CuroAI

План урока по программированию и искусственному интеллекту

20 занятий для школьников: 10 по управлению роботом и 10 по применению ИИ

На основе блоков Cubroid + редактор CuroAI Python.

Примеры рассчитаны на встроенную среду CuroAI/Python Studio. Проверяйте их перед занятием.

Куброид Инк.

Полный обзор учебной программы

Занятие	Единица	Тема урока	Ключевые фрагменты
1	Знакомьтесь, робот	Вперед и назад	dc_motor_basic, dc_motor_loop
2	Знакомьтесь, робот	Управление поворотом и скоростью	dc_motor_turn, dc_motor_speed, dc_motor_individual
3	😊 Выразительный робот	Эмоции со светодиодом	led_face, led_letter, led_text, led_icon
4	😊 Выразительный робот	Звук и мелодия	sound_doremi, sound_melody, sound_fx
5	Робот с датчиками	RGB-подсветка + датчики	master_signal, master_rainbow, master_blink, sensor_proximity, sensor_touch
6	Информационный робот	Визуализация данных датчиков	plotter_sensor
7	Повышение уровня кодирования	Создание функций (def)	def_basic, def_dance
8	Проект роботов	Создание робота-питомца	combo_pet
9	Проект роботов	Создание машины для викторин	combo_quiz
10	Обзор робота	Обзор деталей робота	Самостоятельный выбор
11	Первая встреча с ИИ	Обучаемая машина	tm_basic
12	Компьютерное зрение	Распознавание цвета	color_motor, color_led
13	Разговор с ИИ	Распознавание и синтез речи	voice_motor, tts_basic, tts_sensor
14	Сложный ИИ	Совместная работа средств распознавания	ai_combo
15	Жест ИИ	Распознавание рук	hand_motor, hand_gesture
16	😊 Камера обнаруживает	Обнаружение лица и анализ его	face_count, fm_expression

ЧАСТЬ 1: Основы управления роботом

Занятия 1–10 | Двигатель постоянного тока · Светодиод · Звук · Датчик · Функции · Проекты

CuroAI Программирование и обучение искусственному интеллекту

1

Занятие

Знакомьтесь, робот

Вперед и назад

Сложность: ★☆☆

Цели обучения

1. Понимание структуры платформы CuroAI.
2. Управляйте двигателями постоянного тока с помощью кода Python для перемещения робота вперед/назад.
3. Понять концепцию последовательного выполнения

Введение

Подумай об этом

Как движутся роботы?

Если мы скажем «иди вперед», сможет ли робот понять?

→ Давайте дадим команды роботу с помощью одной строчки кода!

Концепции

Ключевые понятия

1

Последовательное выполнение

2

Цикл for

3

Управление длительностью паузы sleep()

Практика

▸ Вперед/Назад

dc_motor_basic

```
from cubroid import dc_motor, sleep

# motor1 — левое колесо, motor2 — правое колесо
# Вперед 1 сек ⟶ Назад 1 сек
dc_motor.go()
sleep(1)
dc_motor.back()
sleep(1)
dc_motor.stop()
print("Done!")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Повторное вождение

dc_motor_loop

```
1  from cubroid import dc_motor, sleep
2
3 # Повторить 3 раза: вперед⟶назад.
4  for i in range(3):
5      print(f"Loop {i+1}")
6      dc_motor.go()
7      sleep(0.5)
8      dc_motor.back()
9      sleep(0.5)
10 dc_motor.stop()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Как движутся роботы? Посмотреть видео (5 мин)
2	Поиск	Подключитесь к CuroAI, сопряжение BLE, введите код вперед/назад (15 минут)
3	Практика	Составьте повтор движения вперёд и назад с помощью цикла (15 минут).
4	Подведение итогов	Обсудите, какое действие робота задаёт одна строка программы (10 минут).

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Попробуйте изменить паузу sleep: 0,5 → 1,0 → 2,0 секунды и
Прогнозируйте результаты, а затем сравнивайте с реальностью!

➡ Следующее занятие

Далее: поворот и управление скоростью

Фрагменты, использованные в этом занятии

▸ Вперёд и назад (dc_motor_basic) | Повтор движения (dc_motor_loop)

2

Занятие

Знакомьтесь, робот

Управление поворотом и скоростью

Сложность: ★★☆

Цели обучения

1. Понимание поворота через разницу скоростей левого и правого двигателя.
2. Исследуйте траектории, меняя параметры speed().
3. Настраивайте движение раздельным управлением моторами и проверяйте результат.

Введение

Подумай об этом

Машины меняют направление с помощью руля, но...

У робота всего 2 колеса!

