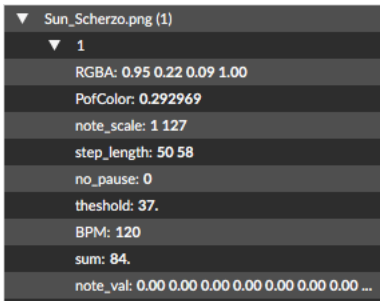
Man kunne anskue disse forandringer gennem Brian Eno's begreb om variationer; 'The *variety* of a system is the total range of its outputs, its total range of behavior.'[4, p. 227]. Eno beskriver yderligere hvordan systemers evolutionære tilpasnings adfærd er et resultat af interaktionen mellem sandsynlighedsprocessen og kravende fra det omgivende miljø hvor i sandsynlighedsprocessen foregår [4]. Det der afgør billedets midi data egenskaber kan siges at have en evolutionær tilpasningsevne i de forskydninger, tilføjelser og varierende tonehøjde værdier der sker hver gang midi data renderes fra omgivende miljø der i min produktion vil være tilsvarende til Čiurlionis billedsonater. Efter Midi-data er renderet i *"MatrixMIDI.maxpat"* kan denne data efterbehandles yderligere med bl.a. ændringer i tempo, skalering af data. Der er også mulighed for at bruger med den røde slider kan tilpasse sekvensens start og stop punkt som det ses i (Figur 8).

Data Capture



Figur 7 "Mainpatcher.maxpat" voilette input i Data Capture(blue) fra ImageProcessing & Image-to-MIDI

De mange forskellige outputs fra hhv. "*imageProcessing.maxpat"* og Image-to-MIDI efterbehandlingen kan gemmes således at bruger videre kan anvende dem i den algoritmiske komposition. I de gemte data kan specifikke karakteristika for hvert billede udledes. Disse data kan potentielt anvendes som kontrol data eller være med til at determinerer opførslen af udvalgte sektioner. Denne funktionalitet muliggøres gennem *dict* objektet.

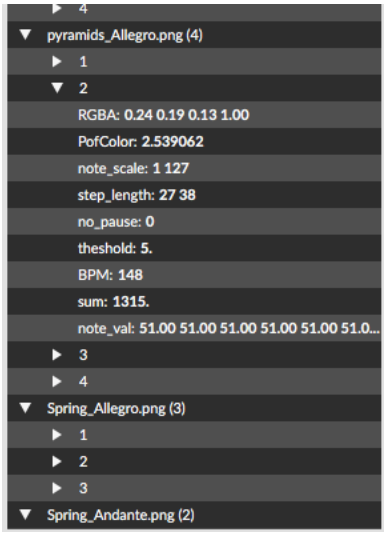


Figur 8 Data capture fra Sun_scherzo.png(1) der gemmes i et dict objekt

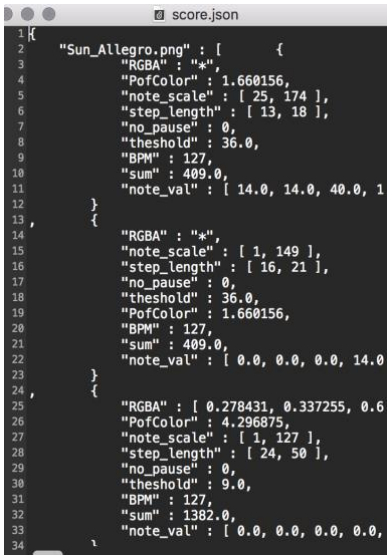
Dict objektet kan anvendes til at genererer og læse "*.yml*" eller "*.json*" filer, og altså gemme arrays som hierarkisk organiserede data strukturer. I det blå område i (Figur 9) ses alle de max objekter der har med indsamlingen af data at gøre. De violette kabler er alle outputs fra hhv. *ImageProcessing.maxpat* og *image-to-MIDI*. Alle disse værdier gemmes i et hierarkisk array i et *dict* objekt med navnet "*score"*. Så snart et af Čiurlionis billeder vælges vil der i *dict* "*score"* blive genereret en hierarkisk liste med først navnet på billedet f.eks. *Sun_Scherzo.png(1)* dernæst en liste med et tal. Under dette tal vil der først være værdier fra RGBA og hvor mange procent den valgte farve udgør

