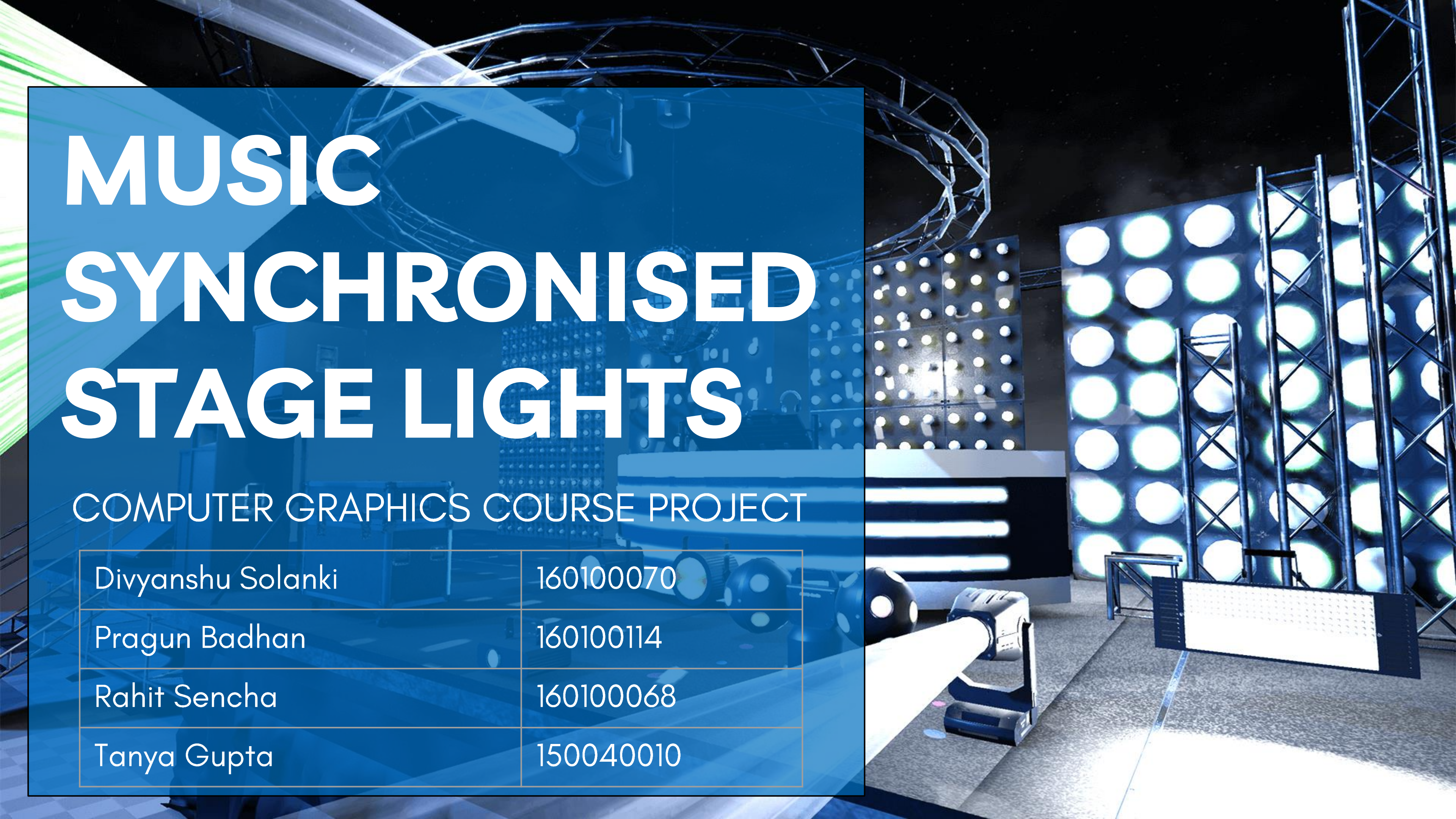