Как поворачивает наш двухколёсный робот?

→ Подсказка: у каждого колеса разная скорость!

Концепции

Ключевые понятия

1

Параметр

2

Поворот за счёт разной скорости колёс

3

Переменные и диапазон

Практика

Поворот .

dc_motor_turn

```
1  from cubroid import dc_motor, sleep
2
3 # Вождение по квадратной дорожке
4  for i in range(4):
5      dc_motor.go()
6      sleep(1)
7      dc_motor.right()
8      sleep(0.5)
9  dc_motor.stop()
10  print("Маршрут по квадрату завершён!")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Занятие 2

Практика

Управление скоростью

dc_motor_speed

```
from cubroid import dc_motor, sleep

# speed(left, right): скорость левого и правого колёс, 0–255
# Прямая: одинаковая скорость с обеих сторон.
for spd in range(50, 256, 50):
    dc_motor.speed(spd, spd)
    print(f"Скорость: {spd}")
    sleep(0.5)

# Поворот налево: налево медленно, направо быстро.
dc_motor.speed(50, 200)
sleep(1)
dc_motor.stop()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Индивидуальное управление двигателем (1/2)

dc_motor_individual

```
from cubroid import dc_motor1, dc_motor2, sleep

# dc_motor1 = левое колесо, dc_motor2 = правое колесо
# Каждым можно управлять независимо!

# Влево только вперед
dc_motor1.go()
sleep(1)
dc_motor1.stop()

# Вправо только вперёд
dc_motor2.go()
sleep(1)
dc_motor2.stop()

# Противоположное направление: вращение на месте.
dc_motor1.go()
dc_motor2.back()
```

Практика

Индивидуальное управление двигателем (2/2)

dc_motor_individual

```
sleep(1)

# Индивидуальное управление скоростью.
dc_motor1.speed(100)
dc_motor2.speed(200)
sleep(1)

dc_motor1.stop()
dc_motor2.stop()
print("Проверка отдельных моторов завершена!")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Сравнение поворота автомобиля и разворота гусеничной машины (5 мин)
2	Поиск	Анализ и выполнение кода управления квадратной траекторией (10 минут)
3	Практика	Создайте свой собственный путь (треугольник/круг) (20 мин)
4	Показ	Демонстрация группового творческого пути (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Создайте треугольную или круговую траекторию и
Реализуйте это в коде!

➡ Следующее занятие

Далее: Эмоции со светодиодом 😊

Фрагменты, использованные в этом занятии

↩ Поворот (dc_motor_turn) | ⚡ Скорость (dc_motor_speed) | Отдельный мотор (dc_motor_individual)

3

Занятие

😊 Выразительный робот

Эмоции со светодиодом

Сложность: ★☆☆

Цели обучения

1. Понимание методов точечно-матричного светодиодного дисплея.
2. Последовательный вывод нескольких выражений с использованием списков и циклов.
3. Реализовать прокрутку текста и отображение значков.

Введение

Подумай об этом

😊😢😫😮😴ᶻᶻᶻ

Как ты выражаешь эмоции?

Роботы тоже могут выражать эмоции с помощью светодиодов 8x8!

→ Давайте сделаем это как смайлики!

Концепции

Ключевые понятия

1

Список

2

Итерация строки

3

Принцип точечной матрицы

Практика

Изменить выражения

led_face

```
from cubroid import led, sleep

# Изменить выражения
faces = ["happy", "sad", "angry", "surprise", "sleepy"]
for f in faces:
    led.face(f)
    print(f"Выражение: {f}")
    sleep(1)
led.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Отображение алфавита/цифры

led_letter

```
1  from cubroid import led, sleep
2
3  # Отображать алфавит A~E
4  for ch in "ABCDE":
5      led.letter(ch)
6      print(f"Буква: {ch}")
7      sleep(0.5)
8
9  # Обратный отсчет числа
10 for n in range(5, 0, -1):
11     led.number(n)
12     sleep(0.5)
13 led.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Прокрутка текста

led_text

