

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА
АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

**МЕТОДИ РОБОТИ З ТЕКСТУРАМИ ЗА ДОПОМОГОЮ ЗАСОБІВ
ОБРОБКИ ЗОБРАЖЕНЬ НА МОВІ PYTHON**

**Текстова частина до курсової роботи за спеціальністю
„Інженерія програмного забезпечення” 121**

Керівник курсової роботи

ас. Корнійчук М. А.

(підпис)

“ ____ ” _____ 2020 р.

Виконав студент ФІ-3

Гамаюн Д. В.

“ ____ ” _____ 2020 р.

Київ 2020

Зміст

Анотація.....	7
ВСТУП.....	8
Розділ 1. Вибір технологій, які найкраще підходять для обраної задачі.	11
1.1 Аналіз наявних технологій та методів програмної обробки зображень.	11
1.1.1 Аналіз бібліотеки OpenCV та її можливостей	11
1.1.2 Аналіз можливостей Google Cloud Vision API	12
1.1.3 Аналіз бібліотеки scikit-image та її можливостей	14
1.2 Підсумки аналізу технологій. Вибір фінальної бібліотеки	15
Розділ 2. Реалізація алгоритму для вирішення основної задачі	16
2.1 Основна задача	16
2.2 Реалізація алгоритму	18
2.2.1 Опис логіки алгоритму.....	18
2.2.2 Тестування кожного кроку алгоритму	20
2.2.3 Тестування фінального алгоритму на різних типах одягу	22
2.3 Тестування накладання результуючих зображень на 3D моделі.....	24
2.4 Висновки	26
Розділ 3. Додавання алгоритму до фінальної програми, тестування цілісної системи	27
3.1 Концепція і основна логіка програми	27
3.1.1 Кінцевий вигляд програми.....	27
3.1.2 Ціль проєкту	27
3.1.3 Сценарій використання	27

3.1.4 Логіка роботи.....	28
3.1.5 Обмеження.....	30
3.2 Функціональні вимоги	31
3.2.1 Вхід в систему	31
3.2.2 Додавання нового елементу одягу	31
3.2.3 Редагування попередньо доданих елементів	32
3.3 Вибір технологій для фінального проєкту	34
3.3.1 Вибір технологій для реалізації бібліотеки.....	34
3.3.1.1 Front-end частина бібліотеки.....	34
3.3.1.2 Back-end частина бібліотеки	35
3.3.2 Вибір технологій для реалізації демо-версії Інтернет-магазину.....	35
3.3.2.1 Front-end частина демо-версії сайту	36
3.3.2.2 Back-end частина демо-версії сайту	36
3.3.2.3 База даних	37
3.4 Демонстрація можливостей фінального проєкту	38
3.4.1 Сторінка вибору комплекту одягу.....	38
3.4.2 Сторінка перегляду комплекту одягу.....	38
3.4.3 Сторінка адміністратора	40
3.4.4 Результат	40
Висновки.....	41
Використані джерела	42

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ

Олександр Петрович,
Зав.кафедри мультимедійних систем,
кандидат ф.-м.н., доцент

_____ О. О. Жежерун

(підпис)

“ ” _____ 2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студенту 3 курсу факультету інформатики Гамаюну Давіду Валерійовичу

ТЕМА: Методи роботи з текстурами за допомогою засобів обробки зображень на мові Python.

Вихідні дані:

- Дослідження можливостей OpenCV в рамках створення алгоритму для перетворення фотографії одягу в частину текстури
- Дослідження створення javascript-бібліотеки та інтеграції бібліотеки в сайт
- Створення сервісу примірки одягу на 3д моделях в браузері для демонстрації роботи створеної бібліотеки

Зміст ТЧ до курсової роботи:

- Анотація
- Вступ
- Розділ 1. Вибір технологій, які найкраще підходять для обраної задачі
- Розділ 2. Реалізація алгоритму для вирішення основної задачі
- Розділ 3. Додавання алгоритму до фінальної програми, тестування цілісної системи
- Висновки
- Список використаних джерел

Дата видачі “ ____ ” _____ 2020 р.

Керівник _____
(підпис)

Завдання отримав _____
(підпис)

Календарний план виконання роботи

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	02.10.2019	
2.	Огляд технічної літератури за темою роботи.	01.11.2019	
3.	Дослідити та проаналізувати наявні технології та фреймворки, які б допомогли при розробці бібліотеки	23.12.2019	
4.	Написання front-end частини бібліотеки	10.02.2020	
5.	Написання back-end частини бібліотеки	10.02.2020	
6.	Аналіз бібліотек та фреймворків для написання сервісу для примірки одягу	29.02.2020	
7.	Написання сервісу для примірки одягу	31.03.2020	
8.	Написання курсової роботи	30.04.2020	
10.	Аналіз виконаної роботи з керівником.	11.05.2020	

Студент Гамаюн Д.В.

Керівник Корнійчук М. А.

“ _____ ”

Анотація

Мета курсової роботи – аналіз наявних підходів і алгоритмів обробки зображень та подальшого їх використання для обробки фото одягу, перетворення їх у фото текстури для 3D моделей. Ця курсова робота являє собою розробку back-end частини для онлайн сервісу примірки одягу на 3D моделях у браузері, глобальним результатом якої, буде тестова версія цього сервісу.

ВСТУП

Глобальна ціль фінального продукту, частиною якого стане ця курсова робота, це зробити інтернет шопінг якомога зручнішим і комфортнішим, звести потребу в очній примірці одягу та інших видах шопінгу до мінімуму.

З кожним роком усе більше і більше людей купують найрізноманітніші речі онлайн і, звісно, одна з базових потреб людини це одяг. Але через те, що кожна людина відрізняється одна від одної, як габаритами (зростом, вагою, довжиною талії тощо), так і вподобаннями, то процес покупки саме одягу, через інтернет, стає таким важким і, інколи, неприємним. Вирішивши основні проблеми, які виникають у людей при покупці одягу онлайн, можна зробити його ще більш популярним.

