МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря Сікорського”
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра інформаційної безпеки

ЗВІТ
з науково-дослідної практики

за спеціальністю: 113 Прикладна математика

На тему:
«Сегментація зображення для прискорення алгоритмів стереобачення»

Виконав: студент 2 курсу
групи ФІ-81мн

Лавягіна О.О.
(ПІБ студента)

(підпис студента)

Керівник:
Панчук І.В.
(ПІБ керівника)

(підпис керівника)

Захистив з оцінкою:

Київ 2020 р.

ЗМІСТ

1 ПОПЕРЕДНІ РОБОТИ ПРИСВЯЧЕНІ ЗАДАЧІ СТЕРЕОБАЧЕННЯ

В першому розділі надано стислий огляд досліджень, що пов'язані із задачею стереобачення. Проводиться аналіз та порівняння деяких існуючих методів розв'язання задачі. Розбір попередніх робіт дає змогу виявити недоліки існуючих рішень та чітко поставити задачу, що розв'язується в другому та третьому розділах дисертації.

1.1 Динамічне програмування

Вважається, що камера не вносить радіальних спотворень, а два зображення отримуються шляхом її ідеального зсуву по горизонталі, тобто пікселям рядка y лівого зображення $L_y : \{1, \ldots, width\} \to C$ відповідають пікселі рядка y правого зображення $R_y : \{1, \ldots, width\} \to C$, де $y = \overline{1, height}$, а C — множина, що задає кольори (або інтенсивності) пікселів. Вважається, що в кожен рядок лівого зображення проектується свій об'єкт, тому вектори зсувів для всіх рядків можна знаходити окремо шляхом мінімізації штрафної функції

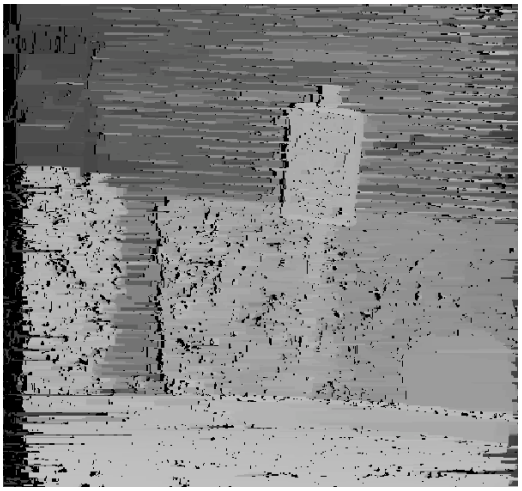
$$E(d, y) = \sum_{x=1}^{width} f\left(L(y, x), R(y, x - d(x))\right) + \sum_{x=1}^{width} g\left(d(x), d(x+1)\right), \quad (1.1)$$

яка називається енергією, де d — вектор зсувів для рядка з номером y. Штрафна функція включає штраф за негладкість карти зсувів g (бінарні штрафи) і невідповідність кольорів співставлених пікселів на двох зображеннях f (унарні штрафи). Унарні штрафи f залежать лише від значення інтенсивності в даному пікселі та вибраного для нього зсуву, бінарні штрафи g — від зсувів двох сусідніх по горизонталі пікселів.

Для кожного рядка зображення будується граф, де кожен піксель x ряд-

ка y є об'єктом, а вершини в об'єктах відповідають можливим зсувам $d(x)$. Назвемо їх мітками. Сусідніми вважаються лише ті об'єкти, які відповідають сусіднім пікселям (у кожного пікселя є лівий і правий сусідні пікселі). Вага мітки в об'єкті x, що відповідає зсуву $d(x)$, задається значенням функції $f(L(y,x), R(y, x - d(x)))$, а вага дужки між двома сусідніми об'єктами x і $x+1$ — значенням функції $g(d(x), d(x+1))$. Методом динамічного програмування знаходится такий вектор зсувів для кожного рядка, що відповідає найкоротшому шляху на побудованому графі [1].

