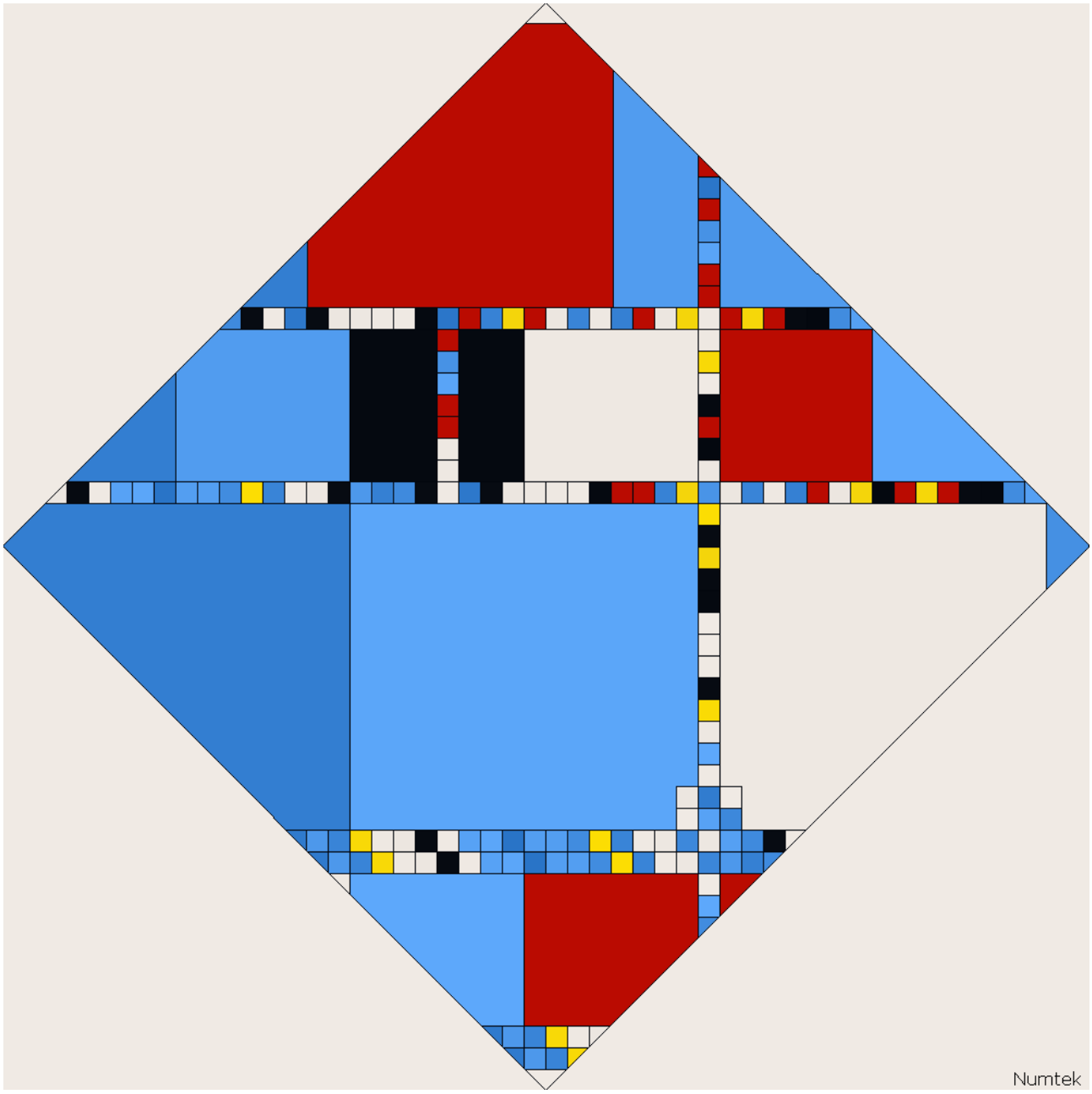


De afbeelding hierboven is de laatste versie en tevens mijn inzending voor de wedstrijd.
Een losse png is hier te vinden :
http://richtlijn.be/~numtek/mondriaan_numtek.png
Het is geschreven in 100% native Python 2.7.3 (build Apr 10 2012, 23:31:26) [MSC v.1500 32 bit (Intel)] on win32) en het script is hieronder te lezen.
Een losse versie van 't script is hier te vinden :
http://richtlijn.be/~numtek/mondriaan_numtek.py
Mijn uitwerking is gebaseerd op het omgevinggeluid van de originele versie. De input is mondriaan_numtek.wav, een losse versie hiervan is te vinden op
http://richtlijn.be/~numtek/mondriaan_numtek.wav
De volledige, 28 minuten durende opname is hier te vinden:
http://richtlijn.be/~numtek/mondriaan_numtek.mp3

Het concept is ontstaan toen ik nadacht over hoe Mondriaan kwam tot zijn originele compositie. Na het luisteren naar Scott Joplin en tijdsgenoten bedacht ik dat het een mooi idee zou zijn om muziek te gebruiken als input, de vormen en kleuren zijn immers duidelijk geïnspireerd door de muziek van zijn tijd. Hoe verleidelijk dit ook was, ik vond het nog steeds een vrij saaie aanpak. Al sinds jaar en dag maak ik visuals die gestuurd worden door muziek en hoe leuk dit soort maakprocessen ook zijn, er is weinig vernieuwends aan.

