

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КІЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря СІКОРСЬКОГО»
Навчально-науковий Фізико-технічний інститут

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА

з кредитного модуля «Інтелектуальні обчислення»

на тему:

**"РОЗПІЗНАВАННЯ РОБОЧОЇ ПОВЕРХНІ СТОЛУ ЗД ПРИНТЕРА ПРЯМИМИ ТА
ІТЕРАЦІЙНИМИ МЕТОДАМИ, НЕЙРОННИМИ МЕРЕЖАМИ"**

Виконав:

студент 3 курсу НН ФТІ
Голуб Михайло Вікторович
номер залікової книжки _____

Перевірив: _____

Оцінка: _____

Зміст

1 Вступ	3
1.1 Актуальність	3
1.2 Мета роботи	3
2 Допограмний етап	4
2.1 Аналіз об'єкту, що розпізнається	4
2.2 Постановка задачі	4
2.3 Створення методів отримання контексту	4
2.3.1 Отримання контексту з колору пікселів	4
2.3.2 Отримання границь, як джерел контексту	6
2.3.3 Використання попередніх результатів розпізнання	7
2.4 Створення фітнес-функцій для оцінки чотирикутників	8
2.4.1 Загальні вимоги до фітнес-функцій	8
2.4.2 Функція оцінки квадратності чотирикутника	8
2.4.3 Функція оцінки наявності границь поруч з сторонами чотирикутника	8
3 Висновки	10

1. Вступ

1.1 Актуальність

3D принтери технології пошарового наплавлення філаменту (Fused filament fabrication, далі – FFF 3D принтери) зараз є найбільш поширеними верстатаами для швидкого протипування. Данна технологія має недолік високої кількості браку. Збільшення кількості завчасних зупинок друку через брак може бути досягнуто при використанні методів що використовують контекст (положення друкованого об'єкту в просторі відносно камери), аніж при використанні нейронних мереж які аналізують зображення без жодної додаткової інформації про 3D модель що друкується. Для розробки і застосування покращених методів необхідно чітко знати положення столу, щоб визначити де на фотографії / відео має знаходитись 3D модель, яка друкується. Першим кроком з створення таких методів є система розпізнавання робочої поверхні столу FFF 3D принтера.

1.2 Мета роботи

Мета роботи – побудувати і порівняти методи розпізнавання робочої поверхні столу для FFF 3D принтера Bambulab A1 mini.



Рис. 1.1: Bambulab A1 mini

2. Допрограмний етап

2.1 Аналіз об'єкту, що розпізнається

Необхідно розпізнати робочу поверхню столу FFF 3D принтера Bambulab A1 mini. Дано модель має декілька різних змінних робочих поверхонь, але найбільш розповсюджена – PEI-пластина.



Рис. 2.1: PEI-пластина. На зображенії справа показано робочу зону

Робоча область PEI-пластини 18x18 см. Колір майже всіх PEI-пластин бронзово-золотистий. Отже, можна спробувати створити методи що шукають на зображенії проекції квадратів, конкретний колір, або проекції і колір одночасно. Також, при використанні кольору як критерію пошуку, можна скористатись тим, що корпус даного 3D принтера в більшості місць білий.

Окрім роботи з значеннями пікселів для визначення кольору, можна скористатись тим, що робоча поверхня дуже контрастна з оточенням, корпусом принтера, і застосувати фільтр границь.

2.2 Постановка задачі

Програма має для вхідного зображення повернути 8 значень – по дві координати чотирьох кутів столу/робочої області.

2.3 Створення методів отримання контексту

2.3.1. Отримання контексту з кольору пікселів

З кольору пікселів можна створити контекст ввівши ідеальне значення кольору і допустимі відхилення. Доцільно це робити використавши HSV (трьохканальна система кольорів "Колірний тон, насиченість, яскравість"), а не RGB (трьохканальна система кольорів "Червоний, зелений, синій"), оскільки в HSV можна проігнорувати яскравість та враховувати лише колірний тон і насиченість. Щоб отримати ідеальне значення кольорів столу і корпусу використано фотографію принтеру в типових умовах освітлення, отриману на камеру телефона.

