УДК 681.518

DOI: https://doi.org/10.20535/0203-3771472024307705

О. В. Збруцький[1], *д.т.н., професор*, **Т. В. Яременко**[2], *магістр*,
А. О. Краснопольський[3], *магістр*

ОЦІНКА ЕФЕКТИВНОСТІ МЕТОДІВ РОЗПІЗНАВАННЯ ОБРАЗІВ ТА СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ ДЛЯ СИСТЕМ ТЕХНІЧНОГО ЗОРУ МАЛИХ БЕЗПІЛОТНИХ АПАРАТІВ

Ua Системи технічного зору є перспективним доповненням для навігації безпілотних рухомих апаратів завдяки їх здатності розпізнавати, відстежувати та локалізувати навігаційні орієнтири, що є критичним для роботи автономних систем автоматичного керування. Проведений аналіз алгоритмів систем технічного зору для розпізнавання навігаційних орієнтирів і використання в умовах обмежених ресурсів. Встановлено, що алгоритми на основі нейронних мереж демонструють високу швидкість та точність, що робить їх оптимальними для прикладного застосування. Однак їх навчання потребує значних ресурсів і ретельної підготовки даних. Оптимізація після навчання дозволяє ефективно використовувати їх на пристроях з обмеженими обчислювальними ресурсами.

Проведені дослідження та виконані експерименти показали ефективність запропонованого алгоритму сегментації зображень для його використання в системі технічного зору сучасних безпілотних транспортних засобів.

En Technical vision systems are a promising addition to the navigation of unmanned mobile vehicles due to their ability to recognize, track and localize navigation landmarks, which is critical for the operation of autonomous automatic control systems. The analysis of the algorithms of technical vision systems for the recognition of navigation landmarks and use in conditions of limited resources was carried out. It has been established that algorithms based on neural networks demonstrate high speed and accuracy, which makes them optimal for real applications. However, their training requires significant resources and careful data preparation. Optimization after training allows them to be used efficiently on devices with limited computational resources.

The conducted research and performed experiments showed the effectiveness of the proposed image segmentation algorithm for its use in the technical vision system of modern unmanned vehicles.

Вступ

Безпілотні рухомі апарати (БПРА) мають широкі області застосування, які стрімко розширюються. Цьому сприяє і застосування сучасних технологій для їх створення, що забезпечують автономність, малогабарит-

[1] *КПІ ім. Ігоря Сікорського*
[2] *КПІ ім. Ігоря Сікорського*
[3] *КПІ ім. Ігоря Сікорського*

ність, захищеність. Використання їх в умовах дії радіоелектронних перешкод показало вразливість і неспроможність навігації на основі *GPS*. Альтернативою є автономні системи на різних фізичних принципах, у тому числі системи технічного зору (СТЗ). Навігація БПРА є одним із перспективних напрямків застосування СТЗ, у тому числі для безпілотних літальних апаратів (БПЛА). СТЗ пропонують потужний інструментарій для вирішення проблем їх навігації. Основні завдання СТЗ у навігації БПРА полягають у розпізнаванні об'єктів – навігаційних орієнтирів, їх відстеженню та локалізації як джерела інформації для роботи системи автоматичного керування (САК), а також сегментації для побудови траєкторії руху та уникнення перешкод. Особливостями використання СТЗ у БПРА, і особливо у БПЛА, є висока швидкодія, надійність, мінімізація обчислювальних ресурсів та апаратних засобів реалізації.

СТЗ активно використовують для виявлення та уникнення перешкод [1], для картографування та моніторингу [2], пошуково-рятувальних операцій [3]. Незважаючи на широкий спектр застосування та значний прогрес у використанні СТЗ, залишаються проблеми, які недостатньо досліджені, або нерозв'язані. Так, погодні умови можуть значно впливати на здатність системи розпізнавати об'єкти та уникати перешкод. Для обробки зображень і відео у реальному часі потрібні значні обчислювальні ресурси, що створює труднощі для компактних і легких БПЛА. Комплексування даних з різних джерел, таких як *GPS* та інерціальні датчики, для покращення САК залишається складною технічною задачею. Навчання моделей технічного зору та адаптація до нових середовищ потребують значного обсягу даних та обчислювальних ресурсів [4].

Сьогодні СТЗ мають вирішальне значення для розвитку САК БПЛА та БПРА. Проте для досягнення повної автономності та надійності необхідно вирішити низку важливих проблем, включаючи точність роботи у складних умовах, обробку даних у реальному часі, інтеграцію із іншими сенсорами та оптимізацію навчання моделей [5].

Постановка задачі

Метою статті є аналіз алгоритмів СТЗ із розпізнавання орієнтирів для навігації безпілотних апаратів, оцінка їх характеристик при використанні в умовах обмежених ресурсів та габаритів, дослідження алгоритмів сегментації ображень.

Викладення основного матеріалу

*Класифікація сучасних алгоритмів технічного зору***.** Алгоритми для СТЗ можна розбити на три основні категорії:

1. Алгоритми на основі нейронних мереж (АНМ) використовують штучні нейронні мережі, які навчаються на великих наборах даних зображень та відповідних їм підписів (*YOLO, SSD, Faster R-CNN, Mask R-CNN* [6, 7]).
2. Алгоритми на основі шаблонів порівнюють нове зображення з базою даних шаблонів відомих об'єктів (*SIFT* (*Scale-Invariant Feature Transform*), *SURF* (*Speeded Up Robust Features*), *Haar cascade* [8, 9]).
3. Алгоритми на основі геометричних ознак виявляють геометричні ознаки на зображенні та використовують ці ознаки для розпізнавання об'єктів (*HOG* (*Histogram of Oriented Gradients*), *Hough Transform, Contours Detection* [10, 11]).