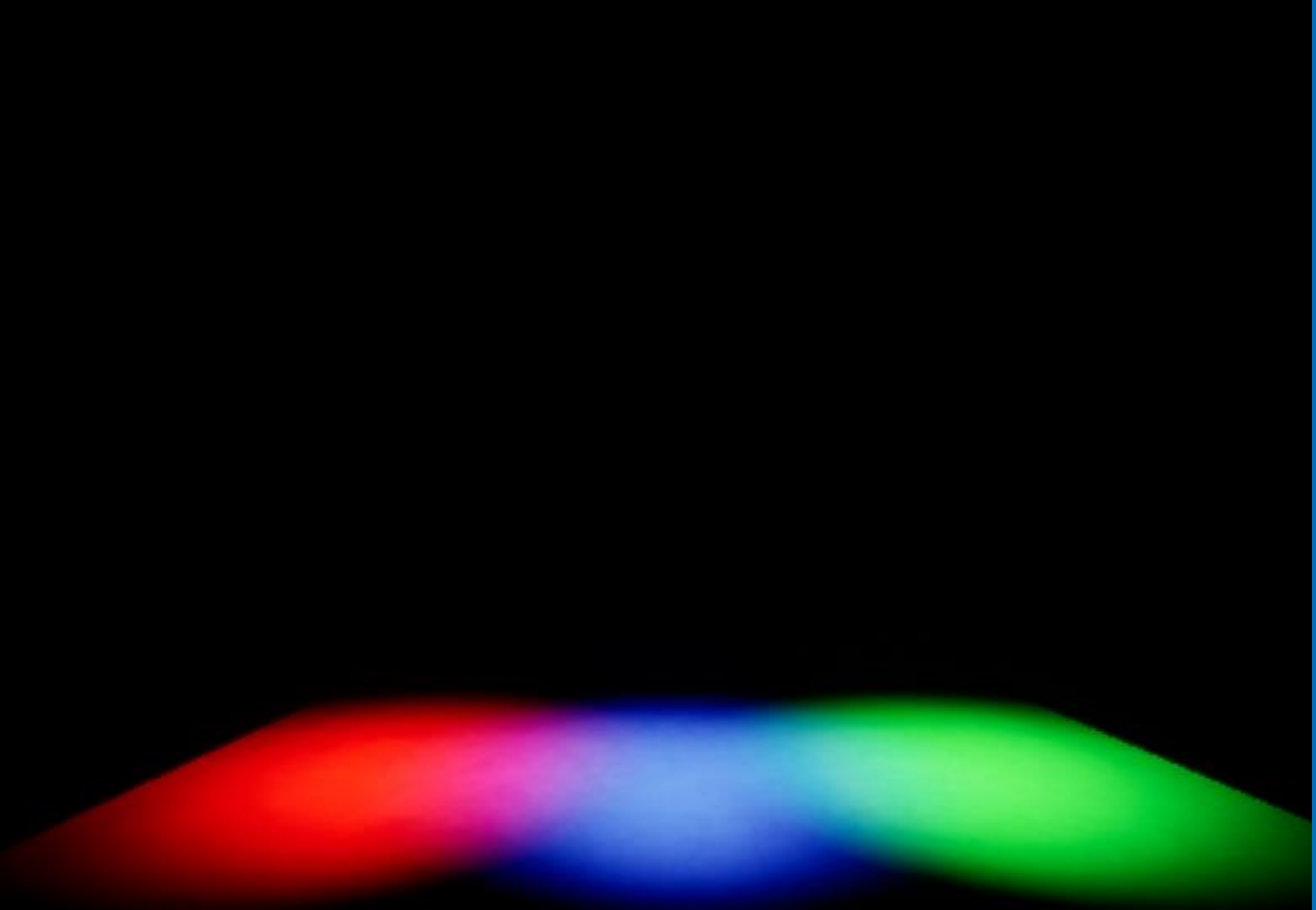