Frederik la Cour [201707873]
Digital Lyd
04-01-2021
Antal tegn: 30.416

af det samlede billede. Efter billedets midi data renderes vil de resterende værdier blive gemt som det ses i (Figur 10). Det tal som værdierne gemmes under, repræsenterer den pågældende iteration af den indsamlede data. Man kan med andre ord lave flere sekvensvariationer ud fra det samme billede. Herefter kan man interagerer med en knap der hedder *New Iteration*, der vil tilføje et nyt nummer under billedets navn, der ligeledes vil blive opdateret så det i vores eksempel vil hedde *Sun_Scherzo.png(2)*.



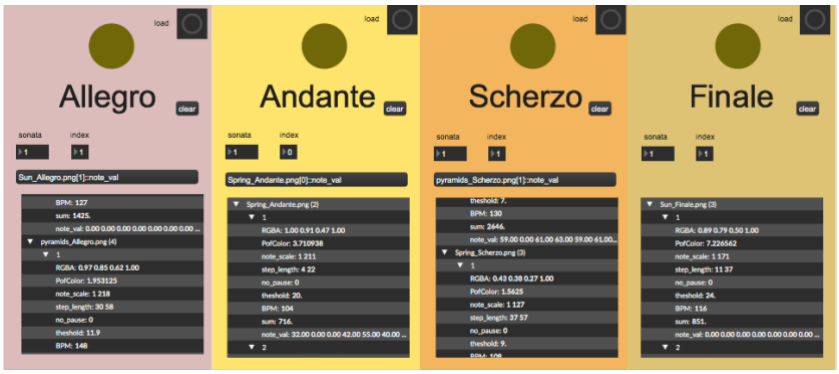
Figur 11 Dict objektet anvendes til at gemme værdier for flere billeder samt deres variationer



Figur 12 Score.json

Således kan en stor mængde forskellige billeder gemmes med en stor variation af samme billeddata behandling med forskelligt indstillede parametre som det ses i (Figur 11). I patchen er der inkluderet en ”*score.json*” hvori værdier fra *dict* objektet er uploadet som det ses i (Figur 12).

Den algoritmiske sonate

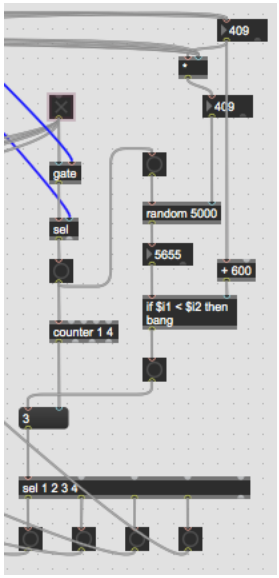


Figur 13 Dict objekterne Allegro, Andante, Scherzo og Finale

Når alle værdierne er blevet indsamlet og indsat i det hierarkiske system, da opdeles de i fire *sonate kanaler*, *Allegro, Andante, Scherzo* og *Finale* som det ses i (Figur 13). I hver af disse kanaler er det muligt at udvælge specifikke billedsonater samt deres specifikke variationer af

de indsamlede værdier. Når en given sonate og variation er valgt, kan denne genloades i det overordnede system og således igen afspilles som før ved at interagerer med *load* knappen i øverste højre hjørne af hver kanal.

Interaktionen med *load* knappen kan enden forekomme ved at en bruger direkte interagerer med knappen for at udvælge en specifik variation. Hvis brugeren af patchen aktiverer knappen ”*turn on sonata cycle”* da vil de gemte databehandlinger blive aktiveret gennem et algoritmisk tilfældighedssystem, der i et hvis omfang benytter sig af de databehandlede værdier fra de aktive sonatekanaler. Systemet hvori de algoritmiske tilfældighedsudregninger foregår er illustreret i (Figur 14).