Є декілька основних проблем при покупці одягу онлайн:

1. Людина не знає, як саме на ньому буде виглядати та, чи інша річ.
2. Людина, не впевнена чи підходить їй той, чи інший розмір.
3. Людина хоче візуально бачити, чи буде певна річ підходити до іншого одягу з її гардеробу. (Цю проблему не завжди вирішує, навіть, очна примірка одягу)

На початковому етапі саме останню проблему і буде вирішувати фінальний продукт.

Для розв'язання цієї проблеми потрібно надати змогу кінцевим користувачам «одягати» 3D модель в браузері як в одяг, що продається на сайті (теоретичний сайт одягу, до якого буде підключена фінальна система), так і завантажувати свої фотографії одягу, які переробляться на текстуру для

моделі, що дасть змогу візуалізувати те, як одяг користувач буде виглядати разом з речами, які продаються на сайті.

Мета і завдання дослідження: Створити метод обробки зображень, який буде перероблювати просту фотографію будь-якого одягу на частину текстури для 3D моделі.

Об'єкт дослідження: Засоби та методи обробки зображень, пошук потрібної інформації на зображенні, видалення зайвого з зображення.

Методи дослідження: Аналіз наявних методів обробки зображень, аналіз та обробка вже наявних підходів до розв'язання подібної проблеми та подальша перевірка впроваджених методів.

Джерела дослідження: Через те, що для вирішення поставленого питання була обрана одна з найпопулярніших бібліотек для обробки зображень OpenCV (обґрунтування вибору буде нижче), то за основу теоретичних знань було взято книги: «Learning OpenCV» авторства Gary Bradski і Adrian Kaehler, та «Mastering OpenCV 4 with Python: A practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7» авторства Alberto Fernandez Villan. Також для вивчення і аналізу більш практичних можливостей технології OpenCV було взято сайт [1]

Практичне значення одержаних результатів: Було розроблено метод перетворення фотографії одягу в частину текстури для 3D моделі, а саме: виокремлення потрібної інформації з фотографії (самого одягу), видалення фону, та зайвих частин фотографії, доведення фотографії до певних розмірів, які потрібні для текстури. Використання цього методу може бути найрізноманітніше. Але з самого початку даний метод розроблявся для

роботи з фотографіями одягу і надання кінцевим користувачам можливості одягати 3D модель в браузері у будь-який одяг.

Робота складається з трьох розділів:

Перший розділ присвячено аналізу наявних методів обробки зображень.

Розглянуто наявні бібліотеки та мови програмування за допомогою яких можливо реалізувати завдання, подібні до поставленого. Із всієї інформації зроблено висновки та обрані фінальні технології для реалізації.

В другому розділі написано власний варіант розв'язання поставленої задачі.

Проаналізовано роботу написаного методу на справжніх фотографіях з подальшим накладанням результуючих текстур на 3D модель.

В третьому розділі написаний метод впроваджується в фінальну програму і тестується робота всієї системи разом.

Створено програмний продукт, який використовується для візуалізації одягу на 3D моделях в браузері.

Постановка задачі:

1. Проаналізувати наявні технології та методи для програмної обробки зображень, обрати технології, які найкраще підходять для вирішення основної задачі.
2. Розробити власний метод, який розв'язує основну задачу за допомогою технологій, обраних в пункті 1. Тестування методу.
3. Додавання отриманого методу до фінальної програми, тестування цілісної системи.

Розділ 1. Вибір технологій, які найкраще підходять для обраної задачі.

1.1 Аналіз наявних технологій та методів програмної обробки зображень.

Найпопулярнішою технологією для вирішення будь-яких завдань обробки зображень є OpenCV, тому наступний аналіз буде будуватися на порівнянні інших технологій з OpenCV.

1.1.1 Аналіз бібліотеки OpenCV та її можливостей

Згідно з інформацією з офіційного сайту [2] OpenCV (Open Source Computer Vision Library) - це бібліотека програмного забезпечення для комп'ютерного зору і машинного навчання з відкритим вихідним кодом. OpenCV був створений для забезпечення загальної інфраструктури для додатків комп'ютерного зору і для прискорення використання машинного сприйняття в комерційних продуктах. Бібліотека має понад 2500 оптимізованих алгоритмів, які включають повний набір як класичних, так і найсучасніших алгоритмів комп'ютерного зору і машинного навчання. Ці алгоритми можуть використовуватися для виявлення і розпізнавання осіб, ідентифікації об'єктів (а саме це і потрібно використовувати, для вирішення поставленої задачі), класифікації дій людини в відео, відстеження рухів камери, відстеження рухомих об'єктів і багато чого іншого. Розгорнуті області застосування OpenCV охоплюють діапазон: від зшивання зображень вулиць до виявлення вторгнень в відеоспостереження в Ізраїлі, спостереження за шахтним

обладнанням в Китаї, допомоги роботам в навігації і підборі об'єктів в Willow Garage, виявлення аварій в результаті утоплення в плавальному басейні в Європі, виконання інтерактивного мистецтва в Іспанія і Нью-Йорк перевіряють злітно-посадочні смуги на наявність сміття в Туреччині, перевіряють етикетки на продуктах на заводах по всьому світу на предмет швидкого виявлення осіб в Японії. Він має інтерфейси C ++, Python, Java і MATLAB і підтримує Windows, Linux, Android і Mac OS. OpenCV в основному орієнтований на додатки для візуалізації в реальному часі і використовує інструкції MMX і SSE, коли вони доступні. Повнофункціональні інтерфейси CUDA і OpenCL зараз активно розвиваються. Існує понад 500 алгоритмів і приблизно в 10 разів більше функцій, які становлять або підтримують ці алгоритми. OpenCV спочатку написаний на C ++ і має шаблонний інтерфейс, який без проблем працює з контейнерами STL.

Бібліотека OpenCV написана на мові C++, тому і основний інтерфейс реалізован на цій же мові, але не дивлячись на це, технологію, тим чи іншим способом, можна використовувати на багатьох найпопулярніших мовах програмування, таких як: Python, JS, Java, C#, Ruby тощо.