Властивості алгоритмів цих категорій наведені у табл. 1.

Як бачимо із табл. 1, АНМ суттєво переважають решту категорій такими якостями: гнучкість у виявленні різноманітних об'єктів, універсальність і ефективність для роботи з різноманітними даними [12]; автоматизований процес навчання на великих обсягах даних та автоматизація процесу розпізнавання об'єктів [13]; виявлення складних закономірностей в даних та підвищення точності розпізнавання об'єктів з часом [14, 15].

***Таблиця* 1.**

Властивості алгоритмів різних категорій

Сімейство алгоритмів	Переваги	Недоліки
Нейронні мережі	Висока точність, універсальність, здатність навчатися на нових даних	Висока обчислювальна складність, потреба у великих наборах даних для навчання
Шаблони	Швидкість, простота реалізації	Низька точність для складних об'єктів, потреба у базі даних шаблонів
Геометричні ознаки	Інваріантність до масштабування та повороту, стійкість до шуму	Низька точність для складних об'єктів

*Навчання АНМ***.** Для навчання моделей АНМ необхідно зібрати великий і різноманітний набір даних, який включає зображення із різними навігаційними орієнтирами у різних умовах освітлення та із різних кутів. Наприклад, для навчання моделі розпізнавати 100 унікальних орієнтирів потрібно зібрати тисячі зображень для кожного орієнтира, щоб забезпечити високу точність і надійність моделі.

Навчання моделі на підготовлених даних передбачає налаштування гіперпараметрів, таких як розмір пакета, швидкість навчання та так звані «епохи». Епохи використовуються під час навчання моделей для того, щоб

модель могла пройти через весь тренувальний набір даних кілька разів, що дозволяє їй поступово покращувати свої ваги та зменшувати похибки. Це допомагає забезпечити стабільне та глибоке навчання моделі, підвищуючи її здатність до точного розпізнавання патернів у даних.

Методи оптимізації АНМ. Після навчання модель оптимізується для роботи на мобільних пристроях та вбудованих системах за допомогою кількох методів. Квантування дозволяє зменшити розрядність параметрів моделі, наприклад, із 32-бітних до 8-бітних. Це значно зменшує розмір моделі і прискорює її виконання без значної втрати точності. Обрізка моделі передбачає видалення маловажливих нейронів або зв'язків у моделі. Це зменшує складність моделі, підвищуючи швидкість обробки та зменшуючи обсяг необхідної пам'яті.

Апаратне прискорення використовує графічні процесори (*GPU*) або спеціалізовані процесори для глибокого навчання (*TPU*) для прискорення обчислень, пов'язаних із навчанням та виконанням моделі. Це значно прискорює процес навчання і виконання, дозволяючи використовувати складніші моделі в реальному часі.

Аугментація даних передбачає використання різних технік для збільшення кількості тренувальних даних шляхом створення варіацій існуючих зображень, таких як обертання, зміна яскравості та масштабування. Це підвищує точність моделі за рахунок збільшення різноманітності тренувальних даних.

Розпаралелювання обчислень включає розподіл обчислювальних задач між кількома процесорами або ядрами процесора для прискорення обробки. Це зменшує час обробки великих наборів даних та підвищує ефективність використання обчислювальних ресурсів [16].

Ці оптимізації знижують вимоги до обчислювальних ресурсів і дозволяють моделям працювати в реальному часі на пристроях з обмеженими ресурсами, що є особливо важливим для безпілотних систем.

Метрики оцінки якості алгоритмів ТЗ. Основними метриками оцінки якості алгоритмів є: точність (*precision*) – відношення кількості правильно розпізнаних об'єктів до загальної кількості об'єктів, які були розпізнані; повнота (*recall*) – відношення кількості правильно розпізнаних об'єктів до загальної кількості реальних об'єктів на зображенні; *mAP* (*Mean Average Precision*) – середнє значення точності, обчисленої для різних рівнів перетину об'єктів на зображеннях, враховує як точність виявлення об'єктів, так і повноту [5]; *F*1 – *score* є гармонійним середнім між *Precision* і *Recall*, що забезпечує баланс між цими двома метриками, коли важливо враховувати обидві [6]; *FLOPs* — міра обчислювальної потужності комп'ютерних систем, зокрема процесорів та графічних процесорів (*GPU*), визначає, скільки операцій із плаваючою комою може виконувати процесор за одну секунду. *FLOPs* використовується для оцінки обчислювальної складності алгоритмі, їх ресурсомісткості і швидкодії на певному

обладнанні. *FLOPs* напряму впливає на швидкість роботи алгоритму. Алгоритми з високими значеннями *FLOPs* потребують більше часу для виконання та більше енергоспоживання, що важливо враховувати при розробці алгоритмів для мобільних та вбудованих систем, таких як БПРА [7].