Figur 14 Central del af Mainpatcher.maxpat hvori det afgøres og udføres hvilken sonate og variation der afspilles og hvor lang tid den fremføres i

Når brugeren interagerer med knappen ”*turn on sonata cycle”* åbnes et *gate* objekt der tillader input fra hvilket step der er aktiveret i den 64 step lange sekvens at passerer igennem. Et *counter* objekt kører i en cyklus gennem tallene fra et til fire der sendes til et max/MSP *messagebox* objekt. Denne *messagebox* aktiveres hver gang den modtager et *bang*, hvilket yderligere er afhængig af et max/MSP *random* objekt, hvis parameter dynamisk ændres afhængig af summen de af 64 slideres iboende værdier ved den aktive sonate multipliceret med procenten af den pågældende sonates farve ud af hele billedet. Et betinget erklærings logik afgør, hvorvidt den tilfældige værdi er mindre end summen af de 64 sliders værdi + 600. Hvis den tilfældige værdi er mindre, vil det resultere i at *messageboxen* aktiveres og udsender det pågældende nummer til endnu et *sel* objekt der så kan aktivere en af de fire sonatekanaler ad gangen.

Nu hvor den komplekse funktionalitet er blevet redegjort fra billedsonater til algoritmisk komposition er der en række spørgsmål i forhold til hvilke betydninger og karakteriseringer dette system kan siges at arbejde med i dets fremførsel og udtryk. Den følgende sektion i denne opgave vil således tilgå den redegjorte funktionalitet fra en mere diskuterende teoretisk vinkel.

Konstitueringer af oversat billeddata

Ideen med at skabe en algoritmisk sonate medfører naturligvis en række spørgsmål i forhold til hvordan en ekstremt hierarkiske kompositions form som sonaten bør sammenkobles med de algoritmiske og generative aspekter vi finder i computermusikkens kompositioner. Den iboende lineære progression, der gennemsyrer sonate-formen som vi kender den fra klassisk musik, med først *allegro* så *andanten* førend vi går til *scherzo* og endelig afslutter med *finale*. Hvordan kan denne forenes med den non-lineære tilgang der kommer ud af sandsynligheds og tilfældigheds baserede systemer, der ligger dybt rodfæstet i computerens materialitet?

Det var vigtigt at den notation der blev genereret fra billederne, skulle besidde genkendelige karaktertræk fra dets oprindelige billede-data. Altså sagt med andre ord var det

vigtigt at den notations data der blev genreret ud fra Čiurlionis billedsonater var tilstrækkelig karakteristisk til at man som bruger ville kunne genkende den. Gennem databehandlingen, sikredes en så direkte oversættelse af billedets iboende strukturer som muligt. Denne koblet med yderligere udtrækninger af data fra billedets øvrige databehandling er med til at konstituere billedets øjeblikkelige identitet og tilstedeværelse i oversættelsen af billedet til notation.

Det er i konstitueringen af brugerens billeddata-analyse at der skabes en relation med øvrige konstitueringer af oversat billeddata og skaber en diversitet mellem billede-sonaterne. Således skabes også en musisk brugbarhed iboende i det indsamlede data [6]. Denne musiske brugbarhed er det fundamentale element i opbyggelsen af den overordnede kompositions struktur som de oversatte billede-data kan siges at være en del af.

