

# Решатель судоку по изображению с камеры

А. А. Муравцев<sup>1</sup>

<sup>1</sup>Высшая школа теоретической механики  
Санкт-Петербургский Политехнический университет Петра Великого

31 мая 2022 г.

# Структура проекта

## Распознавание и обработка судоку на изображении

Цветное в чёрно-белое

Определение местоположения поля судоку на кадре

Распознавание цифр в ячейках с помощью обученной нейронной сети

## Решатель судоку

Рекурсивный перебор

Постановка задачи о точном покрытии

Алгоритм X и его реализация с помощью метода танцующих ссылок

## Телеграм-бот

## Графическое приложение на PyQT

## Возможные пути развития

# Структура проекта

```
sudoku
.
|__ lib
    |__ __image_preprocess.py (чёрно-белое, разделение на ячейки)
    |__ __sudoku_detection.py (обн-ие поля и линий, расп-ие цифр)
    |__ __sudoku_solver.py (р-ль на основе рекурсивного перебора)
    |__ __solver_algorithm_x.py (решатель на основе алгоритма X)
|__ ml_model
    |__ __dataset (датасет с изображениями напечатанных цифр)
    |__ __ml_train.py (код обучения нейросети на основе dataset)
    |__ __neural_net_classifier_2.h5 (обученная нейросеть)
|__ telegram_bot
    |__ __bot.py (код бота)
    |__ __empty_sudoku_field.png (изображение с пустыми ячейками)
|__ frontend
    |__ __sudoku_app.py (запуск графического приложения)
|__ tests
    |__ __sudoku_pytest.py (тесты)
|__ main.py (основной файл с выводом результатов в консоль)
|__ exploration.py (для исследования идей)
```

# Преобразование в чёрно-белое изображение. OpenCV

cv2.cvtColor(...)

(1) В оттенках серого



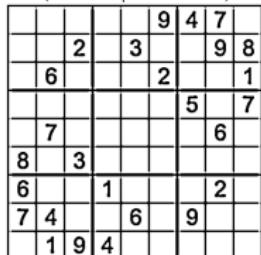
cv2.GaussianBlur(...)

(2) Размытие по Гауссу

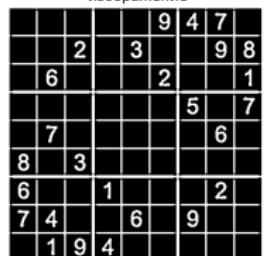


cv2.adaptiveThreshold(...)

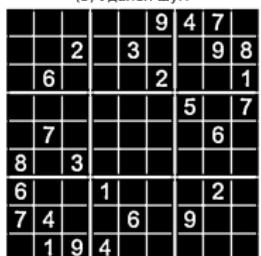
(3) Бинарное изображение  
(только чёрный и белый)



(4) Инвертированное чёрно-белое  
изображение



(5) Удалён шум



(6) Увеличено количество белого

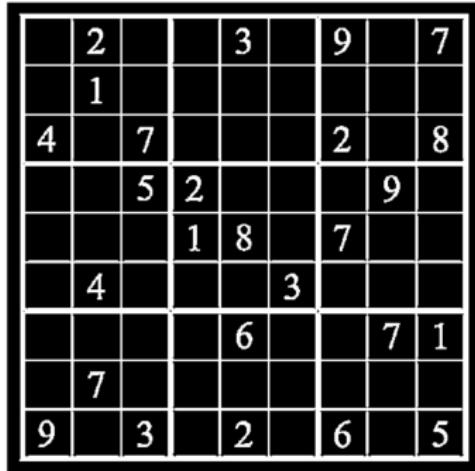
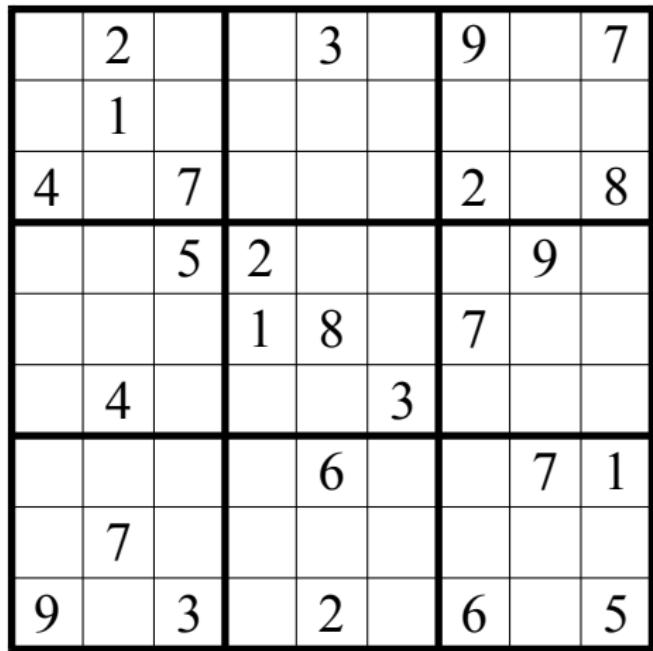