(а) Ліве та праве зображення



(б) Карта глибин

Рисунок 1.1 — Результати, отримані методом динамічного програмування (зображення взяті зі статті [1])

Недоліком методу динамічного програмування є те, що він обробляє всі

рядки зображення незалежно, тобто не враховує узгодженості між рядками, тому в простому випадку метод динамічного програмування не використовує двовимірність задачі та не дає гладкої карти глибин по вертикалі (рис. 1.1). Водночас, алгоритм працює досить швидко, а результати можна значно покращити (рис. 1.2), наприклад розглядаючи вісім сусідів пікселя замість двох та використовуючи сегментацію зображення за кольором [2].

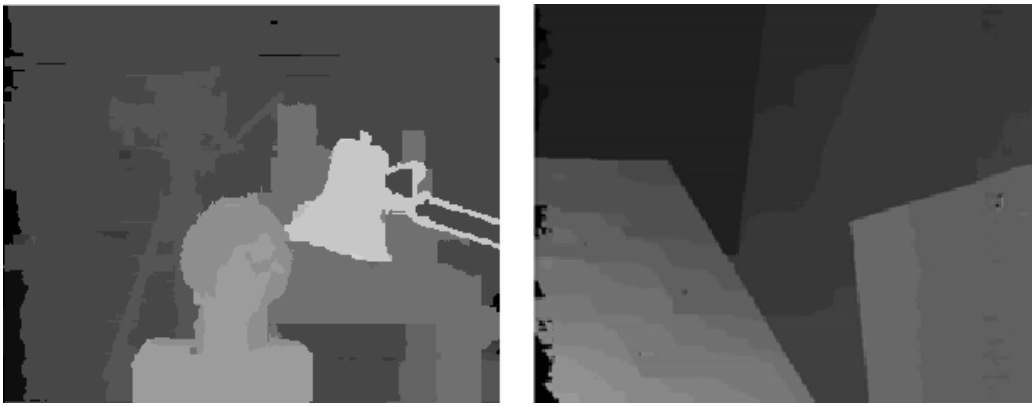


Рисунок 1.2 — Результати, отримані за допомогою покращеного методу динамічного програмування (зображення взяті зі статті [2])

1.2 Нейронні мережі

Нейронна мережа навчається на вхідних даних: парах зображень та відповідній ним карті глибин. Добре розмічені вхідні дані для навчання наявні для зображень дороги. Нейронна мережа, навчена на таких даних може давати погані результати для зображень іншого середовища. На рисунку 1.3 наведено приклад вхідних даних і результуючої карти глибин, що була навчена за допомогою даних з набору «KITTI Stereo» [3].

Рисунок 1.3 — Ліве зображення та карта глибин, отримана за допомогою нейронної мережі (зображення взяті зі статті [4])

1.3 Знаходження мінімального розрізу графа

Задачею є мінімазація енергії 1.1. Будується граф-решітка, у якому вершинами є всі пікселі зображення. Назвемо ці вершини основними. Ваги вершин і дуг задаються як в методі динамічного програмування. До графа додається дві допоміжні вершнии: джерело та стік, які поєднуються з кожними іншими вершинами графу. Кожній основній вершині має бути поставлена у відповідність мітка — значення зсуву координати пікселя по відношенню до цієї ж координати на іншому зображенні зі стереопари. Застосовується алгоритм α-експансії: спочатку фіксується якась мітка в кожній основній вершині. Також випадковим чином обирається ще одна мітка (однакова для всіх вершин). Шляхом знаходження мінімального розрізу графу, в якому дужки направлені та йдуть від джерела до стоку, в кожній вершині обирається одна з двох міток. Обрані мітки подаються на вхід алгоритму на наступній ітерації. Результати, отримані за допомогою такого методу, представлені на рисунку 1.4.

Для більш швидкої глобальної оптимізації виконують сегментацію одного із зображень стереопари. На пікселі в одному сегменті накладаютсья обмеження на зсуви так, що при глобальній оптимізації оновлюється декілька змінних одразу [6].