Методи оцінки якості алгоритмів ТЗ. Для оцінки якості алгоритмів ТЗ використовуються стандартні датасети *COCO* та *ImageNet*. *COCO* є одним із найбільш застосовуваним для задач комп'ютерного зору, що містить понад 200,000 зображень із більш ніж 80 класами об'єктів. Він має детальні анотації, включаючи розмітку об'єктів, ключові точки для людей, сегментацію тощо. I*mageNe*t є великим візуальним датасетом, що містить мільйони зображень, анотованих для тисяч класів об'єктів. Використовується для навчання та оцінки моделей глибокого навчання для задач класифікації та оцінки продуктивності моделей. Процес оцінки алгоритмів на *COCO* та *ImageNet* передбачає попередню обробку зображень, навчання моделей, їх оцінку на валідаційній або тестовій множині, яка не використовувалася під час навчання, та обчислення метрик: для *COCO - mAP* за різних порогах *IoU* (*Intersection over Union*) [8]; для *ImageNet - Top*-1 *Accuracy* (відсоток правильно передбачених класів) та *Top*-5 *Accuracy* (відсоток зображень, для яких правильний клас був серед п'яти класів із найвищою ймовірністю) [9, 10].

Сучасні алгоритми нейронних мереж включають:

YOLO (*You Only Look Once*) – один із найбільш ефективних алгоритмів для розпізнавання об'єктів у режимі реального часу, забезпечує високу швидкість обробки зображень [11].

Faster R-CNN – забезпечує швидке виявлення об'єктів, їх класифікацію та визначення меж [12].

Mask R-CNN є розширенням *Faster R-CNN* на сегментацію об'єктів [13].

MobileNetV2 дозволяє зменшити кількість параметрів і обчислювальних витрат, забезпечує високу ефективність на мобільних пристроях [14].

EfficientNet що одночасно оптимізує ширину, глибину і роздільну здатність мережі та дозволяє досягти кращої продуктивності без значного збільшення обчислювальних витрат [17].

EfficientDet є модифікацією попереднього алгоритму, що дозволяє зменшити кількість параметрів та покращити точність розпізнавання [17].

SqueezeNet значно зменшує кількість параметрів у мережі, що робить його придатним для пристроїв з обмеженими ресурсами [18].

ShuffleNet зменшує кількість обчислень і покращує продуктивність на мобільних пристроях [19].

SSD прогнозує межі об'єктів та їх класів в різних масштабах на різних рівнях просторової роздільної здатності [20].

Порівняння цих алгоритмів (табл. 2) показує, що алгоритм *YOLOv7* має одну з найвищих швидкостей серед розглянутих моделей (60 *FPS*),

***Таблиця* 2.**

Модель	**Кількість параметрів (млн)**	**Розмір (МВ)**	***FLOPs* (млрд)**	***FPS***	***mAP***	***F1-score***
YOLOv3	61,5	236	140	45	57,9	0,76
YOLOv4	64	244	152	30	65,7	0,79
YOLOv7	60	230	150	60	68,1	0,81
Faster R-CNN	42	165	180	10	80,2	0,84
Mask R-CNN	44	170	200	8	81,5	0,86
EfficientDet	12	27	11	40	81	0,85
MobileNetV2	3,4	14	1,3	20	70,6	0,78
EfficientNet	5,3	20	2,4	50	77,1	0,82
SqueezeNet	1,25	0,5	0,8	150	58	0,74
ShuffleNet	5,4	8	2,6	60	69,8	0,79
SSD	34	14	13	20	70,6	0,78

Порівняння сучасних алгоритмів за різними метриками оцінки якості, що робить його придатним для застосувань у системах реального часу, таких як навігація БРПА та БПЛА. *SqueezeNet* має найвищу швидкість (150 *FPS*) серед усіх моделей, однак поступається в точності. *EfficientDet* також демонструє високу швидкість обробки (40 *FPS*), що є достатнім для багатьох реальних додатків.

Найвищу точність обчислень (*mAP* і *F*1-*score*) має алгоритм *Mask R-CNN* (*mAP* 81,5, *F*1-*score* 0,86), що робить його ідеальним для завдань, де потрібна висока точність розпізнавання та сегментації об'єктів. *EfficientDet* демонструє високу точність (*mAP* 81,0) і *F*1-*score* (0,85) і є конкурентоспроможним для точного розпізнавання об'єктів. *YOLOv7* також має високу точність (*mAP* 68,1, *F*1-*score* 0,81) при високій швидкості обробки, що робить його вибір збалансованим, або й оптимальним.

SqueezeNet має найменшу кількість параметрів (0,5 млн), розмір (0,5 МВ) і обчислювальну складність, але й найменшу точність, що є компромісним варіантом для використання на пристроях із дуже обмеженими пам'яттю і обчислювальними ресурсами. *EfficientDet* має помірну кількість параметрів (12 млн) і розмір моделі (27 МВ) і є збалансованим між продуктивністю і точністю. *YOLOv7* має відносно велику кількість параметрів (60 млн) та розмір моделі (230 МВ), але у разі високої точності та швидкості обробки є ефективним для багатьох застосувань.

Експериментальна перевірка розглянутих алгоритмів. Опис експериментальної установки

Алгоритми розпізнавання об’єктів реалізовані на основі бібліотек машинного зору *OpenCV* та *Ultralytics*. Метою експерименту є перевірка характеристик розглянутих алгоритмів, визначення їх метрик якості на локальній машині та порівняння ефективності для об’єктивної оцінки якості алгоритмів. Для експериментів використані треновані моделі алгоритмів на наборі даних *COCO*17. Цей набір включає типові класи об’єктів, такі як люди, тварини чи машини. Цей набір класів достатній для проведення експерименту, так як дає змогу оцінити швидкість та якість роботи алгоритмів, які пройшли відносно однакове навчання. Розглянутий також алгоритм *YOLOv*8 як покращена версія *YOLO*, що пропонує кращу точність та швидкість завдяки вдосконаленій архітектурі і новим методам обробки зображень, але все ще не має офіційного статусу.