MUSIC SYNCHRONISED STAGE LIGHTS

COMPUTER GRAPHICS COURSE PROJECT

Divyanshu Solanki	160100070
Pragun Badhan	160100114
Rahit Sencha	160100068
Tanya Gupta	150040010

TABLE OF CONTENTS



01 INTRODUCTION

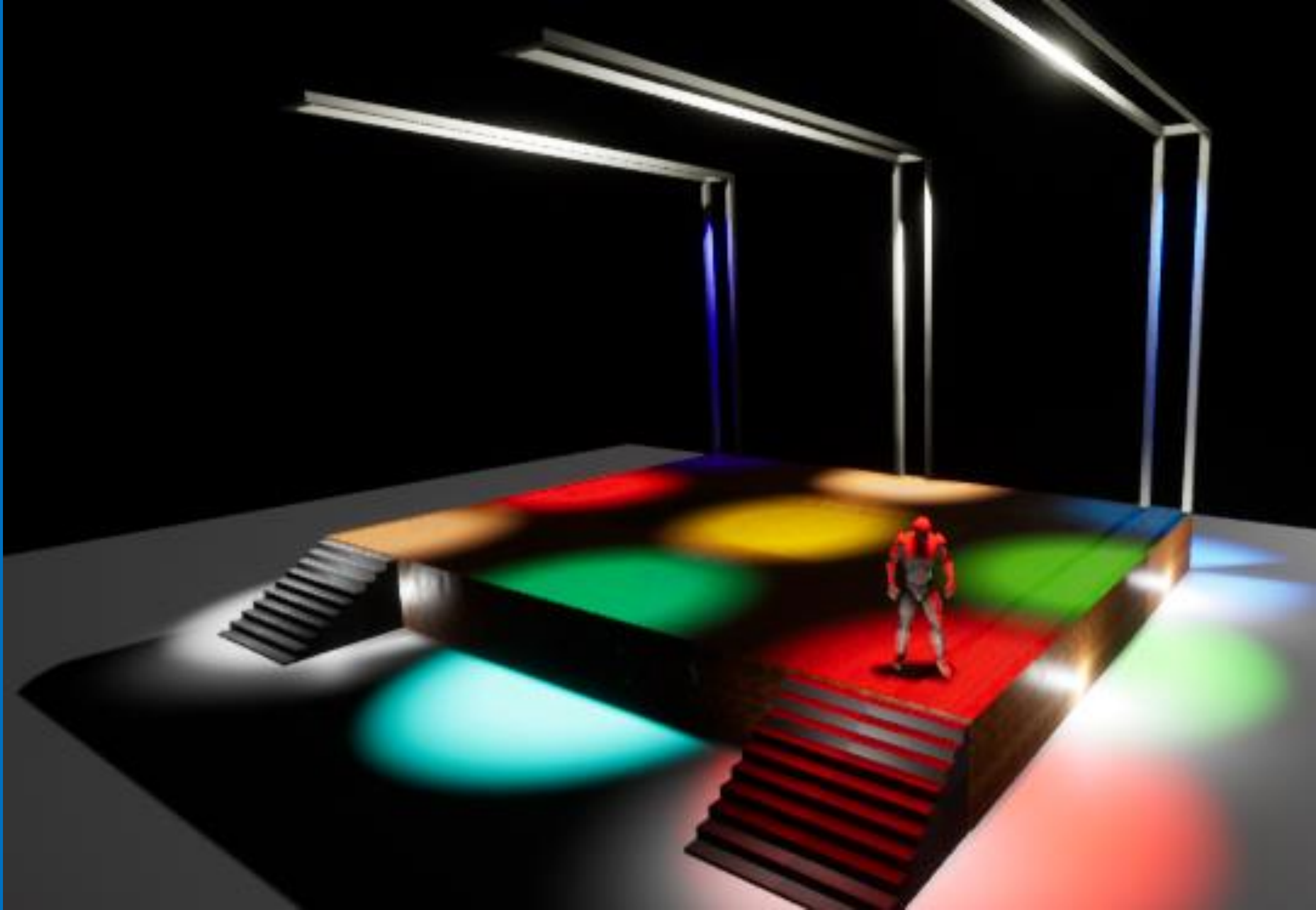
NEED FOR MUSIC SYNCED LIGHTS

Current scenario is that there is a person who controls the lights, its movement and its intensity in association with the music. We would like to remove this human part and automate the process. This will help reduce costs and remove errors.

