МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Комплексна бакалаврська кваліфікаційна робота

на тему: «Розробка системи управління рухомим об'єктом на основі штучної нейронної мережі»

Склад:

Частина 1 «Розробка системи штучного зору для управління рухомим об'єктом»

Виконавець: Шпак Мирослав Ілліч
Керівник: к.т.н., доцент
Перелигін Борис Вікторович

Частина 2 «Розробка координатора для управління рухомим об'єктом»

Виконавець: Дьомін Олександр Володимирович
Керівник: к.т.н., доцент
Перелигін Борис Вікторович

Староста роботи: Дьомін Олександр Володимирович
Провідний керівник проекту: к.т.н., доцент Перелигін Борис Вікторович
Рецензент: Мещеряков Володимир Іванович, д. т. н, професор

Одеса 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Розробка координатора для управління рухомим об'єктом

Виконав студент 4 курсу групи К-45
Спеціальність 122 комп'ютерні науки,
Дьомін Олександр Володимирович

Керівник к.т.н., доцент
Перелигін Борис Вікторович

Консультант

Рецензент д. т. н, професор
Мещеряков Володимир Іванович

Одеса 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Комп'ютерних наук, управління та адміністрування
Кафедра Інформаційних технологій
Рівень вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
С.Д. Кузніченко
"_03_" _____04_____2019 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Дьомін Олександ Володимирович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка координатора для управління рухомим об'єктом
керівник роботи к.т.н., доцент Перелигін Борис Вікторович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "_01_" квітня 2019 року №_49-с_
2. Строк подання студентом роботи 28.05.2019 р.
3. Вихідні дані до роботи Python, Sublime text, opencv

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Машинний зір. основні відомості
2 Використовувані технології
3 Деталі реалізації

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання “ 18 ” квітня 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Вступ	18.04 – 19.04		
2	1 Машинний зір. основні відомості	20.04 – 01.05		
3	2 Використовувані технології	02.05 – 12 05		
4	Рубіжна атестація	13.05 – 19.05		
5	3 Деталі реалізації	20.05 – 27.05		
6	Висновки	28.05		
7				
8				
9				
10				
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)			

Студент _____ **Дьомін О.В.**
(підпис) (прізвище та ініціали)

Керівник роботи _____ **Перелигін Б.В.**
(підпис) (прізвище та ініціали)

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ЕОМ	– електро обчислювальна машина.
МП	– мега піксель.
ООП	– об’єктно орієнтоване програмування.
ПЗ	– програмне забезпечення.
3D-HD	– 3 Dimensional Hight Defination.
CNRI	– Corporation for National Research Initiatives.
GPU	– Graphics Processing Unit.
IBM	– International Business Machines.
JSON	– JavaScript Object Notation.
OpenCV	– OpenSource Computer Vision Library.

ВСТУП

Ми знаємо, що основну частину інформації про зовнішній світ людина отримує за допомогою зору і далі обробляє отриману інформацію за допомогою апарату аналізу та інтерпретації візуальної інформації. Тому ще в минулому столітті постало питання про можливість машинної реалізації даного процесу. Машинний зір – необхідний компонент сучасних технологій. Цей елемент є одним з найбільш перспективних методів автоматизації дій із застосуванням комп'ютерних технологій і робототехніки. Системи машинного зору на увазі перетворення даних, що надходять з пристроїв захоплення зображення, з виконанням подальших операцій на основі цих даних.

В даний час машинний зір найбільш затребуваний в медицині, робототехніці, автомобільної промисловості, військової галузі та біотехнологіях. Це пов'язано з тим, що в цих галузях вже є чітко сформульовані завдання для комп'ютерного зору, вирішенням яких займаються провідні дослідницькі центри і компанії. За рахунок зростання складності розв'язуваних науково-технічних завдань, автоматична обробка і аналіз візуальної інформації стають все більш актуальними питаннями. Крім того, можна сказати, що успіх сучасного бізнесу ґрунтується головним чином на якість пропонованої продукції. А для його забезпечення потрібен візуальний контроль. Так з'явилися цифрові і інтелектуальні камери, а також програмне забезпечення, яке обробляє зображення для виконання всіх перевірок.

Необхідність в машинному зорі виникає в ситуаціях, пов'язаних з ризиком для життя, і буває обумовлена особливостями людини як живого організму, якому властиво швидко втомлюватися, пропускати через себе обмежений обсяг інформації, а також обробляти дані з відносно низькою швидкістю. Звичайно, в ідеалі йдеться про створення універсальної самонавчальної системи, яка б «росла» і «зріла» так само, як це відбувається

з людиною з моменту його народження. Керуючись такими високими цілями, розробники в області комп'ютерного зору сьогодні вирішують непрості завдання. Можна сказати, що область комп'ютерного зору має недовгу за мірками фундаментальних наук, але дуже бурхливу історію зародження і розвитку.

Дана бакалаврська кваліфікаційна робота складається з вступу, переліку скорочень, 3-х розділів, висновків, переліку джерел посилання та 2-х додатків. ПЗ складається з 66 сторінок, 13 рисунків.

1 МАШИННИЙ ЗІР. ОСНОВНІ ВІДОМОСТІ

1.1 Історія розвитку машинного зору

Історія комп'ютерного зору як науки бере свій відлік з 50-х років XX століття. Саме в цей період комп'ютери почали ставати загальнодоступним засобом обробки і аналізу інформації. Однак слід зазначити, що перші системи оцифровки візуальної інформації були досить примітивними, а зображення – малоформатними [1][1)]. В історії розвитку машинного зору можна виділити наступні важливі роки: 1958 рік – вчений Френк Розенблат з університету Корнеля створив комп'ютерну реалізацію персептрона. Це пристрій моделювали схему розпізнавання образів людським мозком. Апаратний варіант Mark I Perceptron (рис. 1) був побудований в 1960 році і призначався для розпізнавання зорових образів.

Рисунок 1 – Mark I Perceptron

[1)] [1] Системы машинного зрения. История, примеры, планы. URL: https://robotics.ua/shows/modernity/5844-sistemy_mashinnogo_zreniya_istoriya_primery_plany (дата звернення 11.05.2019).

Однак розгляд завдань машинного зору носило скоріше умоглядний характер, так як ні техніки, ні математичного забезпечення для вирішення таких складних проблем ще не було. 1960-ті роки – з'явилися перші програмні системи обробки зображень, стали розвиватися прикладні дослідження в області розпізнавання друкованих символів. Однак все ще існували обмеження у розвитку даної галузі науки, такі як відсутність дешевих оптичних систем введення даних, обмеженість і досить вузька спеціалізація обчислювальних систем [1][1)].

До початку 60-х років завдання комп'ютерного зору в основному охоплювали область космічних досліджень, які вимагали обробки великої кількості цифрової інформації. 1970-ті роки – Лауренс Робертс, аспірант MTI, висунув концепцію машинного побудови тривимірних образів об'єктів на основі аналізу їх двовимірних зображень. На даному етапі став проводитися більш глибокий аналіз даних. Почали розвиватися різні підходи до розпізнавання об'єктів на зображенні, наприклад структурні, прізнакові і текстурні.

1979 рік – професор Ганс-Хельмут Нагель з Гамбурзького університету заклав основи теорії аналізу динамічних сцен для розпізнавання рухомих об'єктів в потоці. В кінці 1980-х років були створені роботи, здатні більш-менш задовільно оцінювати навколишній світ і самостійно виконувати дії в природному середовищі. 80-ті і 90-ті роки ознаменувалися появою нового покоління датчиків двомірних цифрових інформаційних полів різної фізичної природи. Удосконалення ж технологій виробництва цих датчиків дозволило істотно знизити їх вартість, а значить, значно розширити область їх застосування.

З початку 90-х років в алгоритмічній аспекті послідовність дій ПЗ обробці зображення прийнято розглядати в згоді з так званої модульної парадигмою. Ця парадигма, запропонована Д. Марром, стверджує, що

[1)] [1] Системы машинного зрения. История, примеры, планы. URL: https://robotics.ua/shows/modernity/5844-sistemy_mashinnogo_zreniya_istoriya_primery_plany (дата звернення 11.05.2019).

обробка зображень повинна спиратися на кілька послідовних рівнів висхідній інформаційної лінії: від неструктурованою форми до їх символічного поданням (векторні і атрибутивні дані в структурованій формі, реляційні структури і т.п.). В середині 90-х років з'явилися перші комерційні системи автоматичної навігації автомобілів. Трохи пізніше в кінці 20-го століття вдалося розробити ефективні засоби комп'ютерного аналізу рухів[1][1)].

2003 рік – на ринок були випущені перші досить надійні корпоративні системи розпізнавання осіб. До теперішнього моменту теорія комп'ютерного зору повністю склалася як самостійний розділ кібернетики, що спирається на солідну наукову і практичну базу знань. Щорічно з даної тематики видаються сотні книг і монографій, проводяться десятки конференцій і симпозіумів, випускається різне програмне та апаратно-програмне забезпечення.

1.2 Області застосування машинного зору

У сучасному світі машинний зір використовується практично у всіх технологічних процесах. Найбільш яскравими прикладами добре фінансованих наукових центрів з розвитку машинного зору можуть служити Лабораторія Штучного Інтелекту Массачусетського Технологічного Інституту (MIT Artificial Intelligence Laboratory), UC Berkeley Computer Vision Group, Vision and Autonomous Systems Center Університету Корнеги-Меллона, Stanford Vision Laboratory [2][2)].

Основними глобальними виробниками систем машинного зору є компанії Cognex, Sensopart, Visionics, Eyematic і інші. У Росії – SPIRIT, Лабораторія комп'ютерного зору Інституту Інформаційних Технологій, Науково-технічний центр «Модуль», «Промінформ» і ряд інших організацій. Особливо затребувані системи машинного зору в робототехніці.

[1)] [1] Системы машинного зрения. История, примеры, планы. URL: https://robo-tic-s.ua/shows/modernity/5844-sistemy_mashinnogo_zreniya_istoriya_primery_plany (дата звернення 11.05.2019).

[2)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

Експерти вважають, що ці технології – найпростіший спосіб навчити апарати автономних дій в природному світі. Так, за допомогою машинного зору роботи орієнтуються в навколишньому середовищі і допомагають здійснювати відеоконференції. У промисловості машинне зір – популярний метод автоматизації виробництва із застосуванням сучасних роботів.

Застосування промислових роботів на виробництві для переміщення предметів вимагає високої точності позиціонування. У цьому допомагають просунуті системи машинного зору. З їх допомогою роботи здатні проводити автоматичний аналіз предметів. Пристрої захоплення зображень, а також пристрій аналізу та обробки зображень в робочій зоні маніпулятора є головними складовими системи машинного зору. Це не тільки прискорює виконання робіт, але і виключає помилки, обумовлені людським фактором [2][1)].

Найбільш часто технології машинного зору застосовуються в автомобільній, харчової та легкої промисловості. Сучасні цифрові камери випускаються переважно в форматі 3D і можуть сприяти головного комп'ютера виконувати одночасно кілька операцій – наприклад, вимірювати рівень рідини в тарі, зчитувати штрих-код і перевіряти на місці чи кришка.

Досить часто для організації машинного зору потрібні додаткові пристрої. В першу чергу це датчики, які виконують операції, недоступні основного пристрою – камері. Це можуть бути сенсори руху, інфрачервоні, фотоелектричні датчики і т.д. Наприклад, компанія Clearpath Robotics використовує в якості основного елемента бачення в своїх роботах-транспортерахдатчик 2D і 3D LIDAR.

Взагалі цей датчик має широке поширення в автомобільній промисловості. За допомогою нього автономні автомобілі можуть «бачити» свій шлях і уникати перешкод. Виробники KUKA, Kawasaki, ABB, Epson використовують технології машинного зору для точної і ефективної роботи своїх маніпуляторів.

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

Важливу роль машинний зір грає в соціальній і домашньої робототехніці. Системами бачення оснащені навіть роботи-пилососи, наприклад, останній пилосос від Samsung POWERbot VR9000H оснащений бортовою камерою і датчиками, які сканують простір і дозволяють роботові легко орієнтуватися в приміщенні. А останній автоматизований пилосос LG HOM-BOT Turbo + має датчики камери Triple Eye, які відстежують місця, вже очищені раніше. Камера розташована на передній частині пристрою і дозволяє виконувати смарт-функцію під назвою Home-Guard. У iRobot також з'явився новий роботизований пилосос Roomba 980з камерою, яка нахиляється вперед і вгору приблизно на 45 градусів і технологією VSLAM для ефективної побудови карти і логічного слідування за маршрутом.

Окреме досягнення – це інноваційна система OMOTE. Це технологія машинного бачення, яка здатна в режимі реального часу визначити форму особи, розпізнати всі об'єкти на ньому і проектувати точно підібрані зображення. Система може відсканувати попередньо особа людини, потім намітити спеціальні точки для маркування тих областей, на які повинні накладатися зображення. Вражаючий ефект технології досягається за допомогою високоточного проектора [2][1)].

Сервісні роботи також є величезне поле діяльності для машинного зору. Візьмемо для початку область охоронних систем для ідентифікації особи, розпізнавання і відстеження рухомих об'єктів. Яскравий приклад – робот-охоронець Bob від G4S, який ефективно сканує офіси за допомогою 3D датчиків, створюючи відображення всієї навколишнього середовища.

Ще один недавно випущений робот-охоронець Knightscope K3 надає комплексні рішення безпеки як усередині приміщень, так і на відкритому повітрі, завдяки обертовому радару на 360 градусів, камерам високої чіткості, тепловізори, системам розпізнавання номерних знаків, людей і осіб.

