|  |  |
| --- | --- |
|  | |
|  | |
|  | |
|  | |
|  | |
| Šimon Raus | |
|  | |
| Aktivity | Popis: fai_logo_cz |

# OBSAH

[1 OBSAH 2](#_Toc196826215)

[2 Úvod 3](#_Toc196826216)

[3 SEZNAM VYTVOŘENÝCH ČÁSTÍ AKTIVITNÍ PRÁCE 4](#_Toc196826217)

[4 5](#_Toc196826218)

[5 seznam obrázků 6](#_Toc196826219)

[6 Seznam Tabulek 7](#_Toc196826220)

[7 Seznam použité literatury 8](#_Toc196826221)

# Úvod

Tato seminární práce se zaměřuje na návrh a implementaci nástroje, který umožňuje konverzi dat mezi dvěma na první pohled zcela odlišnými formáty – rastrovým obrazovým formátem BMP a zvukovým formátem WAV. Cílem není pouze samotná konverze, ale i hlubší porozumění jejich datové struktuře, způsobu ukládání informací a možnostem kreativního i praktického využití těchto převodů.

Konverze mezi obrazem a zvukem není v informatice novým konceptem. Obě domény – vizuální a akustická – jsou nakonec redukovatelné na sekvence číselných hodnot, a tedy na čistá binární data. Tato práce zkoumá, jak lze strukturovaná binární data jednoho formátu převést do jiného bez ztráty integrity dat, a přitom zachovat použitelnost a interpretovatelnost výsledného souboru v běžných přehrávačích či prohlížečích.

Vyvinutý nástroj, postavený na programovacím jazyce Python, nabízí následující funkcionalitu:

* **Načítání a dekódování BMP souborů** s barevnou hloubkou 1, 4, 8 nebo 24 bitů, včetně zpracování palet a převodu do RGBA reprezentace v knihovně NumPy.
* **Načítání a dekódování WAV souborů** s 32 bity na vzorek a převodu do datové struktury knihovny NumPy.
* **Převod BMP na WAV**, při kterém je obrazová informace serializována do 32bitového zvukového signálu bez komprese.
* **Zpětný převod WAV na BMP**, přičemž výstupem je 24bitový bitmapový obrázek rekonstruovaný ze vzorkovaných dat.
* **Generování náhodných BMP a WAV souborů**, a to jak v podobě bílého šumu, tak sinusového signálu pro akustickou demonstraci.
* **Zobrazení dat obou formátů**, např. rozměrů, počtu barev, počtu vzorků, délky zvuku a dalších technických parametrů.
* **Vizualizaci dat** pomocí knihovny Matplotlib, ať už ve formě obrázků, nebo jako grafy průběhu zvukových vln.

# SEZNAM VYTVOŘENÝCH ČÁSTÍ AKTIVITNÍ PRÁCE

|  |  |  |  |
| --- | --- | --- | --- |
| **Název dílčí části** | **[využitý SW] Název/názvy vytvořených výstupů**  **podkladová data** | **Odkaz na použitý zdroj** | **Doba řešení dané části [h]** |
|  |  |  |  |
|  |  |  |  |

# Analýza formátu BMP a Wav

Před samotnou implementací konverzí bylo nutné detailně prostudovat specifikace formátů BMP a WAV. Oba tyto formáty mají dobře definovanou binární strukturu, která umožňuje jejich čtení a zápis i bez použití specializovaných knihoven. V této kapitole jsou popsány jejich základní stavební bloky a praktické aspekty, které byly při vývoji zohledněny.