Проведення експериментів. Підтвердження тестових даних

Запуск тестів *YOLOv*7 локально на датасеті *COCO*17 дозволяє перевірити продуктивність моделі у конкретному середовищі (табл. 3), що відповідає апаратним ресурсам і налаштуванням. Тут *IoU* – міра перекриття між передбаченою областю об’єкта та реальною областю об’єкта. *IoU* = 0,50 означає, що передбачена та реальна області перекриваються на 50%. *Area* – тип розмірів об’єктів, *maxDets* – максимальна кількість об’єктів, яку модель може виявити.

***Таблиця* 3.**

Локальне тестування моделі *YOLOv*7

Конфігурація параметрів	**Результат**
IoU=0.50 | area= all | maxDets=100	0,697
IoU=0.50:0.95 | area= all | maxDets=100	0,688
IoU=0.50:0.95 | area=medium | maxDets=100	0,736
IoU=0.50:0.95 | area= large | maxDets=100	0,839

Результати, наведені у табл. 3, показують значення, що відповідають паспортній документації. Це свідчить про оптимальність налаштувань та означає, що обраний етап забезпечує стабільну продуктивність і точність детекції об’єктів.

Порівняння метрик якості. Обрані три зображення (рис. 1) із різними групами об’єктів: людей (рис. 1, *а*), тварин (рис. 1, *б*) та автомобілів (рис. 1, *в*). Для їх розпізнавання застосовані готові треновані моделі із *Tensor Flow Zoo* [21]. Їх порівняння з можливостями *YOLO* наведені у табл. 4.



а) група людей *б*) група тварин *в*) група авто

Рис. 1. Зображення для порівняння метрик якості

Оцінка результатів експерименту. Як бачимо у табл. 4, алгоритм *YOLOv7* показав себе як один із найбільш ефективних для розпізнавання об'єктів у режимі реального часу, що робить його зручним для застосування у БПЛА. Він показав високу швидкість й точність, універсальність та ефективність використання ресурсів. Цей алгоритм значно продуктивніший і точний у порівнянні із його попередніми версіями та іншими алгоритмами [11].

***Таблиця* 4.**

Локальне тестування моделей розпізнавання об'єктів

Модель	**Середній час, мс**	**Середня впевненість**	**Середня точність**
*centernet_resnet*101_*v*1_*fpn*	710	79%	80%
*ssd_mobilenet_v*2_*fpnlite*	1114	65%	48%
*efficientdet_d*0	1048	70%	80%
*efficientdet_d*2	1082	80%	83%
*yolov*7-*e*6	309	85%	100%
*yolov*7*x*	347	87%	100%
*yolov*8*n*	280	66%	87%
*yolov*8*l*	376	82%	100%

Сегментація зображень у системі технічного зору

Все більшого поширення набувають технології автономної навігації транспортних засобів. Методи та засоби оптимального визначення траєкторії руху постійно розвиваються. Проте із розвитком інших супутніх галузей, таких як технології машинного навчання із використанням багатошарових нейронних мереж, та мініатюризацією обчислювальних засобів виникає можливість по-новому застосовувати вже створені методики [22, 23, 24].

Сучасні СТЗ можна поділити на 2 великі групи: системи монокулярного та системи бінокулярного технічного зору. Останні за рахунок вико-

ристання двох (іноді більше) камер дають можливість побудови тривимірного зображення навколишнього середовища. Проте при цьому, крім ускладнення структури СТЗ [23], часто неможливо отримати дані із достатньо низьким рівнем похибки для прийняття рішень системою автономної навігації.

Спосіб сегментації зображення камери системи монокулярного технічного зору для побудови алгоритму руху по поверхні з перешкодами розглянемо на прикладі дослідного стенду (рис. 2). Стенд включає: обчислювальний модуль *Raspberry Pi* 5 із обсягом операційної пам'яті 4 Гб та 4-ядерним процесором архітектури *ARM* із тактовою частотою 2,5 ГГц; модуль СТЗ із двома сенсорами *Sony IMX*219 роздільною здатністю 3280x2464 пікселів та діагональним кутом поля зору 83°, які підключені за допомогою інтерфейсу *CSI*; малорозмірний монітор із сенсорним екраном та автономним джерелом живлення із літій-іонними акумуляторами. Стенд дає змогу проводити аналіз даних сегментації у реальному часі.

Рис. 2. Дослідний стенд

Отримане необроблене зображення камер СТЗ наведене на рис. 3.

Рис. 3. Зображення, отримане системою технічного зору дослідного стенду

Підготовка СТЗ до роботи включає калібрування камер за допомогою функцій пакету *OpenCV* – визначення параметрів спотворення оптичної схеми камер, які будуть використовуватись для ректифікації зображень перед безпосереднім аналізом [25]. Для цього використаний типовий шаблон «шахівниця» 9 х 6 перехресть (10 на 7 клітинок). Мінімальна кількість тестових зображень для успішного визначення матриці камери та параметрів спотворення складає 30 шт. Для калібрування використані основні функції:

- *findChessboardCorners* – визначення чи наявна у даному зображенні шахівниця заданого розміру;
- *cornerSubPix* – уточнення координат кожного перехрестя між клітинками шахівниці;
- *drawChessboardCorners* – візуалізація знайдених перехресть на шахівниці;
- *stereoCalibrate* – обчислення матриць перетворення камер та коефіцієнтів спотворень їх оптичних схем;
- *stereoRectify* – обчислення матриць перетворення для ректифікації зображень;
- *initUndistortRectifyMap* – обчислення карт відповідності пікселів для ректифікації зображень.