Відомий робот телеприсутності VGo може переміщатися завдяки комп'ютерному зору ПЗ будівлях, дозволяючи людям бачити все навколо і

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

спілкуватися з людьми віддалено. Подібним чином діє і робот телеприсутності з машинним зором Beam від Suitable Technologies. Такі комунікаційні роботи, як Iuro, SociBot, Talking-Ally і новинка 2016 року Aido завдяки вбудованим мікрокамеру, штучного інтелекту і датчикам можуть розпізнавати предмети, розрізняти обличчя людей, розмовляти і навіть виявляти емоції.

У багатьох ресторанах, готелях, торгових центрах і офісах вже давно використовуються просунуті роботи для обслуговування клієнтів. Машинний зір в поєднанні з ПЗ дозволяє цим технологіям розмовляти з людьми і відповідати їх проханням. Найбільш яскравий приклад це гуманоїд SoftBank Pepper, який активно поширюється ПЗ Японії і навіть вже ПЗ Європі, працюючи в магазинах і торгових центрах в ролі консультанта. Серед останніх новинок в цій галузі також варто виділити робота Space Genius, призначеного для роботи в супермаркетах, скануючого всі продукти і штрих-коди і відображає відео на дисплеях в HD дозволі.

Істотний попит на комп'ютерний зір спостерігається з боку сільськогосподарських організацій, де необхідна автоматизація діяльності ПЗ візуальному контролю і сортування продуктів. Тут можуть застосовуватися роботи з машинним баченням, такі як CASER, Ladybird, а також Ibex. Останній оснащений цілим масивом камер і датчиків, щоб спритно, маневренно і з більшою автономією працювати на полях [2][1)].

Увагу заслуговуэ сфера безпілотних літальних апаратів, які оснащені технологіями машинного зору для автономного та безпечного польоту. Aeromapper Talon є безпілотником з кращою оптикою в світі – 16 МП камерою з об'єктивом Voigtlander. Поряд з цими технологіями дрон має систему контролю дальності діапазону і передачі даних на великі відстані (20 км). Компанія SNAP Vision також пропонує безпілотники з високотехнологічними компонентами. Так, дрон StitchCAM оснащений запатентованим датчиком зображення V2, який захоплює дані з високою

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

роздільною здатністю. Коптер DJI Phantom 4 є першим споживчим БПЛА з автономною системою позиціонування, що використовує оптичні, ультразвукові датчики і 4 мікрокамери.

Машинний зір відіграє велику роль для військових безпілотників, так як розпізнавання об'єктів за допомогою комп'ютера необхідно для проведення місій спостереження і розвідки. Такими технологіями оснащені військові безпілотники компаній Northrop Grumman, BAE Systems, Boeing і IAI. Візьмемо наприклад БПЛА Northrop Grumman Bat, який оснащений високоточними датчиками і камерами. Він може літати автономно, проводити розвідку, спостереження, рекогносцировку, визначати саморобні вибухові пристрої і працювати спільно з пілотованої авіацією [2][1)].

Відзначається зростання інтересу до систем машинного зору і в сфері медицини. Ці технології можуть виробляти автоматичний аналіз медичних зображень – рентген, томографія, УЗД. Обробка отриманих за допомогою комп'ютерного зору зображень і відео даних може застосовуватися для більш точної постановки діагнозів. Медичні роботи, оснащені системами комп'ютерного бачення, вже довели свою ефективність на практиці. Хірургічна система da Vinci Xi має розширені основні можливості, в тому числі інструменти візуалізації 3D-HD і інтуїтивного руху. Дана технологія бачення забезпечує хірургу збільшене зображення для точного позиціонування під час операції.

За подібним принципом працюють роботи-хірурги CT Sabre від Parallax Innovations і IGAR від CSII. Медичний робот RP-VITA від iRobot використовується лікарями, щоб періодично спостерігати за пацієнтами, що перенесли інсульт у віддалених лікарнях. Цей робот оснащений тридцятьма датчиками, які дозволяють йому пересуватися ПЗ лікарняних коридорах, автоматично уникати зіткнень з медичним персоналом або іншими перешкодами [2][2)].

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

З останніх новинок – робот-хірург STAR, який є першим автономним інструментом для проведення складних хірургічних операцій. Його зорова система спирається на інфрачервоні маркери області оперованого органу. Спеціальна NIRF камера відстежує ці маркування в той час, як 3D-камери записують зображення всього операційного поля. Поки що апарат застосовується на тварин, але можливо в майбутньому очікувати повноцінні операції з його участю і на людину.

1.3 Комп'ютерний зір, історія розвитку

Комп'ютерний зір, обробка зображень і машинний зір – тісно пов'язані області. Але до сих пір точно не визначено, чи є вони розділами однієї чи більш широкою. При детальному аналізі може здатися, що це лише різні назви однієї і тієї ж області. Щоб не виникало плутанини, прийнято розрізняти їх як напрями, сфокусовані на певному предметі вивчення.

Обробка зображень або аналіз зображень, в основному зосереджені на роботі з двомірними зображеннями, тобто як перетворити одне зображення в інше. Наприклад, попіксельно операції збільшення контрастності, операції ПЗ виділенню країв, усунення шумів або геометричні перетворення, такі як Афінні перетворення. Дані операції припускають, що обробказображення діють незалежно від змісту самих зображень.

Комп'ютерний зір зосереджується на обробці тривимірних сцен, спроектованих на одне або кілька зображень. Наприклад, відновленням структури або іншої інформації про тривимірній сцені ПЗ одному або декільком зображень. Комп'ютерний зір часто залежить від більш-менш складних припущень щодо того, що представлено на зображеннях.

Машинний зір зосереджується на застосуванні, в основному промисловому, наприклад, автономні роботи і системи візуальної перевірки та вимірювань. Це означає, що технології датчиків зображення і теорії

[2)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

управління пов'язані з обробкою відеоданих для управління роботом і обробка даних в реальному часі здійснюється апаратно або програмно [2][1)].

Область комп'ютерного зору може бути охарактеризована як молода, різноманітна і динамічна. І хоча існують більш ранні роботи, можна сказати, що тільки з кінця 1970-х почалося інтенсивне вивчення цієї проблеми, коли комп'ютери змогли управляти обробкою великих наборів даних, таких як зображення.

Однак ці дослідження зазвичай починалися з інших областей, і, отже, немає стандартної формулювання проблеми комп'ютерного зору. Також, і це навіть більш важливо, немає стандартної формулювання того, як повинна вирішуватися проблема комп'ютерного зору. Замість цього, існує маса методів для вирішення різних строго певних завдань комп'ютерного зору, де методи часто залежать від завдань і рідко можуть бути узагальнені для широкого кола застосування.

Багато з методів і додатків все ще знаходяться в стадії фундаментальних досліджень, але все більше число методів знаходить застосування в комерційних продуктах, де вони часто складають частину більшої системи, яка може вирішувати складні завдання (наприклад, в області медичних зображень або вимірювання і контролю якості в процесах виготовлення). У більшості практичних застосувань комп'ютерного зору комп'ютери попередньо запрограмовані для вирішення окремих завдань, але методи, засновані на знаннях, стають все більш загальними.

Важливу частину в області штучного інтелекту займає автоматичне планування або прийняття рішень в системах, які можуть виконувати механічні дії, такі як переміщення робота через деяку середу. Цей тип обробки зазвичай потребує вхідних даних, що надаються системами комп'ютерного зору, що діють як відеосенсор і надають високорівневу інформацію про середовище і роботі. Інші області, які іноді описуються як належать до штучного інтелекту і які використовуються щодо комп'ютерного

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

зору, це розпізнавання образів і навчальні методи. В результаті, комп'ютерний зір іноді розглядається як частина області штучного інтелекту або області комп'ютерних наук взагалі.

Ще однією областю, пов'язаної з комп'ютерним зором, є обробка сигналів. Багато методів обробки одновимірних сигналів, зазвичай тимчасових сигналів, можуть бути природним шляхом розширені для обробки двовимірних або багатовимірних сигналів в комп'ютерному зорі.

Однак через своєрідну природи зображень існує багато методів, розроблених в області комп'ютерного зору і не мають аналогів в області обробки одновимірних сигналів. Особливою властивістю цих методів є їх нелінійність, що, разом з багатомірністю сигналу, робить відповідну подобласть в обробці сигналів частиною області комп'ютерного зору [2][1)].

Крім згаданих підходів до проблеми комп'ютерного зору, багато хто з досліджуваних питань можуть бути вивчені з чисто математичної точки зору. Наприклад, багато методів грунтуються на статистиці, методах оптимізації або геометрії. Нарешті, великі роботи ведуться в області практичного застосування комп'ютерного зору – того, як існуючі методи можуть бути реалізовані програмно і апаратно або як вони можуть бути змінені з тим, щоб досягти високої швидкості роботи без істотного збільшення споживаних ресурсів.

1.4 Штучний інтелект. його вплив на розвиток машинного зору

Коли в 1950-х років з'явилося поняття «думаюча машина» (обчислювальна машина з елементами штучного інтелекту), виникла і велика заклопотаність відносно можливостей і шляхів розвитку цієї нової області. З тих пір в масовій культурі часто розігруються сцени повстання наділених інтелектом машин [3][2)].

[1)] [2] Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).

Застосування штучного інтелекту в системах машинного зору вимагає глибокого навчання машин. В широкому розумінні штучний інтелект – це здатність комп'ютера імітувати людський інтелект. Глибоке машинне навчання дозволяє комп'ютерам діяти без явного програмування, тобто вони можуть вчитися на власному досвіді.

Завдяки кільком знаковим подіям, що сталися за останнє десятиліття, глибоке навчання систем машинного зору з можливості перетворилося в реальність. «Нові методи побудови нейронних мереж, наявність достатньої обчислювальної потужності в графічних процесорах (англ. Graphics Processing Unit, GPU), а також можливість використання« великих даних »відкрили нам шлях до застосування штучний інтелект для обробки зображень (образів)», – каже Олів'є Деспонт (Olivier Despont), що відповідає за розвиток бізнесу в компанії ViDi Systems, швейцарського творця програмного забезпечення для глибокого навчання машин.

Глибоке навчання є вельми багатообіцяючою технологією в порівнянні з традиційними підходами, використовуваними в системах машинного зору. На відміну від традиційних методів із застосуванням програмного забезпечення для обробки зображень, в нових системах використовується підхід, заснований на правилах, «штучний інтелект – це наступний крок, який ми робимо, коли маємо справу з тим, що нелегко охарактеризувати з достатньою точністю і однозначністю, або з нелінійним процесом. У таких випадках для досягнення наступного рівня повторюваності в прийнятті рішень ми повинні впровадити штучний інтелект в машини, – вважає Уоллес Латімер (Wallace Latimer), директор з продажу спеціалізованих оптичних систем в компанії FISBA LLC. – У той час як лінійні алгоритми створюють дуже вузький простір для маневру, поєднання штучний інтелект з глибоким навчанням розширює межі застосування, оскільки воно може надати нам набагато більше варіацій. Завдяки цьому можна ускладнити оцінку:

[2)] [3] Искусственный интеллект и его влияние на машинное зрение. URL: https: //controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).

здійснювати її не тільки за принципом «що таке добре або що таке погано», а й «чому це добре чи погано». З великими можливостями в частині гнучкості ви зможете зосередитися на тому, що дає найбільш ефективний результат у прийнятті рішень, і зменшити число змін ПЗ вхідних параметрів» [3][1)].

В даний час на ринку систем машинного зору існує, принаймні, одна система з глибоким навчанням – ViDi Suite від компанії ViDi Systems. Ця система є першим комерційно доступним програмним забезпеченням, яке виконане на основі глибокого навчання, призначене для аналізу зображень та адаптовано для використання в промисловості. Програмне забезпечення, яке інтегрується зі стандартними бібліотеками обробки зображень, в частині «пізнання світу» діє так, як би це робив дитина.

«Ви не вчіть дитину, використовуючи підхід, заснований на чітких правилах, тобто пояснюючи йому, що таке будинок, – каже Деспонт. – Грунтуючись всього лише на кількох прикладах, наш мозок навіть в ранньому віці може отримати інформацію про те, що робить будинок саме будинком. Наша система працює так само, як і людський мозок».

Програмне забезпечення ViDi Suite складається з трьох інструментів. Перший, ViDi Blue, знаходить і виявляє окремий об'єкт або відразу кілька всередині загального зображення. Цей інструмент локалізує і ідентифікує складні і прості об'єкти, вивчаючи їх за допомогою наявних в його розпорядженні анотованих зображень. Інший інструмент, ViDi Red, виявляє аномалії, аналізуючи нормальний зовнішній вигляд об'єкта, включаючи його варіації, а також сегментує певні області в зображеннях. Нарешті, третій з них, ViDi Green, використовується для класифікації об'єктів. Він навчається розділяти об'єкти в різні класи на основі бази зібраних і вже описаних зображень.

Ще однією важливою перевагою використання технології глибокого навчання в порівнянні з традиційними рішеннями, застосовуваними в

[1)] [3] Искусственный интеллект и его влияние на машинное зрение. URL: https: //controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).

системах машинного зору, є те, що вона може скоротити час, необхідний для розробки програм машинного зору. «При використанні підходу з класичним баченням багато додатків потребують як мінімум в двох місцях розробки і налагодження їх програмного забезпечення, зазначає Деспонт. Використаня продуктів ViDi дозволяє завершити всю розробку буквально за лічені години».