1.1.2 Аналіз можливостей Google Cloud Vision API

Основним конкурентом OpenCV у роботі з обробкою і аналізом зображень є Google Cloud Vision API.

Cloud Vision API - це простий у використанні REST API, який використовує операції HTTP POST для аналізу даних на зображеннях, що відправляються вами в запиті. API використовує JSON як для запитів, так і для відповідей.

Типовий запит JSON Vision API включає в себе вміст зображень, для яких необхідно виконати виявлення, і набір операцій (які називаються функціями) для виконання з кожним зображенням.

Основні можливості API.

FEATURE	PRICE PER 1,000 UNITS, BY MONTHLY USAGE			
	1 - 1,000 UNITS/MONTH	1,001 - 1 MILLION UNITS/MONTH	1,000,001 - 5 MILLION UNITS/MONTH	5,000,001 - 20 MILLION UNITS/MONTH
Label Detection	Free	\$5.00	\$4.00	\$2.00
OCR	Free	\$2.50	\$1.50	\$0.60
Explicit Content Detection	Free	\$2.50	\$1.50	\$0.60
Facial Detection	Free	\$2.50	\$1.50	\$0.60
Landmark Detection	Free	\$2.50	\$1.50	\$0.60
Logo Detection	Free	\$2.50	\$1.50	\$0.60
Image Properties	Free	\$2.50	\$1.50	\$0.60

Скріншот с списком возможностей API з офіційного сайту

Label Detection - аналіз класу до якого належить зображення, аналіз того, що зображено на ньому: коти, собаки, їжа і т.д.

OCR - розпізнавання тексту

Explicit Content Detection – аналіз небажаного контенту на зображенні

Facial Detection – аналіз осіб, рис осіб, особливих точок на обличчях

Landmark Detection - аналіз геолокації за фотографією

Logo Detection - аналіз символів і значків

Через те, що інформація обробляється на серверах Google, то великі обсяги зображень, а саме понад одну тисячу зображень за місяць, вже виходить за рамки безплатних можливостей.

Зважаючи на це і проаналізувавши можливості Google Cloud Vision API на офіційному сайті [3] було зроблено висновки, що ця технологія не зовсім підходить до поставленої задачі.

1.1.3 Аналіз бібліотеки scikit-image та її можливостей

Згідно з інформації на офіційному сайті [4] : scikit-image - це бібліотека обробки зображень з відкритим вихідним кодом для мови програмування Python. Вона включає в себе алгоритми сегментації, геометричних перетворень, управління колірним простором, аналізу, фільтрації, морфології, виявлення ознак і т.д. Він призначений для взаємодії з числовими бібліотеками Python NumPy і SciPy.

Scikit-image являє собою дуже схожу бібліотеку на проаналізовану вище OpenCV, але з набагато меншим обсягом можливостей і тільки для однієї мови програмування, а саме – Python.

Згідно з порівняльним аналізом технологій комп'ютерного зору на сайті [5], користувачі зазначили лише декілька пунктів, за якими бібліотека scikit-image, трохи краща за OpenCV, і це – простота початкових налаштувань і легкість подальшого використання. Ці пункти мають право на існування тільки через те, що OpenCV, як вище було зазначено, може використовуватись багатьма різними мовами програмування, на деяких налаштувати легше, на деяких

складніше, з використанням така ж справа. Для більш менш об'єктивного порівняльного аналізу треба брати до уваги лише OpenCV для мови програмування Python, тому що тільки на цій мові працює бібліотека scikit-image.

Основний список можливостей і способів використання бібліотеки також описано на офіційному сайті [6]:

Порівнявши цей список можливостей і методи їх застосування на практиці з можливостями OpenCV, які описані вище, було зроблено висновок, що ці бібліотеки мають дуже схожий функціонал і реалізують його майже однаково.

В ході порівняльного аналізу і пошуку інформацію про обидві бібліотеки було зроблено основний висновок: бібліотеки дуже схожі по всім параметрам, але у бібліотеки OpenCV є велика перевага – це популярність серед користувачів. Про можливості і методи їх використання OpenCV є дуже багато інформації, написано декілька книжок і гарна офіційна документація, чого не скажеш про scikit-image.

1.2 Підсумки аналізу технологій. Вибір фінальної бібліотеки

Проаналізувавши три основні технології, які використовуються для комп'ютерного зору та обробки зображень, було зроблено такі висновки. Бібліотека OpenCV є повністю безплатна, підтримується багатьма мовами програмування, дуже гарно продокументована і має великий список можливостей, які включають в себе саме те, що потрібно для поставленої задачі.

Через вищезазначені причини було зроблено вибір в сторону бібліотеки OpenCV, саме ця технологія, для реалізації основної задачі, і буде використовуватись далі.

Всю back-end частину фінального продукту було вирішено писати на мові Python, тому відповідно буде краще, якщо основна задача back-end частини, як і базові, буде теж на мові Python.

Розділ 2. Реалізація алгоритму для вирішення основної задачі

2.1 Основна задача

Ще раз опишемо основну задачу, яку потрібно реалізувати:

На вхід програмі подається зображення однієї сторони одягу, як на прикладі нижче:



[7]

*+6-

На виході програма має зробити перетворення фотографії одягу в частину текстури для 3D моделі, а саме: виокремлення потрібної інформації з фотографії (самого одягу), видалення фону (відповідно фотографія має бути у форматі png, який підтримує прозорий фон), та зайвих частин фотографії, доведення фотографії до певних розмірів, які потрібні для текстури.

Тобто найголовнішою частиною є знаходження потрібного контенту на зображенні, а саме одягу, і видалення зайвого фону.

2.2 Реалізація алгоритму

2.2.1 Опис логіки алгоритму

Перше і найголовніше що потрібно зробити це – «Edge detection», тобто знайти контури основного елементу на зображенні, а саме одягу. Використаємо для цього стандартний алгоритм комп'ютерного зору Canny.