```
from cubroid import led, sleep

# Прокручивать текст побуквенно
led.text("HELLO", 0.5)
sleep(1)
led.text("12345", 0.3)
sleep(1)
led.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Галерея иконок

led_icon

```
1  from cubroid import led, sleep
2
3 # Отображать различные значки
4  icons = ["heart", "star", "sun", "moon", "cloud",
5          "rain", "snow", "music", "check", "x",
6          "arrow_up", "arrow_down", "arrow_left", "arrow_right"]
7
8  for icon in icons:
9      led.show(icon)
10     print(f"Значок: {icon}")
11     sleep(0.5)
12 led.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Игра-угадайка эмоций эмодзи (5 мин)
2	Поиск	Запуск фрагментов выражений/букв/значков по порядку (15 минут)
3	Практика	Запрограммируйте свою собственную последовательность эмоций (15 минут)
4	Подведение итогов	Обсуждение удобства списковых типов данных (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Создайте свою собственную историю эмоций, используя более 5 выражений. создайте его и добавьте комментарии в свой код!

➡ Следующее занятие

Далее: Звук и мелодия

Фрагменты, использованные в этом занятии

😊 Изображения (led_face) | Буквы (led_letter) | Текст (led_text) | Значки (led_icon)

4

Занятие

😊 Выразительный робот

Звук и мелодия

Сложность: ★★☆

Цели обучения

1. Различать высоту звука и длительность ноты.
2. Представлять мелодии как данные с помощью списков кортежей.
3. Программировать последовательности нот и повторяющиеся фрагменты.

Введение

Подумай об этом

Слышали ли вы раньше музыку из 8-битных игр?

Компьютеры уже давно создают музыку в виде «данных».

→ Давайте тоже сочинять музыку с помощью кода!

Концепции

Ключевые понятия

1

Кортеж

2

Музыка как данные

3

Темп и длительность нот

Практика

Ре Ми Фа Соль Ла Си До

sound_doremi

```
from cubroid import sound, sleep

# до ре ми фа соль ля си (5-я октава)
notes = ["do", "re", "mi", "fa", "sol", "la", "si", "high_do"]
for n in notes:
    sound.play(n)
    print(f"♪ {n}")
    sleep(0.3)
sound.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Мелодия школьного звонка

sound_melody

```
from cubroid import sound

# Песня школьного звонка (названия нот, доли)
sound.melody([
    ("sol", 1), ("sol", 1), ("la", 1), ("la", 1),
    ("sol", 1), ("sol", 1), ("mi", 2),
    ("sol", 1), ("sol", 1), ("mi", 1), ("mi", 1),
    ("re", 2), ("rest", 1), ("rest", 1),
], bpm=120)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Звуковые эффекты

sound_fx

```
1  from cubroid import sound, sleep
2
3# Сирена: повторить две ноты
4  for i in range(5):
5      sound.play("high_do")
6      sleep(0.15)
7      sound.play("mi")
8      sleep(0.15)
9  sound.off()
10
11  sleep(0.5)
12
13 # Восходящий тон
14  for n in ["do", "re", "mi", "fa", "sol", "la", "si", "high_do"]:
15      sound.play(n)
16      sleep(0.1)
17  sound.off()
```

Деятельность

Ход урока

1	Введение	Введение в историю компьютерной музыки (8-битный звук) (5 минут)
2	Поиск	Воспроизведите гамму и разберите мелодию звонка (10 минут).
3	Практика	Сочините собственную 4-тактную мелодию + создайте звуковые эффекты (20 мин)
4	Показ	Демонстрация групповых мелодий (10 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Сочините свою собственную мелодию из 4+ тактов и

Заполните таблицу сравнения нот!

➡ Следующее занятие

Далее: Мастер-светодиод + датчики

Фрагменты, использованные в этом занятии

Ноты (sound_doremi) | Мелодия (sound_melody) | Сигналы (sound_fx)

5

Занятие

Робот с датчиками

RGB-подсветка ✦ датчики

Сложность: ★★☆

Цели обучения

1. Понимать и использовать цветовую модель RGB.
2. Понимание принципов и применения датчиков приближения и касания.
3. Написание программ реагирования датчиков с условными выражениями (if-elif-else)

Введение

Подумай об этом

Почему светофоры красно-желто-зеленые?

Если дать роботу глаза (датчики) и сигналы (светодиоды)

Разве он не сможет судить самостоятельно?

Концепции

Ключевые понятия

1

Цветовая модель RGB

2

Условное (if/elif/else)

3

Вход датчика → Процесс → Выход

Практика

Светофор

master_signal

```
from cubroid import master, sleep

# Светофор: Красный→Желтый→Зеленый.
master.red()
sleep(2)
master.yellow()
sleep(1)
master.green()
sleep(2)
master.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Цвета радуги

master_rainbow

```
1 from cubroid import master, sleep
2
3#Радуга 7 цветов по порядку
4 master.rainbow(0.5)
5 master.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Мигает

master_blink