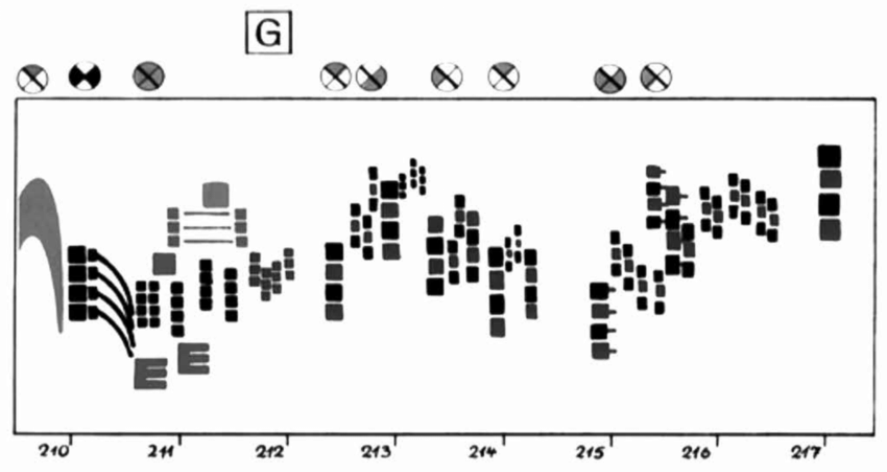
Den algoritmiske sonatens serialisme

Ifølge Stockhausen så er strukturen af en given komposition og strukturen af det materiale der anvendes nødt til at komme fra den samme originale musiske idé, og således kan det siges at strukturen af materialet og strukturen af kompositionen skal være den samme [7]. Hvor det i Stockhausens kapitel omhandler komposition med lyden i sig selv som materialet, ser jeg ingen grund til at det skulle forholde sig anderledes i min komposition, selvom det genererede, ikke direkte er tilskrevet lyd, men har karakter af klassisk notation. Notationen i min produktion er påkrævet en fortolkning og kan opføres af forskellige lydgenererende enheder og ikke kun manifesteres som optagede lydfiler.

Iboende i disse hierarkisk organiserede arrays ligger da også en generel musisk idé, der beror sig på deterministisme og serialisme tilgange til komposition. Om serialismen skriver Breinbjerg at den karakterises ved at den; 'udstiller en analytisk tankegang, der ikke blot opløser de musikalske fænomener i nye målbarer dimensioner, men [også] (…) omsætter musikalske fænomener til tal.'[8, p. 16] I tilfældet med min produktion er det musikalske fænomen sammenkoblet med databehandlingen af Čiurlionis billedsonater. De hierarkiske arrays vi finder i *dict* objektet udstiller da også en analytisk tankegang, der beror sig i at anskue billeder i kraft af de algoritmer der analyserer deres egenskaber, fremfor hvilke symbolske betydninger Čiurlionis måtte have tillagt dem. Endvidere kan serialismen ifølge Breinbjerg siges at åbne op for matematisk funderede manipulationer af de genererede tal [8]. I min egen produktion finder en sådan matematisk manipulation sted i (Figur 14), hvorfra rækkefølgen og længden afgøres af værdier fundet i de aktive *dict* objekters omsatte billeddata og dets variationer.

Dict objektets materialitet er derudover i kraft af sin tillagte funktionalitet også i stand til at gemme dets iboende data som en "*.json*" fil. I min egen produktion medfølger da også en "*score.json*" fil (Figur 12) hvori et eksempel på en dataindsamling af de tre billedsonater er indkodet. "*Score.json*" fungerer som en programmatisk notation der ville være tilsvarende til grafisk notation (Figur 15) som vi kender det fra f.eks. György Ligeti. "*Score.json*" har et iboende behov for at blive fortolket i dets opførsel af computeren, på samme måde som Ligetis grafiske notation har et behov for at blive fortolket af en performer. Forskellen ligger i

graden af åbenhed. De arbitrerer symboler og former synes nemmere at fortolke for mennesker end ”*score.json*” der er langt mindre tilgængelig i forhold til fortolkning. ”*score.json*” henvender sig i højere grad til en maskinel fortolkning system fremfor en menneskelig fortolkning.



Figur 15 et uddrag fra György Ligeti - Artikulation (1958)

Den algoritmiske sonate som et åbent værk?