Алгоритм Кенні (детектор границь Кенні, оператор Кенні) в дисципліні комп'ютерного зору — оператор виділення границь зображення. Був розроблений Джоном Кенні [8]

Алгоритм виявлення границь Кенні можна розділити на 5 різних етапів:

- Застосовуйте фільтр Гаусса, щоб згладити зображення та зняти шум
- Знайдіть градієнти інтенсивності зображення
- Застосовуйте не максимальне придушення, щоб позбутися хибної реакції на виявлення ребер
- Застосовуйте подвійний поріг для визначення потенційних ребер
- Відстеження краю за допомогою гістерезису: Завершити виявлення країв, придушивши всі інші ребра, які слабкі та не з'єднані із сильними краями.

Розмиття зображення для видалення шуму. Оператор Кенні використовує Розмивання Гауса з $\alpha = 1.4$:

$$\mathbf{B} = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \mathbf{A}.$$

Пошук градієнтів. Границі відмічають там, де градієнт набуває найбільшого значення. Вони можуть мати різні напрямки, тому алгоритм Кенні використовує чотири фільтри для визначення горизонтальних, вертикальних і діагональних ребер в розмитому зображенні.

$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2}$$

$$\Theta = \arctg\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right).$$

Кут нахилу градієнту округлюється і може набувати значень 0, 45, 90, 135.

Пошук локальних максимумів (Non-Maximum Suppression)

Подвійна порогова фільтрація.

Алгоритм повніт описаний в роботі самого JOHN CANNY «A Computational Approach to Edge Detection» [9]

Наступним кроком буде виділення з результатів зображення, після обробки методом Кенні, самого контуру, з цим також допоможе стандартний метод обробки зображень. Метод приймає на вхід зображення, оброблене методом Кенні та виділити основну інформацію (наочні приклади нижче).

Після того як ми знайшли фінальний контур контенту, який хочемо залишити на зображенні, в нашому прикладі – елементу одягу, ми маємо створити так звану «маску» цього контура і видалити все з зображення, крім цієї самої маски.

Так ми і отримуємо зображення з видаленим фоном, останній крок залишився – це обрізання зайвого, та зведення зображень до одного розміру.

Алгоритм обрізання фотографії також використовує контур основного зображення і обрізає його до границь цього контуру.

2.2.2 Тестування кожного кроку алгоритму

Протестуємо всі кроки нашого алгоритму на наступному зображенні:



Перший крок: застосуємо алгоритм Canny edge detection:



Виділемо основний конутр і збережемо точки контуру в відсортований масив.
Створимо порожню маску, намалюємо на ній заповнений багатокутник,
відповідний найбільшому збереженому контуру:



Робимо маску більш гладкою і додамо трохи розмиття, це покращить
результат:



Залишаємо на зображенні лише контент, який знаходиться під маскою, а все
інше закрасимо в яскравий колір, для легшого видалення:



Видаляємо яскравий фон:



І останній крок, обрізаємо по контуру:



Це і є фінальний результат роботи алгоритму.

2.2.3 Тестування фінального алгоритму на різних типах одягу

Протестуємо алгоритм на різних типах одягу, в початковій версії бібліотеки було вирішено додати лише два елементи одягу, а саме: футболки і штани, але алгоритм буде працювати для будь-якого одягу.

Протестуємо на штанах:



Вихідний результат:



Вихідний результат повністю збігається з нашими очікуваннями.

Протестуємо на ще одній футболці.

Вхідна фотографія:



Результат:



Результат, такий, як і очікувалось, алгоритм працює коректно.

2.3 Тестування накладання результуючих зображень на 3D моделі

Приклад футболки, яку обрізали за розробленим алгоритмом:



Приклад того, як вона виглядає на 3D моделі футболки в середовищі розробки Blender:



Результат нас повністю задовільняє, алгоритм працює повністю коректно.

2.4 Висновки

Було написано і протестовано програму, яка повністю виконує і задовільняє поставлену задачу, за допомогою зазначених технологій. Наступним кроком буде впровадження цього алгоритму в фінальний проєкт і тестування цілісної системи.

Розділ 3. Додавання алгоритму до фінальної програми, тестування цілісної системи

3.1 Концепція і основна логіка програми

3.1.1 Кінцевий вигляд програми

Фінальна програма являє собою тестування написаної Javascript бібліотеки, яка реалізує примірku одягу на 3д моделях.

На практиці це мало би працювати наступним чином: користувач додає бібліотеку до себе на сайт, на певну сторінку додає головний Canvas з бібліотеки, який і надає всі основні можливості бібліотеки.

3.1.2 Ціль проєкту

Глобальна ціль фінального продукту, частиною якого стане ця курсова робота, це зробити інтернет шопінг якомога зручнішим і комфортнішим, звести потребу в очній примірці одягу і інших видах шопінгу до мінімуму.

3.1.3 Сценарій використання

Сценарій використання (Use Case) — у розробці програмного забезпечення та системному проєктуванні це опис поведінки системи, як вона відповідає на зовнішні запити, вказує на те «хто» і «що» може зробити з розглянутою

системою. Методика різновидів використання застосовується для виявлення вимог до поведінки системи, відомих також як функціональні вимоги. [10]

Користувач - людина, яка хоче купити певний одяг в інтернет магазині, але не впевнена чи будуть естетично гарно виглядати різні елементи одягу (з цього магазину) разом. Користувач має змогу приміряти на 3D манекені будь-який елемент одягу висталений на сайті адміном.

Можливий додатковий варіант:

Користувач - людина, яка хоче купити певний одяг в інтернет магазині, але не впевнена чи будуть естетично гарно виглядати елемент одягу з магазину **разом з одягом який в неї вже є.**

3.1.4 Логіка роботи

Що потрібно буде зробити працівникам інтернет магазину:

Для того, щоб програма коректно працювала при додаванні нового товару на сайт все що потрібно буде зробити це завантажити 2 додаткові фотографії цього елемента одягу спереду і ззаду відповідно. Одяг має лежати на рівній поверхні однотонного яскравого кольору. Приклад:



Після додавання фотографій на серверній частині з кожної фотографії буде видалятися фон і змінюватися розмір до стандартизованої певної величини, яка є оптимальною для текстури. Після всіх маніпуляцій зберігається вже оновлене зображення.