На відміну від систем штучний інтелект, які використовують серверні ферми як основу для свого програмного забезпечення (наприклад, як системи, розроблені компаніями Facebook, Google і IBM), системи компанії ViDi навчаються за допомогою одного високопродуктивного графічного процесора компанії NVIDIA. Причому весь процес, за словами Олів'є Деспонта, займає лічені хвилини, а не дні або місяці, які потрібні для програмування і записи вихідної інформації в параметричної формі за допомогою програмного забезпечення IBM Watson [3][1)].

«Замість того, щоб використовувати мільйони чи мільярди зображень для навчання системи, ми рекомендуємо починати з 30-50 репрезентативних зображень високої якості, – говорить Деспонт. – При цьому для обробки інформації або навчання ми не відправляємо зображення в хмарну серверну ферму. Завдяки цьому можна запускати всі на одному персональному комп'ютері з одним графічним процесором і зберігати інтелектуальні права власності на свої зображення ».

1.5 Можливості та проблеми штучного інтелекту як частини машинного зору

Глибоке навчання дозволяє використовувати системи машинного зору для набагато більш широкого спектра застосувань. «штучний інтелект добре підходить, наприклад, для аналізу продуктів харчування в загальній масі, коли ви хочете перевірити пончики або шматки м'яса, екземпляри яких

[1)] [3] Искусственный интеллект и его влияние на машинное зрение. URL: https: //controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).

істотно відрізняються один від одного», – зазначає Бруно Менар, менеджер ПЗ програмним продуктам в компанії Teledyne Dalsa[3][1)].

При цьому процес ідентифікації можна буде зробити більш складним. Для пояснення Менар наводить як приклад традиційні додатки виявлення дефектів. «Важко запрограмувати комп'ютер з традиційними алгоритмами для визначення дефекту без необхідності повторювати настройки кожного разу, коли з'являється новий дефект, – сказав він. – Але використовуючи штучний інтелект при великій кількості зразків, ви можете отримати дійсно гарне визначення того, що є якісною деталлю або продуктом, а що шлюбом».

Система машинного зору з впровадженим штучним інтелектом знайде застосування в додаткових завданнях контролю і в підсумку вийде за звичні рамки промислової автоматизації. За словами Латімер, глибоке навчання буде вигідно використовувати на таких ринках, як: медицина, медико-біологічні дослідження, продукти харчування, а також для перевірки продукції на справжність з метою виявлення контрафакту і для сортування пиломатеріалів.

«Це як раз ті галузі, в яких є багато сірих плям і умовностей в прийнятті рішень, – зазначає Латімер. – Наприклад, як оцінити, чи достатньо хороше це яблуко? Важко створити якесь лінійне правило, щоб відповісти на це питання. Глибоке навчання дозволить багатьом додаткам стати більш ефективними і мати високу повторюваність в результаті прийняття ними рішень ».

Зі свого боку, Деспонт з компанії ViDiSystems передбачає, що глибоке навчання буде затребуване і в інших галузях: медичній діагностиці, системах відеоспостереження, автономних транспортних засобах, а також в «розумному» сільському господарстві (для перевірки або аналізу карт). «штучний інтелект – це наше майбутнє, і за допомогою цієї технології ми досить швидко і ефективно допоможемо людям вирішувати складні завдання,

---

[1)] [3] Искусственный интеллект и его влияние на машинное зрение. URL: https: //controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).

оскільки обчислювальні можливості процесорів подвоюються практично кожні півтора року», – говорить Деспонт [3][1)].

Багато фахівців в області систем машинного зору визнають всі переваги і ясно розуміють перспективи, які штучний інтелект і глибоке навчання дають індустрії розпізнавання образів, але вважають, що повний потенціал штуіног інтелекту не буде розкритий і реалізований в повній мірі протягом ще як мінімум трьох-п'яти років. Більш того, ІІ не обов'язково буде єдиним універсальним рішенням для всього того, що вплине на традиційні системи машинного зору і обробку зображень.

Менар також відзначає два основних недоліки в системах штуіного інтелекту. «По-перше, потрібно багато тренувань... і необхідно створити своєрідного експерта для досягнення наступного рівня класифікації, – говорить він. – Другий недолік проявляється, коли система, виконана на базі ІІ, вже навчена, але класифікація зазнає невдачі. Цю проблему важко вирішити. І у вас не буде іншої спроби виконати її перепідготовку з новим зразком ».

Перш ніж штуіний інтелект стане звичайним явищем для систем машинного зору, як вважають експерти, індустрія повинна дозволити набагато більшій кількості гравців зробити перші непрості кроки для впровадження цієї технології.

[1)] [3] Искусственный интеллект и его влияние на машинное зрение. URL: https: //controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).

2 ВИКОРИСТОВУВАНІ ТЕХНОЛОГІЇ

2.1 Мова програмування python. Основні відомості

Створення мови Python починалося досить повільно і невпевнено. Головним ентузіастом, який в 1990 році намагався втілити Python в реальність, став Гвідо Ван Россум. Саме ця людина, працюючи над розробкою мови ABC в Голландському інституті CWI, зрозумів, що хотів би створити щось нове. Це послужило стартом для написання нового інтерпретатора; звичайно, не без використання деяких ідей, взятих з ABC.

Цікавим моментом виступає те, що перший робочий прототип Python був створений на домашньому Макінтош Гвідо, та й ще за пару вихідних. Що стосується поширення, то робилося це за допомогою Інтернет [4][1)].

У 1996 році, коли цей проект набирав критичну масу, до розробки підключився Стів Маєвський, який був досить відомим в мережі, так як вів свій блог «Порівняльна критика мов програмування». Стів, як і Гвідо був шанувальником Macintosh, можливо, це і послужило основою їхньої співпраці. Варто відзначити, що мова отримала назву «Python» не в честь види змій, як помилково вважають багато розробників. За часів розробки «Пітона» Гвідо любив дивитися комедійне шоу «Повітряний цирк Монті Пайтона», тому і назвав своєю проект на честь Монті Пайтона.

Так як python мав відмінний потенціал і вільно поширювався через Інтернет, в нього з'явилося ядро послідовників – люди, які були зацікавлені в розвитку Python як мови програмування. На початку свого шляху, ця мова мав вигляд невеликого інтерпретатора з малою кількістю функцій і повною відсутністю ООП, що всіх не влаштовувало і мотивувало на подальший розвиток мови.

Уже в 1991 році стали з'являтися перші кошти ООП розробки.

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

Через деякий временя, Гвідо запропонували посаду в корпорації CNRI, яка знаходиться в Америці. Недовго думаючи, Гвідо покинув Голландію і взявся за роботу. Займаючись проектами компанії, він часто використовував Python для вирішення багатьох завдань, а у вільний час займався його розвитком як інтерпретатора [4][1)].

За такою схемою Пітон розвивався до 1999 року і отримав версію 1.5.2. Після досягнення цієї планки в житті Гвідо почалися зміни. Компанія все більше завантажувала його роботою, що сильно зменшувало час для улюбленого заняття.

Це спонукало замислитися Гвідо про доцільність такої роботи, в результаті чого, він прийняв рішення шукати спонсора, який дасть можливість працювати тільки над розвитком мови. Так як на той час в інтернеті вже існувало чимало співтовариство користувачів, фірма BeOpen вирішила взяти участь в просуванні Python. За контрактом з CNRI Гвідо зобов'язався випустити версію 1.6, що він і зробив перед відходом. Працюючи з BeOpen він показав світу версію 2.0. Багато хто стверджує, що версія 2.0 дала сильний поштовх в соціальному плані. А все тому, що процес розвитку мови став більш відкритим. Гвідо перевів всі дані на SourceForge, що сильно сподобалося спільноті, яке вимагало впровадження можливості участі в розробці коду. Крім цього в той час з'явився Юнікод, а це великий крок вперед. До Юникоде створили новий механізм регулярних виразів,

Через деякий час в компанії BeOpen почалися проблеми. Вони вирішили, що Гвідо повинен працювати старанніше і приносити гроші, а не тільки просити їх, на розвиток проекту. Така поведінка не припало до душі Гвідо – він звільнився і почав міркувати куди йти далі. У багатьох інтерв'ю, Гвідо розповідає, що цей фрагмент життя був переломний.

Цього разу, своє фінансування йому запропонувала компанія Digital Creations – автори Zope. Як не дивно, але це було п'ята пропозиція від них, на

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

що Гвідо погодився. У цій компанії вся команда розробників «пітона» отримала великі можливості, що дало плоди. У тому ж році був випущений версія 2.1. Тепер в Пітоні з'явилися нові об'єкти з мов closures і ієрархія: функції можна вкладати один в одного, зберігаючи при цьому доступ до змінних оточуючих функцій. Це сильно змінить мову, а головне сильно поліпшить його підходи до способу програмування. На даний момент існує версія 3.5.1, що демонструє його розвиток, адже щороку розробники проробляють величезну роботу. Все це перетворило простий інтерпретатор в дуже популярна мова програмування, який використовується як перший в навчанні мільйонів студентів ПЗ всьому світу [4][1)].

Python 1.0 з'явився в січні 1994 року. Основними новими можливостями, включеними в цей реліз, були кошти функціонального програмування: лямбда-числення, map, filter та згортка списку. Ван Россум стверджував, що «Python придбав lambda, reduce (), filter () і map () завдяки любителю Lisp, якому їх не вистачало, і він надав патчі, що реалізують ці функції».

Останньою версією, випущеної Ван Россум під час роботи в центрі математики та інформатики, був Python 1.2. З 1995 року Ван Россум продовжив роботу над Python-му в корпорації національних дослідницьких ініціатив в місті Рестон, штат Вірджинія, де було випущено кілька версій мови.

До версії 1.4 Python включав в себе безліч нових функцій, серед яких найбільш помітними були запозичені в Modula-3 іменовані параметри і вбудована підтримка комплексних чисел. Також в 1.4 з'явилася проста форма приховування даних за допомогою namemangling.

У 2000 році ядро команди розробників Python перейшло в BeOpen.com, сформувавши команду BeOpenPythonLab. Python 2.0 був єдиним релізом

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

BeOpen.com. Після нього Ван Россум і інші розробники PythonLab приєдналися до DigitalCreations.

У версії Python 2.0 з'явилося спискові включення це функція, запозичена з функціональних мов програмування SETL і Haskell. Синтаксис в Python для цієї конструкції дуже схожий на Haskell, за винятком того, що в Haskell вважали за краще використовувати розділові знаки розташовуються, а в Python – ключові слова. Також в Python 2.0 була додана система збирання сміття з підтримкою циклічних посилань [4][1)].

Починаючи з альфа релізу Python 2.1 весь код, технічна документація і специфікації належать не комерційної організації Python Software Foundation (PSF), створеноїв 2001 році за зразком Apache Software Foundation. Реліз включав зміна в специфікацію мови, що підтримує вкладені області видимості, як в мовах із статичною (лексичної) областю видимості.

В Python 2.2 було об'єднання базових типів Python і класів, що створюються користувачем, в одній ієрархії. Це зробило Python повністю об'єктно-орієнтованою мовою.

Python 3.0 (званийтакож "Python 3000" або "Py3K") розроблявся з метою усунення фундаментальних вад у мові. Ці зміни не могли бути зроблені за умови збереження повної сумісності з 2.x версією, тому треба буде зміна головного номера версії. Python 3.0 був випущений 3 грудня 2008 року.

Працюючи з Python, варто мати на увазі, що Python поширений не тільки у вигляді мови, на якому пишуться кінцеві проекти: ця мова часто використовується для автоматизації різних завдань, скажімо, з системного адміністрування (Ansible – один із прикладів написаного на Python, яке використовується при адмініструванні). Тому python часто вже присутній в встановлених операційних системах.

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

Особливо велика ймовірність виявити в системі встановлений Python, коли мова йде про ОС сімейства Linux – а саме, якась з версій Linux буде швидше за все встановлена на машину, на якій ви захочете запустити сервер свого web-додатки або багатокористувацької гри (так, такі теж пишуть на Python!). Тут-то і криється проблема: встановлений разом з операційною системою Python може бути тим самим "другим Пайтон" – так, таке, на жаль, поки зустрічається. І замінити його буде не можна, адже заміна може привести до виходу з ладу всієї ОС.

Для Python є популярні бібліотеки і фреймворки машинного навчання. Дві найбільші з них – scikit-learn і TensorFlow. У scikit-learn вбудовані деякі загальновідомі алгоритми машинного навчання, про які йшла мова вище. TensorFlow – більш низкоуровневая бібліотека, яка дозволяє будувати призначені для користувача алгоритми.

Розробники, яким в їх роботі потрібно застосовувати прийоми статистики або аналізу даних, часто вибирають Python в якості своєї мови. Також він використовується вченими науки про дані – при інтеграції їх завдань з веб-додатками або виробничими середовищами [4][1)].

Але в сфері машинного навчання Python просто «зірка». Комбінація послідовного синтаксису, більш короткого часу розробки і гнучкості робить цю мову відповідним для розробки хитромудрих моделей і Предсказательная движків, які можна безпосередньо впроваджувати в продакшен-системи.

2.2 Бібліотеки python для машинного навчання

Одне з найбільших переваг Python – широкий спектр бібліотек.

Бібліотеки є наборами підпрограм і функцій, написаних цією мовою. Хороший комплект бібліотек може полегшити здійснення складних завдань без необхідності написання багатьох рядків коду.

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

Машинне навчання багато в чому грунтується на математиці. Зокрема, на математичної оптимізації, статистики та теорії ймовірності. Бібліотеки Python допомагають дослідникам / математикам «займатися машинним навчанням», навіть не маючи значних пізнань в розробці.