```
from cubroid import master, sleep

# Мигать красным 5 раз.
for i in range(5):
    master.red()
    sleep(0.3)
    master.off()
    sleep(0.3)
print("Мигание завершено!")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Реакция датчика приближения

sensor_proximity

```
from cubroid import dc_motor, proximity_sensor, master, sleep

# Датчик приближения: если близко, назад + красный
while True:
    if proximity_sensor.is_near():
        master.red()
        dc_motor.back()
        sleep(0.5)
        dc_motor.stop()
    else:
        master.green()
        dc_motor.go()
    sleep(0.1)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Блок освещённости и касания

sensor_touch

```
from cubroid import dc_motor, light_touch_sensor, sound, sleep

# Нажмите → вперед + звуковые эффекты, показать яркость
while True:
    if light_touch_sensor.is_pressed():
        sound.play("do")
        dc_motor.go()
        sleep(1)
        dc_motor.stop()
         sound.off()
    bright = light_touch_sensor.brightness()
    print(f"Освещённость: {bright}%")
    sleep(0.2)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Эксперимент по смешиванию цветов RGB (5 мин)
2	Поиск	Запустить программу светофора, проанализировать код ответа датчика (15 мин)
3	Практика	Комбинация датчик+светодиод: красный, когда близко, зеленый, когда далеко (15 минут).
4	Подведение итогов	Рассмотрение условной структуры, реальные условные примеры (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

с 3+ условными ветвями

Создайте программу реагирования датчиков!

➡ Следующее занятие

Далее: Визуализация данных датчиков

Фрагменты, использованные в этом занятии

Сигналы (master_signal) | Цвета (master_rainbow) | Мигание (master_blink) | Приближение (sensor_proximity) | Касание и свет (sensor_touch)

6

Занятие

Информационный робот

Визуализация данных датчиков

Сложность: ★★☆

Цели обучения

1. Визуализация данных датчиков в реальном времени в виде графиков.
2. Соберите данные, сравните их и объясните, что удалось установить.
3. Понимание циклов while и обработки данных в реальном времени.

Введение

Подумай об этом

Вы видели график ЭКГ в больнице?

Вы также можете просматривать данные датчиков в виде графиков в реальном времени!

→ Давайте посмотрим на мир через датчики с помощью Data Plotter!

Концепции

Ключевые понятия

1

Визуализация данных

2

Бесконечный цикл while True

3

Мониторинг в реальном времени

Практика

График данных датчика

plotter_sensor

```
1  from cubroid import proximity_sensor, light_touch_sensor, sleep
2
3  #    Показания датчика на графике в реальном времени
4  # Нажмите кнопку    рядом с консолью, чтобы открыть график.
5
6  plotter.show()
7  plotter.clear()
8
9  while True:
10     bright = light_touch_sensor.brightness()
11     touch = 100 if light_touch_sensor.is_pressed() else 0
12     near = 100 if proximity_sensor.is_near() else 0
13
14     plotter.add("brightness", bright)
15     plotter.add("touch", touch)
16     plotter.add("proximity", near)
17
18     sleep(0.2)
```

Деятельность

Ход урока

1	Введение	Что такое визуализация данных? Примеры графиков на каждый день (5 минут)
2	Поиск	Наблюдение за графиками датчиков в реальном времени с помощью Data Plotter (10 минут)
3	Практика	Сбор данных о яркости/прикосновении/близости, анализ закономерностей (20 минут)
4	Подведение итогов	О чем нам говорят данные: Написание отчета о наблюдении (10 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Обнаружение закономерностей в данных датчиков и
Напишите отчет о наблюдении!

➡ Следующее занятие

Далее: Создание функций (def)

Фрагменты, использованные в этом занятии

График показаний (plotter_sensor)

7

Занятие

Повышение уровня кодирования

Создание функций (def)

Сложность: ★★☆

Цели обучения

1. Понять концепцию и необходимость функций.
2. Определите свои собственные команды с помощью ключевого слова def.
3. Выделите повторяющееся действие в функцию и используйте её в нескольких частях программы.

Введение

Подумай об этом

Удобно ли каждый раз заново объяснять все шаги приготовления блюда?

Удобно иметь готовый рецепт!

В программировании тоже можно создавать «свои команды».

→ Именно это и есть функция (def)!