Як буде відбуватися відображення певного елемента одягу на 3D манекені:

Після того як користувач обрав певний елемент одягу, та хоче побачити, як він буде виглядати на манекені. З елемента одягу стягується наступна інформація:

1. Який це тип одягу (наприклад футболка), тоді з бази стягується відповідна 3D модель та “одягається” на манекен
2. Фотографії текстур, які накладаються на вже вдягнуту на манекен 3D модель.

Кожна 3D модель поділена на 2 частини (передню і задню, відповідно до текстур), кожна з яких має свою певну UV-розгортку, яка і описує те, як має накладатися текстура на об'єкт. Приклад (накладання обробленої фотографії на 3D модель в редакторі blender):



3.1.5 Обмеження

Треба зазначити декілька основних обмежень, дотримуючись яких програма має працювати коректно:

1. Фото одягу має бути якісним.
2. Фон на фото має бути однотонним та, бажано, яскравого кольору.
3. Елемент одягу, який фотографують має випрямлено лежати на рівній поверхні (наприклад підлозі).

3.2 Функціональні вимоги

Опишемо мінімальний набір функціональності фінальної програми, тобто видуманного сайту одягу, який використовує розроблену бібліотеку для примірки одягу на 3D манекені.

3.2.1 Вхід в систему

В систему можуть зайти два види користувачів:

1. Адмін, який має змогу редагувати старі, та додавати нові товари.
2. Звичайний користувач, який бачить речі, які додав адмін, може їх приміряти

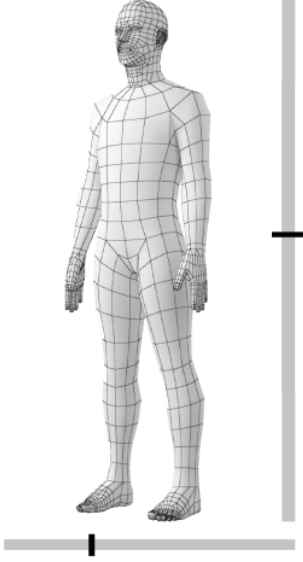
Для авторизації потрібно використати логін та пароль.

3.2.2 Додавання нового елементу одягу

Як було зазначено вище, адмін має змогу додавати нові елементи одягу. Для цього йому обов'язково потрібно буде вибрати тип одягу та завантажити дві фотографії передньої і задньої частини одягу відповідно. Нижче наведено приклад того, як приблизно буде виглядати інтерфейс цієї можливості:

Log In

UPLOAD NEW MODEL



MODEL

MODEL NAME

FRONT IMAGE

BACK IMAGE

DESCRIPTION

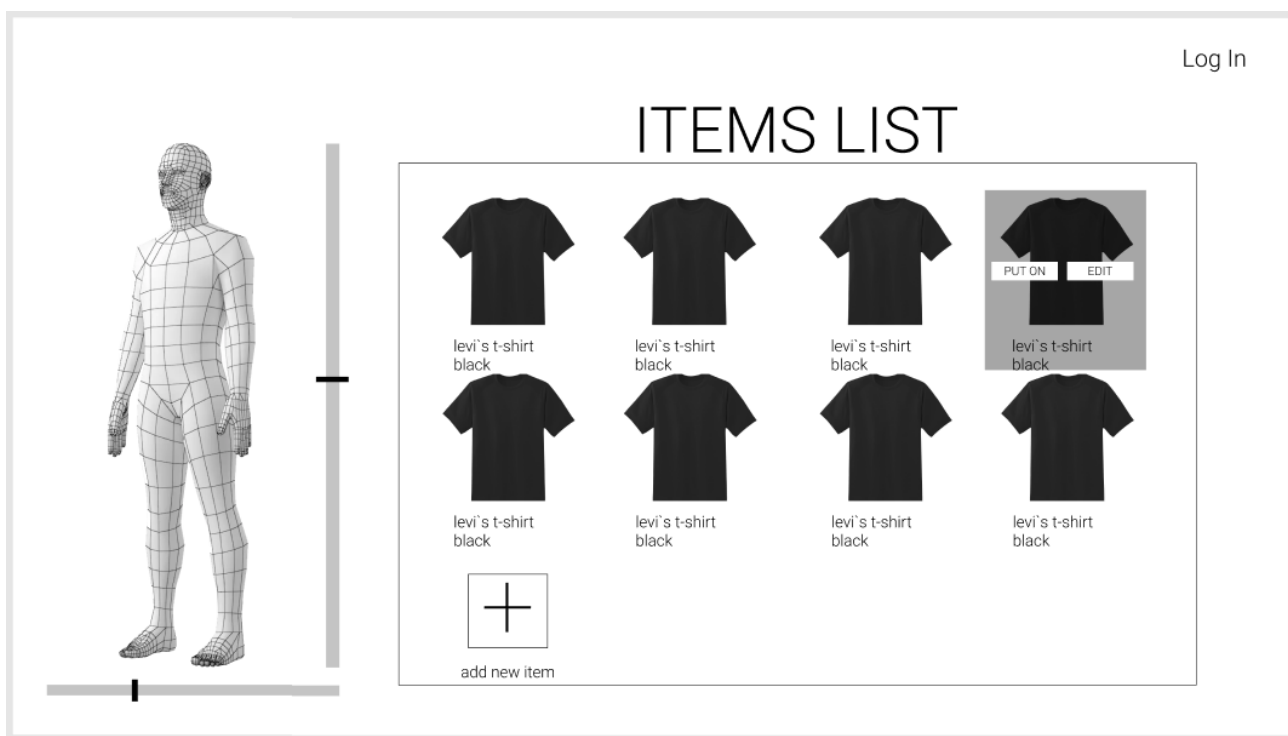
PUT ON

APPLY

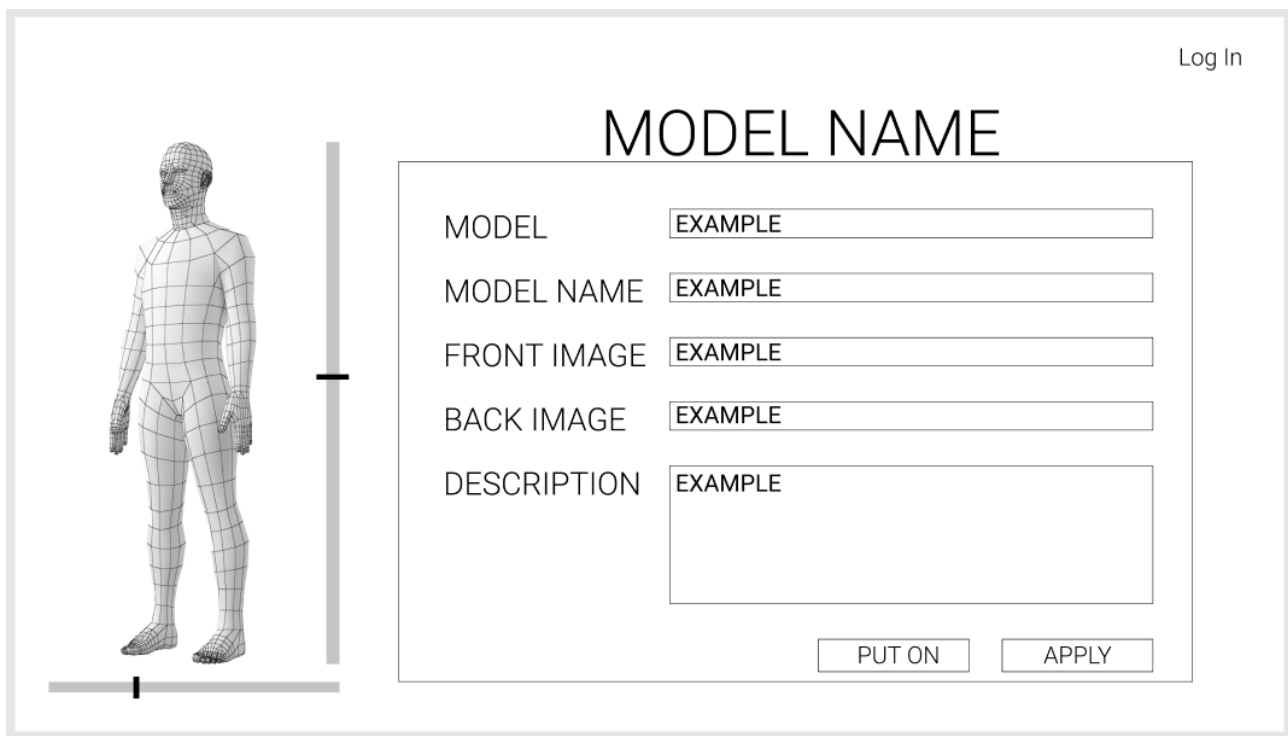
3.2.3 Редагування попередньо доданих елементів

Також адмін має змогу редагувати вже додані раніше товари. Можна редагувати будь яку інформацію про елемент одягу (назва, передня текстура, тип моделі і т.д.)

Нижче наведено приклад того, як приблизно буде виглядати інтерфейс цієї можливості:



(Приклад того як перейти до самого редагування моделі, кнопка «Edit»)



(Приклад самого процесу редагування)

3.3 Вибір технологій для фінального проєкту

Ціль фінального проєкту – продемонструвати приклад роботи бібліотеки на теоретичному інтернет магазині, тому це буде простий веб застосунок, який використовує розроблену бібліотеку.

Усі дані та сутності в веб-застосунку спрощені, оскільки його метою є лише демонстрація роботи бібліотеки.

3.3.1 Вибір технологій для реалізації бібліотеки

Основні задачі бібліотеки:

Front-end : відображати 3д моделі, з накладеними на них текстурами

Back-end: обробляти фотографію одягу та перетворювати її в текстуру

3.3.1.1 Front-end частина бібліотеки

Для реалізації 3д в браузері була обрана бібліотека Three.js

Three.js — це бібліотека JavaScript з кросбраузерністю та інтерфейсом прикладного програмування (API), що використовується для створення та відображення анімованої 3D-комп'ютерної графіки у веб-браузері. Three.js скрипти можуть використовуватися спільно з елементом HTML5 Canvas, SVG або WebGL. Вихідний код бібліотеки Three.js розміщений у сховищі на GitHub.

Three.js дозволяє створювати пришвидшену на GPU, 3D-анімацію, використовуючи мову JavaScript як частину веб-сайту, не покладаючись на власні плагіни браузера. Це можливо завдяки появі WebGL.

Three.js – проста і дуже добре задокументована бібліотека, яка як найкраще підходить для проєкту. [11]

3.3.1.2 Back-end частина бібліотеки

Back-end, частину було вирішено писати на python, разом з бібліотеками flask та OpenCV.

Flask – це WSGI фреймворк для веб-додатків, що призначений для швидкого та легкого початку роботи з можливістю масштабування до складних програм. Він розпочався як простий обгортка навколо Werkzeug та Jinja, але згодом став однією з найпопулярніших Python бібліотек. [12]

Flask – фреймворк для створення веб-застосунків на python. Він дозволяє створити API, за допомогою якого фронт-енд та бекенд обмінюються інформацією.

Бібліотеку OpenCV було проаналізовано і обгрунтовано вище.

3.3.2 Вибір технологій для реалізації демо-версії Інтернет-магазину

Уся робота демо-версії Інтернет-магазину спрощена, оскільки її метою є лише демонстрація роботи бібліотеки.

3.3.2.1 Front-end частина демо-версії сайту

Для реалізації front-end частини сайту був обраний фреймворк Angular.

Angular - це платформа для веб-розробок, яка надає розробникам надійні інструменти для створення клієнтської сторони веб-додатків за допомогою мови TypeScript. Випущений в 2010 році і раніше відомий як AngularJS, Angular - це JavaScript фреймворк, який використовується для створення односторінкових веб-додатків. [13]

Angular на час написання роботи вже була відомою технологією і сама по собі є простою в використанні, тому вибір пав на саме цю технологію.