Started with familiarizing with the software Unreal Engine and implemented a basic stage light control using time delay

DEMO

02 PROJECT DESIGN



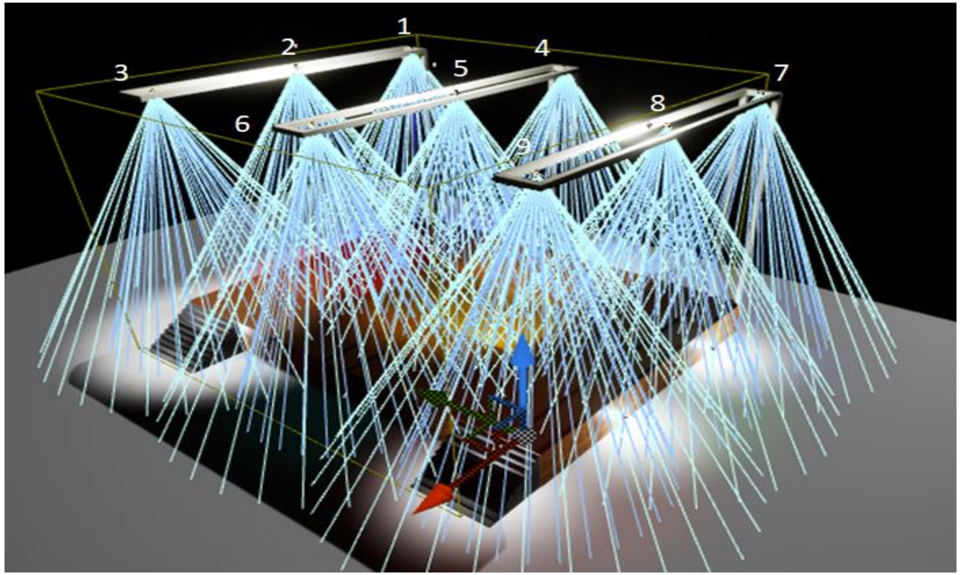
STAGE MODELLING

1. Floor for audience
2. A raised stage for singer and
3. 3*3 spotlight grid whose intensity changes in real time as per characteristics of music.
4. A singer was imported from existing templates
5. Three overhead ceiling bars to support spotlights
6. Static illumination lights around the platform and on ceiling for environment illumination
7. Two entry staircases