Scikit-learn це одна з найпопулярніших бібліотек машинного навчання. Вона підтримує багато контрольованих і неконтрольованих алгоритмів навчання. Наприклад, лінійні і логістичні регресії, дерева прийняття рішень, кластеризації, k-means та т.д [4][1)].

Вона створена на основі двох головних бібліотек Python – NumPy і SciPy. У Scikit-learn доданий набір алгоритмів для поширених завдань машинного навчання і видобутку даних, включаючи кластеризації, регресію і класифікацію. Навіть такі завдання як перетворення даних і вибір функцій можуть бути реалізовані за допомогою всього декількох рядків.

Відносно Tensorflow цікаво те, що при написанні програми на Python можна компілювати і запускати програму як на CPU, так і на GPU. Таким чином для запуску на GPU вам не доводиться писати на C ++ або на рівні CUDA. Бібліотека використовує систему багаторівневих вузлів, яка дозволяє вам швидко налаштовувати, навчати і розгортати штучні нейронні мережі з великими наборами даних. Саме це дозволяє Google визначати предмети на фотографіях і розуміти вимовлені слова в додатках для розпізнавання усного мовлення.

Theano це ще одна хороша бібліотека Python для алгоритму числового розрахунку, схожа на NumPy. Вона дозволяє вам ефективно визначати, оптимізувати і обчислювати математичні вирази, що містять багатовимірні масиви. Виділяє цю бібліотеку використання переваг GPU комп'ютера. Це дозволяє їй робити обчислення з великими обсягами даних в сто разів швидше, ніж при запуску тільки на CPU. Швидкість Theano особливо цінна

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

для глибокого навчання та інших завдань, пов'язаних зі складними обчисленнями.

Останній реліз бібліотеки Theano вийшов в минулому, 2017 році. Це версія 1.0.0 з безліччю нових функцій, змінами інтерфейсу та іншими поліпшеннями.

Pandas це дуже популярна бібліотека, що надає високорівневі структури даних, прості у використанні і інтуїтивно зрозумілі. У ній є багато вбудованих методів для угруповання, комбінування даних і їх фільтрації, а також аналізу часових рядів. Pandas може з легкістю витягувати дані з різних джерел, таких як бази даних SQL, файли CSV, Excel, JSON, і маніпулювати цими даними для здійснення операцій з ними [4][1)].

Matplotlib стандартна бібліотека Python, яку використовують для створення графіків все люди, що займаються наукою про дані. Вона досить низкоуровневая, а значить, вимагає більше команд для генерації добре виглядають графіків і фігур, ніж більш просунуті бібліотеки.

З іншого боку, вона гнучка. Оперуючи достатньою кількістю команд, ви зможете створити практично будь-який графік. Ви можете будувати різноманітні діаграми, від гістограм і діаграм розсіювання до графіків з не-декартовими координатами. Ця бібліотека підтримує GUI-бекенд у всіх операційних системах, а також може експортувати графіки в поширених форматах (PDF, SVG, JPG, PNG, BMP, GIF).

Seaborn це популярна бібліотека візуалізації, яка будує графіки на основі Matplotlib. Це бібліотека вищого рівня, а значить, з її допомогою простіше генерувати певні види графіків, в тому числі теплові карти, тимчасові ряди і скрипкові графіки.

[1)] [4] История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/langua ge_ python.html (дата звернення 11.05.2019).

2.3 Бібліотеки python використані в проекті

2.3.1 Основні відомості про бібліотеку python socket

Сокет (англ. Socket – роз'єм) – назва програмного інтерфейсу для забезпечення обміну даними між процесами. Процеси при такому обміні можуть виконуватися як на одній ЕОМ, так і на різних ЕОМ, пов'язаних між собою мережею. Сокет – абстрактний об'єкт, який представляє кінцеву точку з'єднання.

Слід розрізняти клієнтські і серверні сокети. Клієнтські сокети грубо можна порівняти з кінцевими апаратами телефонної мережі, а серверні – з комутаторами. Клієнтську програму (наприклад, браузер) використовує тільки клієнтські сокети, а серверне (наприклад, веб-сервер, якому браузер посилає запити) – як клієнтські, так і серверні сокети.

Інтерфейс сокетів вперше з'явився в BSD Unix. Програмний інтерфейс сокетів описаний в стандарті POSIX.1 і в тій чи іншій мірі підтримується всіма сучасними операційними системами

Для взаємодії між машинами за допомогою стека протоколів TCP / IP використовуються адреси та порти. Адреса являє собою 32-бітну структуру для протоколу IPv4, 128-бітну для IPv6. Номер порту – ціле число в діапазоні від 0 до 65535 (для протоколу TCP).

Цю пара визначає сокет («гніздо», відповідне адресою і порту).

У процесі обміну, як правило, використовується два сокета – сокет відправника і сокет одержувача. Наприклад, при зверненні до сервера на HTTP -порт сокет буде виглядати так: 194.106.118.30:80, а відповідь буде надходити на mmm.nnn.ppp.qqq: xxxxx.

Кожен процес може створити «слухає» сокет (серверний сокет) і прив'язати його до якого-небудь порту операційної системи (в UNIX непривілейованих процеси не можуть використовувати порти менше 1024).

Хто слухає процес зазвичай знаходиться в циклі очікування, тобто прокидається при появі нового з'єднання. При цьому зберігається можливість

перевірити наявність з'єднань на даний момент, встановити тайм-аут для операції і т. Д.

Кожен сокет має свою адресу. ОС сімейства UNIX можуть підтримувати багато типів адрес, але обов'язковими є INET-адреса і UNIX-адреса. Якщо прив'язати сокет до UNIX-адресою, то буде створений спеціальний файл (файл сокета) ПЗ заданому шляху, через який зможуть повідомлятися будь-які локальні процеси шляхом читання / запису з нього. Сокети типу INET доступні з мережі і вимагають виділення номера порту.

Зазвичай клієнт явно «приєднується» до слухача, після чого будь-читання або запис через його дескриптор передаватимуть дані між ним і сервером.

Відмінності між сокетом (внутрішнє представлення), дескриптором сокета (абстрактним ідентифікатором) і адресою сокету (загальнодоступною адресою) є тонкими, і вони не з великою обачністю розрізняються в повсякденному використанні. Крім того, конкретні визначення "сокета" розрізняються між авторами і часто стосуються інтернет-сокету або TCP-сокета.

Інтернет-сокет характеризується принаймні наступними характеристиками:

– Локальна адреса сокета, що складається з локальної IP-адреси і (для TCP і UDP, але не IP) номера порту

– Протокол: транспортний протокол, наприклад, TCP, UDP, raw IP. Це означає, що (локальні або віддалені) кінцеві точки з TCP-портом 53 і UDP-портом 53 є окремими сокетами, в той час як IP не має портів.

Сокет, який був підключений до іншого сокета, наприклад, під час встановлення з'єднання TCP, також має адресу віддаленого сокета. У операційній системі та програмі, яка створила сокет, сокет посилається на унікальне ціле значення, яке називається дескриптором сокета. Операційна система пересилає корисне навантаження вхідних IP-пакетів на відповідний

додаток, шляхом вилучення інформації адреси сокету з заголовків IP-адрес і транспортного протоколу, та видалення заголовків з даних програми.

У IETF Запит на коментар, Інтернет-стандарти, у багатьох підручниках, а також у цій статті, термін "сокет" означає об'єкт, який однозначно ідентифікується номером сокета. У інших підручниках, термін сокет відноситься до локальної адреси сокету, тобто "комбінації IP-адреси і номера порту". У вихідному визначенні сокет, наведеному в RFC 147, так як він був пов'язаний з мережею ARPA в 1971 році, "'сокет задається як 32-бітове число для парних сокетів, що ідентифікують приймальні сокети. і непарні сокети, що ідентифікують сокети посилань. '"Сьогодні, однак, зв'язки сокета є двонаправленими.

Доступні кілька типів інтернет-сокетів:

– Datagramsockets, також відомі як сокети без з'єднання, які використовують UserDatagramProtocol (UDP).

– Streamsockets, також відомі як орієнтовані на з'єднання сокети, які використовують Протокол керування передачею (TCP), Stream Control Transmission Protocol (SCTP) або Datagram Congestion Control Protocol (DCCP).

– Raw sockets (або raw IP sockets), як правило, доступні в маршрутизаторах та інших мережевих пристроях. Тут транспортний рівень обходиться, і заголовки пакетів стають доступними для програми, і немає номера порту в адресі, тільки IP-адреса.

На практиці сокет зазвичай посилається на сокет в InternetProtocol (IP) мережі (де сокет може називатися Інтернет-сокетом), зокрема для TransmissionControlProtocol(TCP), який є протоколом для з'єднань один до одного. У цьому контексті передбачається, що сокети пов'язані з певною адресою сокета, а саме IP-адреса і номером порту для локального вузла, і є відповідна адреса сокету. На зовнішньому вузлі (іншому вузлі), який сам має асоційований сокет, що використовується зовнішнім процесом. Асоціація сокета з адресою сокету називається binding.

Зауважимо, що в той час, як локальний процес може спілкуватися із зовнішнім процесом, надсилаючи або отримуючи дані на або з зовнішньої сокет-адреси, він не має доступу до чужого сокета сам ПЗ собі, і не може використовувати зовнішній сокет. Дескриптор сокета, оскільки вони є внутрішніми для зовнішнього вузла. Наприклад, у зв'язку між 10.20.30.40:4444 і 50.60.70.80:8888 (локальною IP-адресою: локальним портом, зовнішньою IP-адресою: зовнішнім портом) буде також пов'язаний сокет на кожному кінці, відповідний внутрішньому поданню з'єднання стеком протоколів на цьому вузлі. Вони локально називаються дескрипторами числових сокетів, скажімо 317 на одній стороні і 922 на іншій. Процес на вузлі 10.20.30.40 може запросити зв'язатися з вузлом 50.60.70.80 на порту 8888 (запит, щоб стек протоколу створив сокет для спілкування з цією адресою), і як тільки він створив сокет і отримав дескриптор сокета (317), він може спілкуватися через цей сокет за допомогою дескриптора (317). Стек протоколу буде пересилати дані на вузол 50.60.70.80 і з нього на порт 8888. Однак процес на вузлі 10.20.30.40 не може запитувати зв'язок на основі дескриптора зовнішнього сокету (наприклад, «socket 922» або «socket 922 на вузлі» 50.60.70.80 "), оскільки вони є внутрішніми для іншого вузла і не можуть використовуватися стеком протоколів на вузлі 10.20.30.40.

Модуль socket забезпечує можливість працювати з сокетами з Python. Сокетивикористовують транспортний рівень згідно семирівневій моделі OSI (Open SystemsInterconnection, взаємодія відкритих систем).

Кожен сокет відноситься до одного з комунікаційних доменів. модуль socketпідтримує домени UNIX та Internet. Кожен домен має на увазі своє сімействопротоколів і адресацію. Дане виклад стосуватиметься тільки домен Internet, асаме протоколи TCP / IP і UDP / IP, тому для вказівки комунікаційного доменапри створенні сокета буде вказуватися константа socket.AF_INET.

2.3.2 Основні відомості про бібліотеку python numpy

На сайті [5][1)] надається наступне визначення, NumPy – це розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Попередник NumPy, Numeric, був спочатку створений Jim Hugunin. NumPy – відкрите програмне забезпечення, Взяти участь в розробці може будь-який бажаючий.

Оскільки Python – інтерпретована мова, математичні алгоритми часто працюють в ньому набагато повільніше, ніж в компільованих мовах, таких як C і чи Java. NumPy намагається вирішити цю проблему для великого кількості обчислювальних алгоритмів, забезпечуючи підтримку багатовимірних масивів і безліч функцій і операторів для роботи з ними. Таким чином, будь-який алгоритм, який може бути виражений в основному як послідовність операцій над масивами і матрицями, працює так само швидко, як еквівалентний код, що виконується в MATLAB, а після спеціальної оптимізації швидкість може досягти швидкості компільованих мов типу C [5][2)].

NumPy можна розглядати як хорошу вільну альтернативу MATLAB, оскільки мова програмування MATLAB зовні нагадує NumPy: обидва вони інтерпретуються, і обидва дозволяють користувачам писати швидкі програми, поки більшість операцій виробляються над масивами або матрицями, а НЕ над скалярами. перевага MATLAB в великому кількості доступних додаткових тулбоксов, включаючи такі як пакет Simulink. Основні пакети, що доповнюють NumPy, це: SciPy – бібліотека, що додає більше MATLAB – подібної функціональності ; Matplotlib – пакет для створення графіки в стилі MATLAB. Внутрішньо як MATLAB, Так і NumPy засновані

[1)] [5] NumPy – это… Что такое NumPy? URL: https://dic.academic.ru/dic.nsf/ruwik-/1271758 (дата звернення 11.05.2019).

[2)] [5] NumPy – это… Что такое NumPy? URL: https://dic.academic.ru/dic.nsf/ruwik-/1271758 (дата звернення 11.05.2019).

на бібліотеці LAPACK, призначеної для вирішення основних завдань лінійної алгебри.

NumPy заснований на двох більш ранніх пакетах для Python. Спочатку був Numeric, цілком стабільний і повний, доступний ПЗ сей день, але застарілий. Він був написаний в 1995 році програмістом Jim Hugunin при участі багатьох людей, серед яких Jim Fulton, David Ascher, Paul DuBois, і Konrad Hinsen. Більш нова версія під назвою Numarray це повністю переписаний Numeric який тепер теж не рекомендується до використання NumPy об'єднує в собі ці два пакета, він побудований на базовому коді Numeric і доповнений можливостями Numarray.