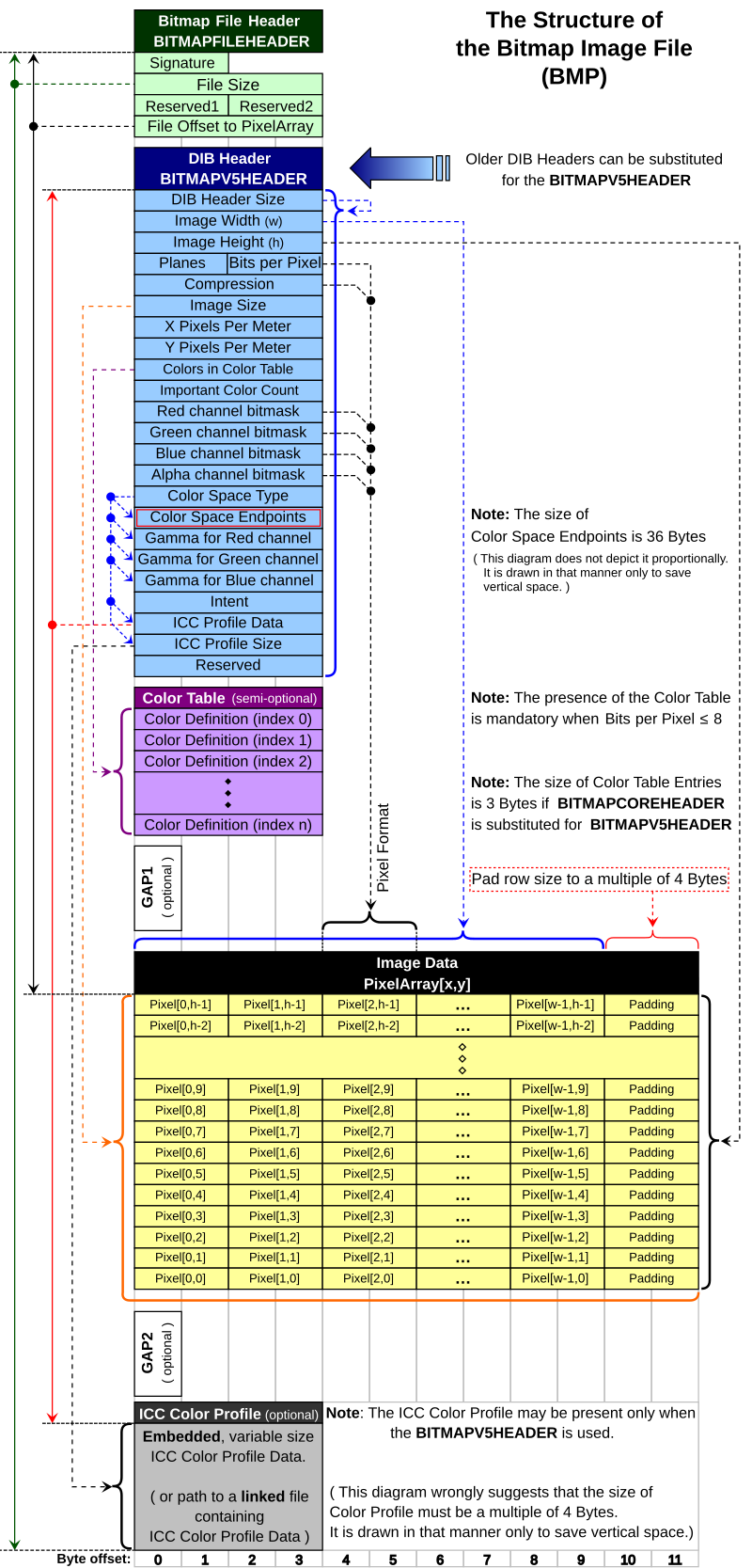
## BMP – Bitmap Image File

Formát BMP (Bitmap) je jednoduchý rastrový formát používaný především v systémech Windows. Obsahuje fixní hlavičku, volitelnou paletu a samotná obrazová data.

**Struktura BMP souboru**

Tabulka 1 Formát BMP

|  |  |  |
| --- | --- | --- |
| Název | Velikost v bytech | Popis |
| BMP identifier | 2 | Identifikátor BMP |
| File size | 4 | Velikost souboru |
| Reserved bytes | 4 | Rezervované byty |
| Palette offset | 4 | Odsazení palety |
| Header size | 4 | Velikost hlavičky |
| Width | 4 | Šířka obrázku |
| Height | 4 | Výška obrázku |
| Planes | 2 | Počet barevných vrstev |
| Bits per pixel | 2 | Barevná hloubka |
| Compression | 4 | Kompresní metoda |
| Image size | 4 | Velikost obrázku |
| Horizontal resolution | 4 | Horizontální rozlišení |
| Vertical resolution | 4 | Vertikální rozlišení |
| Number of colors | 4 | Počet barev |
| Important colors | 4 | Počet důležitých barev |



Obrázek 1 BMP formát[1]

### Příklad BMP formátu

A table with numbers and a red text

AI-generated content may be incorrect.

Obrázek 2 Příklad BMP formátu [2]

**Bitová hloubka a formát pixelů**

* **1 bit** – 2 barvy, každý bit reprezentuje jeden pixel.
* **4 bity** – 16 barev, každý bajt obsahuje 2 pixely.
* **8 bitů** – 256 barev, každý bajt je index do palety.
* **24 bitů** – Přímé RGB hodnoty (3 B na pixel).

**Zarovnání a pořadí řádků – scanline**

Každý řádek pixelů musí být zarovnán na 4 bajty. To znamená, že za každým řádkem mohou být přidány vyrovnávací bajty (padding). Obrázky jsou navíc uloženy odspodu – první řádek v souboru odpovídá spodnímu řádku obrazu.



Obrázek 3 Výpočet ScanLine [1]

## WAV – Waveform Audio File Format

Formát WAV je kontejnerový formát postavený na RIFF (Resource Interchange File Format) a nejčastěji obsahuje nekomprimované PCM zvukové stopy.

