DOI: 10.59647/978-617-520-936-3/1

Автоматизований інструмент для перетворення фотографій або ескізів у піксель-арт

Анастасія Суткова, здобувач вищої освіти[1] (ORCID: 0009-0004-0649-9604)
Олена Доля, доцент, доцент кафедри IT [1] (ORCID: 0000-0002-8320-4038)

[1] *Київський національний університет будівництва і архітектури, проспект Повітряних сил, 31, м.Київ, Україна*

АНОТАЦІЯ

У даній роботі розглядаються ключові аспекти розробки інструменту для перетворення фотографій або ескізів у піксель-арт. Проаналізовано, як впровадження такого цифрового рішення може покращити ефективність творчого процесу та розширити можливості художників і дизайнерів. Розглянуто ефективні підходи до створення алгоритмів автоматичного перетворення зображень у піксель-арт, а також інструментів для налаштування рівня деталізації та колірних палітр.

Ключові слова: перетворення зображення, піксель-арт, автоматизований інструмент.

1. ВСТУП

У сучасному цифровому світі, автоматизація творчих процесів стала важливим інструментом для художників і дизайнерів. Одним із таких інструментів є система для перетворення фотографій або ескізів у піксель-арт. Цей стиль, популярний у відеоіграх і ретро-мистецтві, та в реалізації вимагає значної рутинної ручної праці, що у свою чергу займає нераціонально багато часу. Тож, створення автоматизованого інструменту для конвертації зображень дозволить прискорити процес створення піксельного малюнку, зберігаючи його якість та загалом художню цінність.

2. АВТОМАТИЗАЦІЯ ПРОЦЕСУ СТВОРЕННЯ ПІКСЕЛЬ-АРТУ

Автоматизація процесу створення піксель-арту значно спрощує перетворення складних зображень у стилізовану ретро-графіку, що зменшує трудомісткість та час на виконання завдань. Для цього використовуються алгоритми, які аналізують фотографії або ескізи, спрощують їх структуру та адаптують під обмежену палітру кольорів і низьку роздільну здатність, характерну для піксель-арту. Такий підхід дозволяє не лише художникам та розробникам ігор швидко створювати контент, а й сприяє стандартизації процесу, підвищуючи загальну якість графіки [1]. Автоматичні інструменти можуть пропонувати налаштування рівня деталізації, що робить їх універсальними як для професіоналів, так і для початківців, дозволяючи їм легко створювати піксельні зображення без глибоких знань або досвіду в ручній графіці [2].

3. ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ

Програмна реалізація інструменту для перетворення зображень у піксель-арт вимагає ретельного вибору технологій та методів. Важливо забезпечити ефективність алгоритмів, зручність інтерфейсу та надійність роботи на різних платформах

3.1. Вибір технологічного стека

Для серверної частини доцільно використовувати платформу Node.js з бібліотекою Jimp або Sharp для обробки зображень. Node.js є популярною платформою для розробки серверних застосунків, оскільки вона дозволяє використовувати JavaScript як на клієнті, так і на сервері. Бібліотека Jimp [3] надає функції для зменшення роздільної здатності зображень, зміни палітри кольорів та інших операцій, необхідних для пікселізації. Sharp [4] може бути більш продуктивним вибором для роботи з великими зображеннями, оскільки вона оптимізована для швидшої обробки графічних файлів. Також можна інтегрувати бібліотеку OpenCV.js, яка забезпечує додаткові можливості для складніших графічних маніпуляцій.

Для бази даних можна використати MongoDB, оскільки цей інструмент є документно-орієнтованою NoSQL базою, яка дозволяє зберігати та працювати з великими обсягами даних у вигляді JSON-подібних документів. Вона добре підходить для зберігання інформації про проекти користувачів, файли та налаштування зображень. MongoDB легко масштабується та інтегрується з іншими компонентами стеку.

Для клієнтської частини варто використовувати React.js або Vue.js — обидві ці JavaScript-бібліотеки підходять для створення динамічних інтерфейсів користувача. React.js забезпечує компонентну архітектуру, що полегшує створення багатофункціональних інтерфейсів для взаємодії з користувачем, таких як налаштування зображень перед конвертацією [5]. Vue.js, зі своєю простотою та гнучкістю, дозволяє швидко будувати інтерактивні елементи для управління процесом перетворення [6].

Обидві бібліотеки використовують віртуальний DOM, що значно підвищує продуктивність і швидкість роботи інтерфейсу, дозволяючи швидко оновлювати дані в режимі реального часу. Для налаштування інтерактивних компонентів користувач може використовувати слайдери та інші елементи керування для зміни розміру пікселів, вибору палітри кольорів тощо.

3.2. Розробка компонентів серверної частини

Серверна частина відповідає за обробку зображень, збереження проектів користувачів та інтеграцію з хмарними сховищами.Основні компоненти включають:

- *Завантаження та обробка зображень.* Сервер повинен мати функціонал для обробки зображень, який включає зменшення роздільної здатності для пікселізації та застосування фільтрів для налаштування кольорової гами. Цей процес виконується за допомогою бібліотеки Pillow або OpenCV, що дозволяє проводити обробку зображень на сервері.