Tot ik ineens bedacht dat de compositie, de uiteindelijke kleurkeuze zo opvallend zijn omdat ze vaan Mondriaans hand zijn. Het was dus zaak om terug te gaan naar de basis, terug naar het origineel. Gewapend met een recorder ben ik naar het Gemeentemuseum in Den Haag gegaan om aan het schilderij te vragen hoe het ontstaan is. Dit leverde een 28 minuten durende soundscape op bestaande uit flarden van de kassa, video-loops die spelen rondom de Victory Boogie Woogie en een klas schoolkinderen die zich afvraagt hoe veel geld de stukjes tape zouden kosten die er op geplakt zijn. Hiervan heb ik korte versie gemaakt van 4 minuten en deze is bepalend voor de kleur, vorm en compositie van mijn inzending.

Om zo veel mogelijk het origineel te evenaren heb ik de audio ook gekoppeld aan subtiele kleurvariaties per vlak. Omdat een scherm nu eenmaal niet de kracht heeft van verf op canvas heb ik gekozen om de vlakken te voorzien van een outline met zwarte pixels.

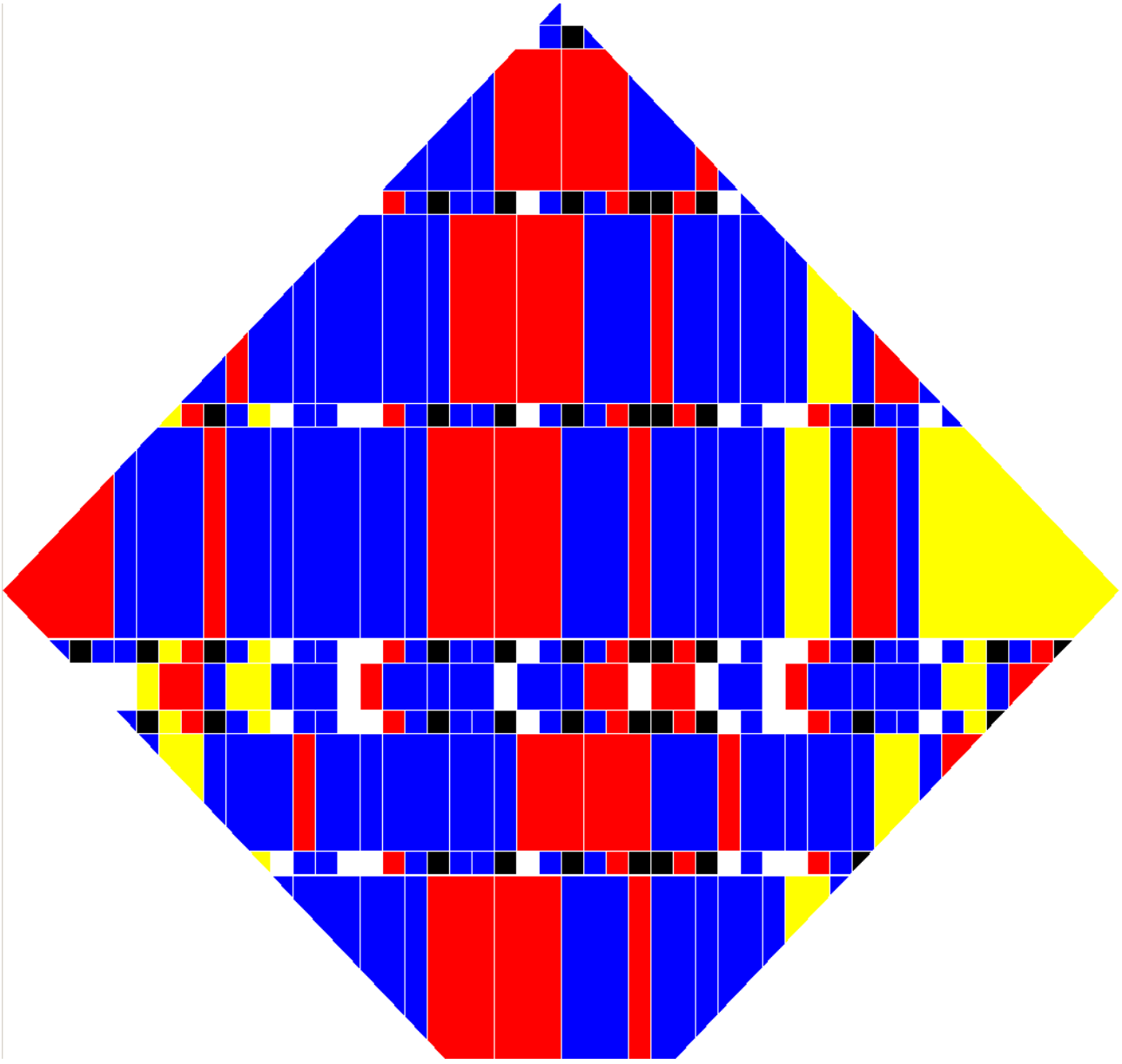
```
#!/usr/bin/python
import struct, random, wave,operator
from Tkinter import *
samplebar = []
samplepos=0
sample=0
x=0
x1=0
x2=0
x3=0
x4=0
x5=0
color1=""
color2=""
color3=""
size=0
temp_size=0
startx_vert=0
canvas_size=1000
lastpos=0
lastpos_blue=0
startx=0
starty=0
endx=0
endy=0
redbar = []
count=0
count2=0
outline_width=0
def define_color(r,g,b):
        rgb = r, g, b
        new_color = '#%02x%02x%02x' % rgb
        return(new_color)

def set_color(audio_sample): #kies de kleur op basis van de audio, subtiele variaties voor
verhoogd realisme
        r = 35 + audio_sample % 60
        g = 110 + audio_sample % 60
        b = 193 + audio_sample % 60
        blue = define_color(r,g,b)