**Struktura WAV souboru**

Tabulka 2 Formát WAV

|  |  |  |
| --- | --- | --- |
| Název | Velikost v bytech | Popis |
| Chunk ID | 4 | Obsahuje ‘RIFF’ v ASCII |
| Chunk size | 4 | Velikost zvůkových dat |
| Format | 4 | Obsahuje ‘WAVE’ v ASCII |
| Subchunk 1 ID | 4 | Identifikátor sub-bloku |
| Subchunk 1 size | 4 | Velikost hlavičky |
| Audio Format | 2 Kompresní metoda |  |
| NumChannels | 2 | Počet kanálů |
| Sample Rate | 4 | Snímkovací frekvence |
| Bytes per second | 4 | Byte Rate |
| Block Align | 2 | Počet bytů na sample |
| Bits per sample | 2 | Bitová hloubka audio signálu |
| 'Data' | 4 | Obsahuje ‘Data’ v ASCII |
| Subchunk 2 size | 4 | Velikost audio dat |

### Příklad WAV formátu

52 49 46 46 24 08 00 00 57 41 56 45 66 6d 74 20 10 00 00 00 01 00 02 00

22 56 00 00 88 58 01 00 04 00 10 00 64 61 74 61 00 08 00 00 00 00 00 00

24 17 1e f3 3c 13 3c 14 16 f9 18 f9 34 e7 23 a6 3c f2 24 f2 11 ce 1a 0d

A screenshot of a computer code

AI-generated content may be incorrect.

Obrázek 5 WAV formát [3]

**Parametry důležité pro projekt**

* **Sample Rate (Hz)** – Počet vzorků za sekundu (např. 44 100 Hz).
* **Bit Depth** – Počet bitů na jeden vzorek (např. 32 bitů = 4 bajty).
* **Počet kanálů** – Používá se 1 (mono) pro zjednodušení převodů.
* **Formát vzorku** – Pro tento projekt se používá 32bitové signed integer PCM.

**Zápis a čtení vzorků**

V projektu jsou zvuková data uložena jako int32, což odpovídá čtyřem bajtům na jeden vzorek. Díky tomu lze data relativně snadno přemapovat na barvy (8 bitů na složku RGBA) nebo opačně při rekonstrukci BMP z WAV.

# Funkce Převodníku

Tato kapitola popisuje konkrétní funkce, které aplikace poskytuje, a technologické schopnosti, které umožňují obousměrný převod mezi obrazovými a zvukovými daty, jejich generování a vizualizaci.

## Generace náhodného obrázku

Aplikace umožňuje generovat náhodné rastrové obrázky ve formátu BMP s definovanou šířkou a výškou. Obrázek je tvořen polem pixelů, kde každý pixel obsahuje náhodně vygenerované hodnoty pro jednotlivé barevné kanály (červený, zelený, modrý). Tyto hodnoty jsou reprezentovány jako 8bitová čísla, takže každý pixel zabírá celkem 24 bitů.

**Vlastnosti**

* **Rozlišení obrázku** je zadáváno uživatelem pomocí šířky a výšky v pixelech.
* **Barevné složky** (R, G, B) jsou generovány nezávisle pro každý pixel, což zaručuje vizuálně náhodné a různorodé obrázky.
* **Formát výstupu** je 24bitový nekomprimovaný BMP.
* **Použití**: Výstupní obrázek může sloužit jako vstup pro převod na WAV, nebo jako testovací soubor při vývoji.

**Parametry příkazové řádky**

py ./main.py -gb <sirka> <vyska> <vystupni\_cest> [--show -s] [--show-with-axes -sx] [--test]

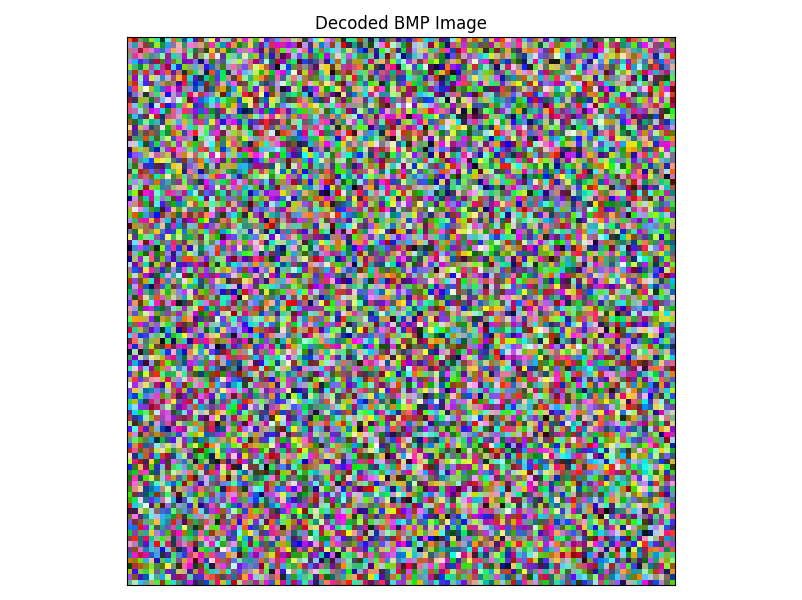
**Volitelné přepínače**

* -s zobrazí vygenerovaný obrázek po jeho vytvoření.
* -sx zobrazí vygenerovaný s osami obrázek po jeho vytvoření.
* --test uloží zobrazení do testovací složky místo zobrazení v okně.