3.3.2.2 Back-end частина демо-версії сайту

Для реалізації back-end частини сайту була обрана технологія Node.js.

Node.js — платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript.

Node.js має наступні властивості:

- асинхронна одно-нитева модель виконання запитів;
- неблокуючий ввід/вивід;
- система модулів CommonJS;

- рушій JavaScript Google V8;

Для керування модулями використовується пакетний менеджер npm (node package manager). [14]

Технологію Node.js як і Angular було обрано через наявність досвіду роботи.

3.3.2.3 База даних

На роль бази даних в проєкті була обрана MySQL.

MySQL — компактний багатопотоковий сервер баз даних. Характеризується високою швидкістю, стійкістю і простотою використання.

MySQL вважається гарним рішенням для малих і середніх застосувань. Сирцеві коди сервера компілюються на багатьох платформах. Найповніше можливості сервера виявляються в UNIX-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому.

Можливості сервера MySQL:

- простота у встановленні та використанні;
- підтримується необмежена кількість користувачів, що одночасно працюють із БД;
- кількість рядків у таблицях може досягати 50 млн;
- висока швидкість виконання команд;

- наявність простої і ефективної системи безпеки.

[15]

3.4 Демонстрація можливостей фінального проєкту

На сервісі є три сторінки:

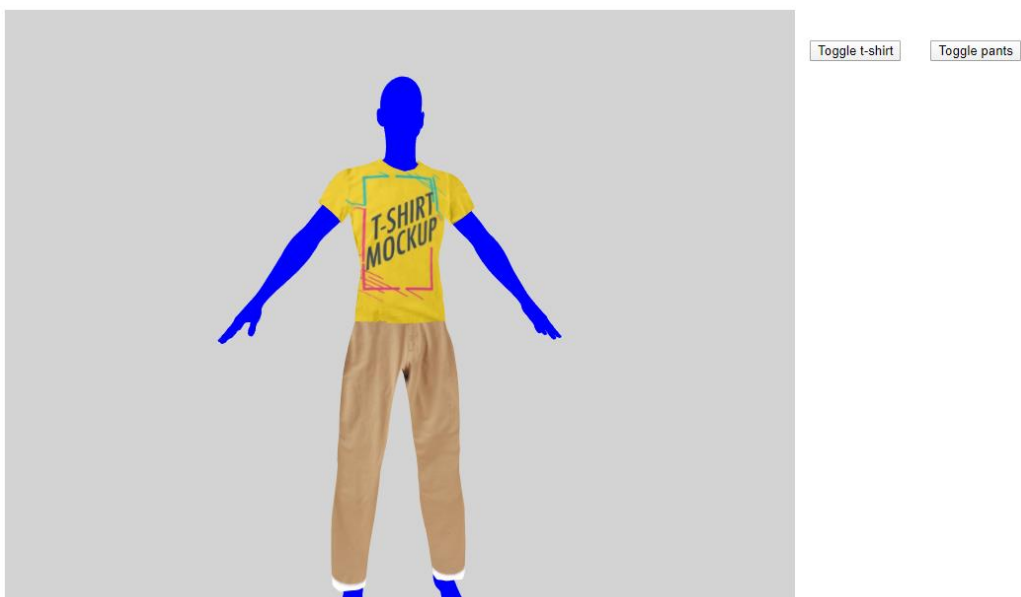
- Сторінка вибору одягу
- Сторінка візуалізації манекена, який одягнений в обраний одяг
- Сторінка адміністратора, де можна додавати та редагувати елементи одягу

3.4.1 Сторінка комплекту одягу

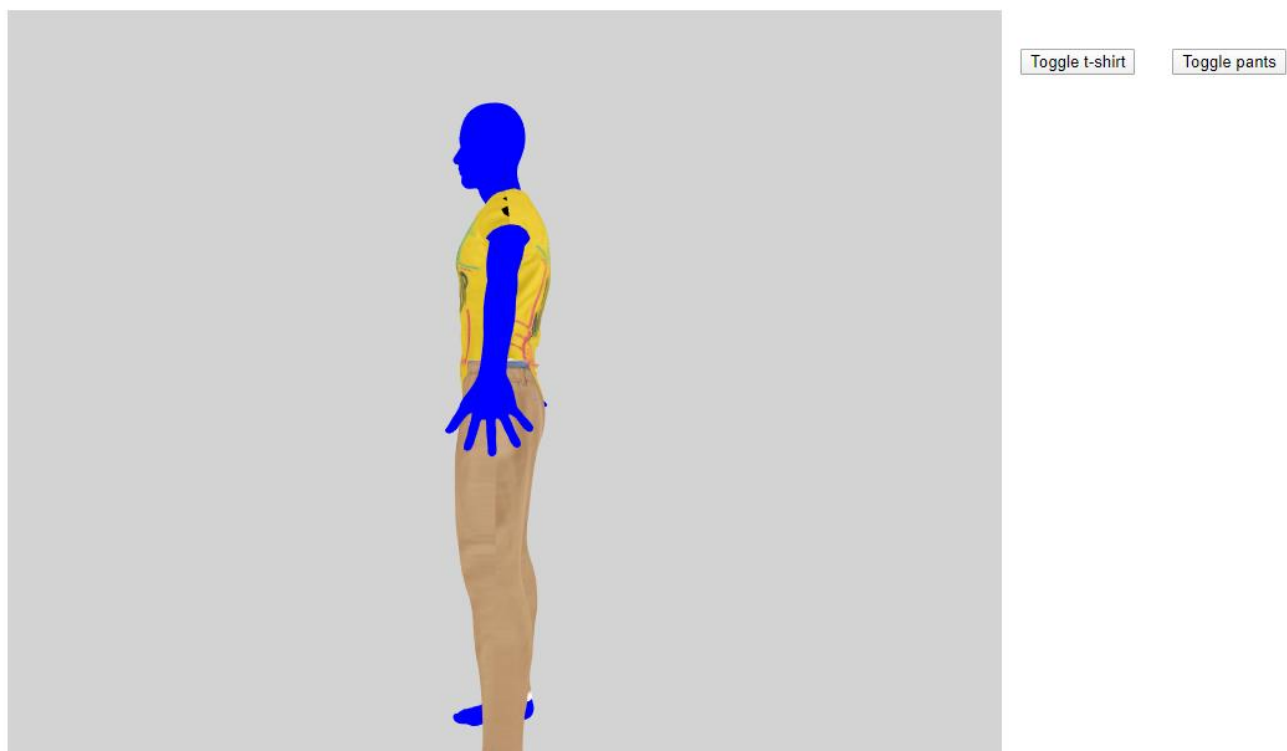
На сторінці відображаються елементи одягу які наявні на сайті (тобто які попередньо додав адміністратор). Будь який наявний елемент одягу можна приміряти на манекені. Також є можливість комбінувати різні типи одягу.