LightMass Importance Volume – To reduce graphics rendering load on GPU

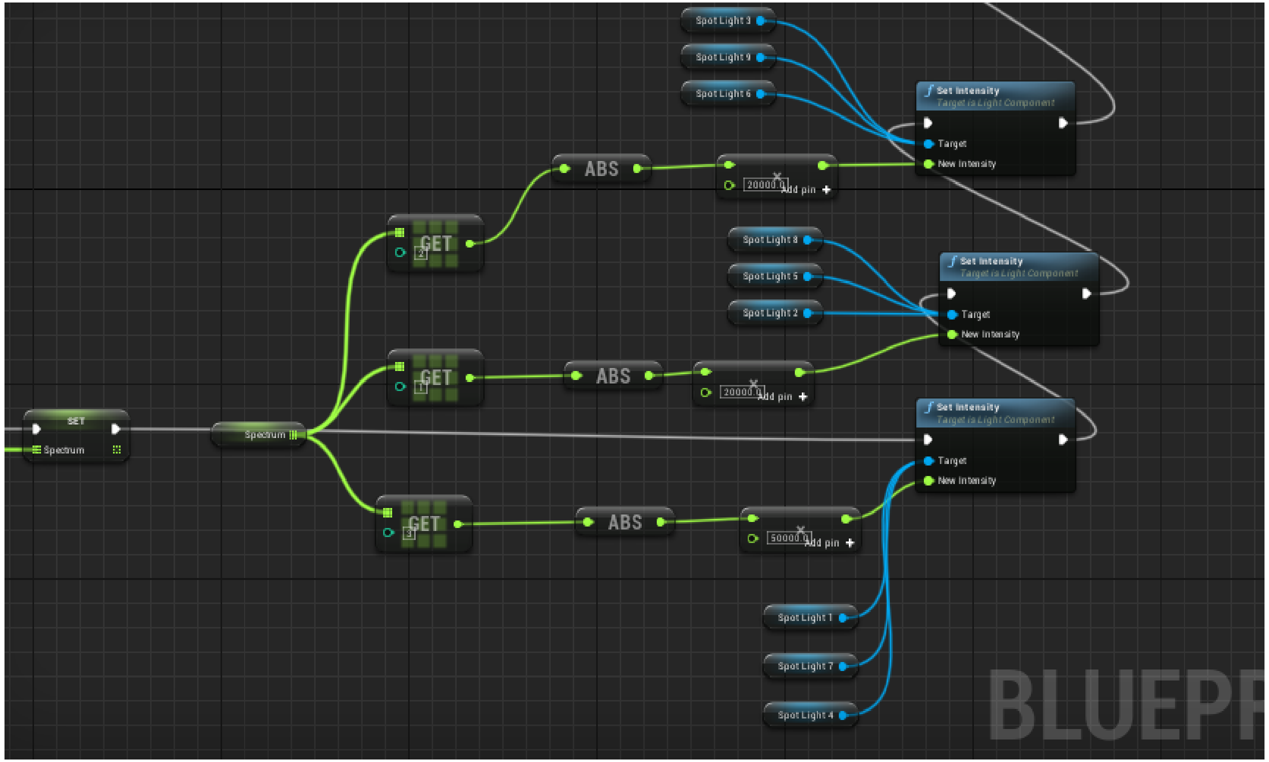
LIGHTS

– **Intensity** determines how much energy the light outputs into the scene.
– **Light Color** will adjust the color of the light and the sprite that represents the light in the editor will change its color to match.
– **Attenuation Radius** of the light sets the reach of the light and defines what objects it will affect
– **Source Radius** and **Source Length** define the size of specular highlights on surfaces.
– **Static lights** are those whose intensity remains constant throughout the simulation
– **Outer cone angle** and **Inner cone angl**e of spotlights for directing light as required



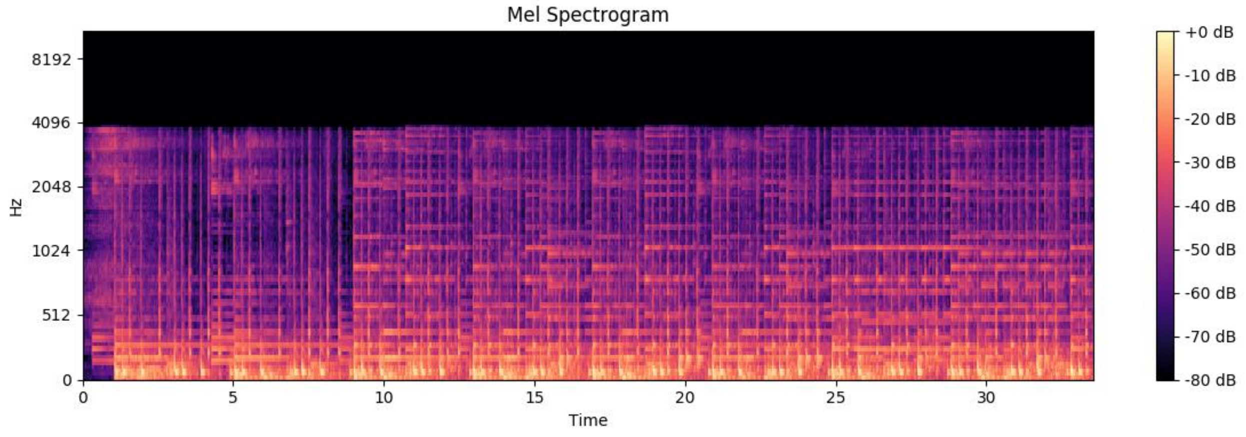
SPOTLIGHT ANIMATION DESIGN

Blueprint allows for visual scripting of simulation events.