**Příklad použití**

py ./main.py -gb 100 100 ./content/media/out/random.bmp -s

Tento příkaz vygeneruje náhodný obrázek velikosti 100×100 pixelů, uloží jej jako random.bmp a okamžitě jej zobrazí v grafickém okně.



Obrázek 6 Generace náhodného obrázku

**Interní průběh**

1. Inicializace pole np.ndarray typu uint8 s rozměry (výška, šířka, 4).
2. Náhodné vyplnění hodnot v rozsahu 0–255 pro každý kanál.
3. Serializace dat do BMP souboru pomocí interní BMP třídy.
4. Volitelně: vykreslení obrázku pomocí matplotlib.

## Generace náhodného zvuku

Aplikace umožňuje generovat náhodný zvukový signál ve formátu WAV. Tento signál je tvořen posloupností náhodných 32bitových celých čísel, která představují jednotlivé vzorky zvuku. Výsledný soubor může sloužit jako testovací vstup pro převod do bitmapového obrázku nebo pro vizualizaci dat.

**Vlastnosti**

* **Délka zvuku** je zadána uživatelem v počtu vzorků.
* **Vzorkovací frekvence** je pevně nastavena na 44 100 Hz.
* **Počet kanálů**: 1 (mono).
* **Bitová hloubka**: 32 bitů na vzorek.
* **Formát výstupu**: WAV PCM (nekomprimovaný).

**Parametry příkazové řádky:**

py ./main.py -gw <pocet\_vzorku> <vystupni\_cesta> [--show -s] [--show-with-axes -sx] [--test]

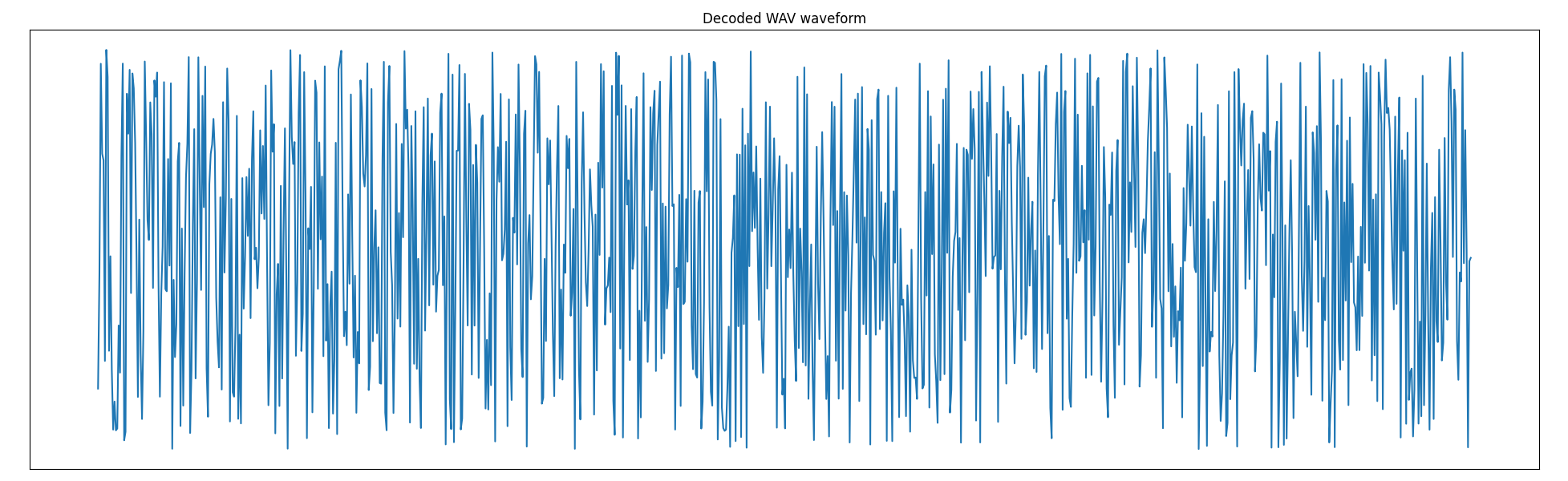
**Volitelné přepínače**

* -s zobrazí vygenerovanou zvukovou vlnu (vizualizaci) po jejím vytvoření.
* -sx zobrazí vygenerovanou zvukovou vlnu (vizualizaci) s osami po jejím vytvoření.
* --test uloží zobrazení do testovací složky místo jeho zobrazení na obrazovce.

**Příklad použití**

py ./main.py -gw 1000 .\content\media\out\random\_wave.wav -s

Tento příkaz vygeneruje náhodný zvukový signál s 1000 vzorky, uloží jej jako random.wav a zobrazí jeho grafický průběh.