        r = 252 - audio_sample % 10
        g = 221 - audio_sample % 10
        b = 3 + audio_sample % 10
```

```
        yellow = define_color(r,g,b)

        r = 3 + audio_sample % 5
        g = 7 + audio_sample % 5
        b = 14 + audio_sample % 5
        black = define_color(r,g,b) #volledig zwart bestaat niet in de echte wereld

        r = 240 - audio_sample % 5
        g = 234 - audio_sample % 5
        b = 228 - audio_sample % 5
        white = define_color(r,g,b)  #volledig wit bestaat niet in de echte wereld


        if audio_sample<(22000-4*(44000/6)):
                color1=black
        if audio_sample<(22000-4*(44000/6)):
                color1=black
        if (audio_sample>(22000-4*(44000/6))) and (audio_sample<(22000-3*(44000/6))):
                color1=white
        if (audio_sample>(22000-3*(44000/6))) and (audio_sample<(22000-2*(44000/6))):
                color1=blue
        if (audio_sample>(22000-2*(44000/6))) and (audio_sample<(22000-1*(44000/6))):
                color1=yellow
        if audio_sample>(22000-1*(44000/6)):
                color1=red
        return(color1)

def draw_vert(startx_vert,starty,endy):
        temp_size=0
        startx_vert=(int(canvas_size * startx_vert) / 20 ) * 20 #deelbaar door 20 anders valt
het niet goed op 't grid
        for i in range((int(canvas_size * starty) / 20 ) * 20+1,(int(canvas_size * endy) / 20 ) * 20
,20): #vert
                x=samplebar[i+startx_vert+i]
                color1=set_color(x)
                w.create_rectangle(startx_vert,i,startx_vert+20,i+20,fill=color1,outline=black)
#draw vert
        return
if __name__ == '__main__':

        #------------------------------
        #Init, keuren en canvas