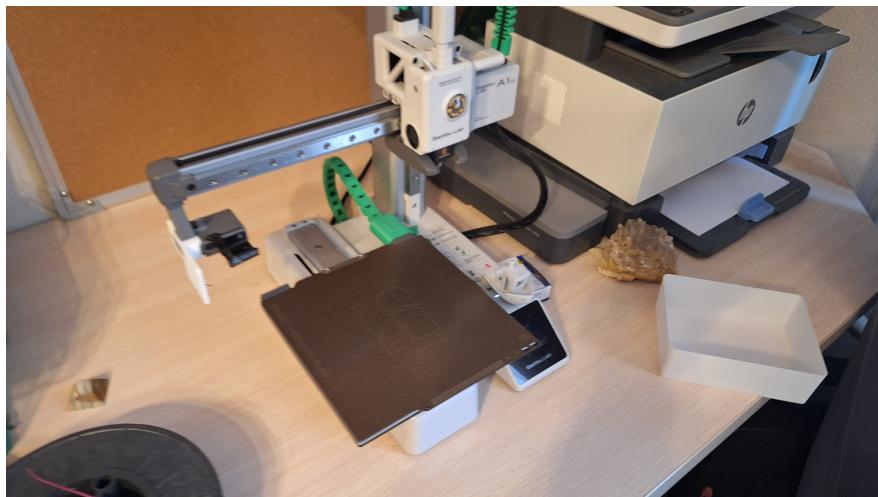


Рис. 2.2: Зображення принтера отримане за допомогою камери телефона

Отримані значення RGB: (89, 65, 55) для столу і (219, 208, 202) для корпусу. Після переходу в HSV отримано (0.05, 0.382, 0.349) і (0.0583, 0.078, 0.859) відповідно. Покладено допустимі відхилення тону кольору і насиченості в 0.1 для столу. Покладено що насиченість для корпусу має бути менша за 0.2, при будь-якому значенні тону. Дані правила застосовані для даного зображення.