DEMO

03 AUDIO ANALYSIS



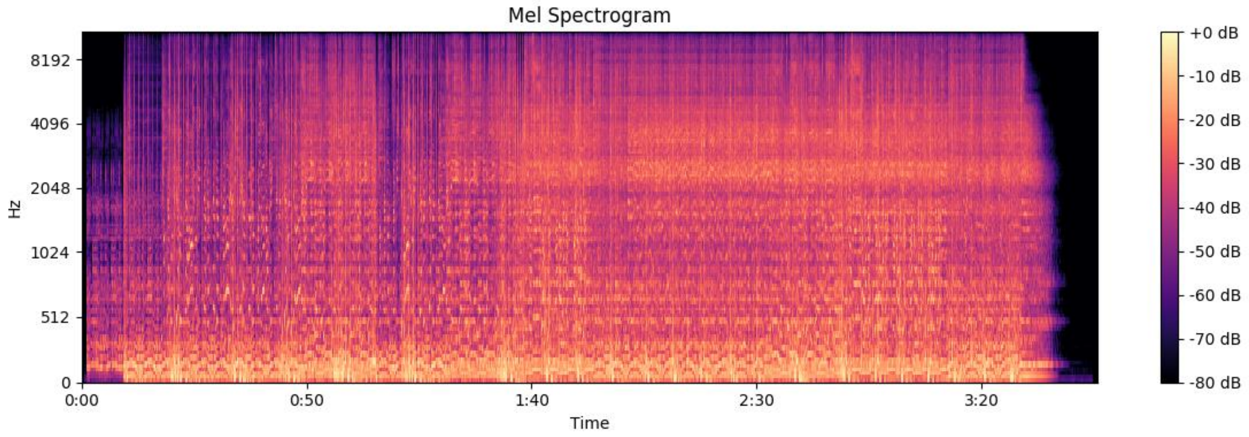
The analysis of audio was performed in MATLAB and then, due to limited functionalities of MATLAB and open-source nature of Python, using the Librosa library of Python.

MEL SPECTOGRAM

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. A Mel scale is a widely used scale in music which converts the frequencies (Hz) to mels as:

$$m = 2595 * \log\left(1 + \left(\frac{f}{700}\right)\right)$$

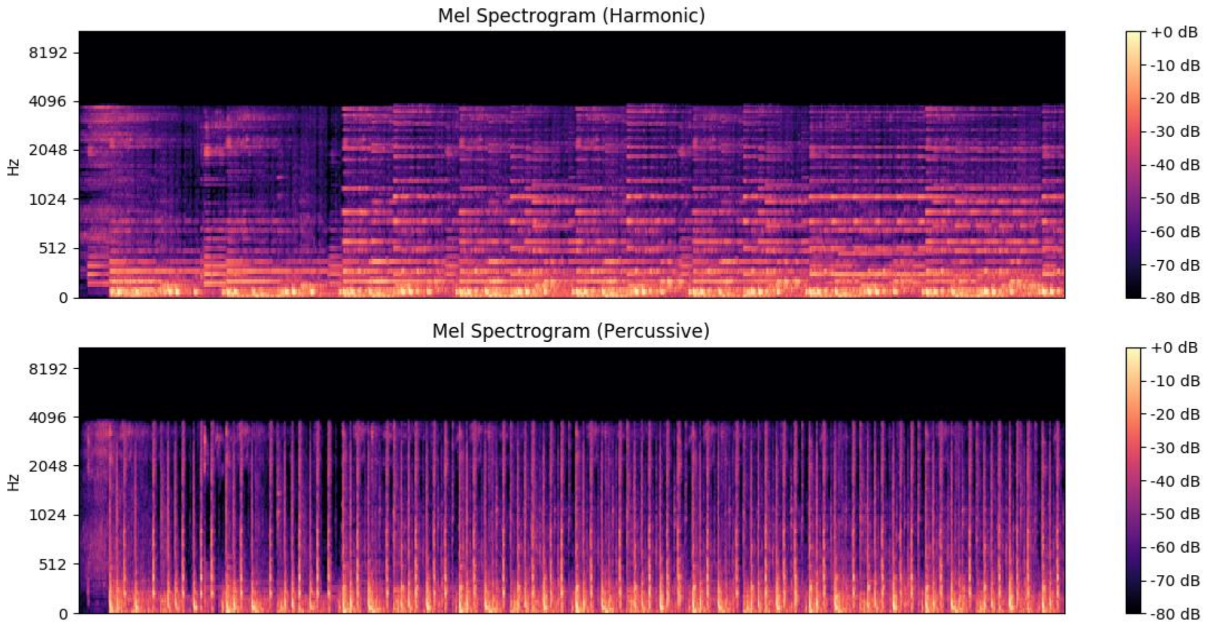
A Mel Spectrogram shows which frequencies occur in the audio and at which time with what power/intensity. Hence it is a 3D graph and Librosa uses colour gradients to show the power values.