- *Зберігання файлів та проектів.* Після обробки зображення та його конвертації в піксель-арт, користувач може зберігати свої роботи на сервері. Для цього використовуються бази даних, такі як MongoDB, що дозволяє зберігати як вихідні зображення, так і налаштування для подальшого редагування. Крім того, можна використовувати хмарні сховища, такі як Amazon S3, для зберігання великих файлів.
- *Інтегрованість з іншими сервісами.* Наприклад, інтеграція з хмарними сервісами для збереження даних і з платформами для спільного редагування дозволить розширити можливості інструменту.

3.3. Розробка компонентів клієнтської частини

Клієнтська частина — це інтерфейс, з яким взаємодіє користувач для завантаження зображень, налаштування параметрів пікселізації та перегляду результатів. Основні функції інтерфейсу:

- Попередній перегляд результату. Важливо забезпечити функцію попереднього перегляду, щоб користувач міг побачити, як виглядатиме кінцеве зображення після застосування фільтрів та налаштувань.
- Можливість збереження та експорту. Після завершення роботи користувач може зберегти результати на свій комп'ютер або поділитися ними через соціальні мережі чи інші платформи.
- Завантаження зображень та налаштування параметрів. Користувач повинен мати можливість завантажити зображення у форматах JPEG, PNG, GIF. Інтерфейс має включати інструменти для вибору розміру пікселів, кількості кольорів у палітрі та інших параметрів, таких як різкість або контрастність.

3.4. Інтеграція сторонніх сервісів

Для покращення функціональності інструменту важливо забезпечити інтеграцію з такими сторонніми сервісами, цього можна досягти застосовуючи хмарні сервіси для зберігання та синхронізації проектів, наприклад, Google Drive або Dropbox, щоб користувачі могли зберігати свої файли безпосередньо у хмарі та мати доступ до них з будь-якого пристрою. Також для подальшого редагування результатів доцільно буде використовувати API графічних редакторів, таких як Adobe Photoshop або GIMP. Інтеграція з соціальними мережами дозволить користувачам ділитися своїми роботами безпосередньо з інтерфейсу програми.

3.5. Хостинг

Для стабільної роботи інструменту важливо вибрати відповідне середовище для хостингу. Використання хмарних платформ, таких як Amazon Web Services (AWS) [7], дозволить автоматично масштабувати ресурси в залежності від навантаження на систему. Це особливо важливо, оскільки піксель-арт може бути затребуваним серед великої аудиторії.

Хмарні рішення забезпечують також управління базами даних і зберігання файлів користувачів, що дозволяє налаштувати автоматичне резервне копіювання даних і швидке відновлення у випадку збоїв. Використання CDN-сервісів (Cloudflare або Fastly) дозволить прискорити завантаження веб-додатку для користувачів з різних регіонів.

3.6. Безпека та резервне копіювання

З огляду на те, що інструмент працює з персональними зображеннями користувачів, важливо забезпечити високий рівень безпеки. Це включає шифрування даних за допомогою SSL-сертифікатів, впровадження безпечної аутентифікації через OAuth або JWT, а також захист від атак, таких як SQL-ін'єкції та XSS.

Крім того, регулярне резервне копіювання даних є ключовим для збереження файлів користувачів. Використання хмарних рішень для автоматичного резервування дозволяє уникнути загрози втрати даних.

3.7. Тестування та підтримка

Перед випуском інструменту необхідно провести ретельне тестування його роботи. Це включає перевірку функціонування алгоритмів обробки зображень, зручність інтерфейсу користувача та продуктивність системи під високим навантаженням. Після запуску важливо забезпечити регулярне оновлення та додавання нових функцій на основі відгуків користувачів. та нових технологічних вимог.

4. ВИСНОВКИ

Розробка автоматизованого інструменту для перетворення фотографій або ескізів у піксель-арт є важливим кроком у спрощенні творчого процесу. Вибір правильного технологічного стека, розробка ефективних алгоритмів та забезпечення зручного інтерфейсу дозволять створити інструмент, який буде корисним для художників та дизайнерів. Такий підхід допоможе зробити процес створення піксельної графіки швидким і доступним, зберігаючи при цьому високу якість результату та сприяючи створенню нових видів цифрового мистецтва.

Список літератури

[1] CADVision “Latest trends in Design Process Automation”. URL: https://cadvisionengineers.com/latest-trends-in-design-process-automation/.

[2] Pixelixe “Boosting Creativity - How Graphic Automation Frees Up Time for Designers”. URL: https://pixelixe.com/blog/how-graphic-automation-frees-up-time-for-designers/.

[3] FileFormat “Комплексні операції обробки зображень через JavaScript API ”. URL: https://products.fileformat.com/uk/image/javascript/jimp/

[4] DigitalOcean “How To Process Images in Node.js with Sharp”. URL: https://www.digitalocean.com/community/tutorials/how-to-process-images-in-node-js-with-sharp

[5] React “Посібник: знайомство з React”. URL: https://uk.legacy.reactjs.org/tutorial/tutorial.html

[6] Vuejs.de “Introduction”. URL: https://vuejs.org/guide/introduction.html

[7] Spot “AWS Auto Scaling: Scaling EC2, ECS, RDS, and more”. URL: https://spot.io/resources/aws-autoscaling/scaling-ec2-ecs-rds-and-more/

.