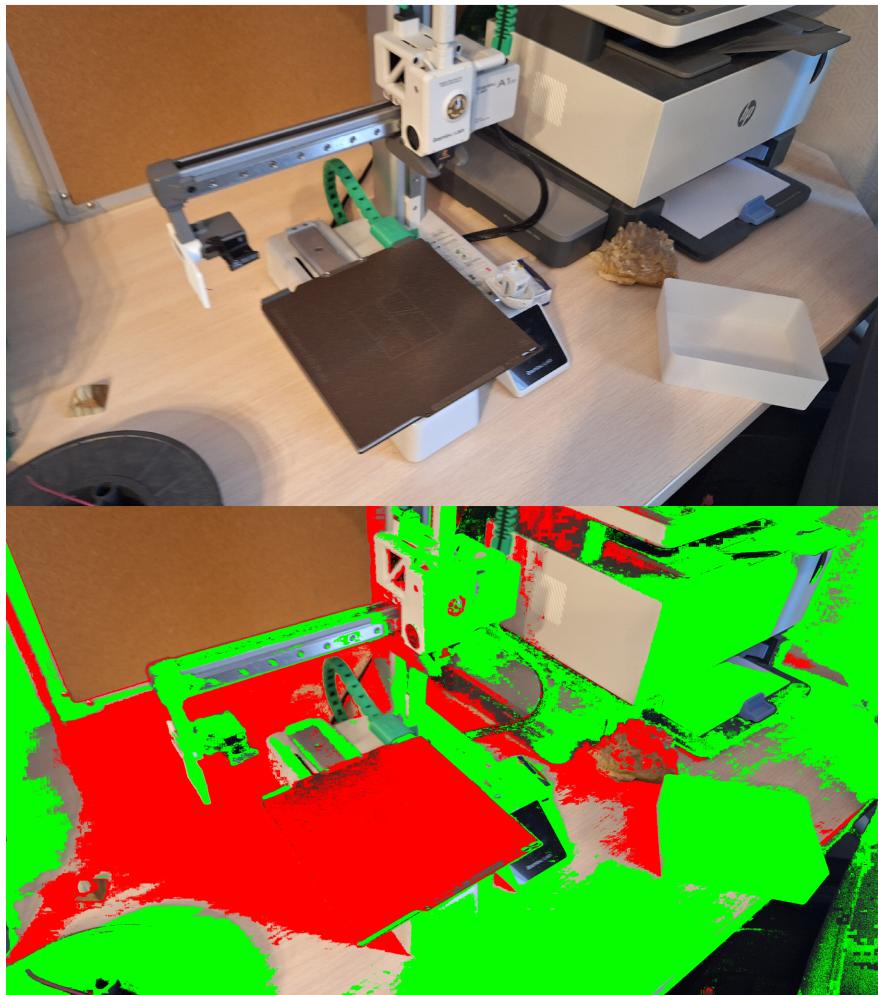


Рис. 2.3: Перший набір правил застосований до зображення: правила для столу червоним, правила для корпусу білим

Застосування даних правил значно зменшило область пошуку столу, проте вона все ще зали-

шалася великою. З порівняння зображень видно, що правила для столу спрацьовують для світлих ділянок, а правила для корпусу – для темних. Щоб уникнути таких спрацювань, було додано правило на яскравість: для столу яскравість має бути меншою за середню на зображені, а для корпусу – більшою.

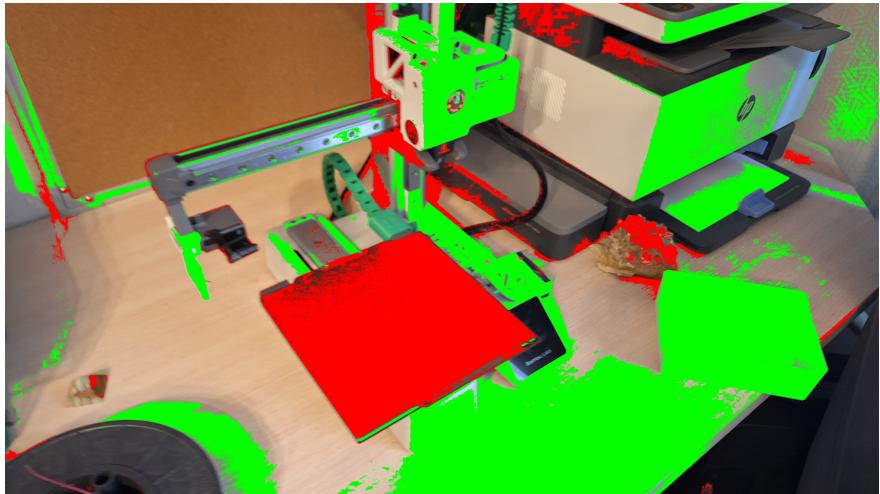


Рис. 2.4: Другий набір правил застосований до зображення

Дане розширення правил ще раз значно зменшило область пошуку. Проте, частина столу не фарбується в червоний, відповідно необхідно збільшити допустимі відхилення тону і насиченості. Допустимі відхилення збільшено вдвічі.