Obrázek 7 Generace náhodné vlny

**Interní průběh**

1. Generování pole náhodných 32bitových celých čísel (np.int32).
2. Vytvoření hlavičky WAV podle parametrů.
3. Serializace dat do WAV souboru.
4. Volitelně: vizualizace signálu pomocí matplotlib.

## Generace sinusového signálu

Aplikace umožňuje vytvořit čistý sinusový signál ve formátu WAV. Tento typ signálu je ideální pro testování kvality převodu a vizualizace zvukových dat, protože má jednoduchý a předvídatelný průběh.

**Vlastnosti**

* **Frekvence signálu**: Uživatelsky definovatelná (např. 440 Hz).
* **Vzorkovací frekvence**: 44 100 Hz (pevně daná).
* **Počet vzorků**: Uživatelsky definovatelný počet vzorků.
* **Počet kanálů**: 1 (mono).
* **Bitová hloubka**: 32 bitů na vzorek.
* **Formát výstupu**: WAV PCM (nekomprimovaný).

**Parametry příkazové řádky**

py ./main.py -gw <pocet\_vzorku> <výstupní\_cesta> -f <frekvence> [--show -s] [--show-with-axes -sx] [--test]

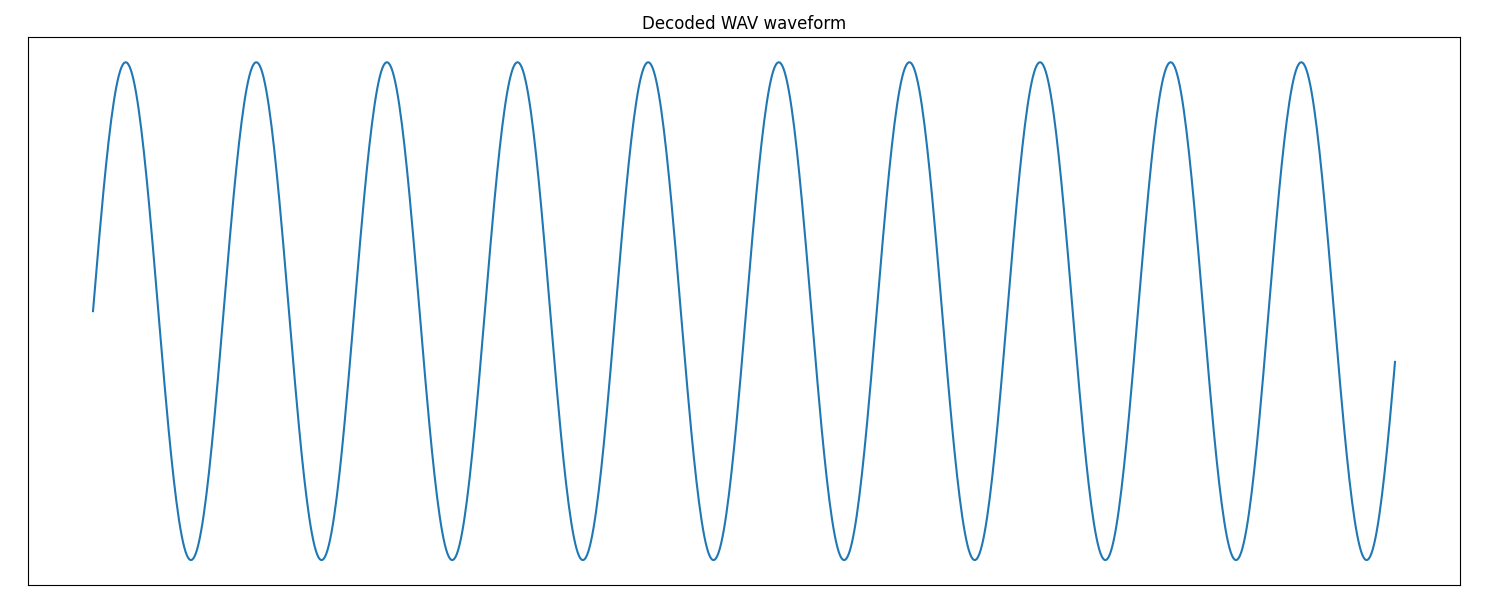
**Volitelné přepínače**

* -s zobrazí vygenerovanou vlnu (sinusový průběh) po jejím vytvoření.
* -sx zobrazí vygenerovanou zvukovou vlnu (vizualizaci) s osami po jejím vytvoření.
* --test uloží výstupní graf místo jeho přímého zobrazení.

**Příklad použití**

py ./main.py -gw 1000 .\content\media\out\sine\_wave.wav -f 440 -s

Tento příkaz vytvoří 1000 vzorků sinusového signálu o frekvenci 440 Hz (komorní A) a zobrazí jeho průběh.