Концепции

Ключевые понятия

1

Определение и вызов функции

2

Абстракция кода

3

Модульность и повторное использование

Практика

Создавайте свои собственные движения

def_basic

```
1  from cubroid import dc_motor, sleep
2
3  # Создавайте свои собственные команды
4  def forward():
5      dc_motor.go()
6      sleep(1)
7      dc_motor.stop()
8
9  def spin():
10     dc_motor.speed(200, -200)
11     sleep(0.5)
12     dc_motor.stop()
13
14 # Использовать созданные команды
15 forward()
16 spin()
17 forward()
18 print("Done!")
```

Практика

Танцевальный робот (Функция+Цикл) (1/2)

def_dance

```
from cubroid import dc_motor, led, sound, sleep

# Создание танцевальных движений как функций
def step_right():
    dc_motor.speed(200, 50)
    sleep(0.3)
    dc_motor.stop()

def step_left():
    dc_motor.speed(50, 200)
    sleep(0.3)
    dc_motor.stop()

def wiggle():
    for i in range(3):
        step_right()
        step_left()

```

Практика

Танцевальный робот (Функция✦Цикл) (2/2)

def_dance

```
def celebrate():
    led.face("love")
    sound.play("high_do")
    sleep(0.5)

# Время танцев!
print("  Начинаем танец!")
for i in range(2):
    print(f"Dance {i+1}")
    wiggle()
    celebrate()

dc_motor.stop()
led.face("happy")
print("  Танец завершён!")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Объясните функцию на примере рецепта (5 минут).
2	Поиск	Создайте свои собственные функции действий (вперед, вращение) (10 мин)
3	Практика	Танцевальный робот: создайте хореографию, комбинируя функции (20 минут)
4	Показ	Демонстрация групповой танцевальной хореографии + обзор кода (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Определите 3+ функции и объедини их, чтобы создать танцевальную хореографию!

➡ Следующее занятие

Далее: Создание робота-питомца

Фрагменты, использованные в этом занятии

Свои действия (def_basic) | Танцующий робот: функции и цикл (def_dance)

8

Занятие

Проект роботов

Создание робота-питомца

Сложность: ★★★

Цели обучения

1. Согласование действий робота с показаниями датчиков
2. Создайте робота-питомца, сочетая условия, движение, матрицу и звук.
3. Спланируйте проект, соберите модель, составьте программу и проверьте её.

Введение

Подумай об этом

Что, если бы у вас был робот-компаньон?

Радуюсь, когда ты приближаешься,

Робот-питомец, который засыпает, когда никого нет рядом!

→ Давайте объединим все, что мы узнали!

Концепции

Ключевые понятия

1

Программирование на основе состояний

2

Связь входных данных, обработки и действия робота

3

Процесс разработки проекта

Практика

Робот -питомец (1/2)

combo_pet

```
from cubroid import dc_motor, proximity_sensor, led, sound, master, sleep

# Робот-питомец: эмоциональный робот, реагирующий на датчики.
led.face("happy")
master.green()
print("   Робот-питомец проснулся!")

while True:
    if proximity_sensor.is_near():
# Радуюсь, когда кто-то приближается
        led.face("love")
        master.rgb(255, 100, 200)
        sound.play("sol")
        sleep(0.2)
        sound.play("high_do")
        sleep(0.3)

# Радостно поприветствовать (потрясти влево/вправо)
```

Практика

Робот -питомец (2/2)

combo_pet

```
        dc_motor.speed(150, 50)
        sleep(0.3)
        dc_motor.speed(50, 150)
        sleep(0.3)
        dc_motor.stop()
    else:
# Сон, когда никого нет рядом
        led.face("sleepy")
        master.rgb(50, 50, 100)
    sleep(0.3)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Посмотрите видео о будущих роботах-компаньонах (5 минут).
2	Дизайн	Разработка сценария поведения питомца-робота (карта эмоций) (10 мин)
3	Реализация	Написать код и проверить работу (20 мин)
4	Демо	Демонстрация группового робота + обратная связь (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Создайте свою собственную карту эмоций и
Реализуйте 4+ эмоциональных реакции!

➡ Следующее занятие

Далее: Создание машины для викторин

Фрагменты, использованные в этом занятии

Робот-питомец (combo_pet)

9

Занятие

Проект роботов

Создание машины для викторин

Сложность: ★★★

Цели обучения

1. Управление данными викторины с помощью списков и индексации.
2. Использовать переменную счёта и условные операторы.
3. Разработать дизайн пользовательского интерфейса (светодиод + звуковая обратная связь).

Введение

Подумай об этом

Есть ли у вас любимая викторина или игровое шоу?

Выберите «верно» или «неверно» с помощью кнопки и

Используется светодиод и звук для обозначения правильных/неправильных ответов.

Давайте создадим свою собственную машину для викторин!