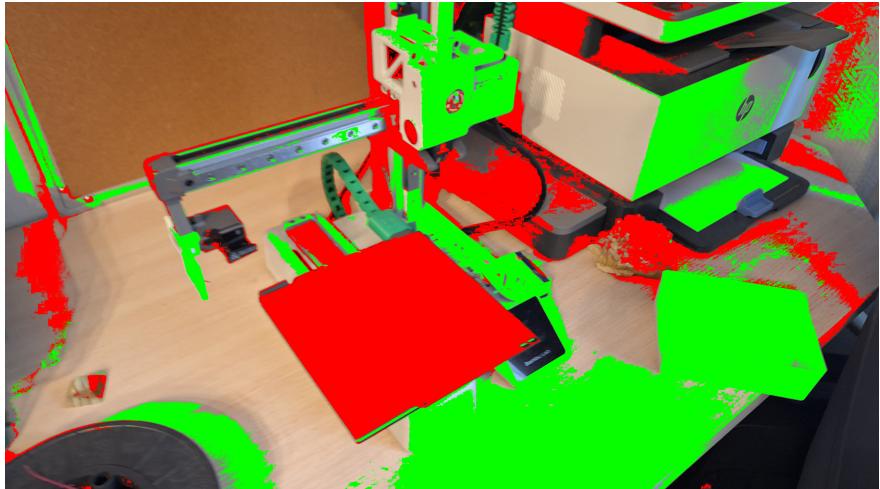


Рис. 2.5: Третій набір правил застосований до зображення

2.3.2. Отримання границь, як джерел контексту

Для отримання контексту у вигляді границь об'єктів необхідно застосувати матричний фільтр границь:

$$kernel = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Даний фільтр застосовано до різних зображень: оригінального, оригінального в HSV та, чорно-білого. Після чого було взято середнє всіх трьох каналів. Для кращого відображення границь, розмір зображення зменшено в 8 разів.

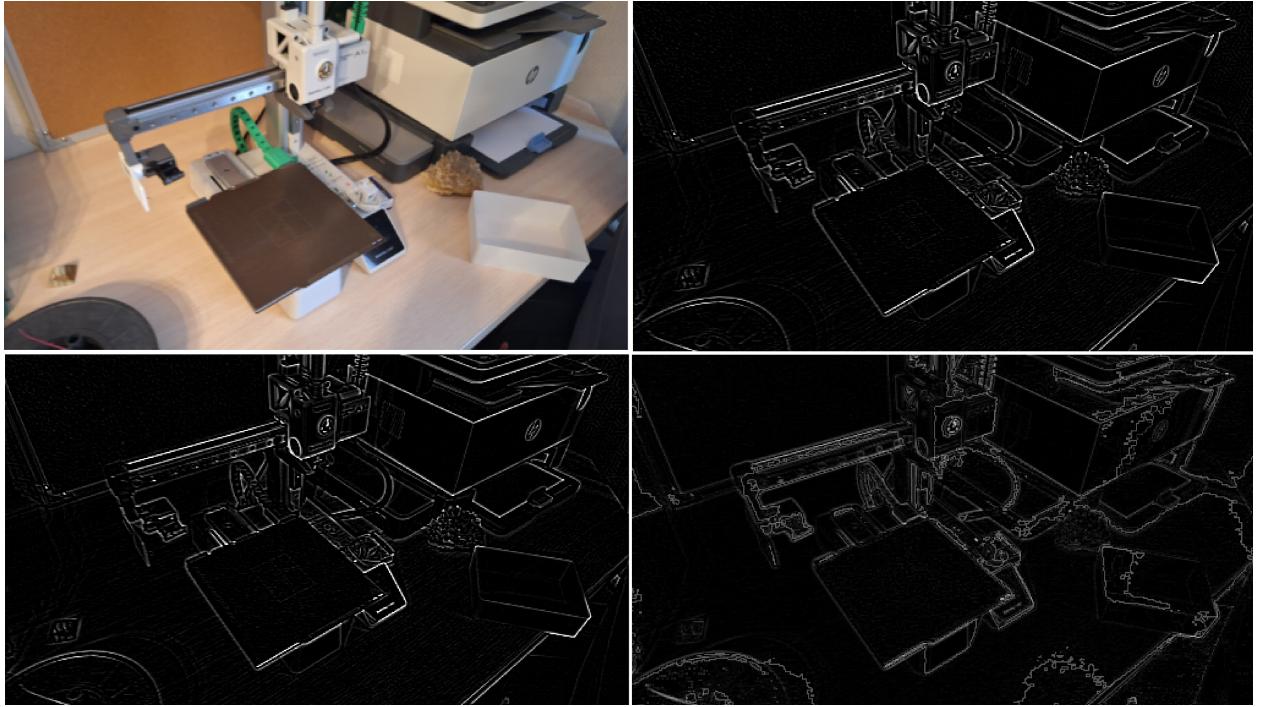


Рис. 2.6: Перший ряд: оригінальне зображення, застосування фільтру до RGB, застосування фільтру до середнього значення каналів RGB, застосування фільтру до HSV

З зображень видно, що фільтр HSV має занадто нечіткі лінії щоб використовуватись далі.