Obrázek 8 Generace sinusového signálu

**Interní průběh**

1. Vytvoření časové osy podle vzorkovací frekvence a délky.
2. Výpočet hodnot sinusové funkce v čase (amplituda ±1).
3. Normalizace a převod do formátu np.int32.
4. Vytvoření WAV hlavičky a serializace dat.
5. Volitelně: vizualizace pomocí matplotlib.

## Převod BMP na WAV

Aplikace umožňuje převod rastrového obrázku ve formátu BMP na zvukový soubor WAV. Každý pixel obrázku je převeden na jeden zvukový vzorek, přičemž hodnoty barevných složek (RGB) jsou serializovány do celého čísla a reprezentují amplitudu signálu.

**Vlastnosti**

* **Podporované BMP formáty**: 1bit, 4bit, 8bit, 24bit (RGB se doplňuje).
* **Zachování informací**: Přesná bitová reprezentace pixelových dat včetně jejich pořadí.
* **Výstupní WAV**
  + Vzorkovací frekvence: 44 100 Hz
  + Hloubka: 32 bitů
  + Mono
  + Neobsahuje audiální reprezentaci, ale datovou (nepoužitelné k poslechu)

**Parametry příkazové řádky**

python main.py --convert <vstup.bmp> <vystup.wav> [--show -s] [--show-with-axes -sx] [--test]

**Volitelné přepínače**

* --show: zobrazí zvukový průběh odpovídající datům z obrázku (ne sinus, ale hodnota amplitud).
* --test: uloží vizualizaci místo zobrazení v okně.

**Příklad použití**

python main.py --convert ./content/media/bmp/1bit.bmp ./content/media/out/1bit.wav –show



Obrázek 9 1bitový BMP obrázek převeden na zvukovou vlnu

**Interní průběh**

1. Načtení BMP pomocí interního dekodéru (vlastní parser).
2. Převod jednotlivých RGBA hodnot pixelů na 32bitová celá čísla (bitové operace).
3. Sestavení WAV hlavičky s metadaty (včetně rozměrů obrázku).
4. Serializace do WAV souboru.
5. Volitelně: zobrazení hodnot jako průběh (osa x: index, osa y: hodnota).

**Využití**

* Možnost „ukrýt“ obrázek ve zvukovém souboru.
* Slouží jako základ pro zpětný převod do obrazu.

## Převod WAV na BMP

Aplikace umožňuje převést zvukový soubor ve formátu WAV zpět na rastrový obrázek BMP. Tento proces využívá toho, že zvukový soubor obsahuje serializovaná data pixelů, která byla do něj dříve zapsána při konverzi z BMP.

**Vlastnosti**

* **Převod amplitud zpět na RGB hodnoty**.
* **Rekonstrukce původního rozměru obrázku** – pokud je uložen v hlavičce WAV.
* **Zajištění zpětné kompatibility** s konverzí BMP → WAV.

**Parametry příkazové řádky**

python main.py --convert <vstup.wav> <vystup.bmp> [-c width height] [--show -s] [--show-with-axes -sx] [--test]

**Volitelné přepínače**

* -c width a height: volitelné, pouze pokud není rozměr v hlavičce WAV.
* --show: zobrazí vygenerovaný obrázek.
* --test: uloží náhled obrázku jako display\_bmp\_output.png.

**Příklad použití**

py .\main.py --convert .\content\media\out\1bit.wav ./content/media/out/1bit.bmp -s



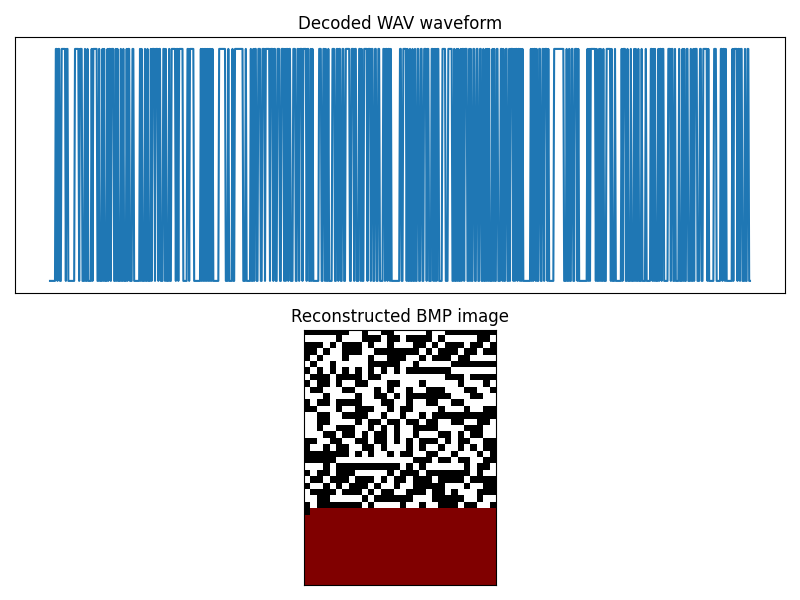
Obrázek 10 WAV formát převeden BMP – bez rozměrů

py .\main.py --convert .\content\media\out\1bit.wav ./content/media/out/1bit.bmp -s -d 30 20