Концепции

Ключевые понятия

1

Индексирование списка

2

Переменное накопление (оценка)

3

Булевы и логические операции

Практика

Машина для викторин (1/3)

combo_quiz

```
from cubroid import led, sound, master, light_touch_sensor, sleep

# Машина для викторин: отвечайте на О/Х касанием
questions = [
    "Q1: 1+1=2?",
    "Q2: 3>5?",
    "Вопрос 3: Python — язык программирования?",
]
answers = [True, False, True]
score = 0

for i in range(len(questions)):
    print(f"\\n{questions[i]}")
    led.text(f"Q{i+1}", 0.3)
    sleep(1)
    print("Нажатие — верно, без нажатия — неверно. На ответ три секунды.")
    master.yellow()

```

Практика

Машина для викторин (2/3)

combo_quiz

```
    touched = False
    for t in range(30):
        if light_touch_sensor.is_pressed():
            touched = True
        sleep(0.1)

    correct = (touched == answers[i])
    if correct:
        score = score + 1
        master.green()
        sound.play("high_do")
        led.face("happy")
        print("Correct!")
    else:
        master.red()
        sound.play("do")
        led.face("sad")
        print("Wrong!")
```

Практика

Машина для викторин (3/3)

combo_quiz

```
    sleep(1)

led.off()
master.off()
print(f"\\nИтоговый счёт: {score}/{len(questions)}")
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Поделитесь любимыми приложениями-викторинами/игровыми шоу (5 минут)
2	Анализ	Разберите программу викторины: проследите изменение переменных (10 минут).
3	Практика	Добавьте 5 собственных вопросов викторины, отрегулируйте сложность (20 минут).
4	Опыт	Обменяйтесь и решите викторины с партнером (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Добавьте 5+ вопросов и
Добавьте отображение доли правильных ответов.

➡ Следующее занятие

Далее: итоговый проект по управлению роботом.

Фрагменты, использованные в этом занятии

Викторина (combo_quiz)

10

Занятие

Обзор робота

Обзор деталей робота

Сложность: ★★★

Цели обучения

1. Обзор и подведение итогов занятий 1–9.
2. Реализуйте творческие программы для роботов с помощью самостоятельных проектов.
3. Обсудите работы в парах: что получилось и что можно улучшить.

Введение

Подумай об этом

Мобилизуйте все, что вы узнали!

Моторы, матрица, звук, датчики, построение графиков и функции…

Свободно комбинируйте все это, чтобы

Создадим своего оригинального робота!

Концепции

Ключевые понятия

1

Комплексный обзор

2

Планирование и реализация проекта

3

Экспертная оценка

Деятельность

Ход урока

1	Подведение итогов	Повторение ключевых понятий управления роботом (10 минут).
2	Свой проект	Групповое программирование роботов на произвольную тему (25 мин)
3	Показ	Групповые презентации + взаимооценка (10 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Создать программу-робот на свободную тему и
Подготовьте план, программу и демонстрацию.

➡ Следующее занятие

Далее: знакомство с Teachable Machine — обучением модели на примерах.

ЧАСТЬ 2: Расширенная интеграция с ИИ

Занятия 11–20 | ТМ · Цвет · Голос · Рука · Лицо · Поза · Предмет · Обзор

11

Занятие

Первая встреча с ИИ

Обучаемая машина

Сложность: ★★☆

Цели обучения

1. Основы машинного обучения: примеры, обучение и классификация
2. Обучение модели классификации изображений с помощью Teachable Machine.
3. Соедините обученную модель с роботом.

Введение

Подумай об этом

Как ИИ различает объекты?

Отличать яблоки от бананов

Вы говорите ИИ: «Это яблоко, это банан».

тебе нужно это сказать.

→ Давайте сами обучать ИИ!

Концепции

Ключевые понятия

1

Основы машинного обучения

2

Данные обучения и классификация

3

Развертывание модели (URL)

Практика

Классификация TM → Мотор (1/2)

tm_basic

```
from cubroid import dc_motor, master, sleep
from cubroid.ai import camera, teachable

#1. Запустить камеру.
camera.start()
sleep(1)

# 2. Загрузка модели Teachable Machine.
#    Введите URL-адрес на правой панели и загрузите для автоматического подключения.
teachable.load("PASTE_TM_URL_HERE")
sleep(2)

#3. Действие по результату классификации
while True:
    result = teachable.predict()
    name = result["label"]
    score = result["score"]
    print(f"Результат классификации: {name} ({score}%)")
```

Практика

Классификация TM → Мотор (2/2)

tm_basic

```
19
20  if name == "Class 1":
21      dc_motor.go()
22      master.green()
23  elif name == "Class 2":
24      dc_motor.back()
25      master.red()
26  else:
27      dc_motor.stop()
28      master.off()
29  sleep(0.5)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Как ИИ различает объекты? (5 мин)
2	Опыт	Обучите модель Teachable Machine различать два класса (15 минут).
3	Подкл.	URL модели → Подключение CuroAI, управление двигателем по классам (15 мин)
4	Подведение итогов	Обсуждение важности обучающих данных (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Обучите модель различать не менее двух классов и
Проверьте результаты на новых примерах.