        #------------------------------
        red = define_color(186,11,1)
```

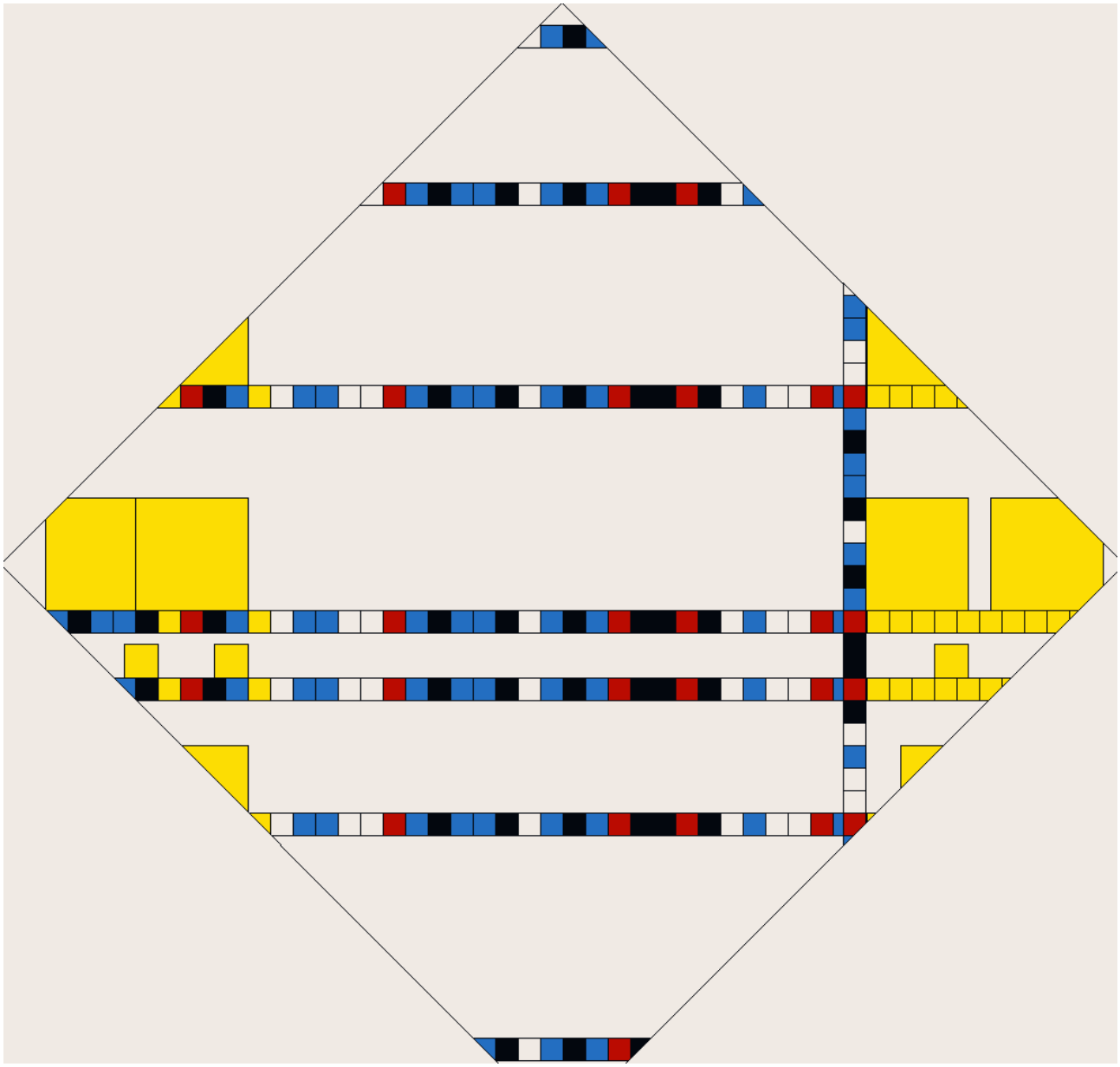
```
	blue = define_color(35,110,193)
	yellow = define_color(252,221,3)
	white = define_color(240,234,228) #volledig wit bestaat niet in de echte wereld
	black = define_color(3,7,14) #volledig zwart bestaat niet in de echte wereld
	waveFile = wave.open('mondriaan_numtek.wav', 'r')
	length = waveFile.getnframes()
	for i in range(1,canvas_size*10000,20): #inladen, 10000x canvas size om genoeg punten te nemen
		count=count+1
		waveData = waveFile.readframes(1)
		if count==canvas_size/10:
			data = struct.unpack("<h", waveData)
			data=data[0]*2
			samplebar.append(data)
			count=0
	master = Tk()
	w = Canvas(master, width=canvas_size, height=canvas_size)
	w.pack()
	w.create_rectangle(0,0,canvas_size*2,canvas_size*2,fill=white,outline=white) #canvas white
	#------------------------------
	#Teken kleine verticale regels
	#------------------------------
	startx_vert=0
	for i in range(1,canvas_size,20): #vert
		temp_size=temp_size+1
		x = samplebar[i]
		color1=set_color(x)
		if color1==red:#als een vertical blok rood is (max) teken dan een horizontale regel
			redbar.append(i)
			size=(temp_size-2)*20
			temp_size=0
			for i2 in range(0,canvas_size,20): #horizontaal klein

				#------------------------------
				#Teken horizontale regels
				#------------------------------

				x3=samplebar[i2-i] #-1 om per regel nieuwe wave-data op te halen
				color2=set_color(x3)
				startx=startx_vert+i2
				starty=i
```

```
                    endx=startx_vert+20+i2

                    endy=i+20


        w.create_rectangle(startx,starty,endx,endy,fill=color2,outline=black) #draw