cv2.bitwise\_not(...)

cv2.morphologyEx(...)

cv2.dilate(...)

# Распознавание поля судоку. Разделение на ячейки



Преобразовано с помощью библиотеки для  
обработки изображений OpenCV

# Датасет

Обычный MNIST не подошёл, так как в нём представлены рукописные, а не напечатанные цифры.

389 штук



388 штук



389 штук



414 штук



396 штук



412 штук



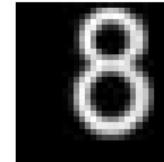
368 штук



349 штук



367 штук



# Датасет

Возможны помехи в пустых ячейках: случайный штрих или блик света. При обучении нейросети это 0 (означает пустую ячейку в поле судоку).

пример №1



пример №2



пример №3



пример №4



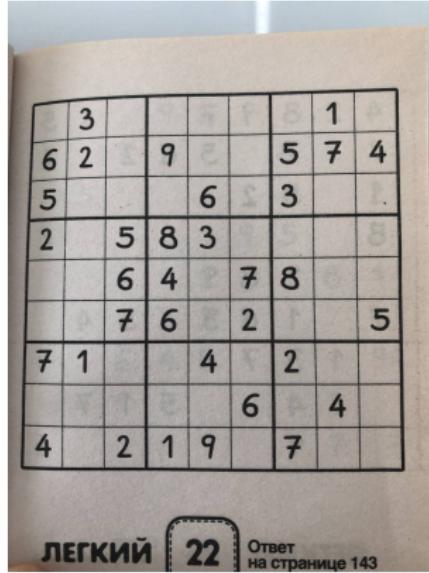
пример №5



# Датасет

Возможны и вовсе не случайные помехи: например, просвечивает страница в сборнике судоку.