➡ Следующее занятие

Далее: Распознавание цвета

Фрагменты, использованные в этом занятии

Классификация Teachable Machine и мотор (tm_basic)

12

Занятие

Компьютерное зрение

Распознавание цвета

Сложность: ★★☆

Цели обучения

1. Понимать принципы определения цвета компьютерным зрением.
2. Реализация распознавания цветов на основе RGB и условного ветвления.
3. Проследите, как изображение с камеры обрабатывается системой ИИ и вызывает действие робота.

Введение

Подумай об этом

Люди и компьютеры видят цвета одинаково?

Наши глаза: «Он красный!»

Компьютер: «R=255, G=20, B=30!»

→ Давайте распознавать цвета глазами ИИ!

Концепции

Ключевые понятия

1

Основы компьютерного зрения

2

Цветовое пространство RGB

3

Входные данные, обработка ИИ и действие робота

Практика

Цвет → Мотор (1/2)

color_motor

```
1  from cubroid import dc_motor, master, sleep
2  from cubroid.ai import camera, color
3
4# Запустить камеру (распознавание цвета происходит автоматически)
5  camera.start()
6  sleep(1)
7
8 # Управление роботом по цвету
9  while True:
10     c = color.name_en()
11     print(f"Определён цвет: {c}")
12
13     if c == "red":
14         dc_motor.stop()
15         master.red()
16     elif c == "green":
17         dc_motor.go()
18         master.green()
```

Практика

Цвет → Мотор (2/2)

color_motor

```
19    elif c == "blue":
20        dc_motor.back()
21        master.rgb(0, 0, 255)
22    sleep(0.3)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Цвет → Светодиод

color_led

```
from cubroid import master, led, sleep
from cubroid.ai import camera, color

camera.start()
sleep(1)

# Отображение обнаруженного цвета на главном светодиоде
while True:
    r, g, b = color.rgb()
    master.rgb(r, g, b)
    c = color.name_en()
    if c:
        led.letter(c[0].upper())
    sleep(0.3)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Дебаты: различия в распознавании цветов человеком и компьютером (5 минут)
2	Поиск	Запустите цвет → код управления двигателем, проверьте различные цвета (15 минут)
3	Практика	Цвет → Сопоставление основного светодиода + добавление собственных правил цвета (15 минут)
4	Подведение итогов	Изучите значения RGB, обсудите ограничения распознавания цветов (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Карта 4+ цветов и

Заполните лист записи значений RGB!

➡ Следующее занятие

Далее: Распознавание и синтез речи

Фрагменты, использованные в этом занятии

Цвет и мотор (color_motor) | Цвет и подсветка (color_led)

13

Занятие

Разговор с ИИ

Распознавание и синтез речи

Сложность: ★★☆

Цели обучения

1. Понимание принципов STT и TTS.
2. Написать программу для управления роботом с помощью голосовых команд.
3. Искать ключевое слово в строке с помощью оператора in.

Введение

Подумай об этом

«Привет, Siri», «Alexa», «Привет, Google»

Как голосовые помощники обрабатывают нашу речь?

→ Распознавание речи выполняет компьютер; языковую поддержку проверяют отдельно.

Он даже может говорить!

Концепции

Ключевые понятия

1

CTT/TTC

2

Поиск строки (внутри)

3

Основы обработки естественного языка

Практика

Голос → Мотор (1/2)

voice_motor

```
from cubroid import dc_motor, master, led, sleep
from cubroid.ai import voice

# Запустить распознавание речи
voice.start()
print("Команды примера: go, stop, left, right. Проверьте язык распознавания.")

while True:
    text = voice.text()
    if text:
        print(f"Результат распознавания: {text}")
        if "go" in text or "forward" in text:
            dc_motor.go()
            master.green()
            led.face("happy")
        elif "stop" in text or "halt" in text:
            dc_motor.stop()
            master.yellow()
```

Практика

Голос → Мотор (2/2)

voice_motor

```
        led.face("sleepy")
    elif "left" in text:
        dc_motor.left()
        master.rgb(0, 0, 255)
    elif "right" in text:
        dc_motor.right()
        master.rgb(255, 0, 255)
    sleep(1)
sleep(0.2)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Робот говорит

tts_basic