Для утворення чітких границь, необхідно їх перетворити у бінарні значення 0 і 1. Один з більш простих способів такого перетворення – прогове перетворення:

$$1, \text{pixel} \geq \text{threshold}$$

$$0, \text{pixel} < \text{threshold}$$

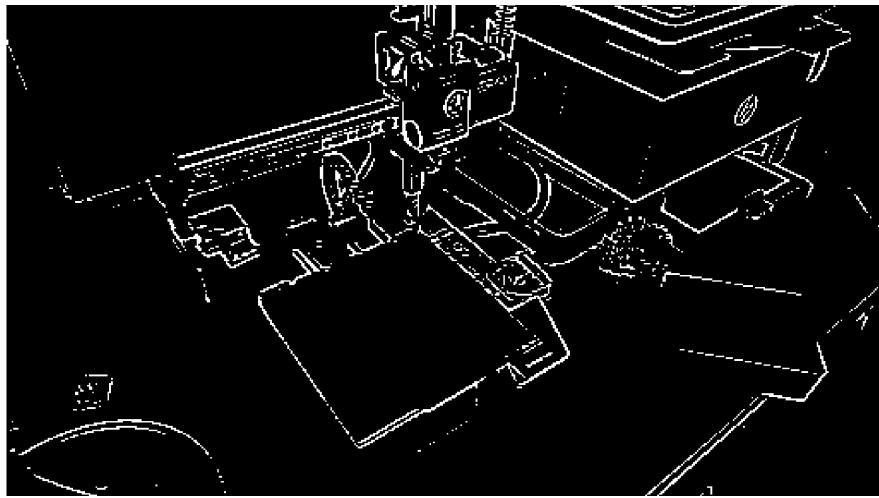


Рис. 2.7: Порогова бінаризація RGB границь з порогом 0.2

2.3.3. Використання попередніх результатів розпізнання

Нехай програма вже успішно змогла розміннати стіл в попередньому моменті часу (наприклад на попередньому кадрі відео), тоді відомо масив точок що входили в регіон столу. Оскільки хід столу вперед-назад не перевищує розміру самого столу, то якась частина попереднього масиву точок мають

опинитись в новому регіоні столу. Можна впровадити простий критерій, що хоча б одна точка з попереднього регіону столу, має бути в новому регіоні столу.

Якщо попередні моменти часу відсутні, можна попросити людину, або більш потужний розпізнавач розпізнати початкове положення столу. У разі використання людини користувача/оператора, її достатньо вказати якусь точку столу, щоб звичайний розпізнавач за цією точкою знайшов стіл на першому зображенні.

2.4 Створення фітнес-функцій для оцінки чотирикутників

2.4.1. Загальні вимоги до фітнес-функцій

Фітнес-функція має приймати на вхід вершини чотирикутника, окрім вершин функція може приймати значення вершин в попередній момент часу, початкову точку та константи пов'язані з оброблюванням зображенням.

2.4.2. Функція оцінки квадратності чотирикутника

Квадрат є паралелограмом, з більшості кутів огляду (викривленням лінзи можна знехтувати). Паралелограми мають наступні властивості: їх протилежні сторони рівні, їх протилежні сторони паралельні.

Оцінка рівності протилежних сторін:

$$\phi_E = \frac{(E_1 - E_3)^2}{E_1 + E_3} + \frac{(E_2 - E_4)^2}{E_2 + E_4}, \text{ де } E_i - \text{довжина ребра.}$$

В оцінці рівності протилежних сторін присутнє ділення на суму цих протилежних сторін, інакше оцінка буде працювати по-різному для різних довжин ребер.