Изображение с просвечиванием

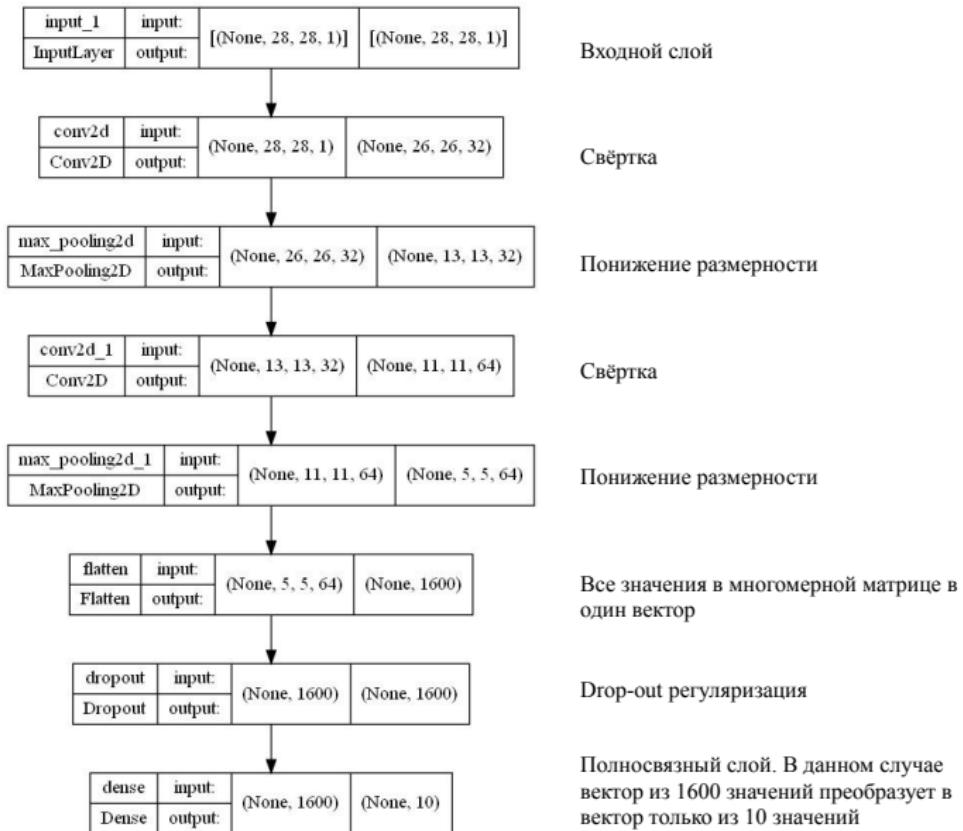


При сильном просвечивании  
нейросеть может уловить  
паттерн просвечивающей цифры.

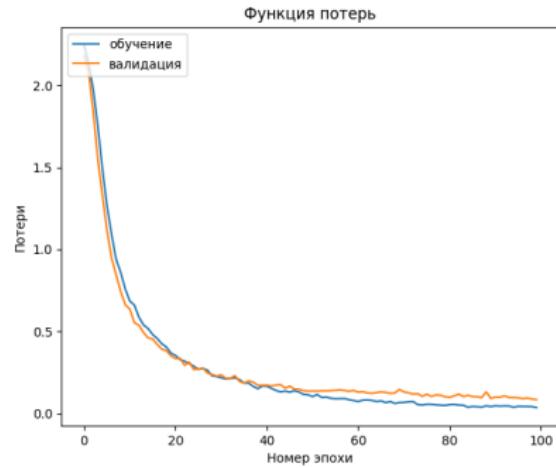
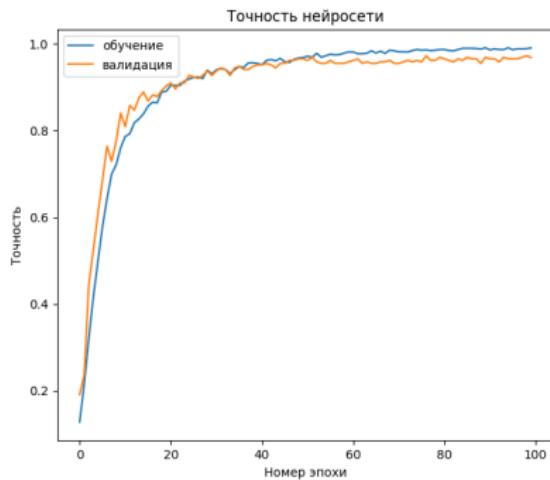
Дополнительно видим, что есть  
разные начертания одних и тех  
же цифр ⇒ необходимость  
расширения датасета.