Det kan diskuteres hvor tilgængelig ”*score.json*” er at fortolke, og det er netop spørgsmålet om tilgængelighed eller åbenhed i kompositionen der er genstand for refleksion i dette afsnit. kan den algoritmiske sonate karakteriseres som et ’*open work*’[5, p. 3]? Umberto Eco definerer begrebet som et værk der ikke kan siges at være færdiggjort, eller have forskrevne repetitioner [5]. Endvidere er det essentielt at værket kun kan bringes mod en konklusion såfremt en performer interagerer med kompositionens bestanddele [5]. Yderligere karakteriserer Eco også den traditionelle tilgang til komposition i klassisk musik; ’(…) an assemblage of sound units which the composer arranged in a closed, well-defined manner before presenting it to the listener’[5, p. 2f]. Brugerens interaktion med kompositionen fordrer en vekslen mellem systemet som komponist og brugeren som komponist.

Ifølge Eco så er modtagelsen af en komposition både en fortolkning og en performance, idet der i enhver modtagelse anlægges et nyt perspektiv fra beskuer [5]. I forhold til kompositionen som en performance, så kan det diskuteres hvori rollen som performer findes. Systemet kan ligesom brugeren også aktiverer notationen. Dog kan der argumenteres for at der forud for computerens performative egenskaber nødvendigvis må forekomme en menneskelig interaktion der også udgør en form for performance. Hvad end denne består i selv at udfører en dataindsamling af data fra Čiurlionis billedsonater eller loade ”*score.json*” ind i systemet. I forhold til kompositionen som en fortolkning, så kan der forekomme en menneskelig fortolkning af den endelige opførsel af den genererede notation og dennes relation til Čiurlionis billedsonater. Computerens fortolkning er her en reducering af Čiurlionis billedsonater til de genererede indsamlede data der gemmes og kaldes fra *dict* objekterne. Heri ligger fortolkningen fra RGBA-værdier til 64 step midi sekvenser, samt de

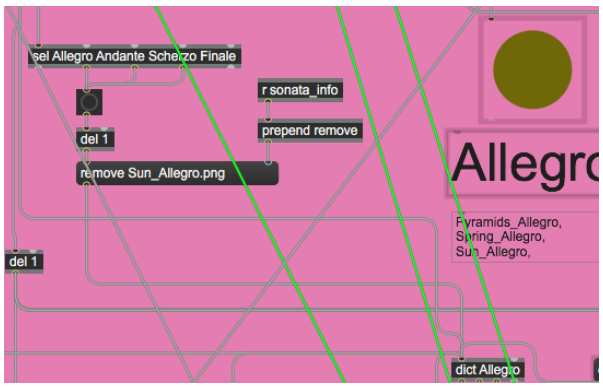
øvrige arbitrerer data der er med til at karakteriserer og differentierer de forskellige notationer og dets variationer fra hinanden.

Det er denne grad af handlekraft, der tildeles beskueren i samspil med systemet, der er nødvendig for at kunne opleve den ellers ufærdige komposition. Selvom ”*score.json*” findes som en fil tilhørende produktionen, så kan beskueren i sin interaktion med systemet vælge selv at foretage en data indsamling og gemme den som sin egen ”*.json*” fil. Således kan ”*score.json*” siges blot at være en potentiel variation af sammenfletningen af billedsonaterne. Brugeren har muligheden for at konstruerer flere ”*.json*” filer man i opførslen kunne skifte imellem for at tilføje yderligere variation i den endelige komposition.

Den algoritmiske sonate kan altså tilskrives at være et ’*open work*’[5, p. 3] for så vidt at det kræves af brugeren at interagerer med systemet. I denne interaktion indtager brugeren flere roller i form af komponist og performer for at opleve kompositionens opførsel og forholde sig æstetisk til dette. Dog er det vigtigt at understrege at kompositionen i kraft af dens iboende serialisme og determinisme, da indeholder magtstrukturer i de styrende og bestemmende parametre kompositionens materiale består af.