Obrázek 11 WAV formát převeden BMP – s rozměry (30 x 20)

py .\main.py --convert .\content\media\out\1bit.wav ./content/media/out/1bit.bmp -s -d 30 40



Obrázek 12 WAV formát převeden BMP – s rozměry (30 x 40)

**Interní průběh**

1. Načtení WAV a ověření formátu (mono, 32bit, 44 100 Hz).
2. Získání délky datového segmentu a rozměrů obrázku (z hlavičky WAV nebo vstupu).
3. Pokud WAV obsahuje méně dat než je potřeba pro celý obrázek, automaticky se doplní nulovými vzorky (s varováním).
4. Každý 32bitový vzorek se dekomprimuje zpět do čtveřice RGBA hodnot.
5. Vytvoření BMP obrázku a uložení.

**Ošetření chyb**

* Pokud WAV nemá dostatek dat → doplnění nulami a varování uživateli.
* Pokud WAV nemá uložené rozměry a nejsou zadány → chyba.

**Využití**

* Rekonstrukce skrytých obrázků.
* Ověření bezeztrátového převodu v testovacím režimu.
* Možnost upravit zvukový soubor a následně vidět změny v obraze.

## Zobrazování dat souborů

Tato funkce aplikace umožňuje vizualizaci dat jak z WAV, tak z BMP souborů, což usnadňuje pochopení struktury a obsahu souborů, a zároveň slouží jako diagnostický nástroj pro kontrolu správnosti konverze.

**Vlastnosti**

* **Zobrazení zvukových dat**: Umožňuje vizualizovat vlnovou formu zvuku z WAV souboru.
* **Zobrazení obrazových dat**: Po konverzi WAV do BMP lze zobrazit výsledný obrázek.
* **Dynamické přepínání mezi režimy**: Zobrazení pouze zvukových dat nebo pouze obrazových.
* **Označení os (volitelné)**: Možnost vypnout/zapnout zobrazení os pro lepší přehlednost.

**Parametry příkazové řádky**

py ./main.py -i <soubor.wav nebo soubor.bmp> [--show -s] [--show-with-axes -sx] [--test]

**Volitelné přepínače**

* -s zobrazí soubor bez os po jeho načtení.
* -sx zobrazí soubor s osami po jeho načtení.
* --test uloží výstupní graf místo jeho přímého zobrazení.

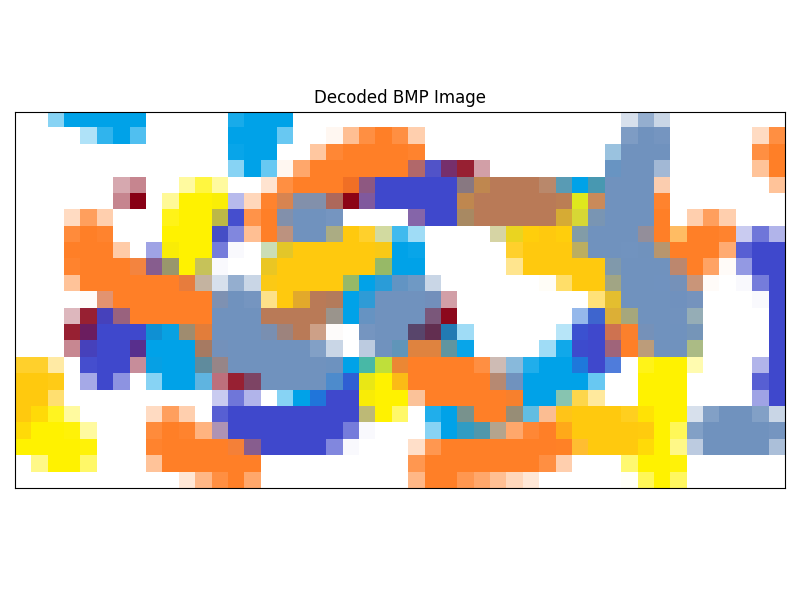
**Příklad použití**

py .\main.py -i .\content\media\bmp\24bit.bmp -s

A screenshot of a computer

AI-generated content may be incorrect.