# Свёрточная нейросеть Keras

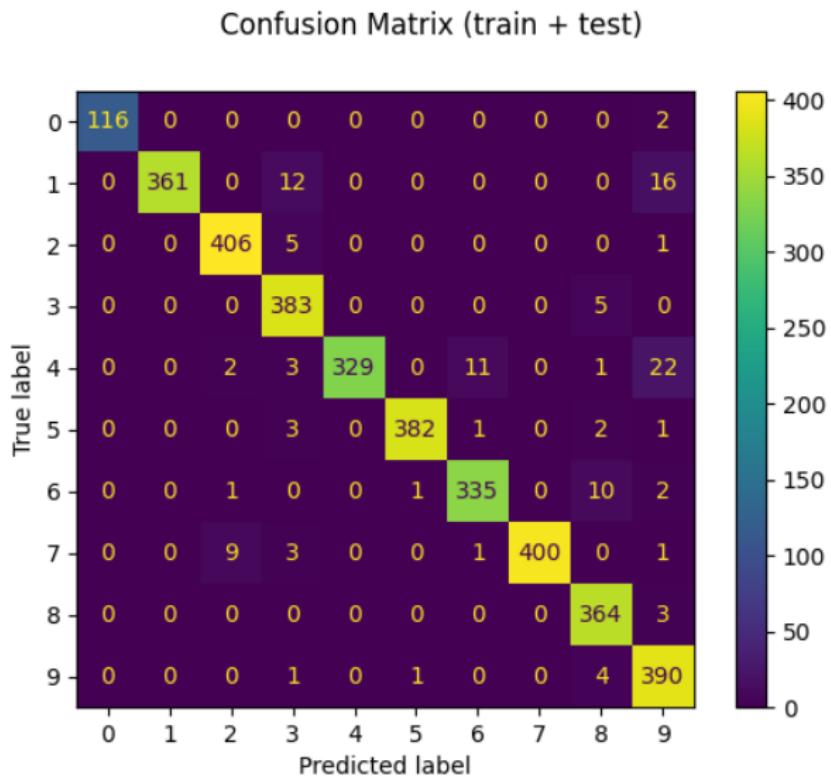
## Архитектура



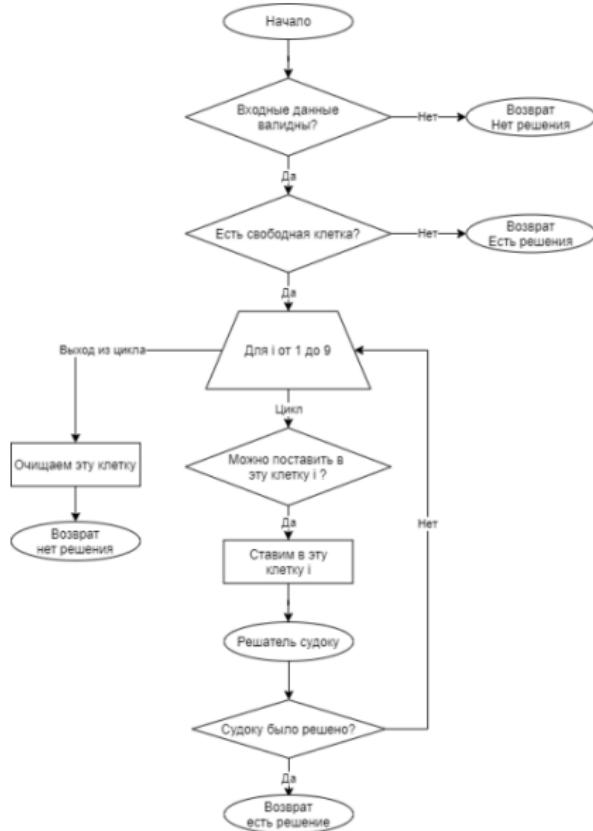
# Обучение нейросети



# Матрица ошибок обученной нейросети



# Рекурсивный перебор, поиск с возвратом (бэктрекинг)



## Задача о точном покрытии

Дано множество  $X$  и другое множество  $S$ , каждый элемент которого есть подмножество множества  $X$ . Требуется найти такое подмножество  $S^*$  множества  $S$ , чтобы каждый элемент из  $X$  был ровно в одном элементе выбранного подмножества.

Пример.

Пусть  $X = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ .

Пусть  $S = \{A, B, C, D, E\}$ , где  $A = \{0, 1, 4, 6\}$ ,  $B = \{2, 3, 5\}$ ,  
 $C = \{0, 4, 8\}$ ,  $D = \{0, 3, 3, 4, 5, 7, 8\}$ ,  $E = \{1, 6, 7\}$ .

Тогда  $S^* = \{B, C, E\}$  удовлетворяет поставленным условиям.

# Удобное представление данных задачи о точном покрытии

$$X = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$$

$$S = \{A, B, C, D, E\}, \text{ где } A = \{0, 1, 4, 6\}, B = \{2, 3, 5\}, \\ C = \{0, 4, 8\}, D = \{0, 2, 3, 4, 5, 7, 8\}, E = \{2, 6, 7\}$$

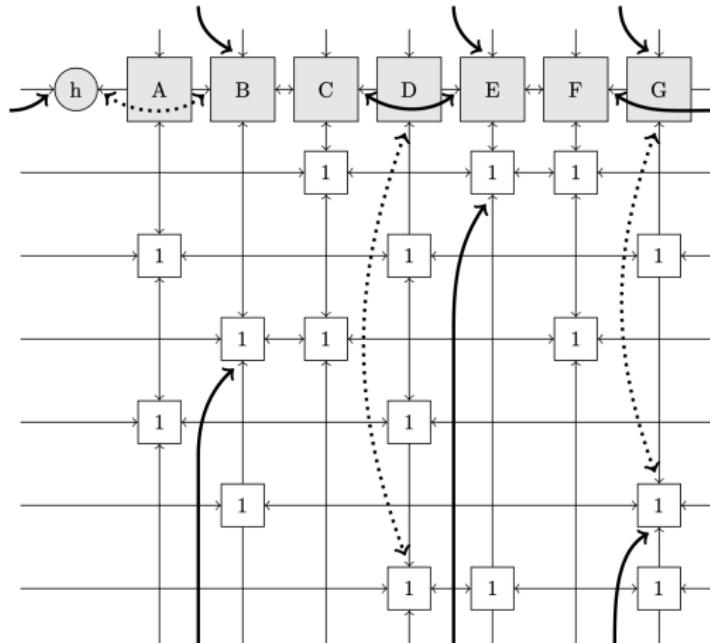
	0	1	2	3	4	5	6	7	8
A	1	1	0	0	1	0	1	0	0
B	0	0	1	1	0	1	0	0	0
C	1	0	0	0	1	0	0	0	1
D	1	0	1	1	1	1	0	1	1
E	0	1	0	0	0	0	1	1	0

# Судоку в терминах задачи о точном покрытии

Есть ли цифра в ячейке				Есть ли в строке с данным номером определённая цифра									Есть ли в столбце с данным номером определённая цифра									Есть ли в квадрате с данным номером определённая цифра								
Ячейка №0	Ячейка №1	...	Ячейка №80	В строке №0 цифра 1	В строке №0 цифра 2	...	В строке №1 цифра 1	...	В строке №8 цифра 9	В столбце №0 цифра 1	В столбце №0 цифра 2	...	В столбце №1 цифра 1	...	В столбце №8 цифра 9	В квад-те №0 цифра 1	В квад-те №0 цифра 2	...	В квад-те №1 цифра 1	...	В квад-те №8 цифра 9									
В строке 0 в столбце 0 стало цифру 1	1	0	...	0	1	0	...	0	...	0	1	0	...	0	...	0	1	0	...	0	...	0								
В строке 0 в столбце 0 стало цифру 2	1	0	...	0	0	1	...	0	...	0	0	1	...	0	...	0	0	1	...	0	...	0								
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...								
В строке 0 в столбце 1 стало цифру 1	0	1	...	0	1	0	...	0	...	0	0	0	...	1	...	0	1	0	...	0	...	0								
В строке 0 в столбце 1 стало цифру 2	0	1	...	0	0	1	...	0	...	0	0	0	...	0	...	0	0	1	...	0	...	0								
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...								
В строке 1 в столбце 0 стало цифру 1	0	0	...	0	0	0	...	1	...	0	1	0	...	0	...	0	1	0	...	1	...	0								
В строке 1 в столбце 0 стало цифру 2	0	0	...	0	0	0	...	0	...	0	0	1	...	0	...	0	0	1	...	0	...	0								
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...								
В строке 8 в столбце 8 стало цифру 9	0	0	...	1	0	0	...	0	...	1	0	0	...	0	...	1	0	0	...	0	...	1								