Оцінка паралельності протилежних сторін:

$$\phi_\alpha = \min^2(|\alpha_1 - \alpha_3|, |\alpha_1 + \alpha_3 - 2\pi|) + \min^2(|\alpha_2 - \alpha_4|, |\alpha_2 + \alpha_4 - 2\pi|), \text{ де } \alpha_i - \text{кут нахилу ребра в радіанах.}$$

Об'єднання оцінок рівності і паралельності протилежних сторін можна виконати безліччю методів. Простішими з них є сума і множення. Сума оцінок зберігає незалежність оцінок і дозволяє покращувати одну оцінку, навіть якщо інша занулена. Але сума потребує балансування оцінок, інакше алгоритми і моделі будуть намагатись оптимізувати оцінку з найбільшим абсолютним значенням. Множення не має необхідності балансування оцінок, проте має проблеми, якщо якась з оцінок занулюється. Щоб уникнути занулення, до оцінок можна додати невелику константу.

Оцінки об'єднані наступним чином:

$$\phi_{\text{пар.}} = (\phi_E + c_1) \cdot (\phi_\alpha + c_2), \text{ де } c_i - \text{константи, які необхідно визначити і збалансувати експериментально.}$$

2.4.3. Функція оцінки наявності границь поруч з сторонами чотирикутника

Оскільки по периметру столу 3D принтера майже завжди знаходяться границі, можна рахувати кількість пікселів границі поруч з сторонами чотирикутника. Нехай "поруч" це "сторона проходить через піксель". Для визначення пікселів через які проходить відрізок існує алгоритм Брезенхейма. Загальне рівняння прямої через дві точки (x_0, y_0) , (x_1, y_1) :

$$\frac{y - y_0}{y_1 - y_0} = \frac{x - x_0}{x_1 - x_0}$$

Алгоритм Брезенхейма, для всіх 8 напрямків прямої:

1. r – пустий масив точок;
2. $\Delta x = |x_1 - x_0|$; $\Delta y = |y_1 - y_0|$;
3. $sx = x_1 \geq x_0$; $sy = y_1 \geq y_0$;

4. $x = x_0; y = y_0;$
5. $D = \text{argmax}(\Delta x, \Delta y)$ – домінантна вісь за якою відбуваються кроки, S – інша вісь (X, Y = D, S або Y, X = D, S);
6. $Error = \text{floor}(\Delta d/2);$
7. Доки $d \neq d_1$:
8. Записати (x, y) в p
9. $Error = Error - \Delta s$
10. Якщо $Error < 0$, то: $s = s + ss$; $Error = Error + \Delta s$
11. $d = d + sd;$
12. Повернутись до "Доки";
13. Записати (x_1, y_1) в p;
14. Повернути p.

Для обрахунку кількості білих пікселів, алгоритм модифіковано наступним чином:

1. $p = 0;$
2. $\Delta x = |x_1 - x_0|; \Delta y = |y_1 - y_0|;$
3. $sx = x_1 \geq x_0; sy = y_1 \geq y_0;$
4. $x = x_0; y = y_0;$
5. $D = \text{argmax}(\Delta x, \Delta y)$ – домінантна вісь за якою відбуваються кроки, S – інша вісь (X, Y = D, S, або Y, X = D, S);
6. $Error = \text{floor}(\Delta d/2);$
7. Доки $d \neq d_1$:
8. $p = p + \text{Image}[x][y];$
9. $Error = Error - \Delta s$
10. Якщо $Error < 0$, то: $s = s + ss$; $Error = Error + \Delta s$
11. $d = d + sd;$
12. Повернутись до "Доки";
13. $p = p + \text{Image}[x_1][y_1];$
14. Повернути p.

3. Висновки

Бібліографія

- [1] Автор1. Назва книги. Видавництво, рік.
- [2] Автор2. Назва статті // Назва журналу. – Рік. – Том. – С. 100-200.