Obrázek 13 Data o souboru 24bit.bmp



Obrázek 14 Reprodukovaný obrázek z dat

py .\main.py -c .\content\media\bmp\24bit.bmp .\content\media\out\wave.wav

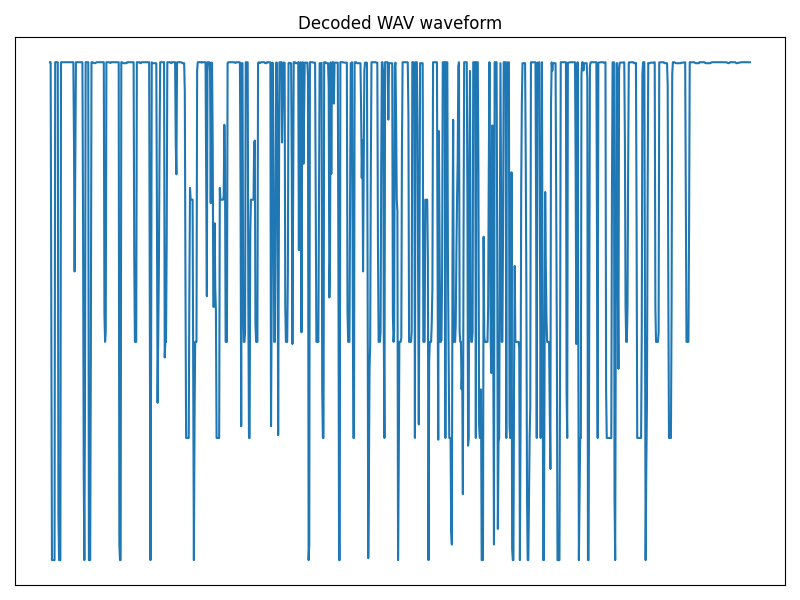
py .\main.py -i .\content\media\out\wave.wav -s

Pomocí těchto dvou příkazů se převedl 24bitový BMP (obrázek 12 a 13) na WAV a ten následně zobrazil (obrázek 14 a 15)

A screen shot of a computer

AI-generated content may be incorrect.

Obrázek 15 Data zvukové vlny wave.wav



Obrázek 16 Zobrazený WAV soubor wave.wav

**Interní průběh**

1. **Načtení souboru**: Aplikace zjistí formát souboru (WAV nebo BMP).
2. **Zobrazení zvukových dat**: Pokud je soubor ve formátu WAV, načtou se vzorky zvuku a zobrazí se jejich amplitudy ve formě grafu.
3. **Zobrazení obrázku**: Pokud je soubor BMP, aplikace načte pixelová data a zobrazí je jako obraz.

**Ošetření chyb**

* Pokud soubor není v podporovaném formátu (WAV nebo BMP), aplikace vyhodí chybu a uživateli se zobrazí informace o neplatném formátu.
* Při zobrazování více datových vrstev (zvuk + obraz) se provádí kontrola, zda jsou všechny potřebné komponenty (např. obrazová data v WAV) přítomny.

**Použití**

* **Vizualizace WAV souboru** pro analýzu zvukových vzorců.
* **Kontrola kvality konverze BMP na WAV** a naopak.
* **Testování funkčnosti převodníku** při generování náhodných nebo testovacích souborů.

# seznam obrázků

[Obrázek 1 BMP formát[1] 6](#_Toc196954868)

[Obrázek 2 Příklad BMP formátu [2] 7](#_Toc196954869)

[Obrázek 3 Výpočet ScanLine [1] 7](#_Toc196954870)

[Obrázek 4 WAV formát [3] 9](#_Toc196954871)

[Obrázek 5 Generace náhodného obrázku 11](#_Toc196954872)

[Obrázek 6 Generace náhodné vlny 12](#_Toc196954873)

[Obrázek 7 Generace sinusového signálu 14](#_Toc196954874)

[Obrázek 8 1bitový BMP obrázek převeden na zvukovou vlnu 15](#_Toc196954875)

[Obrázek 9 WAV formát převeden BMP – bez rozměrů 17](#_Toc196954876)

[Obrázek 10 WAV formát převeden BMP – s rozměry (30 x 20) 17](#_Toc196954877)

[Obrázek 11 WAV formát převeden BMP – s rozměry (30 x 40) 18](#_Toc196954878)

[Obrázek 12 Data o souboru 24bit.bmp 20](#_Toc196954879)

[Obrázek 13 Reprodukovaný obrázek z dat 20](#_Toc196954880)

[Obrázek 14 Data zvukové vlny wave.wav 21](#_Toc196954881)

[Obrázek 15 Zobrazený WAV soubor wave.wav 21](#_Toc196954882)

# Seznam Tabulek

[Tabulka 1 Formát BMP 5](#_Toc196954953)

[Tabulka 2 Formát WAV 8](#_Toc196954954)

# Seznam použité literatury

1. Wikipedia contributors. BMP file format [online]. Wikipedia, The Free Encyclopedia, 21. září 2003 [cit. 1. května 2025]. Dostupné z: https://en.wikipedia.org/wiki/BMP\_file\_formatUser13974146. Initializing a BMP file in MIPS [online]. Stack Overflow, 13. května 2021 [cit. 1. května 2025]. Dostupné z: https://stackoverflow.com/questions/67523940/initializing-a-bmp-file-in-mips
2. Soundfile.sapp.org. WAVE PCM soundfile format [online]. [cit. 30. 4. 2025]. Dostupné z: <http://soundfile.sapp.org/doc/WaveFormat/>