729 строк и 324 столбца

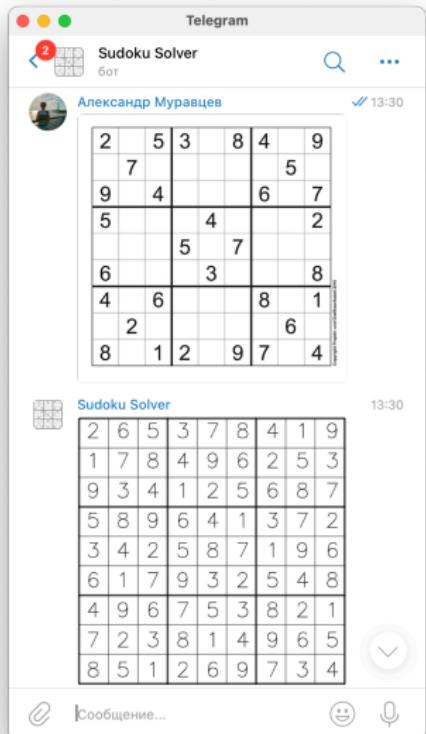
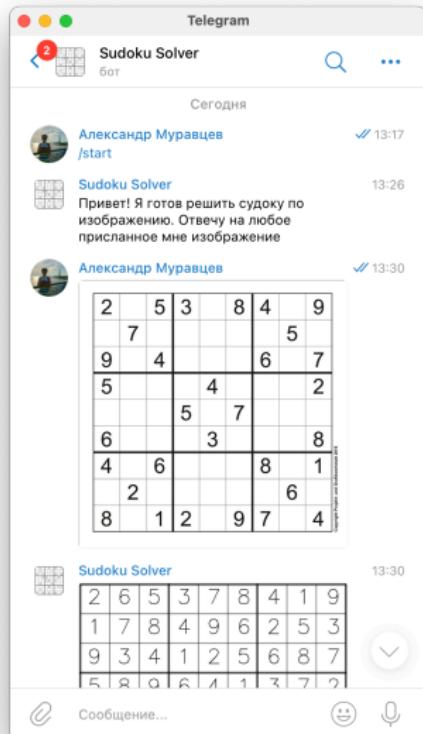
# Алгоритм X



Представление бинарной  
матрицы в виде  
многосвязного списка

Метод танцующих ссылок

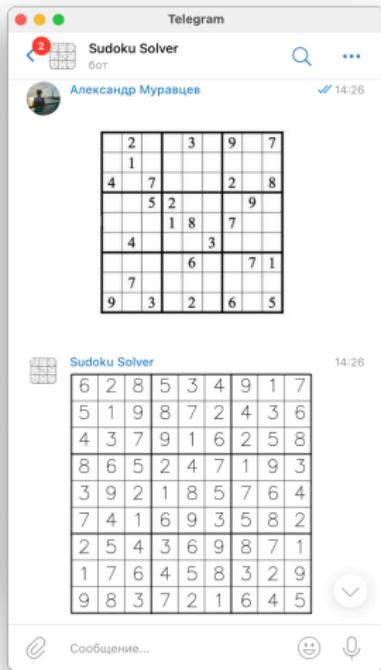
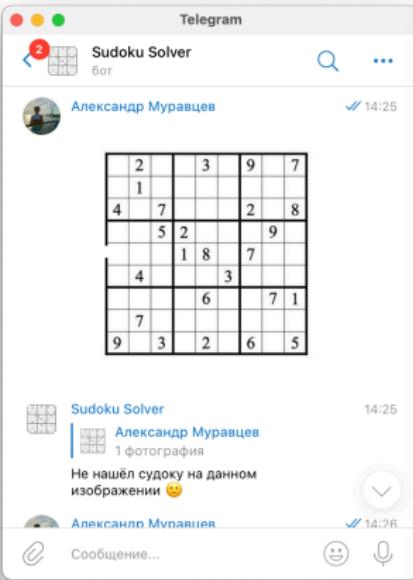
# Телеграм-бот с помощью aiogram