Були висловлені побажання додати Numeric в стандартну бібліотеку Python, але Гвідо Ван Россум (автор Python) чітко дав зрозуміти що код в його тодішньому стані було неможливо підтримувати. Інша проблема полягала в тому що на великих масивах Numeric був дуже повільним. В Внаслідок був створений Numarray. Він швидше на великих масивах, але повільніше на малих. Деякий час використовувалися обидва пакета. Остання версія Numeric v24. 2 була випущена 11 Ноября 2005 року, а остання версія numarray v1. 5. 2 вийшла 24 серпня 2006 [5][1)].

В початку 2005 Travis Oliphant захотів об'єднати співтовариство навколо одного пакета матричних обчислень. Код Numeric був переписаний так щоб його було легше підтримувати і щоб він міг включити нові можливості Numarray. Цей новий проект був частиною SciPy. Щоб НЕ завантажувати весь пакет SciPy заради створення масиву, NumPy був виділений в окремий пакет. Хоча вихідний код знаходиться в відкритому доступі і містить велику кількість документації, є також докладний Путівник ПЗ NumPy. NumPy version 1. 3. 0, випущений 5 Апреля 2009 року і підтримує Python 2. 6. Підтримка Python 3 реалізована починаючи з версії 1. 5. 0.

NumPy можна розглядати як вільну альтернативу MATLAB. Мова програмування MATLAB зовні нагадує NumPy: обидва вони

[1)] [5] NumPy – это… Что такое NumPy? URL: https://dic.academic.ru/dic.nsf/ruwik-/1271758 (дата звернення 11.05.2019).

інтерпретуються, обидва дозволяють виконувати операції над масивами (матрицями), а не над скалярами. Перевага MATLAB в наявності великої кількості пакетів («тулбоксов»), наприклад, Simulink (англ.). Для NumPy теж існують подібні «пакети», наприклад, бібліотека SciPy надає більше MATLAB-подібної функціональності, бібліотека Matplotlib дозволяє створювати графіки в стилі MATLAB. І MATLAB, і NumPy для вирішення основних завдань лінійної алгебри використовують код, заснований на коді бібліотеки LAPACK.

2.3.3 Основні відомості про бібліотеку python base64

У форматі електронної пошти MIMEbase64 – це схема, ПЗ якій довільна послідовність байт перетвориться в послідовність друкованих ASCII символів. Використовуються тільки символи латинського алфавіту в верхньому і нижньому регістрі – символи (A-Z, a-z), цифри (0-9), і символи «+» і «/», з символом «=» в якості спеціального коду суфікса.

Повна специфікація цієї форми base64 міститься в RFC 1421 і RFC 2045. Ця схема використовується для кодування послідовності октетів (байт).

Для того, щоб перетворити дані в base64, перший байт поміщається в найстарші вісім біт 24-бітного буфера, наступний – в середні вісім і третій – в молодші значущі вісім біт. Якщо кодується менш, ніж три байта, то відповідні біти буфера встановлюються в нуль. Далі кожні шість біт буфера, починаючи з найстарших, використовуються як індекси рядка:

«ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345 6789 + /»

і її символи, на які вказують індекси, поміщаються у вихідний рядок. Якщо кодуються тільки один або два байти, в результаті виходять тільки перші два або три символи рядка, а вихідна рядок доповнюється двома або

одним символами «=». Це запобігає додавання додаткових бітів до відновлених даних. Процес повторюється над рештою вхідними даними.

Наприклад, цитата з "Левіафана" Томаса Гоббса :

Людина відрізняється не тільки своїм розумом, але й цією особливою пристрастю від інших тварин, яка є прагненням розуму, що завдяки наполегливості насолоди в тривалому і невгамовному поколінні знань перевищує коротку жвавість будь-якої плотської насолоди.

Будучи перекодованої з ASCII в base64, виглядає наступним чином:

TWFuIGlzIGRpc3Rpbmd1aXNoZWQsIG5vdCBvbmx5IGJ5IGhpcyByZWFzb24sI
GJ1dCBieSB0aGlzIHNpbmd1bGFyIHBhc3Npb24gZnJvbSBvdGhlciBhbmltYWxzLCB3a
GljaCBpcyBhIGx1c3Qgb2YgdGhlIG1pbmQsIHRoYXQgYnkgYSBwZXJzZXZlcmFuY2U
gb2YgZGVsaWdodCBpbiB0aGUgY29udGludWVkIGFuZCBpbmRlZmF0aWdhYmxlIGdl
bmVyYXRpb24gb2Yga25vd2xlZGdlLCBleGNlZWRzIHRoZSBzaG9ydCB2ZWhlbWVuY
2Ugb2YgYW55IGNhcm5hbCBwbGVhc3VyZS4 =

UTF-7 являє собою змінений варіант Base64. Ця схема кодування використовується для файлів UTF-16 як проміжний формат в MIME. UTF-7 призначений для того, щоб дозволяти використовувати unicode в e-mail без використання поділу транспортного кодування вмісту. Головна відмінність цього варіанту base64 від MIME в тому, що символ «=» не використовується для доповнення, так як потрібно багаторазове екранування цього символу. Замість цього біти октету доповнюються нулями.

У сервер-сервер протоколі, використовуваному в IRC і сумісному програмному забезпеченні, версія base64 використовується для кодування клієнт / серверних числових і двійкових IP адрес. Клієнтські і серверні числові дані мають фіксовані розміри, які точно збігаються з кількістю знаків base64, тим самим, немає необхідності в додатку. Двійкові IP-адреси для відповідності розширюються провідними нульовими бітами. Набір символів незначно відрізняється від MIME використанням [] замість + /.

Кодування Base64 може бути корисно, якщо в оточенні HTTP використовується інформація, довжину якої можна точно визначити. Також багатьом додаткам необхідно кодувати двійкові дані для зручності включення в URL, приховані поля форм, і тут Base64 зручно не тільки для компактного представлення, але і відносною нечитаності для спроби з'ясування випадковою людиною-спостерігачем природи даних.

Використання URL-кодувальника над стандартом Base64, незважаючи на це, незручно, так як він перетворює символи /і +в спеціальні шістнадцяткові послідовності. Якщо пізніше цей рядок використовується разом з базою даних або через гетерогенні системи, вони припиняють роботу на символі %, створеному URL-кодувальником (бо символ %також використовується в ANSI SQL як шаблон).

У наслідок цього існує змінений Base64 для URL, де не використовується заповнення символом = і символи +і / відповідно замінюються на *і, так що використання кодерів / декодерів URL перестає бути не обхідним і не має ні якого впливу на довжину закодованого значення, залишаючи туж саму закодовану форму, не ушкоджену для використання вреляційних базах даних, веб-формах і ідентифікатори об'єкта взагалі. Стандартом Base64-кодування URL адрес визнається варіант, коли символи +і /замінюються, відповідно, на -і _(RFC 3548, розділ 4).

Інший варіант називається змінений Base64 для регулярних виразів, використовує ! замість *, для того, щоб замінити стандартний Base64 +/, томущо обидва + і * можуть бути зарезервовані для регулярних виразів.

Є інші варіанти, які використовують _ або._, якщо рядок Base64 повинна бути використана разом з ідентифікаторами для програм, або.-для використання в токенах імен XML (Nmtoken), або в більш обмежених ідентифікаторах XML (Name). У деяких випадках для URL застосовується Base58, який не використовує символи +/.

Для кодування URL в деяких системах використовується Base58, що відрізняється від Base64 відсутністю в кінцевому тексті символів, які можуть

сприйматися людиною не однозначно. Виключені 0 (нуль), O (заголовна латинська o), I (заголовна латинська i), l (маленька латинська L). Також виключені символи + (плюс) і / (коса риска), які при кодуванні URL можуть призводити до невірної інтерпретації адреси.

Radix-64 – різновид кодування Base64 двійкових даних в текстовий формат, що використовується в PGP. Від Base64 відрізняється тим, що в кінець додається контрольна сума в 24 біта.

Операційні системи сімейства Unix зберігають обчислені за допомогою crypt-хеші паролі в в файл / etc / passwd використовуючи кодування B64. Вона схожа на radix-64, але суфікс вирівнювання «=» не використовується і в алфавіті небуквених символи розташовані спочатку.

Существует множество вариантов применения Base64. Например, Thunderbird и MozillaSuite использовали Base64 для сокрытия паролей в POP3. Base64 может использоваться как метод для сокрытия секретов без издержек на криптографическое управление ключами, однако этот подход является абсолютно небезопасным и не рекомендуется к использованию.

Сканери спаму, що не декодують повідомлення в base64, часто пропускають повідомлення в Base64, так як вони здаються досить випадковими, або не містять ключові слова в тексті Base64, щоб бути прийнятими за спам. Це використовують спамери для обходу основних антиспамових інструментів.

Зараз нам здається само собою зрозумілим, що 1 байт = 8 біт, що в байті можна закодувати 256 різних значень. Але колись було зовсім не так. Історія пам'ятає і семібітние кодування, і шестібітние, і навіть більш екзотичні системи (наприклад – ЕОМ «Сетунь», яка використовувала трійкову логіку, тобто один потрійний біт – Тріт міг мати три, а не два значення, для неї було справедливо співвідношення 1 Трайтен = 6 тритію). Але якщо залишити осторонь будь-яку екзотику, то мейнстримом все-таки були кодування, в яких 6, 7 або 8 біт в байті.

Шестібітная кодування (наприклад – BCD) дозволяла закодувати в одному байті 64 різних значення, що, як здавалося, було цілком достатньо для кодування алфавітно-цифрових символів, а «зайвий» сьомий біт розширював кодування вже до 128 символів.

Затвердження восьмибітних кодувань як стандарту де-факто принесло багато проблем. До цього моменту вже існувала певна інфраструктура, яка використовує саме семібітние кодування, і holy wars розгорілися з новою силою.

До нас вони дійшли у вигляді проблем з «обрізанням восьмого біта» в системі електронної пошти. Затвердження восьмібітного байта дало 256 різних значень для одного байта, що, в свою чергу дозволило вмістити в одній кодовій таблиці і загальноприйняті символи (цифри, знаки пунктуації, латиницю) та символи, скажімо кирилиці. Здавалося б – суцільне зручність, текст можна набирати хоч російськими буквами, хоч англійськими, а якщо потрібно – і для німецьких Умлаут місце знайдеться!

Але, як завжди, диявол крився в деталях. Вже накопичений і працює хард-н-софт часто був пристосований для кодувань семібітних, що призводило до різноманітних проблем.

Наприклад, поштовий сервер при передачі листа міг абсолютно спокійно обнулити старші біти в кожному байті повідомлення, що не могло не привести до проблем, часто інформація просто катастрофічно губилася.

Для тимчасового вирішення цієї проблеми було запропоновано кілька варіантів. Одним з них стала кодування «КОІ-8». Рішення, потрібно визнати, дуже елегантне – в цьому кодуванні російські літери розташовувалися ПЗ порядку латинських і відрізнялися від них рівно на той самий старший біт. Таким чином при обрізанні цього біта російська «А» перетворювалася в латинську «А», «Б» – в «В» і так далі, повідомлення просто транслітерувати і його все-таки можна було прочитати. Правда, і тут не обійшлося без скелета в шафі – сортування в російській алфавітному порядку в «ЯКІ» ставала кошмаром...

А що було робити іншим мовам, народам і кодувань? А бінарні дані? Все одно кодування з транслітерацією вирішували фундаментальну проблему – втрату восьмого біта, втрату частини інформації. Так народилася кодування (а точніше – алгоритм) Base64.

Алгоритм base64 і донині застосовується там, де немає можливості гарантувати дбайливого поводження з вашою інформацією – наприклад при кодуванні вкладень електронної пошти. У PGP алгоритм base64 використовується для кодування бінарних даних.

Можна уявити собі й інші застосування base64 – наприклад при збереженні в базу даних, якщо заздалегідь невідомо оточення (ох уже ці magic_qoutes в PHP!) І немає необхідності в індексації та пошуку ПЗ тексту, можна скористатися base64.

Base64 цілком може використовуватися для отримання хеш, наприклад за алгоритмом md5, як засіб проти табличного підбору хеша, якщо дані, наприклад пароль користувача в системі, попередньо будуть перетворені в base64.

В python бібліотека base64 так і називається.

2.3.4 Основні відомості про бібліотеку python json

Вікіпедія [6][1)] надає наступне визначення JSON (англ. JavaScript Object Notation) – це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дозволяє описувати об'єкти та інші структури даних. Цей формат головним чином використовується для передачі структурованої інформації через мережу (завдяки процесу, що називають серіалізацією).

Розробив і популяризував формат Дуглас Крокфорд.

[1)] [6] JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).

JSON знайшов своє головне призначення у написанні веб-програм, а саме при використанні технології AJAX. JSON, що використовується в AJAX, виступає як заміна XML (використовується в AJAX) під час асинхронної передачі структурованої інформації між клієнтом та сервером. При цьому перевагою JSON перед XML є те, що він дозволяє складні структури в атрибутах, займає менше місця і прямо інтерпретується за допомогою JavaScript в об'єкти.

JSON з'явився через необхідність обміну даними з сервером у реальному часі без використання плагінів для браузерів, flash-додатків або Java-аплетів, які використовувались скрізь на початку 2000-х років.

Дуглас Крокфорд був тим, хто активно просував новий на той час формат. Він з колегами хотів створити технологію, яка використовувала б можливості звичайного браузера давала б веб-розробникам можливість створювати веб-додатки із постійним двостороннім зв'язком із веб-сервером. JSON вперше був використаний в проекті в Communities.com для Cartoon Network, він дозволяв обмінюватись повідомленнями і одночасно маніпулювати DHTML-елементами [6][1)].