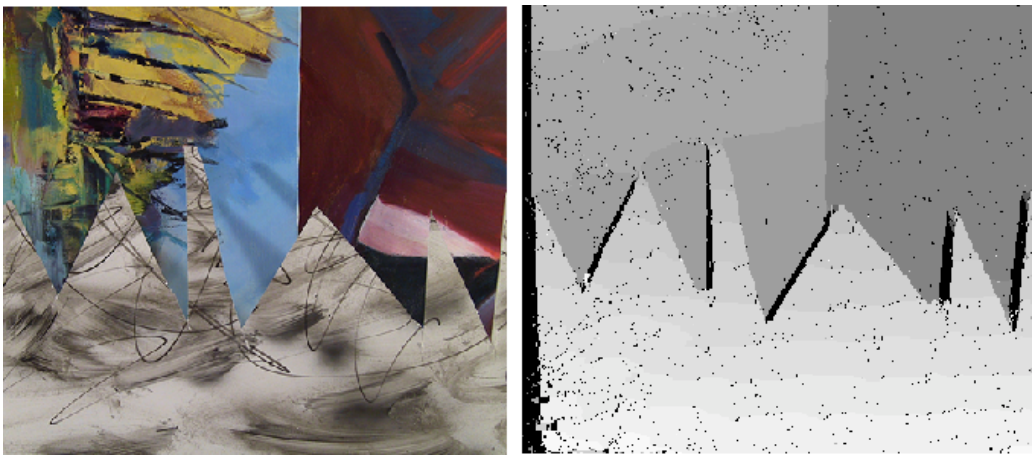


Рисунок 1.4 — Ліве зображення та карта глибин, отримана за допомогою знаходження мінімального розрізу графу (зображення взяті зі статті [5])

1.4 Алгоритм дифузії

Будується граф-решітка та мінімізується енергія 1.1 шляхом розв'язання двоїстої задачі [7]. Докладніше алгоритм описано в наступному розділі.

1.5 Постановка задачі

Виходячи з аналізу статей, запишемо постановку задачі, яка буде розв'язана в наступних розділах.

Є два зображення: ліве

$$L \ : \ \{1, \ldots, height\} \times \{1, \ldots, width\} \to C$$

та праве

$$R \ : \ \{1, \ldots, height\} \times \{1, \ldots, width\} \to C,$$

де $height$ — висота зображення (кількість пікселів по вертикалі), $width$ — ширина зображення (кількість пікселів по горизонталі), $C = \{0, 255\}$ — множина інтенсивностей пікселів. Таким чином, $L\,(y, x)$ — це інтенсивність пікселя лі-

вого зображення з координатою y по вертикалі та x по горизонталі.

Множину всіх координат пікселів зображення

$$\{1, \ldots, height\} \times \{1, \ldots, width\}$$

будемо позначати через T.

Вважається, що рядку y лівого зображення відповідає рядок y правого зображення. Кожен піксель лівого зображення (y, x) зсувається по горизонталі вліво на величину $d\,(y, x)$ для отримання координати відповідного пікселя на правому зображенні. Множину всіх можливих зсувів для об'єкту (y, x) з горизонтальною координатою x позначимо через

$$D_x = \{0, \ldots, \max D_x\}\,.$$

Будується $|T|$-дольний граф-решітка, де кожній долі відповідає піксель $(y, x) \in T$. Будемо називати долі графу об'єктами.

Кожен об'єкт $(y, x) \in T$ має $|D|$ вершин $(y, x, d)\,,\ d \in D_x$, які відповідають всім можливим зсувам d відповідного пікселя (y, x).

Кожна вершина графа $(y, x, d)\,,\ d \in D_x$ має вагу $f\,(L\,(y, x)\,, R\,(y, x - d))$.

Кожен об'єкт має не більше чотирьох сусідніх об'єктів: верхній, нижній, лівий і правий. $\mathcal{N}\,(y, x)$ — множина всіх сусідніх об'єктів до об'єкта (y, x). Всі вершини в об'єкті поєднані дужками з усіма вершинами в сусідніх об'єктах. Вага дужки між об'єктом (y, x) з міткою d й об'єктом (y', x') з міткою d' дорівнює $g\,(d, d')$, де $(y', x') \in \mathcal{N}\,(y, x)$, тобто $((y, x)\,, (y', x'))$ — сусідні об'єкти. Множину всіх сусідніх об'єктів позначимо через $\mathcal{N}$.