horizontaal

                w.create_rectangle(startx_vert,i,startx_vert+20,i+20,fill=color1,outline=black)
#draw vert

        #grote vul blokken, kijk naar rood, teken daar, check size en vul de regel, door naar
de volgende rode
        size=(temp_size*20)-20 #20 per row minus de row zelf
        temp_size=0

        #------------------------------
        #Teken grote horizontale regels
        #------------------------------
        lastpos=0
        size=0
        count=0
        for i2 in redbar:
                count=count+1
                size=i2-lastpos
                for i3 in range(10):
                        x3=samplebar[count*20*i3] #-1 om per regel nieuwe wave-data op te
halen
                        color2=set_color(x3)
                        startx=i3*size
                        starty=lastpos+20
                        endy=lastpos+(i2-lastpos)
                        endx=startx+size
                        if endx > canvas_size: #wel binnen de lijntjes eindigen
                                endx=canvas_size-1
                        if startx > canvas_size: #wel binnen de lijntjes starten
                                startx=canvas_size-1

                        w.create_rectangle(startx,starty,endx,endy,fill=color2,outline=black)
#draw grote vul blokken
                lastpos=i2

        for i2 in range(1,canvas_size,20): #
                x3=samplebar[i2-i] #-1 om per regel nieuwe wave-data op te halen
```

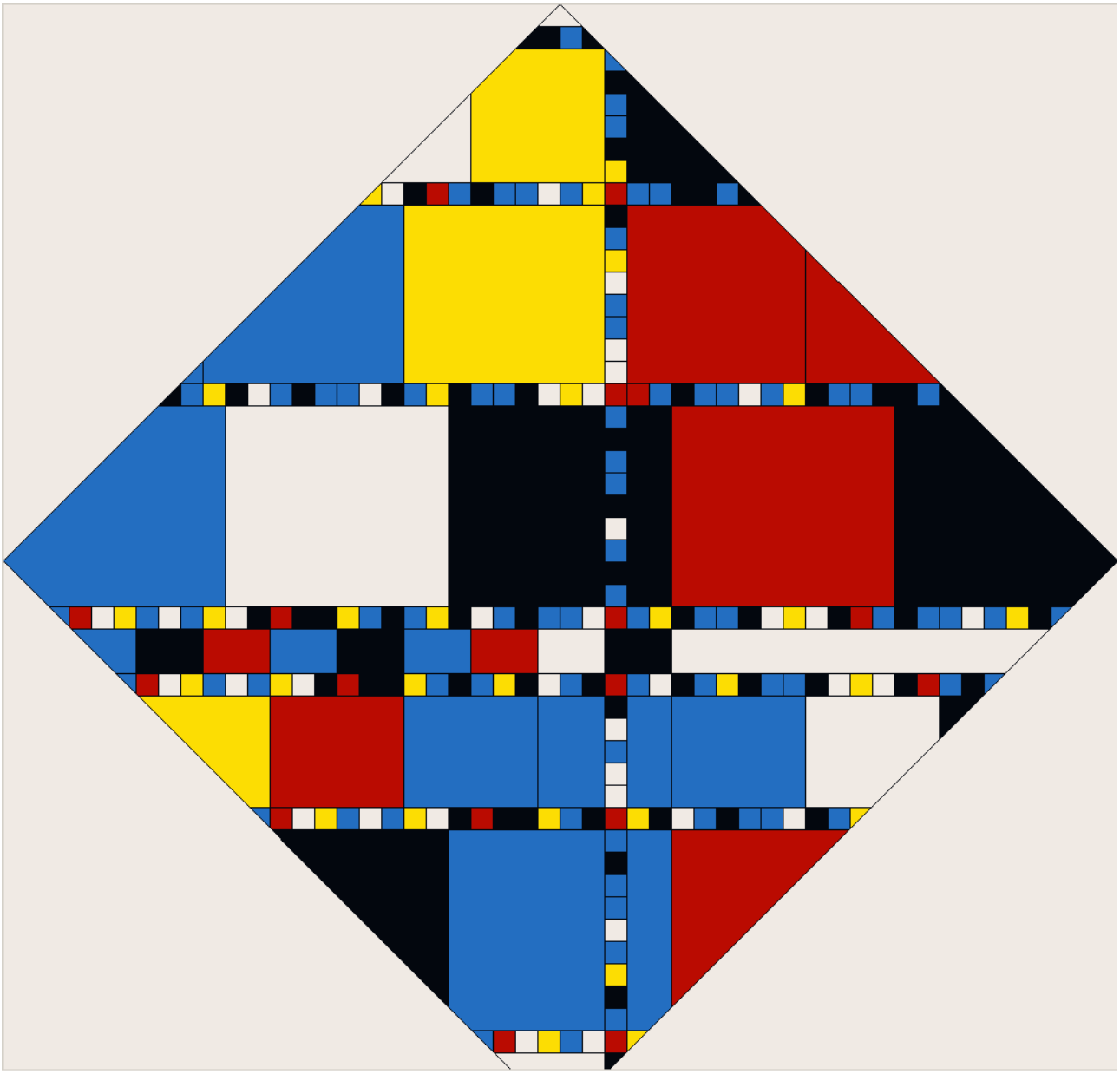
```
                color2=set_color(x3)

        #------------------------------
        #Teken losse verticale regels
        #------------------------------
        draw_vert(0.65,0,1)