Веб-сайт JSON.org було запущено 2002 року. У грудні 2005-го року Yahoo! почав переводити деякі зі своїх веб-сервісів на роботу з JSON. Google взялася до роботи з технологією у своєму веб-протоколі GData у грудні 2006-го року.

За рахунок своєї лаконічності в порівнянні з XML, формат JSON може бути більш придатним для серіалізації складних структур.

Якщо говорити про веб-застосунки, в такому ключі він доречний в задачах обміну даними як між браузером і сервером (AJAX), так і між самими серверами (програмні HTTP-інтерфейси). Формат JSON так само добре підходить для зберігання складних динамічних структур в реляційних базах даних або файловому кеші.

JSON будується на двох структурах.

[1)] [6] JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).

Набір пар назва/значення. У різних мовах це реалізовано як об'єкт, запис, структура, словник, хеш-таблиця, список з ключем або асоціативним масивом.

Впорядкований список значень. У багатьох мовах це реалізовано як масив, вектор, список, або послідовність.

Це універсальні структури даних. Теоретично всі сучасні мови програмування підтримують їх у тій чи іншій формі. Оскільки JSON використовується для обміну даними між різними мовами програмування, то є сенс будувати його на цих структурах.

У JSON використовуються такі їхні форми:

Об'єкт – це послідовність пар назва/значення. Об'єкт починається з символу { і закінчується символом }. Кожне значення слідує за : і пари назва/значення відділяються комами.

Масив – це послідовність значень. Масив починається символом [і закінчується символом]. Значення відділяються комами.

Значення може бути рядком в подвійних лапках, або числом, або логічними true чи false, або null, або об'єктом, або масивом. Ці структури можуть бути вкладені одна в одну [6][1)].

Рядок – це послідовність з нуля або більше символів юнікода, обмежена подвійними лапками, з використанням escape-послідовностей, що починаються зі зворотної косої риски \. Символи представляються простим рядком.

Тип Рядок (String) дуже схожий на String в мовах C і Java. Число теж дуже схоже на C- або Java-число, за винятком того, що вісімкові та шістнадцяткові формати не використовуються. Пропуски можуть бути вставлені між будь-якими двома лексемами.

Останні версії веб-браузерів мають вбудовану підтримку JSON і здатні його обробляти без виклику функції eval(), що призводить до описаної проблеми. Обробка JSON у такому разі зазвичай здійснюється швидше.

[1)] [6] JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).

Так у червні 2009 року вбудовану підтримку JSON мали такі браузери:

– Mozilla Firefox 3.5+, SeaMonkey 2, та Thunderbird 3;
– Microsoft Internet Explorer 8Native JSON in IE8. Архів оригіналу за 12.02.2012;
– Opera 10.5+;
– браузери, засновані на WebKit (наприклад, Google Chrome, Apple Safari).

Принаймні дві популярні бібліотеки JavaScript використовують вбудований JSON у разі його доступності:

– jQuery;
– Dojo.

JSONP або «JSON з підкладкою» є розширенням JSON, коли назва функції зворотного виклику вказується як вхідний аргумент. Спочатку ідея була запропонована в блозі MacPython в 2005 році, і в наш час використовується багатьма Web 2.0 застосунками, такими, як DojoToolkitApplications, GoogleToolkitApplications і zanoxWebServices. Подальші розширення цього протоколу були запропоновані з урахуванням введення додаткових аргументів, як, наприклад, у разі JSONPPза підтримки S3DB вебсервісів.

Оскільки JSONP використовує скрипт-теги, виклики ПЗ суті відкриті світові. З цієї причини JSONP може бути недоречними для зберігання конфіденційних даних [6][1)].

Включення скриптових тегів від віддалених сайтів дозволяє їм передати будь-який контент на сайті. Якщо віддалений сайт має вразливості, які дозволяють виконати ін'єкції JavaScript, то початковий сайт також може бути ними зачеплений.

BSON – це бінарна форма представлення простих структур даних і асоціативних масивів (які називають об'єктами або документами). Назва

[1)] [6] JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).

«BSON» заснована на визначенні JSON і неофіційно означає «Binary JSON» (бінарний JSON).

Як функціонально, так і синтаксично JSON є підмножиною мови YAML. Зокрема, специфікація YAML 1.2 указує, що «будь-який файл у форматі JSON є коректним файлом у форматі YAML». Найпоширеніший парсер YAML здатний обробляти й JSON. Специфікація YAML до версії 1.2 не повністю покривала JSON, насамперед через відсутність рідної підтримки UTF-32 в YAML, а також вимоги пропуску після роздільника-коми. Крім того, специфікація JSON включала коментарі в стилі /* */.

JSON Schema – одна з мов опису структури JSON документа. Використовує синтаксис JSON. Базується на концепціях XML Schema, RelaxNG, Kwalify. JSON Schema – самоописова мова: при її використанні для обробки даних і опису їхньої допустимості можуть використовуватись однакові інструменти серіалізації/десеріалізації [6][1)].

В python бібліотека так і назива’ється json.

2.3.5 Основні відомості про бібліотеку python cv2

OpenCV (англ. Open Source Computer Vision Library, бібліотека комп'ютерного зору з відкритим кодом) – бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), відстежування руху об'єктів, перетворення зображень, застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях [7][2)].

[1)] [6] JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).

[2)] [7] OpenCV – Вікіпедія. URL: https://uk.wikipedia.org/ wiki/OpenCV (дата звернення 11.05.2019).

Бібліотека розроблена Intel і нині підтримується Willow Garage та Itseez. Сирцевий код бібліотеки написаний мовою C++ і поширюється під ліцензією BSD. Біндинги підготовлені для різних мов програмування, таких як Python, Java, Ruby, Matlab, Lua та інших. Може вільно використовуватися в академічних та комерційних цілях.

Офіційно проект OpenCV був запущений у 1999 році за ініціативою Intel Research з ціллю розвивати CPU-ресурсомісткі додатки. Основними вкладниками у проект була Intel's Performance Library Team та певна кількість експертів з чисельної оптимізації у Inter Russia. На перших етапах розвитку OpenCV основними задачами бібліотеки були:

- розвивати дослідження у напрямку комп'ютерного зору, забезпечуючи добре оптимізований та відкритий код бібліотеки;
- поширювати знання у сфері комп'ютерного зору, забезпечуючи загальну інфраструктуру, яку б могли розвивати розробники, таким чином код ставатиме більш легким для сприйняття та обміну;
- розвивати засновані на роботі з комп'ютерним зором комерційні додатки, створюючи не залежну від платформи, оптимізовану та безкоштовну бібліотеку.

Для цього використовувалася ліцензія, яка не вимагала від таких комерційних додатків бути відкритими [7][1)].

Перша альфа-версія OpenCV була оприлюднена на IEEE конференції з комп'ютерного зору й розпізнавання образів у 2000 році, і п'ять бета-версій було випущено у період між 2001 і 2005 роками. Перша версія 1.0 була випущена у 2006 році. У середині 2008 року, OpenCV отримала корпоративну підтримку від Willow Garage і знову перейшла у стадію активної розробки. «Пре-релізна» версія 1.1 була випущена у жовтні 2008 року [8][2)].

---

1) [7] OpenCV – Вікіпедія. URL: https://uk.wikipedia.org/ wiki/OpenCV (дата звернення 11.05.2019).

2) [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/ files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

Другий великий випуск OpenCV відбувся у жовтні 2009 року. OpenCV 2 включала у себе серйозні зміни у інтерфейсі C++. Ці зміни спрямовані на більш прості, тип-безпечні моделі, додавання нових функцій, і кращу реалізацію існуючих моделей в плані швидкодії (особливо на багатоядерних системам). Офіційні релізи надалі відбуваються кожні 6 місяціві розробкою займається незалежна команда з Росії, яка підтримується комерційними корпораціями.

У серпні 2012 року, підтримку OpenCV було передано некомерційній організації, OpenCV.org.

Бібліотека містить понад 2500 оптимізованих алгоритмів, серед яких повний набір як класичних так і практичних алгоритмів машинного навчання і комп'ютерного зору [8][1)]. Алгоритми OpenCV застосовують у таких сферах:

- аналіз та обробка зображень;
- системи з розпізнавання обличчя;
- ідентифікації об'єктів;
- розпізнавання жестів на відео;
- відстежування переміщення камери;
- побудова 3D моделей об'єктів;
- створення 3D хмар точок зі стерео камер;
- склеювання зображень між собою, для створення зображень всієї сцени з високою роздільною здатністю;
- система взаємодії людини з комп'ютером;
- сошуку схожих зображень із бази даних;
- усування ефекту червоних очей при фотозйомці зі спалахом;
- стеження за рухом очей;
- аналіз руху;
- ідентифікація об'єктів;
- сегментація зображення;

---

1) [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

- трекінґ відео;
- розпізнавання елементів сцени і додавання маркерів для створення доповненої реальності.

OpenCV написана на C++ і її основний інтерфейс також реалізовано на C++, але бібліотека і досі представляє старіший C інтерфейс. На даний момент реалізовано інтерфейс на мовах Python, Java і MATLAB / OCTAVE (починаючи з версії 2.5). API для цих інтерфейсів можна знайти в онлайн документації Оболонки для інших мов, таких як C#, CH,Ruby були розроблені з метою охоплення ширшої аудиторії.

Всі нові розробки та алгоритми OpenCV у даний момент розробляються у C++ інтерфейсі.

При проектуванні OpenCV закладалася максимальна обчислювальнаефективність з упором на додатки реального часу. Вона написана на оптимізованому C ++ і може користуватися перевагами багатоядерних процесорів. Якщо ви хочете добитися ще більшої швидкодії на платформахз архітектурою Intel, то можете купити розроблені Intel бібліотеки IntegratedPerformance Primitives (IPP) [IPP], що включають оптимізовані низько рівневі процедури для багатьох алгоритмів. OpenCV автоматично використовує відповіднубібліотеку IPP, якщо вона встановлена. Починаючи з версії OpenCV 3.0, Intel передалакоманді і спільноті OpenCV безкоштовне підмножина IPP (що отримало назвуIPPICV), яке за замовчуванням вбудовано в OpenCV і забезпечує прискорення [8][1)].

Одна з цілей OpenCV – надати просту для використання інфраструктуру комп'ютерного зору, яка дозволила б швидко створювати щодоскладні додатки. Бібліотека OpenCV налічує понад 500 функцій, що охоплюють багато областей комп'ютерного зору, в т. Ч. Контроль якості продукції, медичні зображення, безпеку, призначені для користувача інтерфейси, калібрування камери, стереозрения і робототехніку. Оскільки

[1)] [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

комп'ютерний зірі машинне навчання часто йдуть рука об руку, в OpenCV включена також універсальна бібліотека машинного навчання (модуль ML). Акцент у цій підбібліотеках зроблений на статистичному розпізнаванні образів і кластеризації. модуль MLвиключно корисний для основної місії OpenCV, але разом з тим має достатню спільністю для вирішення будь-яких завдань машинного навчання.

Багато фахівців в області теоретичної інформатики та програмісти-практики знайомі з тієї чи іншої гранню комп'ютерного зору, але мало хто знає провсіх способах його застосування. Так, більшість людей чуло про використаннякомп'ютерного зору в охоронних системах і багато в курсі того, що воно все частішезастосовується для обробки зображень і відео в інтернеті. Хтось бачив, як комп'ютерний зір інтегрується в ігрові інтерфейси. Ще менше народу розуміють, щов більшості зображень, отриманих за допомогою аерофотозйомки, а також в панорамах (як в «Перегляді вулиць» від Google) широко використовується калібрування камериі методи зшивання зображень. Дехто знає про нішевих застосуваннях в області моніторингу безпеки, безпілотних літальних апаратах та аналізі біомедичних даних. Але лише деякі усвідомлюють, наскільки глибоко комп'ютерний зірпроникло в виробництво: практично всі товари масового виробництва на якомусьетапі проходять автоматичний контроль за допомогою системи комп'ютерного зору [8][1)].

Ліцензія, ПЗ якій поширюється OpenCV, допускає використанняOpenCV повністю або частково в комерційних продуктах. Ви не зобов'язані розкривати вихідний код свого продукту або робити внесені вами удосконаленнязагальнодоступними, хоча ми на це сподіваємося. Почасти через таких ліберальних умов навколо OpenCV склалося велике співтовариство, яке включає і представників великих компаній (IBM, Microsoft, Intel, SONY, Siemens, Google – і це далеконе всі) і дослідних

[1)] [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

центрів (Стенфордський університет, Массачусетськийтехнологічний інститут, університет Карнегі-Меллон, Кембридж і державний інститут досліджень в інформатиці та автоматики – INRIA).

У групахYahoo існує форум (https://groups.yahoo.com/neo/groups/ OpenCV/info), що налічує понад 50 000 учасників. OpenCV популярна у всьому світі, але особливовеликі спільноти склалися в Китаї, Японії, Росії, Європі та Ізраїлі.

З моменту виходу альфа-версії в січні 1999 року OpenCV знайшла застосуванняв багатьох додатках, продуктах і наукових дослідженнях, наприклад: зшиваннязображень для отримання супутникових і веб-карт, поєднання результатів сканування зображень, придушення шумів на медичних зображеннях, аналізоб'єктів, системи спостереження і виявлення вторгнень, автоматичний моніторинг в системах технічної безпеки, системи контролю виробничихпроцесів, калібрування камер, військові додатки, а також безпілотні повітряні,наземні і підводні транспортні засоби. Методи комп'ютерного зору застосовуються навіть до аналізу спектрограм для розпізнавання звуку і музики. OpenCVлежала в основі системи зору робота Stanley, створеного в Стенфордському університеті і завоював заснований агентством DARPA приз у 2 мільйони доларівна змаганнях автомобілів-роботів в пустелі [8][1)].