Фокусування камер здійснюється максимізацією дисперсії матриці, яка утворена шляхом перетворення вхідного зображення на монохромне, та подальшим застосуванням оператора Лапласа.

Алгоритм сегментації включає наступні задачі:

- визначення найближчих до транспортного засобу перешкод у полі зору СТЗ (рис. 4, *а*);
- визначення наявності безпосередньо перед транспортним засобом поверхні для руху;
- вивід результатів сегментації для подальшої обробки у системі прийняття рішень;
- візуалізація даних сегментації.

Для розробки алгоритму використані мова програмування *Python*3, пакет допоміжних функцій обробки зображень *OpenCV* [26] та методи обробки зображень [10, 24, 27]. Використання такого набору засобів розробки дозволяє легко перевіряти гіпотези, які виникають в процесі аналізу зображень, та швидко прототипувати результати для застосування у промислових проектах.

Послідовність алгоритму сегментації:

1. *Визначення відносної глибини оточуючого середовища та формування відповідного зображення.* Використовуємо результати [28], які дають змогу проводити відносну оцінку глибини оточуючого світу за допомо-

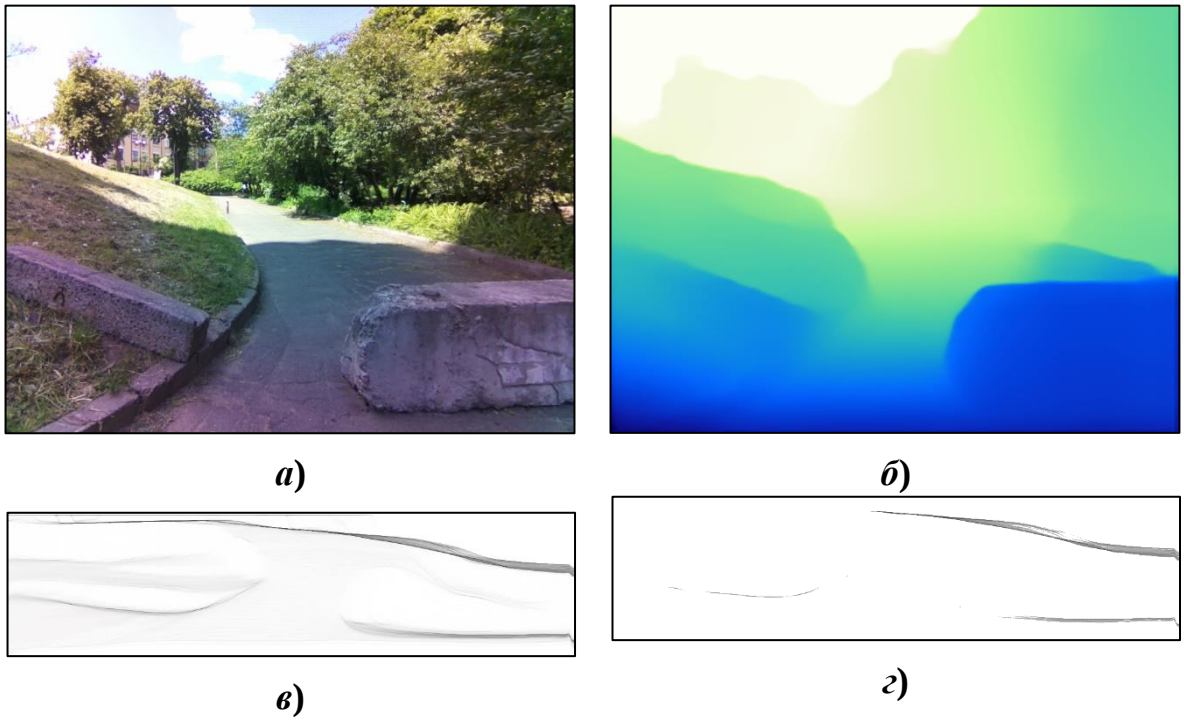


Рис. 4. Зображення СТЗ: *а*) типові перешкоди; *б*) зображення глибини; *в*) горизонтальна гістограма; *г*) горизонтальна гістограма після порогування

гою багаторівневих штучних нейронних мереж із використанням вкрай широкої гами навчальних даних, та найбільш швидкодіючу із навчених моделей *MiDaS v*2.1 *Small*. При використанні обчислювального модуля із прискорювачем ШІ-обчислень, доцільним є використання більших моделей, таких як *MiDaS v*3 *Hybrid* або *MiDaS v*3 *Large*, а також найбільш точної на сьогодні моделі, яка була навчена на понад 62 млн. зображень [29]. Обраний спосіб оцінки глибини вимагає використання лише однієї камери. У результаті обробки вхідного зображення від камери отримуємо монохромне зображення відносної глибини. З метою більш однозначної інтерпретації даних відносної глибини під час подальшого аналізу, необхідно виконати нормалізацію, тобто привести значення отриманої глибини у діапазон значень інтенсивності від 0 до 255, де найбільш близькі об’єкти мають максимальну інтенсивність, а далекі – мінімальну (рис. 4, *б*).