Відображення $\overline{d} : T \to D$ називається розміткою. Задача полягає в такому виборі розмітки $\overline{d}$ (тобто виборі зсуву $d \in D_x$ в кожному об'єкті $(y, x) \in T$),

яка мінімізує штрафну функцію

$$E\left(\overline{d}\right) = \sum_{y=1}^{height} \sum_{x=1}^{width} f\left(L\left(y,x\right), R\left(y, x - d\left(y,x\right)\right)\right) + \\ + \sum_{y=1}^{height} \sum_{x=1}^{widht} \sum_{(y',x') \in \mathcal{N}(y,x)} g\left(d\left(y,x\right), d\left(y',x'\right)\right). \tag{1.2}$$

Вибір функцій f і g залежить від вибору алгоритму, що буде застосовуватись для оптимізації штрафної функції та характеру поверхні на вхідних зображеннях.

Висновки до розділу 1

Проведено огляд алгоритмів, які використовуються для розв'язання задачі стереобачення. Поставлена задача, розв'язання якої наведено в наступних розділах дисертації.

2 АЛГОРИТМ ДИФУЗІЇ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ СТЕРЕОБАЧЕННЯ

Другий розділ присвячено розв'язку задачі стереобачення алгоритмом дифузії. Описуються властивості алгоритму та з'ясовується придатність розв'язку для поставленої задачі.

2.1 Алгоритм дифузії для розв'язання задачі стереобачення

Алгоритм дифузії є блочно-координатним спуском [7]. Кожній вершині графу (y, x, d), де $(y, x) \in T$, $d \in D_x$, ставиться у відповідність блок змінних $\left(\varphi_{(y,x),(y',x')}(d) : (y', x') \in \mathcal{N}(y, x),\, d \in D_x\right)$.

Введемо репараметризовані ваги вершин

$$f^{\varphi}(L(y,x), R(y,x-d)) := \\ := f(L(y,x), R(y,x-d)) - \sum_{(y',x')\in\mathcal{N}(y,x)} \varphi_{(y,x),(y',x')}(d), \quad (y,x) \in T, \quad d \in D_x$$

та репараметризовані ваги дужок

$$g^{\varphi}(d,d') := g(d,d') + \varphi_{(y,x),(y',x')}(d) + \varphi_{(y',x'),(y,x)}(d'), \\ ((y,x),(y',x')) \in \mathcal{N}, \quad d \in D_x, \quad d' \in D_{x'}.$$

Елементарний крок алгоритму дифузії складається з двох операцій для кожного об'єкту (y, x):

$$\forall (y',x') \in \mathcal{N}(y,x) \quad \forall d \in D_x \\ \varphi^{t+1}_{(y,x),(y',x')}(d) := \varphi^{t}_{(y,x),(y',x')}(d) - \min_{d'\in D_{x'}} g^{\varphi^t}(d,d'), \tag{2.1}$$

$$\forall (y',x') \in \mathcal{N}(y,x) \quad \forall d \in D_x$$
$$\varphi^{t+2}_{(y,x),(y',x')}(d) := \varphi^{t+1}_{(y,x),(y',x')}(d) + \frac{f^{\varphi^{t+1}}\left(L(y,x), R(y,x-d)\right)}{|\mathcal{N}(y,x)|}, \tag{2.2}$$

де через t позначено номер ітерації.

Для фіксованої вершини (y,x,d) перша операція 2.1 знаходить мінімальну вагу дужки між заданою вершиною та всіма вершинами (y',x',d'), де $d' \in D_{x'}$, з сусідніх до неї об'єктів $(y',x') \in \mathcal{N}(y,x)$, віднімає її від ваг усіх цих дужок $((y,x,d),(y',x',d'))$ та додає до ваги вершини (y,x,d). В результаті мінімальна вага дужки між вершиною (y,x,d) та кожним її сусідом $(y',x') \in \mathcal{N}(y,x)$ дорівнює нулю.

Після цього друга операція 2.2 перерозподіляє репараметризовані ваги вершин $f^{\varphi}\left(L(y,x), R(y,x-d)\right)$ між дужками до всіх сусідніх об'єктів $(y',x') \in \mathcal{N}(y,x)$. В результаті вага вершини (y,x,d) стає рівною нулю, а мінімальні ваги дужок з цієї вершини до всіх сусідніх об'єктів стають рівними одна одній.