OpenCV складається з декількох рівнів. На самому верху знаходиться операційна система, в якій працює бібліотека. Нижче розташовані мовні прив'язки і приклади додатків. Ще нижче – наданий код зі сховищ opencv_contrib,який містить велику частину високорівневою функціональності. далі -ядро OpenCV, а на самому нижньому рівні – різні апаратні оптимізації, складові рівень апаратного прискорення (hardware acceleration layer – HAL). Ця організація показана на рис. 2.

[1)] [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

OpenCV з'явилася як плід досліджень Intel, зроблених для просуваннядодатків, які потребують потужному процесорі. З цією метою Intel почала багатопроектів, включаючи трасування променів в режимі реального часу і тривимірні відео стіни. Один з авторів, Гері Бредскі [Bradski], в той час працював в Intel, бувчастим гостем в університетах і звернув увагу, що в деяких провідних навчальних закладах сформувалися групи, наприклад MIT Media Lab, з розвиненою і відкритою для внутрішнього користування інфраструктурою комп'ютерного зору – кодпередавався від одного студента іншому, і таким чином кожен новий студент отримував відправну точку для розробки власного додатка. Замість того щобзаново винаходити базові функції, студенти розвивали те, що було створено до них .

Таким чином, OpenCV замислювалася як спосіб надати інфраструктурукомп'ютерного зору в загальне користування. За підтримки групи розробки бібліотеки підвищення продуктивності (Performance Library Team) з Intellбулазакладена основа OpenCV у вигляді вже написаного коду та специфікацій алгоритмів. Все це було відправлено членам російської групи розробників у складі Intel.Тут і знаходяться витоки OpenCV: вона була народжена в дослідницькій лабораторії Intel у співпраці з групою розробки бібліотек підвищення продуктивності, а реалізацією і оптимізацією зобов'язана досвіду програмістів з Росії.Головним серед росіян був Вадим Писаревський, який керував розробкою, кодував сам і оптимізував більшу частину бібліотеки, він і до цього днязнаходиться в центрі більшості робіт, пов'язаних з OpenCV. У розробці ранніх варіантів інфраструктури йому допомагав Віктор Єрухіма, а Валерій Курякін керував російською лабораторією і всіляко підтримував зусилля колег [8][1)]. З самогопочатку перед OpenCV було поставлено кілька цілей:

[1)] [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

- сприяти дослідженням в області комп'ютерного зору, надавшине просто відкритий, а й оптимізований код базової інфраструктури (досить винаходити велосипед);
- поширювати знання про комп'ютерному зорі, надавши єдину інфраструктуруяку можна надбудовувати (таким чином, код стане більш зрозумілим і стерпним);
- просувати комерційні додатки на основі комп'ютерного зору, безкоштовнонадавши стерпний оптимізований код – за ліцензією, що невимагає розкриття або безоплатність самих комерційних додатків.

Ці цілі складають відповідь на питання «навіщо». Наявність додатків комп'ютерного зору викликає потребу в більш швидких процесорах. Стимулюванняпереходу на більш швидкі процесори фінансово вигідніше для Intel, ніж продаж додаткового програмного забезпечення. Бути може, саме тому відкритий і вільний код був написаний в надрах компанії з виробництва обладнання, а не ПО. Іноді у виробника обладнання виявляється більше причиндля інновацій в області софта.

У будь-якому проекті ПЗ (див.рис. 2) з відкритим вихідним кодом важливо досягти критичноїмаси, після чого проект підтримує себе сам. На сьогоднішній день OpenCV булазавантажена приблизно 11 мільйонів разів, і це число щомісяця збільшується в середньомуна 160 000. У розробку OpenCV вносять вклад багато користувачів, а центр управління розробкою вийшов за межі Intell [8][1)].

[1)] [8] Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/ files/ PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.

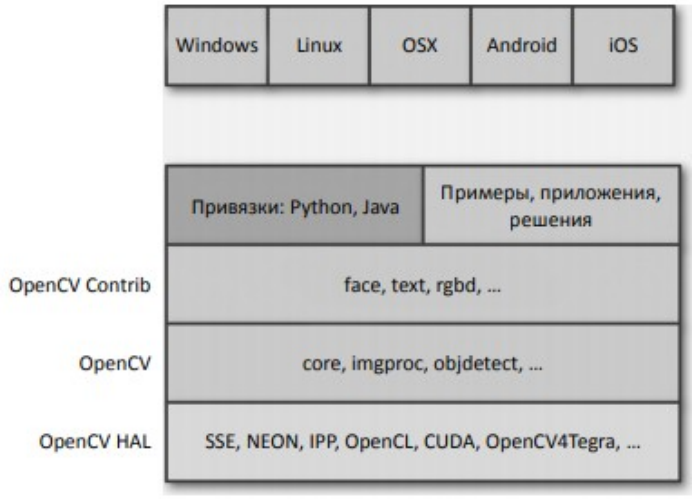

Рисунок 2 – Архітектурна діаграма OpenCV
і підтримуваних операційних систем

На OpenCV також вплинули бум «доткомів» і численні зміни керівництва і напрямків розвитку. Бували періоди, коли над OpenCVне працював взагалі ніхто з Intell. Але з появою багатоядерних процесорів і рядунових додатків комп'ютерного зору значення OpenCV знову стало рости. Поштовх подальшому розвитку дав також швидкий прогрес в області робототехніки.

Перетворившись в бібліотеку з відкритим вихідним кодом, OpenCV кілька років розроблялася при активному сприянні Willow Garage, а тепер підтримується фондом OpenCV (http://opencv.org/). В наші дні розробка ведеться силами як фонду, так і декількох публічних і приватних організацій.

3 ДЕТАЛІ РЕАЛІЗАЦІЇ

Уся програма працює в безкінечному циклі, на кожній ітерації програма зчитує зображення з камери пристрія. Потім обробляє його та знаходить на ньому шарообразну фігуру. З знайденою фігурою ми отримаємо координати цієй фігури та радіус. Ми порівнюємі ці параметри з граничними значеннями. Граничними значеннями служить квадрат розміром 3/5 від ширини та висоти зображення. Якщо знайдена фігура виходить за рамки квадрата то на сервер пристрія посилається запрос рухатися у потрібному напрямку з потрібною потужністю на кожний мотор. Після цього зображення кодується та посилається на сервер комп'ютера, який виводить одержане зображень на монітор.

Для відправки зображення використовуються сокети так як це знизить нагрузку на процесор пристрія. Зображення кодується у форматі base64. Для відправки координат для руху використовуються http запроси.

Напрямок обирається залежно від розташування знайденої фігури на екрані

Мотори пристрія після одержання команди працюють 0.1 секунду, тому щоб не трапилося колізії программа не продовжує свою працю, поки не отримає відповідь від сервера. До зображення додаються квадрати, які описують знайдену фігуру, а також квадрат який показує граничні контури.

Сервер, який працює на комп'ютері прослуховує порт 9797. Якщо приходить запрос, він декодую зображення та виводить його на монітор. Для підвищення точності знаходження фігури зображення переводиться в сірий відтінок. Також використовуються функції для зменшення шуму та згладжування. Також використовується гауссовий поріг для виявлення гострих країв.

У результаті ми можемо управляти рухом пристрою за допомогою руху шарообразної фігури відносно камери.

Для того щоб перетворити зображення, його треба отримати. Для цього використовується функція cv2.VideoCapture(0). Вона приймає один параметр типу Boolean. Якщо передається 0 то зображення отримується від вбудованої камери, якщо 1 то від сторонньої, яка підключена через usb порт.

Зображення отримується у вигляді двовимірного масиву, де кожний елемент відповідає за один піксель. В бібліотеці opencv це спрощує обробку зображень та дозволяє використовувати бібліотеку numpy, яки використовується для математичних перетворень та операцій.

Після того як зображення отримане воно переводиться у сірий відтінок. Це спрощує пошук шарів на зображенні. Для цього використовується наступний фрагмент коду:

```
cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Першим параметром передається зображення отримане з камери, другим передається константа яка відповідає за потрібний відтінок. У нашому випадку за сірий відтінок відповідає константа COLOR_BGR2GRAY. Вона знаходиться у пространстві імен бібліотеки cv2.

На рисунку 3 наведен приклад зображення до використання функції cvtColor. А на рисунку 4 приклад зображення перетвореного з оригінального, за допомогою функці cvtColor.

Також функція cvtColor може приймати наступні константи:

- COLOR_BGR2GRAY
- COLOR _RGB2GRAY
- COLOR _GRAY2BGR
- COLOR _GRAY2RGB

Кожен з них використовується для різних форматів зображення, але усі вони переводять зображення у сірий відтінок.

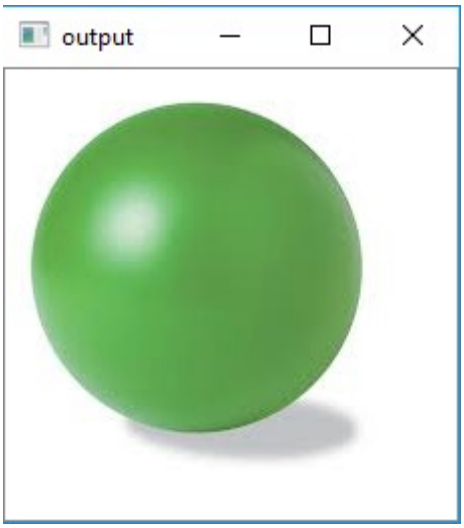

Рисунок 3 – приклад зображення використаного до застосування функції cvtColor

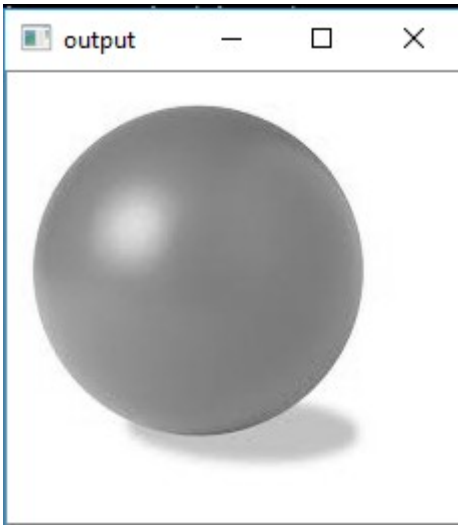

Рисунок 4 – приклад зображення після використання функції cvtColor

Далі нам потрібно видалити шуми. Це родиться за допомогою наступного коду:

```
cv2.GaussianBlur(gray,(5,5),0)
```

В цій функції використовується фільтр Гаусса. Ми повинні вказати ширину і висоту ядра, які повинні бути позитивними і непарними. Ми також повинні вказати стандартне відхилення в напрямках X та Y, sigmaX і sigmaY відповідно. Якщо вказано тільки sigmaX, sigmaY приймається рівним sigmaX. Якщо обидва дані у вигляді нулів, вони розраховуються з розміру ядра. Гауссова фільтрація дуже ефективна для видалення гауссовского шуму із зображення.

Розмивання Гауса – це метод фільтрації зображення за допомогою функції Гауса, який призводить до розмивання зображення. Даний ефект широко використовується в графічних програмах, як правило, для зменшення зашумленості зображенні та зниження деталізації. Візуальний ефект цієї фільтрації розмивання аналогічний погляду на зображення крізь напівпрозорий екран. На рисунку 5 наведен приклад зображення перетвореного за допомогою функції GaussianBlur.

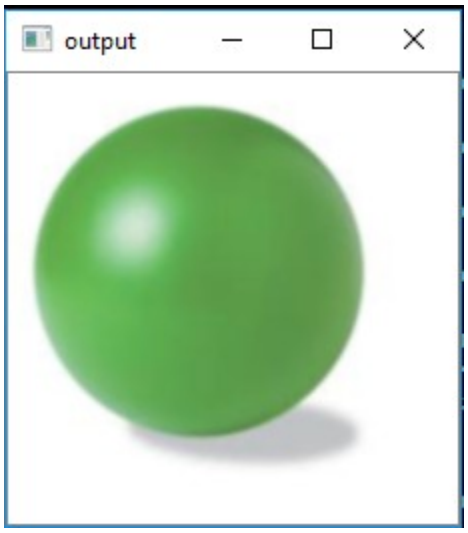

Рисунок 5 – приклад зображення після використання функції GaussianBlur

Не дуже замітно, але зображення стале менш чутким. На рисунку 6 наведен приклад використання функції GaussianBlur у сірому відтінку.

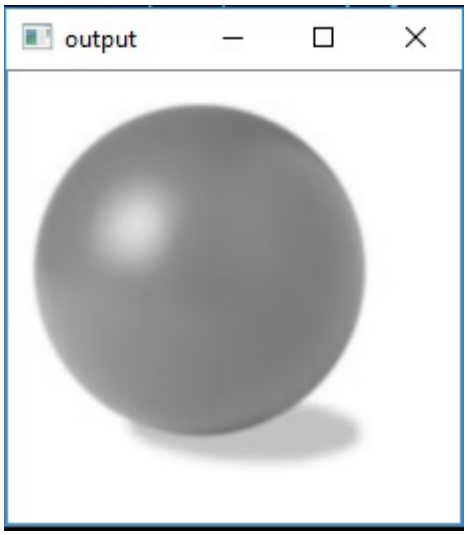

Рисунок 6 – приклад зображення після використання функції GaussianBlur у сірому відтінку

Далі ми повинні згладити шуми за допомогою медіанного фільтру.

Медіанна фільтрація - метод нелінійної обробки сигналів, розроблений Тьюки. Цей метод виявляється корисним при придушенні шуму на зображенні. Одновимірний медіанний фільтр являє собою ковзне вікно, яке охоплює непарне число елементів, зображення. Центральний елемент замінюється медіаною всіх елементів зображення у вікні.

В opencv медіанний фільтр реалізується в функції medianBlur. Наступний код це демонструє:

```
cv2.medianBlur(gray,5)
```

Приклад наведений на рисунку 7.

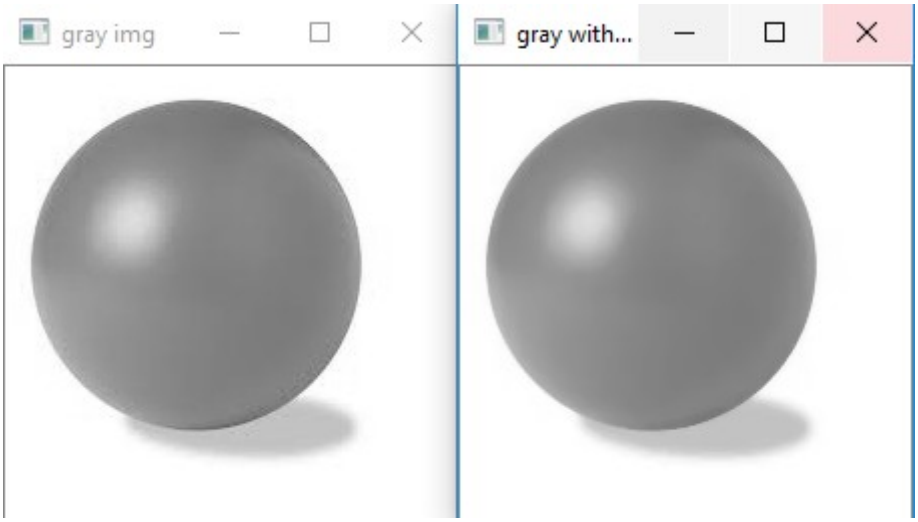

Рисунок 7 – приклад зображення до і після використання медіанного фільтру

На рисунку 7 ми можемо бачити приклад використання медіанного фільтру. Різниця ледве помітна, але якщо придивитися на тінь то можемо помітити, що на лівому зображенні, на якому не використовувався медіанний фільтр тінь менш чітка ніж на правому

Далі використовується порогова функція. Це дуже важлива функція, яка дозволяє виділити усі контури, це полегшує знаходження шарів на зображенні. Порогова функція в opencv реалізується через функцію adaptiveThreshold. Приклад наведений у наступному коді:

```
gray = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH _GAUSSIAN_C, cv2.THRESH_BINARY,11,3.5)
```

Якщо значення пікселя більше порогового значення, йому присвоюється одне значення (може бути білим), в іншому випадку йому присвоюється інше значення (може бути чорним). Першим аргументом є вихідне зображення, яке повинно бути зображенням в градаціях сірого. Другий аргумент - це граничне значення, яке використовується для класифікації значень пікселів. Третій аргумент - це maxVal, який представляє

значення, яке буде дано, якщо значення пікселя більше (іноді менше) порогового значення. OpenCV надає різні стилі порогових значень, і це визначається четвертим параметром функції.

Приклад наведено на зображенні 8.

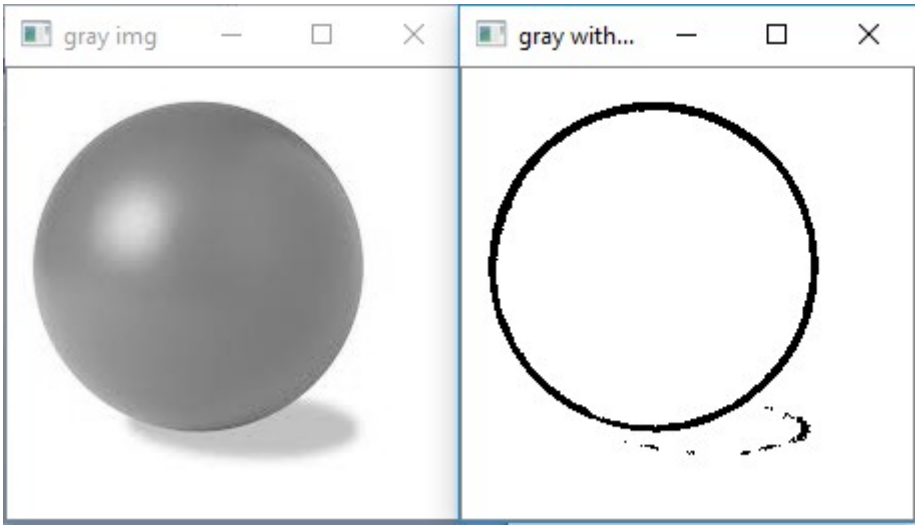

Рисунок 8 – приклад зображення до і після використання функції adaptiveThreshold

На зображенні 8 ми бачимо що на картинці залишилися лише контури.

Далі ми застосовуємо функції дилатації та ерозії. За своїм призначенням вони схожі, але діють абсолютно по різному.

Дилатація полягає в згортанні зображення A з деяким ядром B, який може мати будь-яку форму або розмір, зазвичай квадрат або коло. Ядро B має певну опорну точку, зазвичай це є центром ядра. Оскільки ядро B сканується по зображенню, ми обчислюємо максимальне значення пікселя, що перекривається з B і замінюємо піксель зображення в позиції точки прив'язки на це максимальне значення. Ця операція максимізації змушує яскраві області на зображенні зростати.

Операція ерозії аналогічна дилатації але вона навпаки змушує білі зони зменшуватися а темні збільшуватися.

Приклад наведено на рисунку 9.

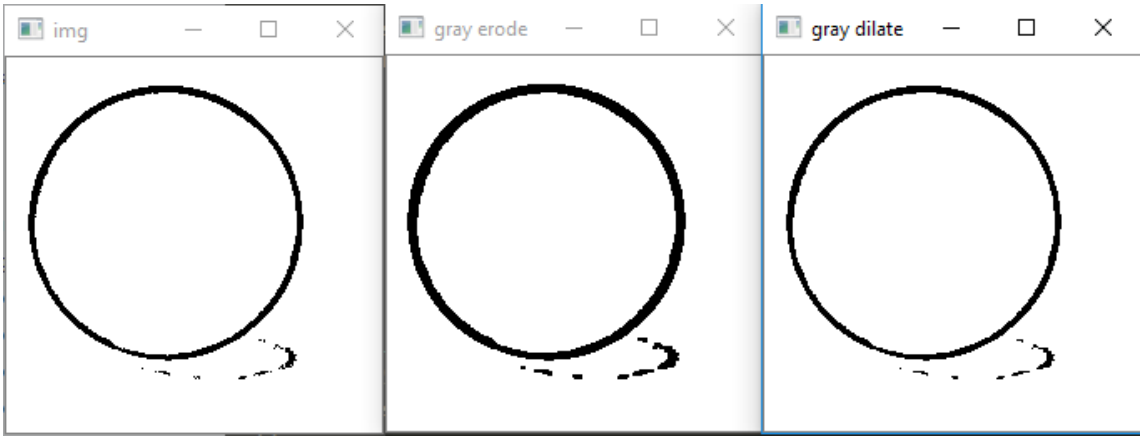

Рисунок 9 – приклад використання ерозії та дилатації

На зображенні 9 ми бачимо три кола. Спочатку була використана ерозія, що збільшила темні зони, а потім була використана дилатація, що їх навпаки зменшила. Різниця першого і останнього кола ледве помітна, але білі крапки які могли бути в контурах зникли, це важливо для розпізнавання кола.

У коді ця операція виглядає наступним образом:

```
kernel = np.ones((2,3),np.uint8)
gray = cv2.erode(gray,kernel,iterations = 1)
gray = cv2.dilate(gray,kernel,iterations = 1)
```

Тепер коли усі перетворення здійснені можна приступати до розпізнавання. Розпізнавання шарів в бібліотеці opencv здійснюється за допомогою функції HoughCircles. Вона використовую попередньо навчені нейронні мережі. В коді це виглядає наступним образом:

```
circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, 20, param1=100, param2=50)
```

Ця функція повертає двовимірний масив знайдених кругів. Кожен елемент містить три параметри: радіус, позицію по X та позицію по У.

Приклад наведено на рисунку 10.

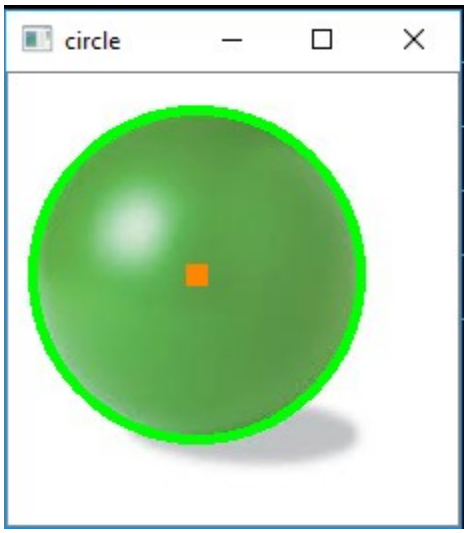

Рисунок 10 – приклад знаходження кола

На рисунку 10 на знайдене коло наноситься контур та центр. Контур та центр наносяться на оригінальне зображення, так як зображення яке піддавалося обробці знаходиться у сірому діапазоні кольорів, тому нанесена на нього мітка буде зливатися з самим контуром.

Далі, коли коло знайдене, визначається куди потрібно їхати пристрію. Відвовідна команда передається на сервер пристрія. А зображення з камери кодується методом base64 та передається через сокети до комп’ютера, який може його розкодувати та вивести на монітор.

ВИСНОВКИ

Машинне бачення має великі перспективи, щоб перевершити людини в найближчі десять років. Уже зараз роботи можуть бачити крізь стіни і на кілометри вперед. Навіть таке складне завдання, як розшифровка відеоінформації скоро буде під силу комп'ютерів. Розумними роботами стануть не тільки автомобілі, але і роботи-поїзда і роботи-літаки. А ще роботи-хірурги, рука яких ніколи не здригнеться і не зірве скальпель, а їх пильне око вчасно помітить артерію. Будемо сподіватися, що люди зможуть направити розвиток машинного зору в вірне русло.

В майбутньому ідеальна система машинного зору буде грунтуватися повністю на цифрових технологіях з використанням інтелектуальних камер і бюджетного обладнання для обробки і розпізнавання зображень. Ключовим елементом тут, звичайно, буде програмне забезпечення, яке здатне гнучко і швидко налаштовуватися на потрібну область з динамічним розширенням функціональних можливостей. Хірохиса Хірукава, дослідник з Національного інституту перспективних наукових досліджень і технологій, заявляє, що виробництво роботів в XXI столітті може стати найбільшою галуззю промисловості, подібно виробництву автомобілів в XX столітті.

При цьому вже до 2025-го, в крайньому випадку, до 2050 року варто очікувати масового поширення роботів, службовців для виконання домашніх робіт. Керівник проектного відділу компанії Microscan Боб Таплетт також заявляє з цього приводу: «Можна стверджувати, що в майбутньому системи машинного зору перетворяться в системи збору даних. Пристрої для зчитування штрих-кодів підуть в минуле, і в значній мірі це буде обумовлено тим, що системи машинного зору здатні вирішувати набагато більше завдань».

Розібравшись в машинному зорі, ми переконалися, що багато висоти вже досягнуто, але багато ще попереду. Розвиток машинного зору може допомогти роботам досягти небувалих результатів і розвинути величезну

міць, випереджаючи людини. Хоча до цього ще далеко. Недосконаліть машинного зору зумовлено почасти технічними причинами, однак йде бурхливий розвиток інформаційних технологій і знаходиться все більше рішень цих технічних проблем. Системи машинного зору стають все більш актуальними, так як покликані вирішувати найбільш значущі проблеми людства, такі як безпека, медицина, промислові завдання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Системы машинного зрения. История, примеры, планы. URL: https://robotics.ua/shows/modernity/5844-sistemy_mashinnogo_zreniya_istoriya_primery_plany (дата звернення 11.05.2019).
2. Компьютерное зрение – Википедия. URL: https://ru.wikipedia.org/wiki/Компьютерное_зрение (дата звернення 11.05.2019).
3. Искусственный интеллект и его влияние на машинное зрение. URL: https://controlengrussia.com/tehnicheskoe-zrenie/iskusstvennyj-intellekt-i-mashinnoe-zrenie/ (дата звернення 11.05.2019).
4. История создания языка программирования Python. URL: https://web.informatics.ru/works/17-18/web_online/barabanov_n_v/language_ python.html (дата звернення 11.05.2019).
5. NumPy – это… Что такое NumPy? URL: https://dic.academic.ru/dic.nsf/ruwiki/1271758 (дата звернення 11.05.2019).
6. JSON – Вікіпедія. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 11.05.2019).
7. OpenCV – Вікіпедія. URL: https://uk.wikipedia.org/ wiki/OpenCV (дата звернення 11.05.2019).
8. Learning OpenCV3. - Келєр А., Бредски Г. - ISBN 978-5-97060-471-7. - 978-5-97060-471-7.pdf [Електронний ресурс]. - Режим доступу: https://dmkpress.com/files/PDF/978-5-97060-471-7.pdf. - Дата звернення 11.05.2019.