2. *Обчислення горизонтальної гістограми зображення відносної глибини.* Для визначення перешкод у полі зору камери та поверхні для руху користуємось методиками [30]. Побудуємо горизонтальну гістограму зображення СТЗ. Для цього для кожного окремого стовпця нормалізованого зображення глибини обчислюється гістограма насиченості. Для забезпечення оптимальної швидкодії використана функція calcHist бібліотеки OpenCV. Висота отриманої гістограми має 256 рядків (відповідає варіації значень інтенсивності), а ширина дорівнює ширині вихід-

ного зображення. Така гістограма дозволяє відносно легко оцінити наявність умовно-однорідних об'єктів на зображенні (рис. 4, в).

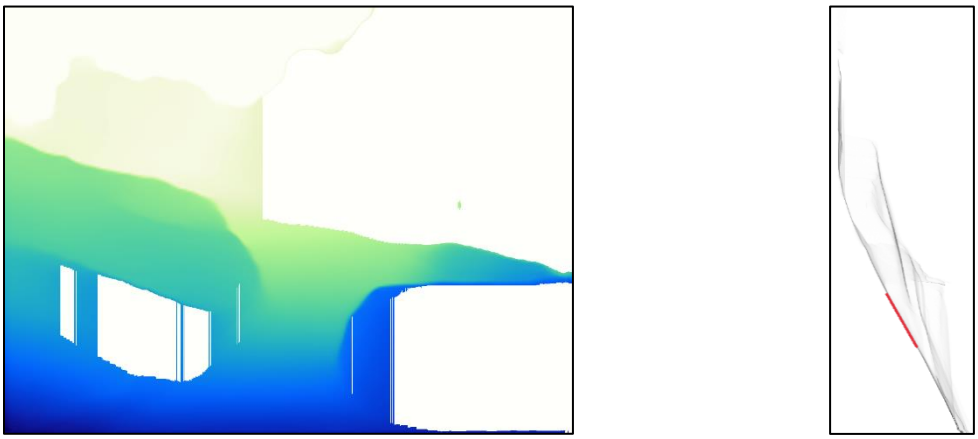
3. *Фільтрація та аналіз горизонтальної гістограми з метою визначення перешкод.* Перешкодами за горизонтальною віссю можна вважати ділянки зображення, для яких на гістограмі спостерігаються горизонтальні лінії певної інтенсивності, при чому інтенсивність ліній на гістограмі безпосередньо залежить від висоти перешкоди, а віддаленість перешкоди визначається вертикальною координатою на гістограмі. Неважко бачити, що найближчим перешкодам (бетонним блокам на зображенні, рис. 4, а) відповідають горизонтальні лінії у нижній частині гістограми (рис. 4, в). Відкинемо далекі перешкоди застосувавши алгоритм порогування до горизонтальної гістограми. В даному контексті під порогуванням мається на увазі комбінований алгоритм, який передбачає фактичне встановлення нулів у рядки горизонтальної гістограми які відповідають найвіддаленішим об'єктам (10% відносної глибини), з подальшим застосуванням класичного алгоритму Оцу [31] для усунення неоднорідних об'єктів. Після порогування маємо гістограму (рис. 3, г) на якій кожен піксель з інтенсивністю відмінною від нуля відповідає інтенсивності перешкоди на зображенні глибини [32].

   Отримані дані можна безпосередньо використати для зонування зображення СТЗ. Для цього визначимо інтервали за горизонтальною віссю в яких відсутні будь-які значення окрім нулів (ігноруємо значення які відповідають 30% найвіддаленіших об'єктів зображення відносної глибини). Такі інтервали будуть відповідати зонам де перешкоди відсутні. Враховуючі можливі похибки під час перетворення зображень знехтуємо несуттєвими за довжиною інтервалами в яких наявні перешкоди (менше ніж 1% пікселів за горизонталлю) та об'єднаємо суміжні інтервали без перешкод.

4. *Маскування зображення відносної глибини.* Використаємо дані, які було отримано на попередньому кроці, для того, щоб отримати зображення глибини з якого будуть видалені зони які відповідають перешкодам. Так, для кожного стовпця зображення глибини, встановимо у нуль значення пікселів для яких у відповідному стовпці горизонтальної гістограми існують значення відмінні від нуля. У результаті отримаємо зображення глибини з якого усунуті усі попереднього визначені перешкоди (рис. 5, *а*), що дасть змогу провести більш точний аналіз наявності горизонтальної поверхні для руху транспортного засобу [30].

5. *Обчислення вертикальної гістограми маскованого зображення відносної глибини.* Для цього побудуємо вертикальну гістограму насиченості залишкового зображення глибини (рис. 5, *б*). Для кожного окремого стовпця залишкового зображення глибини обчислимо гістограму насиченості. Висота отриманої гістограми дорівнює висоті вихідного

зображення, а ширина – 256 стовпців (відповідає варіації значень інтенсивності).