@sudoku\_grids\_solver\_bot



# Телеграм-бот



# Интерфейс GUI-приложения

Sudoku Solver

— □ ×

Camera image

Current scanned solvable sudoku field

[0. 2. 0. 0. 3. 0. 9. 0. 7.]
[0. 1. 0. 0. 0. 0. 0. 0. 0.]
[4. 0. 7. 0. 0. 0. 2. 0. 8.]
[0. 0. 5. 2. 0. 0. 0. 9. 0.]
[0. 0. 0. 1. 8. 0. 7. 0. 0.]
[0. 4. 0. 0. 0. 3. 0. 0. 0.]
[0. 0. 0. 0. 6. 0. 0. 7. 1.]
[0. 7. 0. 0. 0. 0. 0. 0. 0.]
[9. 0. 3. 0. 2. 0. 6. 0. 5.]]

Run Sudoku Solver

Select image with sudoku field manually

# Слоты PyQt

Слот преобразования openCV изображения в PyQt изображение:

```
@pyqtSlot(np.ndarray)
def update_image(self, cv_img):
    """Updates the image_label with a new openCV image"""
    qt_img = self.convert_cv_to_qt(cv_img)
    self.frame_field.setPixmap(qt_img)
```

Слот обновления текста внутри QTextEdit:

```
@pyqtSlot(np.ndarray)
def update_text(self, sudoku_to_solve):
    """Updates the image_label with a new openCV image"""
    self.scanned_sudoku.setText(np.array2string(sudoku_to_solve))
```

# Поток с результатом

Класс с выводом результата потока

```
class ThreadWithResult(threading.Thread):
    def __init__(self, group=None, target=None, name=None,
                 args=(), kwargs={}, *, daemon=None):
        self.result = None

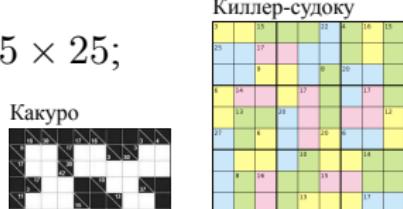
    def function():
        self.result = target(*args, **kwargs)
    super().__init__(group=group, target=function,
                     name=name, daemon=daemon)
```

Создаётся свой поток для каждого из трёх последовательных изображений, полученных с камеры. Если результаты после выполнения каждого из потоков совпадают, то отсканированный судоку записывается в текстовое поле PyQt приложения.

# Возможные пути развития

Дополнить программу методами сканирования и решения креативных судоку-подобных пазлов:

- ▶ больших размеров:  $16 \times 16$ ,  $12 \times 12$ ,  $25 \times 25$ ;
- ▶ двойные судоку;
- ▶ чёт-нечёт;
- ▶ х-судоку; пазл-судоку; виндоку;
- ▶ судоку больше-меньше;
- ▶ какуро;
- ▶ киллер-судоку;
- ▶ судоку-цветок



Судоку 12 на 12

3				1							
11	10	5	7	4	8	6					
4	3	10	1	5							
10	12	11		5	3						
12		2	6	5	7	11					
7	4					12					
7			2	6							
12	11	6	7	1		9					
5	6		11	10	2						
4	2	12	5	6							
9	6	12	4	2	10	8					
	10					4					

Судоку-цветок



Многие из представленных типов судоку могут быть сведены к задаче о точном покрытии.

# Возможные пути развития

Реализовать другой способ взаимодействия пользователя с решателем:

- ▶ встроить ругаме для наглядной визуализации скана судоку;
- ▶ веб-приложение.

Запустить собственный генератор классических судоку с заданным уровнем сложности.