        #draw_vert(0.61,0.68,0.78)
        draw_vert(0.62,0.72,0.78)
        draw_vert(0.67,0.72,0.78)
        #draw_vert(0.68,0.68,0.78)
        draw_vert(0.4,0.28,0.45)
        #------------------------------
        #Teken witte border
        #------------------------------
        w.create_polygon([(1, 1), (1, canvas_size/2), (canvas_size/2,
1)],fill=white,outline=black) #make border
        w.create_polygon([(1, canvas_size), (1, canvas_size/2), (canvas_size/2,
canvas_size)],fill=white,outline=black) #make border
        w.create_polygon([(canvas_size, 1), (canvas_size, canvas_size/2), (canvas_size/2,
1)],fill=white,outline=black) #make border
        w.create_polygon([(canvas_size, canvas_size), (canvas_size, canvas_size/2),
(canvas_size/2, canvas_size)],fill=white,outline=black) #make border
        x=canvas_size-70
        y=canvas_size-10
        w.create_text(x,y, anchor=W, font="Verdana",text="Numtek")
        mainloop()
```



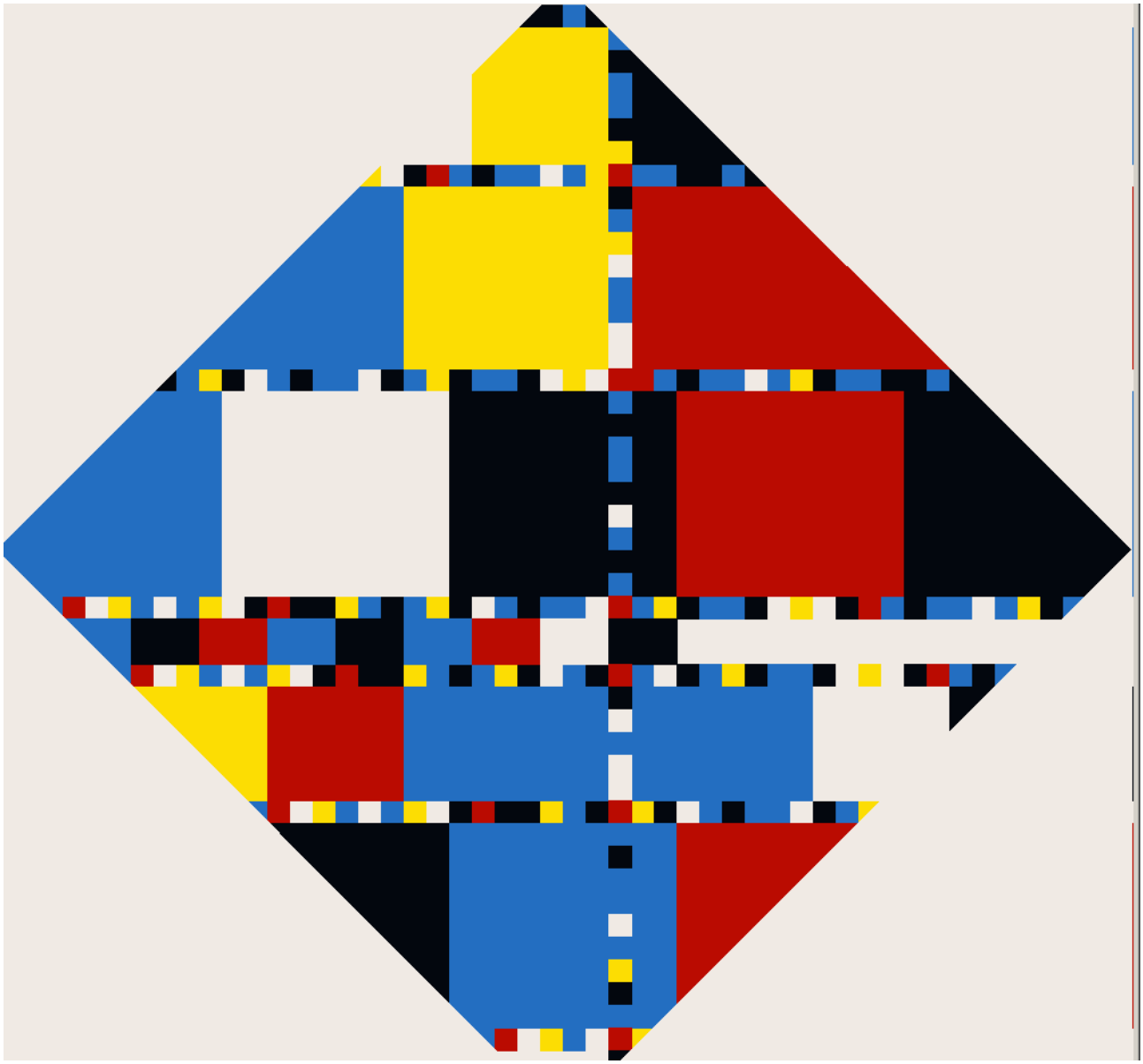
Dit is de eerste versie, nog aangestuurd door een oude opname die ik maakte met een walkman met cassetetapes. Aangezien ik van huis uit geen programmeur ben was dit de versie die mij genoeg zelfvertrouwen gaaf om door te gaan met de wedstrijd. Immers, als het me niet zou lukken om binnen afzienbare tijd op de proppen te komen met iets dat enigszins overeenkomsten vertoont met het origineel, dan was deze uitdaging misschien niet voor mij weggelegd. Normaal gesproken laat ik namelijk het ontwikkelen van software graag over aan mensen die hier voor gestudeerd hebben omdat dit nu eenmaal gewoon wat sneller gaat. Toen ik dit eenmaal zag om mijn scherm had ik genoeg motivatie om verder te gaan.