а) ***б***)

Рис. 5. *а*) зображення відносної глибини із видаленими визначеними перешкодами, *б*) його вертикальна гістограма

6. *Фільтрація та аналіз вертикальної гістограми з метою визначення горизонтальної поверхні.* Наявність на вертикальній гістограмі похилих прямих ліній (рис. 5, *б*) дозволяє визначити зони в яких спостерігається горизонтальна площина для руху транспортного засобу [32]. Знайдемо такі лінії застосувавши комбінацію алгоритму Кенні [33] та рандомізованого перетворення Гафа [34]. В результаті отримаємо масив пар координат початків та кінців визначених прямих. Сформулюємо достатні умови наявності необхідної для руху транспортного засобу горизонтальної площини які мають бути накладені на пряму що їй відповідає на вертикальній гістограмі: кут нахилу до горизонталі 75 градусів; довжина у пікселях складає 10 % від кількості рядків гістограми; початок прямої не розташований у верхній частині гістограми (верхні 30 %); кінець прямої розташований нижче її початку.

Якщо вдалось визначити принаймні одну таку пряму, то вважається, що рух транспортного засобу можливий.

Комбінація зонування та ознаки наявності горизонтальної площини для руху дозволяють визначити можливі напрямки для руху транспортного засобу (рис. 6). Отже результатом роботи системи технічного зору в нашому випадку буде 2 параметри, один із котрих являє собою булеву ознаку наявності горизонтальної площини, а інший – масив діапазонів за координатою *x* системи технічного зору в яких відсутні перешкоди.

Висновки

Алгоритм *YOLOv*7 є оптимальним для застосувань, де потрібна висока швидкість і достатня точність розпізнавання об'єктів у режимі реального

часу. *Mask R-CNN* може бути використаний для задач, де важлива висока точність розпізнавання та сегментації об'єктів, але менш важлива швидкість обробки. *EfficientDet* є збалансованим алгоритмом для завдань, де потрібна висока точність і достатня швидкість. *SqueezeNet* є хорошим алгоритмом для пристроїв із дуже обмеженими ресурсами. Для мобільних додатків та систем із обмеженими обчислювальними ресурсами також можуть бути прийнятними алгоритми *MobileNetV2* та *ShuffleNet*, маючи полегшену модель та даючи баланс між точністю та швидкістю.

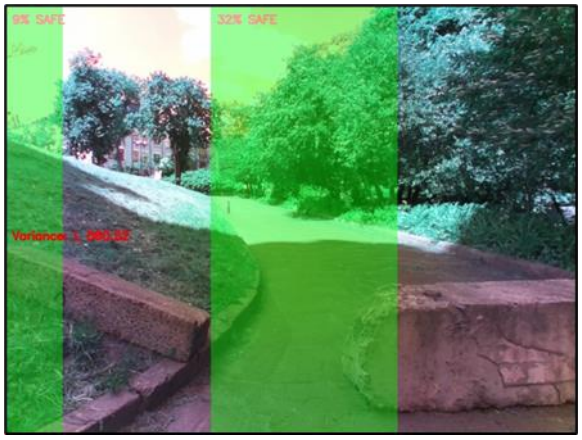


Рис. 6. Візуалізація результатів сегментації

Проведені дослідження та виконані експерименти показали ефективність запропонованого алгоритму сегментації зображень для його використання в системі технічного зору сучасних безпілотних транспортних засобів. Перевагою практичної реалізації алгоритму у реальному часі є застосування малогабаритних, швидкодіючих та бюджетних обчислювальних модулів, використання досягнень мови програмування *Python* та бібліотеки для роботи із системами машинного навчання *PyTorch* із використанням нейронних мереж.

Список використаних джерел

1. *Zhefan Xu, Yumeng Xiu, Xiaoyang Zhan, Baihan Chen, Kenji Shimada*. Vision-aided UAV navigation and dynamic obstacle avoidance using gradient-based B-spline trajectory optimization// [Online]. 15. 09. 2022. Available: https://arxiv.org/abs/2209.07003 7 p.
2. *Xin Wu, Wei Li, Danfeng Hong, Ran Tao, Qian Du*. Deep Learning for UAV-based Object Detection and Tracking: A Survey// [Online]. 25. 10. 2021. Available: https://arxiv.org/abs/2110.12638 24 p.
3. *Martin Messmer, Benjamin Kiefer, Leon Amadeus Varga*, Andreas Zell. UAV-Assisted Maritime Search and Rescue: A Holistic Approach// [Online]. 21. 03. 2024. Available: https://arxiv.org/abs/2403.14281 8 p.