The spectrogram plot for Nirvana's 1992 song "Come As You Are"

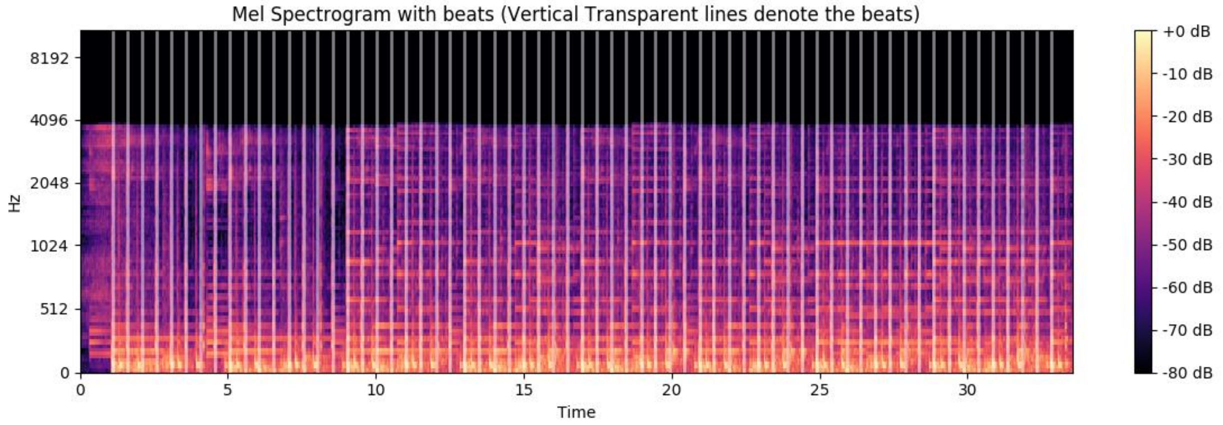
SEPARATING HARMONIC AND PERCUSSION COMPONENTS

- The percussive components of the audio correspond to the sound from two colliding objects, for example, drums. Percussive sounds do not have a pitch but a very clear localization in time.
- Harmonic sounds are perceived to have a certain pitch, for example a violin sound. Usually, a note played on an instrument (say piano) has onset of percussion (hammer strung on piano strings) followed by a harmonic tone (from the vibration of the strings)



BEAT TRACKING AND TEMPO

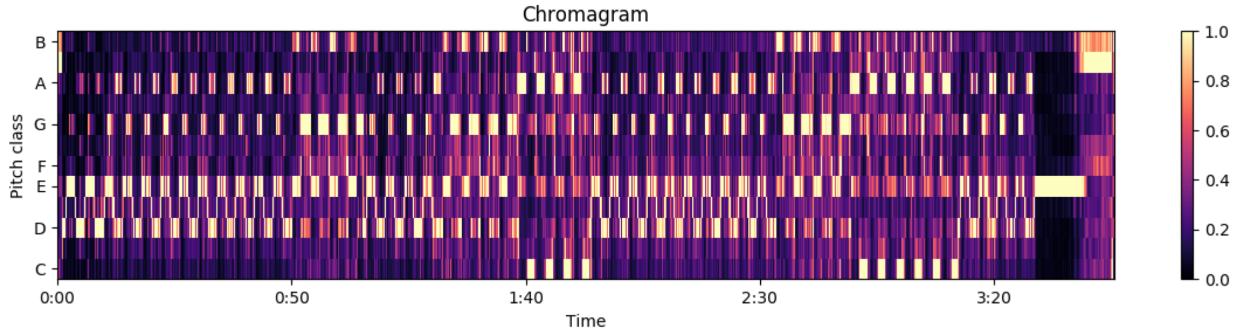
Beat tracking simply means finding the time stamps where a listener “taps feet”, which is either the onset from a musical instrument which is regularly spaced as a part of the musical rhythm.