Іншими словами, після операцій 2.1 і 2.2 для об'єкта $(y,x) \in T$ виконується

$$f^{\varphi}\left(L(y,x), R(y,x-d)\right) = 0, \quad \forall d \in D_x,$$

$$\min_{d' \in D_{x'}} g(d,d') = \min_{d'' \in D_{x''}} g(d,d''), \quad \forall (y',x'), (y'',x'') \in \mathcal{N}(y,x), \quad d \in D_x.$$

Відомо [7], що операції 2.1 і 2.2 максимізують двоїсту функцію Лагранжа за змінними $\left(\varphi_{(y,x),(y',x')}(d) : (y',x') \in \mathcal{N}(y,x), d \in D_x\right)$, а тому мінімізують штрафну функцію 1.2.

Алгоритм полягає в ітеративному повторі елементарного кроку дифузії до виконання критерію зупинки.

На першій ітерації вважається, що $\varphi_{(y,x),(y',x')}(d) = 0$ для всіх об'єктів $(y,x) \in T$, всіх міток $d \in D_x$ та всіх сусідніх об'єктів $(y',x') \in \mathcal{N}(y,x)$.

2.2 Вибір найкращої розмітки

Після завершення якоїсь кількості ітерацій алгоритму дифузії, перевіряється, чи досягається необхідна точність за допомогою алгоритма викреслювання другого порядку. На вже побудованому графі розв'язується $(\bigvee, \bigwedge)$-задача.

Тепер кожна вершина і дужка графу може бути допустимою або ні. $\tilde{f}_{(y,x)}(d) \in \{0,1\}$ означає допустимість мітки $d \in D_x$ в об'єкті $(y,x) \in T$. Аналогічно, $\tilde{g}_{(y,x),(y',x')}(d,d') \in \{0,1\}$ означає допустимість пари міток $d \in D_x$ і $d' \in D_{x'}$ в парі сусідніх об'єктів $((y,x),(y',x')) \in \mathcal{N}$.

На початку всі вершини вважаються допустимими, тобто $\tilde{f}_{(y,x)}(d) = 1$ для всіх міток $d \in D_x$ в об'єкті $(y,x) \in T$. В кожній парі сусідніх об'єктів знаходиться дужка з мінімальною вагою. Між цією парою об'єктів допустимими є ті дужки, вага яких відрізняється від мінімальної не більше наперед заданої малої величини ε.

Алгоритм полягає в багатократному застосуванні операцій «викреслювання вершини»

$$\tilde{f}_{(y,x)}(d) := \tilde{f}_{(y,x)}(d) \bigwedge_{(y',x') \in \mathcal{N}(y,x)} \bigvee_{d' \in D_{x'}} \tilde{g}_{(y,x),(y',x')}(d,d')$$

та «викреслювання дужки»

$$\tilde{g}_{(y,x),(y',x')}(d,d') := \tilde{g}_{(y,x),(y',x')}(d,d') \bigwedge \tilde{f}_{(y,x)}(d) \bigwedge \tilde{f}_{(y',x')}(d').$$

Алгоритм завершує роботу зі скінченну кількість ітерацій. Якщо після завершення роботи алгоритма деякі вершини залишились невикресленими, то було зроблено досить ітерацій дифузії, і з множини невикреслених вершин можна побудувати розмітку. Інакше треба продовжити ітерації дифузії.

Для гарантії збіжності алгоритму дифузії ваги дужок g вихідного графу

повинні мати властивість супермодулярності [8]. Прикладом вагової функції для дужок з властивістю супермодулярності є

$$g\left(d\left(y,x\right),d\left(y',x'\right)\right)=\left|d\left(y,x\right)-d\left(y',x'\right)\right|.$$

Висновки до розділу 2

Пред'явлено розв'язок задачі стереобачення за допомогою алгоритма дифузії. За певних штрафних функцій алгоритм гарантовано збігається, проте на практиці алгоритм працює досить довго, а отже потребує оптимізації.

3 СЕГМЕНТАЦІЯ ЗОБРАЖЕННЯ ДЛЯ ПРИСКОРЕННЯ АЛГОРИТМІВ СТЕРЕОБАЧЕННЯ

У третьому розділі наведено спосіб прискорення алгоритмів стереобачення на прикладі алгоритму дифузії за допомогою сегментації зображення. Представлено практичні результати застосування алгоритму дифузії з сегментацією та без із зазначенням вхідних даних.