4. *Yulin Wang, Yizeng Han, Chaofei Wa*ng, *Shiji Song, Qi Tian, Gao Huang*. Computation-efficient Deep Learning for Computer Vision: A Survey// [Online]. 27. 08. 2023. Available: https://arxiv.org/abs/2308.13998 47 p.
5. *Joseph Redmon,* Santosh Divvala, *Ross Girshick, Ali Farhadi*. You Only Look Once: Unified Real-Time Object Detection// [Online]. 09. 05. 2016. Available: https://arxiv.org/abs/1506.02640 10 p.
6. *Keiron O'Shea, Ryan Nash*. An Introduction to Convolutional Neural Networks// [Online]. 02. 12. 2015. Available: https://arxiv.org/abs/1511.08458 11 p.
7. *Abolfazl Younesi, Mohsen Ansari, Mohammad Amin Fazli, Alireza Ejlali, Muhammad Shafique, Jörg Henkel*. A Comprehensive Survey of Convolutions in Deep Learning: Applications, Challenges, and Future Trends// [Online]. 28 02 2024. Available: https://arxiv.org/abs/2402.15490 37 p.
8. *Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik*. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation// [Online]. 22. 10. 2014. Available: https://arxiv.org/abs/1311.2524 21 p.
9. *Karen Simonyan,* Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition// [Online] 04. 09. 2014. Available: https://arxiv.org/abs/1409.1556 14 p.
10. *R. Szeliski*. Computer Vision: Algorithms and Applications, 2nd ed., The University of Washington: Springer, 2022.
11. *Chien-Yao Wang, Alexey Bochkovskiy, H. Y. M. Liao*. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors// [Online]. 06. 01. 2021. Available: https://arxiv.org/abs/2207.02696 15 p.
12. Shaoqing Ren, *Kaiming He, Ross Girshick, Jian Sun*. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks// [Online]. 06. 01. 2016. Available: https://arxiv.org/abs/1506.01497 14 p/
13. *Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick*. Mask R-CNN. // [Online] 20.03. 2017. Available: https://arxiv.org/abs/1703.06870 12 p.
14. *Sandler, M., Howard, A., Zhu, M., Zhmoginov, A.,* and *Chen, L*. MobileNetV2: Inverted Residuals and Linear Bottlenecks// [Online] 13. 01. 2018. Available: https://arxiv.org/abs/1801.04381 14 p.
15. *Shiwen Zhao, Wei Wang, Junhui Hou,* Hai Wu. Hybrid Pooling and Convolutional Network for Improving Accuracy and Training Convergence Speed in Object Detection// [Online] 02 01 2024. Available: https://arxiv.org/abs/2401.01134 10 p.
16. *Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, Erich Elsen*. Rigging the Lottery: Making All Tickets Winners// [Online]. 20. 11. 2019. Available: https://arxiv.org/abs/1911.11134 19 p.

17. *Mingxing Tan, Ruoming Pang, Quoc V. Le*. EfficientDet: Scalable and Efficient Object Detection// [Online]. 20. 11. 2019. Available: https://arxiv.org/abs/1911.09070 10 p.
18. *Forrest N. Iandola, Song Han*, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size// [Online]. 24. 02. 2016. Available: https://arxiv.org/abs/1602.07360 13 p.
19. *Zhang, X., Zhou, X., Lin, M.*, and *Sun, J*. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices// [Online]. 04. 07. 2017. Available: https://arxiv.org/abs/1707.01083 9 p.
20. *Wei Liu, Dragomir Anguelov*, et al. SSD: Single Shot Multibox Detector// [Online]. 08 12 2016. Available: https://arxiv.org/abs/1512.02325 17p
21. TensorFlow 2 Detection Model Zoohttps://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md
22. Moghadam P., Sardha W., Moratuwage D. Towards A Fully-Autonomous Vision-based Vehicle// 2011. [Online]. Available: https://www.researchgate.net/publication/224216657_Towards_a_fully-autonomous_vision-based_vehicle_navigation_system_in_outdoor_environments.
23. *Anantharam P*. Disparity Map Computation in Python and C++// 22. 05. 2020. [Online]. Available: https://pramod-atre.medium.com/disparity-map-computation-in-python-and-c-c8113c63d701.
24. *Iloie A., Giosan I., Nedevsch S*. UV disparity based obstacle detection and pedestrian classification in urban traffic scenarios//- 2014.- IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 119-125.
25. OpenCV Open Source Computer Vision. Camera Calibration and 3D Reconstruction// 27. 12. 2023. [Online]. Available: https://docs.opencv.org/4.9.0/d9/d0c/group__calib3d.html.
26. *Kaustubh S*. Stereo Camera Depth Estimation With OpenCV (PythonC++)// 05. 04. 2021. [Online]. Available: https://learnopencv.com/depth-perception-using-stereo-camera-python-c.
27. *Mancini M, Costante G., Valigi P., Ciarfuglia T. A*. Fast Robust Monocular Depth Estimation for Obstacle Detection with Fully Convolutional Networks// 21. 07. 2016. [Online]. Available: https://arxiv.org/abs/1607.06349.
28. *Ranftl R., Lasinger K., Hafner D., Schindler K., Koltun V*. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer//Computer Vision and Pattern Recognition (cs.CV), 2020. [Online]. Available: https://arxiv.org/abs/1907.01341.
29. *Yang L., Kang B., Huang Z., Xu X., Feng J., Zhao H*. Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data// 07. 04. 2024. [Online]. Available: https://arxiv.org/abs/2401.10891. [Дата звернення: 08 05 2024].

30. *Musleh B., De la Escalera A., Armingol J.* U-V Disparity Analysis in Urban Environments// Computer Aided Systems Theory. – EUROCAST 2011. EUROCAST 2011. Lecture Notes in Computer Science, vol 6928, Springer, Berlin, Heidelberg, 2011.
31. *Otsu N.* A Threshold Selection Method from Gray-Level Histograms// IEEE Trans. Syst. Man Cybern.-V. 9, pp. 62-66, 1979.
32. *Yuan J., Jiang T., He X., Wu S.,* Liu J., *Guo D.* Dynamic obstacle detection method based on U–V disparity and residual optical flow for autonomous driving//10. 05. 2023. [Online]. Available: https://doi.org/10.1038/s41598-023-34777-6.
33. *Canny J.* A computational approach to edge detection,» IEEE Transactions on Pattern Analysis and Machine Intelligence, т. 8, № 6, p. 679–698, 1986.
34. *Xu L., Oja E., Kultanen P.* A new curve detection method: Randomized Hough transform (RHT)// Pattern Recognition Letters .-Volume 11, № Issue 5, pp. 331-338, 1990.