Den algoritmiske sonates magtstrukturer

Som Eco argumenterer for i hans kapitel, så er et ’*open work*’[5, p. 3] ikke ensbetydende med, at den handlekraft der tildeles beskueren er lig ubetinget frihed. Derimod er den deliberativt begrænset af iboende faktorer i det ”åbne værk” fra skaberen af systemet selv, med henblik på at styrer brugerens fokus imod bestemte dele i værket [5]. Magtstrukturerne der er styrende for brugeren i min produktion kan siges at ligge et sted mellem de udvalgte billedsonater, deres data, de algoritmer der fortolker og komponere denne data og ikke mindst de udvalgte parametre i systemet bruger er i stand til at interagerer med.



Figur 16 udsnit af Sonata Splitter funktionen ved Allegro

Der er ikke umiddelbart nogen måde hvorpå andre billeder kan analyseres i min produktion og således er Čiurlionis billedsonater et centralt element hvorfra produktionen er konstrueret. De billeder systemet behandler, må hører indenfor en form for sonate struktur i deres navngivningskonventioner, da denne ellers ikke ville kunne indgå i systemet vi ser i (Figur 16). Dette skyldes den type algoritme vi ser i (Figur 16), hvori *dict* ”*Allegro*” får frasorteret alle arrays der ikke har med ”*allegro”* at gøre. Med andre ord vil et billede der ikke tilhørte sonateformen altså blive frasorteres med ”*sonate splitter*” funktionaliteten. Hvis ikke det indsamlede data kan sorteres ud i disse fire *dict* objekter ”*allegro, andante, scherzo eller finale”* da kan den data ikke automatiseres eller fremføres gennem tilfældighedssystemet i (Figur 14).

Ligeledes er det også kun muligt at fortolke billede-data fra venstre mod højre, idet algoritmerne ikke læser lodret ned eller opadgående. Når systemet i (Figur 14). aktiveres kan

de forskellige variationer af sonatebestanddelene heller ikke afbrydes i deres opførsel, den pågældende sekvens vil altid færdiggøres førend den fortsætter til den næste sekvens, medmindre brugeren intervenerer. Selvom bruger i rollen som performer har mulighed for at ændre i tempo, step-længde, skalering af midi data i realtid så vil brugerens performative ændringer ikke gemmes i sekvensen undervejs. Dvs. at den indsamlede billeddata oversat til notation er fastfrosset i tid, idet man som bruger ikke har mulighed for at gemme de performative ændringer man foretager i opførslen af den indsamlede data.

Lyden af den algoritmiske sonate

Hovedfokusset i produktionen har været at udvikle et system der var i stand til at genererer midi notations data fra Čiurlionis billedsonater. Den genererede data kan opføres af en indbygget sampler, synthesizer, eksternt system forbundet med midi eller en kombination af disse. Afhængig af hvilken af disse lydgenererende enheder der opfører notationen kan disse enheder blive påvirket i større eller mindre grad af den indsamlede data. Med andre ord er den algoritmiske komposition som sådan ikke afhængig af et bestemt lydligt udtryk men kan anvendes og fortolkes frit af brugeren.

Dette har jeg yderligere forsøgt at demonstrerer gennem den optagelse på 3 minutter der medfølger opgaven, hvoraf hvert minut er notationen spillet gennem først den indbyggede FM-synthesizer med filter, herefter sampleren fra en tidligere produktion ”*morphologocial manufacturing”*, og endeligt en ekstern semi-modular hardware synthesizer. For at beskrive hvordan den algoritmiske sonate kan lyde, tager jeg i dette afsnit udgangspunkt i anvendelsen af den indbyggede digitale synthesizer med filter og envelope.

Frekvensen bestemmes her af midi tonerne fra de generede sekvenser der omsættes gennem *mtof* objektet der omsætter midi til frekvens data. Hvorvidt FM er aktiveret, afgøres af logik der tjekker hvorvidt procent af den givne farve er over eller under 2. Hvis denne er over 2 da aktiveres FM funktionaliteten i synthesizeren, hvoraf modulationsfrekvensen styres af en skalering af den pågældende step-længde. Cutoff frekvensen i det tilhørende filter bestemmes af gennemsnittet af RGBA-værdierne. Hvert billede kan således sige at have et individualiseret udtryk der er med til at gøre dem forskellige fra hinanden.