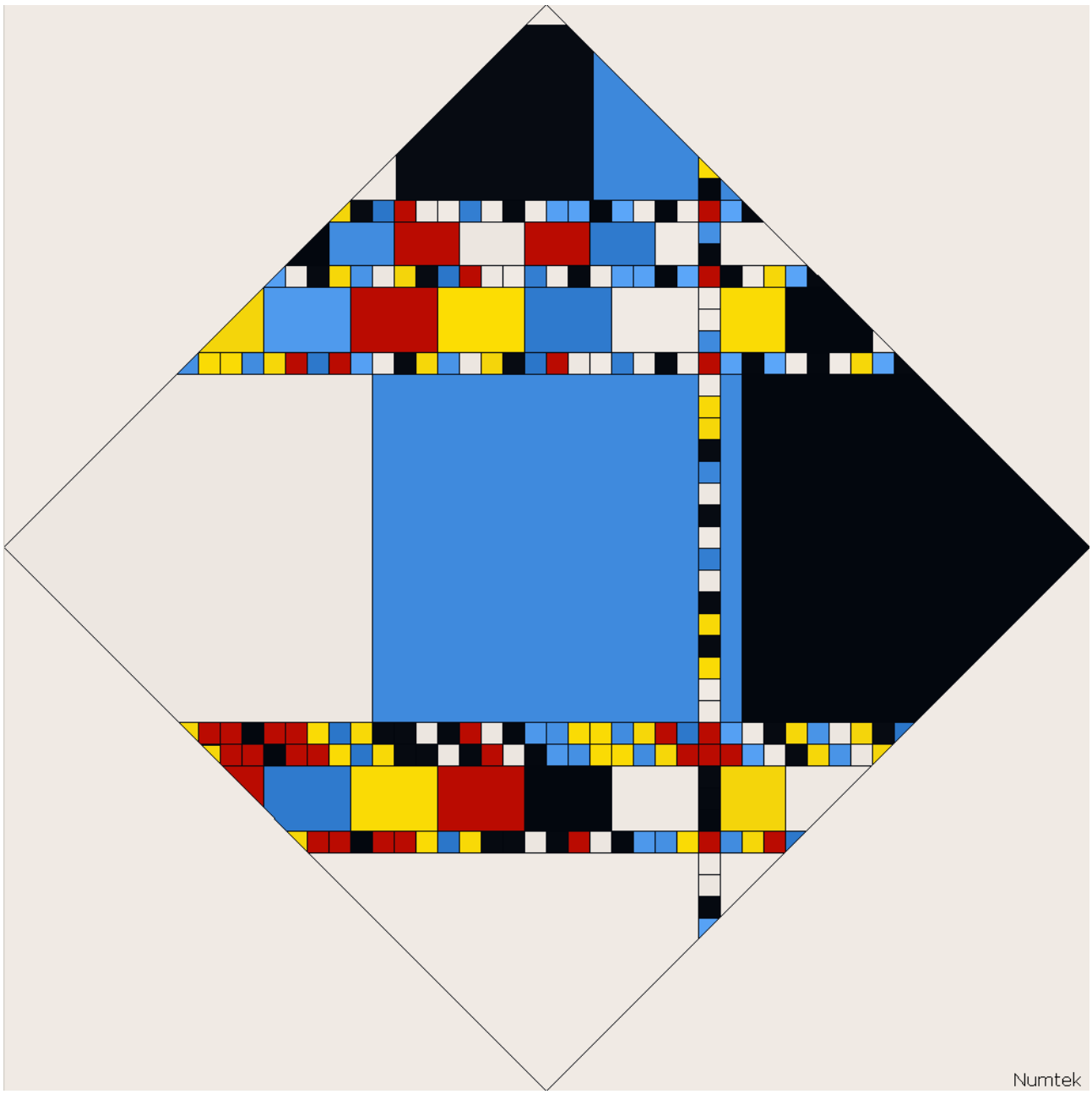
Een aantal versies later kwam dit er per ongeluk uit. Ik moest onwillekeurig glimlachen om de gelijkenis met Broadway Boogie-Woogie en overwoog bijna om deze output te gebruiken voor mijn inzending. De rust en de leegte, de gele vlakken die net niet de bovenliggende horzontale lijnen er boven raken leverden in mijn optiek een prachtige compositie op. Ik kon me zo Mondriaan voor me zien in zijn atelier, tevreden zuchtend met het eindresultaat maar in zijn hoofd wel al weer twintig stappen verder zijn. Alleen zal hij dan nooit een verticale lijn gebruiken die niet naadloos aansluit op een horzizontale,.