3.1 Сегментація зображення для прискорення алгоритму дифузії

Зображення розбивається на прямокутку решітку. Розмір кожної комірки решітки однаковий.

Назвемо суперпікселем набір пікселів, які належать до однієї комірки та мають якісь загальні властивості.

Всі пікселі кожної комірки діляться на дві частини, тобто на два суперпікселя, за середньою інтенсивністю пікселів, які належать до комірки. До першого суперпікселя відносяться всі пікселі комірки, інтенсивність яких не перевищує середньої інтенсивності пікселів у цій комірці, а до другого суперпікселя — всі інші пікселі комірки.

Так як пошук карти глибин виконується для лівого зображення, то сегментується тільки ліве зображення.

Після запропонованої сегментації будується $(m \cdot n)$-дольний граф, де m — кількість суперпікселів по вертикалі, n — кількість суперпікселів по горизонталі. Тепер кожна доля відповідає одному суперпікселю. Кожен суперпіксель (y_s, x_s), де $1 \leq y_s \leq m$, $1 \leq x_s \leq n$, містить $|D_{x_s}|$ вершин, що відповідають можливим зсувам, тобто всі пікселі, що належать одному сіперпікселю, будуть мати однаку глибину в результуючій карті глибин.

Теперь кожен суперпіксель (або об'єкт) може мати до восьми сусідів: по

два суперпікселя з правої, нижньої, верхньої та лівої комірок решітки, а також другий суперпіксель, що належить цій же комірці.

Штраф за вибір мітки $d \in D_{x_s}$ у суперпікселі (y_s, x_s) задається як сума штрафів за вибір мітки d в усіх пікселяї, що належать даному суперпікселю

$$f_s(x_s, y_s, d) = \sum_{(y,x)\in(y_s,x_s)} f\left(L(y,x), R(y, x-d)\right).$$

Штрафна функція, що накладається на дуги графу, не змінюється.

Таким чином, штрафна функція задачі 1.2 набуває наступного вигляду

$$\begin{aligned} E_s\left(\overline{d}\right) = \sum_{y_s=1}^{m}\sum_{x=1}^{n} f_s\left(x_s, y_s, d\left(x_s, y_s\right)\right) + \\ + \sum_{y_s=1}^{m}\sum_{x_s=1}^{n}\sum_{(y'_s,x'_s)\in\mathcal{N}(y_s,x_s)} g\left(d\left(y_s, x_s\right), d\left(y'_s, x'_s\right)\right). \end{aligned} \tag{3.1}$$

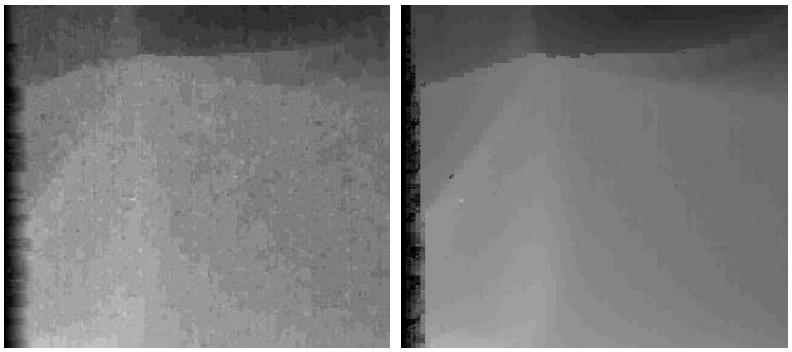
Далі задача розв'язується методом, що описаний у попередньому розділі.

3.2 Практичні результати

Результати роботи програми зображені на рисунку 3.1. Зліва зображена карта глибин, отримана за допомогою алгоритму, описаного в попередньому розділі, де кожній долі графа відповідає один піксель. Справа зображена карта глибин, отримана за допомогою об'єднання частин зображення в суперпікселі.

Вхідні зображення (рис. 3.2) були взяті з набору зображень стереопар, що були зроблені в Міддлберійскому коледжі в 2006 році [9] [10].