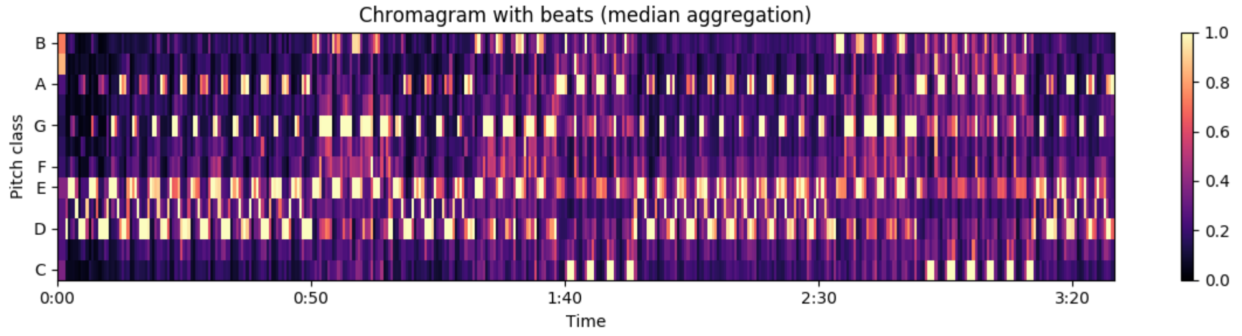


The spectrogram with beats shown (transparent vertical lines) is generated

CHROMAGRAM

A chromagram represents what pitch classes belong in a sound and its various components. A pitch class is the set of all pitches which are a whole number of octaves apart. A chromagram is a 3-D graph showing the power/intensity of each pitch class at a time instant in the audio sample. This is particularly useful for us as the onset of an instrument can be tracked in the chromogram and correspondingly the light can be moved and the duration for which it is lit.





The chromagram generated for Nirvana's 1992 song "Come As You Are"

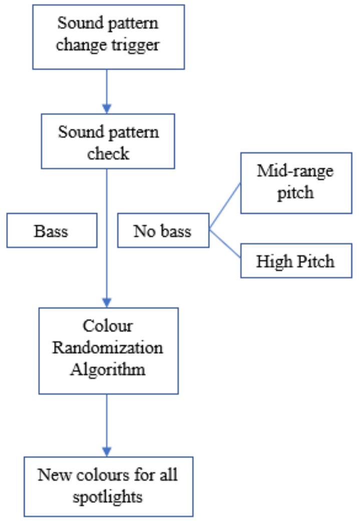
DEMO

04 SPOTLIGHT RANDOMISATION

SPOTLIGHT CONTROL

- **Randomize colors** of each spotlight when a beat or sound pattern is changed
- Control the **flickering frequency** based on beat frequency
- Control the **flicker intensity** based on beat amplitude for sections of concert with beats in the song
- For beat-less, slow sections, control **fade in and fade out** of lights

$$Rate\ of\ fading = \frac{\Delta frequency}{\Delta time}$$


Sound pattern change trigger
Sound pattern check
Bass
No bass
Mid-range pitch
High Pitch
Colour Randomization Algorithm
New colours for all spotlights

Sound Pattern check analyze every time step in Chronogram for bass/ mid-frequency / shrill sound

Colour Randomization Algorithm

2 layered N X 3 matrix, where N represents number of spotlights

Layer 1 stores in each row (which corresponds to particular spotlights) 3 randomly generated values corresponding to ON (1) or OFF (0) state of colours R, G, & B. It is ensured that RGB state of any spotlight is never 000, that is spotlight is not OFF.

Layer 2 stores randomized intensity value of R, G, & B between 1 to 255 (1 being least intense, 255 being max intensity) for every spotlight.

This way a particular colour in form of an RGB intensity mix can be assigned to each spotlight

DEMO

05 FUTURE WORK

EXPANSION ON CURRENT WORK

MACHINE LEARNING
Genre identification in music can be effectively implemented using Neural Networks

POST-PROCESSING
Strobes, black out, bright white flashes, hue shifts to make lights match the music

REAL-TIME INPUT
Currently, we are limited to music files already stored in system

CONNECTING
Automatic connection of the spotlight control code with audio analysis code

INCLUDE CONTROLS
Control of the rotation and solid angle of the spotlight cone, lasers & artificial fog

THANK YOU!