3.4.2 Сторінка візуалізації манекена, який одягнений в обраний одяг

На основній сторінці можна подивитися на манекен, одягнений в обраний на попередній сторінці одяг. Манекен, за бажанням, можна крутити та приближати\віддаляти.



Манекен, одягнений в широкі штани і звичайну футболку

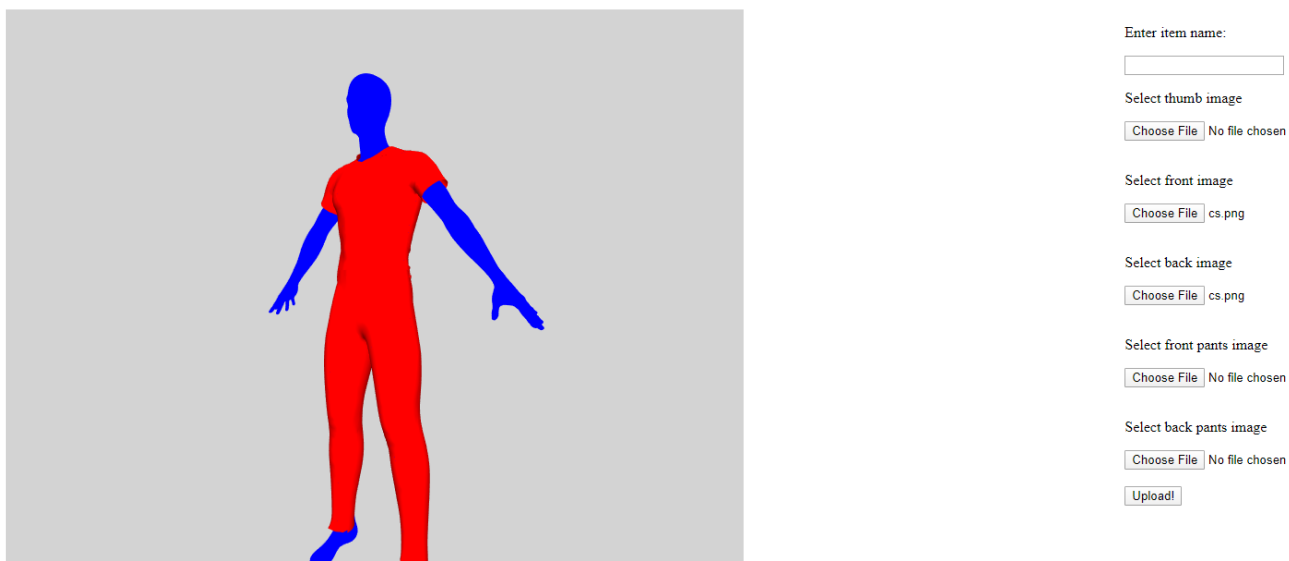


Манекен, одягнений в звичайну футболку і вузькі штани

3.4.3 Сторінка адміністратора

Сторінка адміністратора дозволяє працювати з елементами одягу. Є можливість додати новий елемент, для якого можна задати розмір. Також є можливість видаляти раніше додані елементи.

На цій сторінці, адміністратор може добавляти новий комплект одягу в базу даних, який згодом можна буде переглядати на манекені. Адміністратору пропонується надати назву комплекту, зображення комплекту, а також 4 зображення для футболки та штанів, спереду і ззаду.



Користувацький інтерфейс адміністратора

3.4.4 Результат

Було розроблено теоретичний магазин одягу на якому і було продемонстровано і протестовано всі можливості розробленої бібліотеки. Все працює коректно, результат задовільний.

Висновки

В ході виконання роботи було досліджено технології та алгоритми комп'ютерного зору. В результаті був створений алгоритм, який дозволяє перетворювати фотографію будь-якого одягу в частину текстури для 3D моделі. Фінальним результатом стала javascript-бібліотека, написана з використанням цього алгоритму. Бібліотека дозволяє користувачам інтернет-магазинів приміряти одяг на 3D манекенах в браузері. Для тестування створеної бібліотеки було створено сервіс для примірки свого одягу на віртуальних 3D моделях. Створена бібліотека може бути корисною в тих областях, де потрібна візуалізація одягу, аби покращити взаємодію з користувачами та клієнтами.

Використані джерела

- 1 - <https://www.learnopencv.com/>
- 2 - <https://opencv.org/about/>
- 3 - <https://cloud.google.com/vision#industry-leading-accuracy-for-image-understanding>
- 4 - <https://scikit-image.org/>
- 5 - <https://www.g2.com/products/opencv/competitors/alternatives>
- 6 - https://scikitimage.org/docs/stable/auto_examples/index.html
- 7 - <https://www.teepublic.com/t-shirt/4205347-baka-naruto>
- 8 - https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html
- 9 - <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf>
- 10 - <https://www.usability.gov/how-to-and-tools/methods/use-cases.html>
- 11 - <https://threejs.org/>
- 12 - <https://github.com/pallets/flask>
- 13 - <https://yalantis.com/blog/when-to-use-angular/>
- 14 - <https://nodejs.org/en/about/>
- 15 - <https://dev.mysql.com/doc/index-about.html>