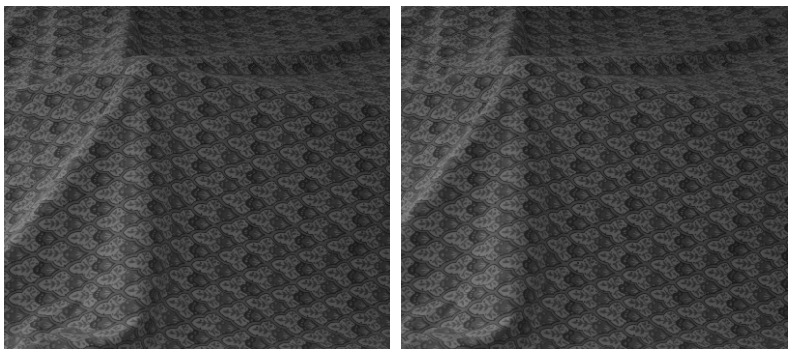
Програмне забезпечення було написано на мові програмування Rust спеціально для цієї роботи.



(а) Карта глибин, отримана алгоритмом дифузії без застосування сегментації зображення

(б) Карта глибин, отримана алгоритмом дифузії після застосування сегментації зображення

Рисунок 3.1 — Практичні результати



(а) Ліве зображення стереопари

(б) Праве зображення стереопари

Рисунок 3.2 — Вхідні зображення

Висновки до розділу 3

Проведено опис способу прискорення розв’язання задачі стереобачення за допомогою алгоритма дифузії. Прискорення алгоритму базується на зменшенні розміру графу, на якому розв’язується задача, так, щоб не втратити багато інформації та не погіршити результуючу карту глибин.

ВИСНОВКИ

В результаті виконання роботи було досліджено алгоритм дифузії для розв’язання задачі стереозору, а також запропоновано метод його прискорення, що використовує сегментацію вхідного зображення.

Було реалізовано програмне забезпечення, що за допомогою алгоритму дифузії будує карту глибин за ректифікованою стереопарою з використанням сегментації та без.

ПЕРЕЛІК ПОСИЛАНЬ

1 A Maximum Likelihood Stereo Algorithm / Ingemar J. Cox, Sunita L. Hingorani, Satish B. Rao, Bruce M. Maggs // *Comput. Vis. Image Underst.* — 1996. — May. — Vol. 63, no. 3. — Pp. 542–567. https://doi.org/10.1006/cviu.1996.0040.

2 Wang, Fuzhi. Stereo Matching Using Iterative Dynamic Programming Based on Color Segmentation of Images / Fuzhi Wang, Changlin Song, Qiang Du // *Journal of Computers*. — 2014. — 06. — Vol. 9.

3 Geiger, Andreas. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite / Andreas Geiger, Philip Lenz, Raquel Urtasun // Conference on Computer Vision and Pattern Recognition (CVPR). — 2012.

4 Luo, Wenjie. Efficient Deep Learning for Stereo Matching / Wenjie Luo, Alexander G. Schwing, Raquel Urtasun // *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. — 2016. — Pp. 5695–5703.

5 Kolmogorov, Vladimir. Graph Cut Algorithms for Binocular Stereo with Occlusions / Vladimir Kolmogorov, Ramin Zabih // Handbook of Mathematical Models in Computer Vision. — 2005.

6 Feng, Liting. Superpixel-based graph cuts for accurate stereo matching / Liting Feng, Kaihuai Qin // *IOP Conference Series: Earth and Environmental Science*. — 2017. — 06. — Vol. 69. — P. 012161.

7 Savchynskyy, Bogdan. Discrete Graphical Models — An Optimization Perspective / Bogdan Savchynskyy // *Foundations and Trends® in Computer Graphics and Vision*. — 2019. — Vol. 11, no. 3-4. — Pp. 160–429. http://dx.doi.org/10.1561/0600000084.

8 Шлезингер, М. И. Анализ алгоритмов диффузии для решения оптимизационных задач структурного распознавания / М. И. Шлезингер, К. В. Антонюк // *Кибернетика и системный анализ*. — 2011.

9 Scharstein, Daniel. Learning Conditional Random Fields for Stereo / Daniel Scharstein, Chris Pal. — 2007. — 06.

10 Hirschmüller, Heiko. Evaluation of Cost Functions for Stereo Matching / Heiko Hirschmüller, Daniel Scharstein. — 2007. — 06.