Konklusion

I opgaven har jeg arbejdet med et udvalg af den Litauiske komponist M.K. Čiurlionis billedsonater gennem programmeringssproget max/MSP. Udgangspunktet i opgaven var at konstruere en algoritmisk komposition med afsæt i fornævnte udvalg af Čiurlionis billedsonater. I den algoritmiske komposition er der arbejdet særligt med hvordan man udtrække billeddata, hvordan billeddata kan omsættes til sekventielt midi data, hvorvidt den omsatte data kan siges genkendelig fra dennes oprindelses sted, hvordan man kan indsamle og gemme komplekse hierarkiske arrays i ”*.json*-filer” og hvordan tilfældigheds og sandsynligheds matematiske del elementer i systemet kan anvendes i struktureringen og manipuleringen af forskellige lydgenererende enheder.

Derfra det blevet diskuteret hvordan et hierarkisk kompositionssystem som sonaten kan forenes med den algoritmiske kompositions iboende organiske og tilfældige udtryksformer? Hvordan brugerens rolle i interaktion med systemet hele tiden skifter mellem komponist, performer og publikum. Diskussionen af *dict* objektet rolle i forhold til den algoritmiske kompositions notation. Dertil diskussionen af "*score.json's"* fortolkningstilgængelighed der udmunder i afsnittet der diskuterer hvorvidt den algoritmiske sonate kan siges at være et 'open work'[5, p. 3]. Hvorefter de iboende magtstrukturer der ligger i den algoritmiske sonate også bliver behandlet. For at afslutte den diskuterende del af opgaven med hvordan man kunne anvende den omsatte billeddata notation til at styrer udvalgte parametre i en digital synthesizer med filter, med henblik på dermed også at argumenterer for at lyden af notationen er arbitrær idet den ikke er bundet til fremførslen af et bestemt instrument, men kan fortolkes og opføres på mange forskellige måder, hvilket er op til brugeren af systemet at afgøre.

Med andre ord så indeholder min produktion mange forskelligartede spørgsmål. Hvis jeg skulle arbejde videre med disse, ville jeg nok særligt vælge at arbejde videre med hvordan brugerens rolle skifter i sin interaktion med det komplekse system, hvilke faktorer er det der medvirker til sådanne skift? Har vi som instrument konstruktører nogen muligheder for at præge vores interaktive artefakter i en retning hvor vi kan kontrollere disse rolleskift? Herunder kunne det være spændende at reflekterer over forskellen på den algoritmiske sonate komposition og det komplekse digitale bruger-fjendtlige instrument. Kan der overhovedet siges at være nogen forskel?

Litteraturliste

[1] R. Andriušytė-Žukienė, 'Painting', *Painting*. http://ciurlionis.eu/en/painting/.
[2] J. W. Bernard, 'Messiaen's Synaesthesia: The Correspondence between Color and Sound Structure in His Music', *Music Percept.*, vol. 4, no. 1, pp. 41–68, Oct. 1986, doi: 10.2307/40285351.
[3] 'Scriabin's Color Symbolism in Music', *Interlude*, Feb. 05, 2016. https://interlude.hk/scriabins-color-symbolism-music/ (accessed Dec. 27, 2020).
[4] B. Eno, 'Generating and Organizing Variety in the Arts', p. 4.
[5] U. Eco, 'The Poetics of the Open Work', in *The Open Work*, Cambridge, MA: Harvard University Press, 1989.
[6] M. V. Mathews, 'The Digital Computer as a Musical Instrument', *Sci. New Ser.*, vol. 142, no. 3592, pp. 553–557, 1963.
[7] K. Stockhausen, 'Electronic and Instrumental Music', p. 5.
[8] M. Breinbjerg, *Musikkens interfaces*. Århus: Center for Digital æstetik-forskning, 2006.