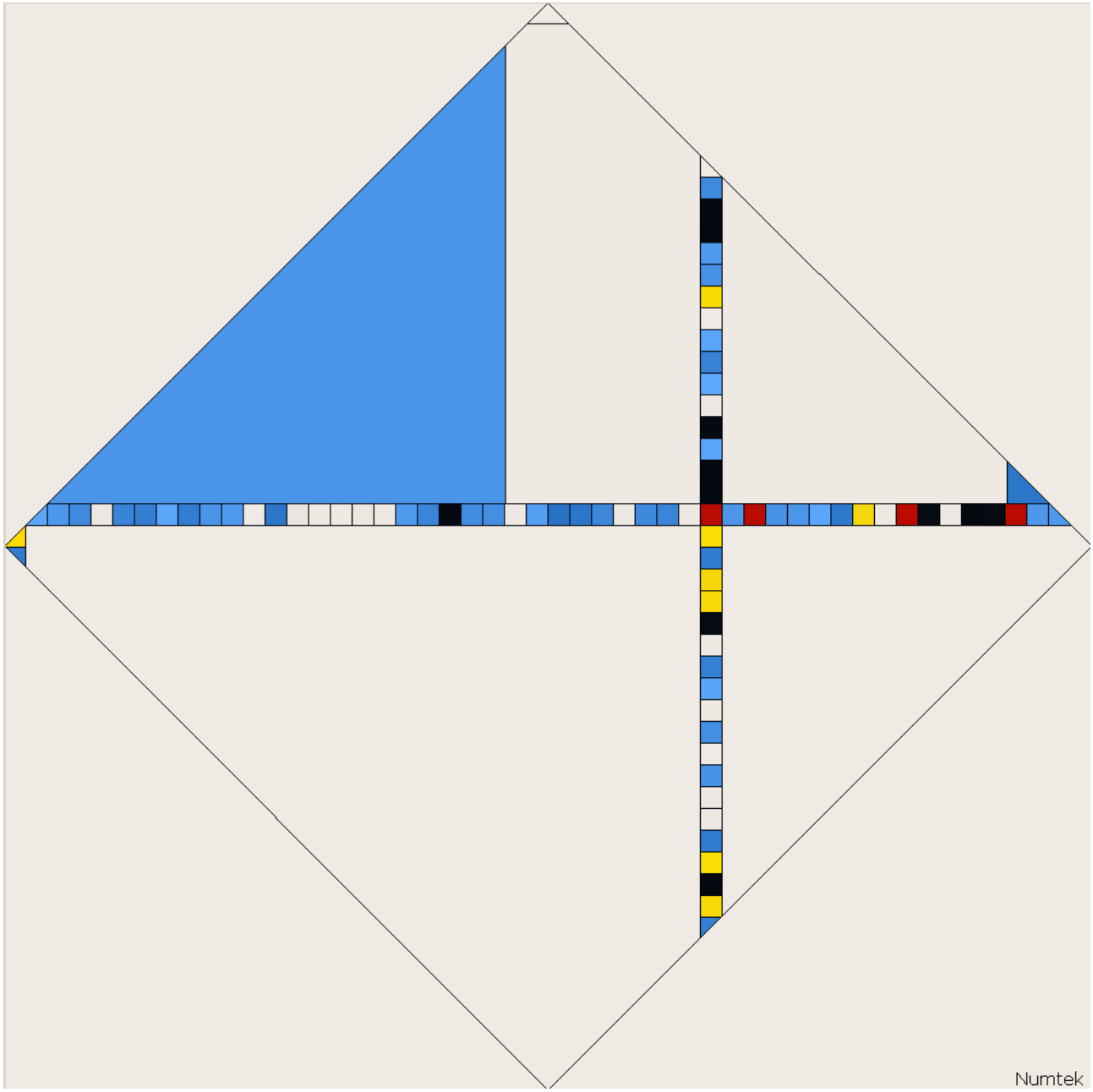
Ook al was deze versie nog steeds gemaakt met een audio opname van belabberde kwaliteit, het einddoel kwam al behoorlijk in zicht. Dit is gemaakt rond versie 30 van het script, het uiteindelijke eindresultaat was versie 47. Vanaf dit punt heb ik mij vooral gericht op het herschrijven van het script, omdat er visueel gezien niet zo heel veel meer aan verbeterd kon worden nar mijn mening. Dit was ook de versie die ik eerder had verzonden naar de jury om aan te geven dat ik bezig was met mijn inzending. Wat nog wel duidelijk ontbreekt zijn zaken die pas opvallen na een wat aandachtiger blik. Wat mist is de rust, de compositie is niet speels genoeg en ook zijn de kleuren per vlak zijn nog steeds hetzelfde.



Gedurende een discussie met andere mensen die ook participeerden in deze wedstrijd kwam het punt ter sprake van de zwarte outlines. Om mijn keuze te motiveren had ik deze versie gemaakt zonder outlines.



Een van de eerste versie die ik maakte aan de hand van mijn nieuwe audio-opnames. Typerend is de dubbele horizontale lijn die ook in mijn uiteindelijke inzending te zien is. De witte vlakken aan de linker- en onderkant geven naar mijn mening een heel prettig, ruimtelijk gevoel.



Ineens was er dit, een bijna leeg canvas dat in de verte doet denken aan een molen, of misschien wel een zeilboot. De rode vlakken bepalen wanneer er een verticale lijn getekend gaat worden, en deze versie had blijkbaar besloten dat de audio onvoldoende input opleverde om voldoende vlakken te vullen.