```
from cubroid.ai import tts
from cubroid import led, sleep

# Робот говорит (TTS)
led.face("happy")
tts.say("Здравствуйте! Я робот Curo.")
sleep(2)
tts.say("Давайте заниматься вместе!")
sleep(2)
led.off()
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Сенсор → Говорить

tts_sensor

```
1  from cubroid import proximity_sensor, master, sleep
2  from cubroid.ai import tts
3
4  # Реакция датчика приближения + TTS
5  print("Поднесите руку к датчику!")
6  count = 0
7
8  while True:
9      if proximity_sensor.is_near():
10         count = count + 1
11         master.green()
12         tts.say(f"Обнаружение номер {count}! Здравствуйте!")
13         sleep(2)
14         master.off()
15     sleep(0.2)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Как голосовые помощники обрабатывают нашу речь? (5 минут)
2	Поиск	Голос → управление двигателем + заставить робота говорить с помощью TTS (15 минут)
3	Практика	Создайте свой набор голосовых команд + добавьте ответы TTS (15 минут)
4	Подведение итогов	Исследование и презентация реальных приложений голосового искусственного интеллекта (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

После проверки языковой поддержки составьте набор из пяти голосовых команд и
Добавьте отзыв TTS!

➡ Следующее занятие

Далее: Мультисенсорный ИИ

Фрагменты, использованные в этом занятии

Голос и мотор (voice_motor) | Синтез речи (tts_basic) | Датчик и речь (tts_sensor)

14

Занятие

Сложный ИИ

Совместная работа средств распознавания

Сложность: ★★★

Цели обучения

1. Понимание сложных систем, использующих одновременно несколько функций искусственного интеллекта.
2. Согласуйте две функции распознавания. Голосовые возможности используйте только после проверки языковой поддержки.
3. Разработка приоритетов обработки нескольких входов.

Введение

Подумай об этом

В беспилотных автомобилях используются камеры, лидар, радар...

Мы используем сразу несколько датчиков!

Наш робот может использовать распознавание цвета + речь + TTS.

Можем ли мы использовать их все сразу?

Концепции

Ключевые понятия

1

Мультимодальный ИИ

2

Обработка нескольких входов

3

Проект системной интеграции

Практика

Цвет ✦Голос✦TTS (1/2)

ai_combo

```
from cubroid import dc_motor, master, sleep
from cubroid.ai import camera, color, voice, tts

# Распознавание цвета + голос + TTS в сочетании
camera.start()
voice.start()
sleep(1)

print("Робот готов к работе!")
tts.say("Здравствуйте! Покажите цвет или подайте команду!")
sleep(2)

while True:
    text = voice.text()
    if text:
        print(f"Речь: {text}")
        if "forward" in text:
            dc_motor.go()
```

Практика

Нет ✦Голос✦TTS (2/2)

ai_combo

```
19            tts.say("Движение вперёд!")
20        elif "back" in text:
21            dc_motor.back()
22            tts.say("Движение назад!")
23        elif "stop" in text:
24            dc_motor.stop()
25            tts.say("Стоп!")
26        sleep(1)
27
28    c = color.name_en()
29    if c == "red":
30        master.red()
31    elif c == "green":
32        master.green()
33    elif c == "blue":
34        master.rgb(0, 0, 255)
35    sleep(0.3)
```

Деятельность

Ход урока

1	Введение	Мультисенсорные приложения в автономных транспортных средствах (5 мин)
2	Поиск	Анализ сложной структуры кода искусственного интеллекта (нарисование блок-схемы входных данных) (10 минут)
3	Практика	Внедрение и тестирование интегрированного робота цвета+голоса+TTS (20 мин)
4	Подведение итогов	Дискуссия: Мультимодальная обработка в системах искусственного интеллекта (10 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Интеграция двух+ функций искусственного интеллекта и Покажите обработку входных данных на блок-схеме.

➡ Следующее занятие

Далее: Распознавание рук

Фрагменты, использованные в этом занятии

Цвет, голос и синтез речи (ai_combo)

15

Занятие

Жест ИИ

Распознавание рук

Сложность: ★★☆

Цели обучения

1. Понимание принципов обнаружения ориентиров рук.
2. Управляйте роботом с помощью счета пальцев/жестов.
3. Использовать модуль random для случайного выбора в игре.

Введение

Подумай об этом

Разведите 5 пальцев... вперед!

Сожми кулак... остановись!

Управляйте роботом, используя только руки.

Он даже может играть в камень-ножницы-бумагу!

Концепции

Ключевые понятия

1

Обнаружение ориентиров

2

случайный модуль

3

Логика игры (победа/проигрыш)

Практика

Пальцы → Мотор (1/2)

hand_motor

```
from cubroid import dc_motor, master, led, sleep
from cubroid.ai import camera, hand

camera.start()
hand.start()
sleep(2)

print("Управляйте роботом числом поднятых пальцев!")

while True:
    fingers = hand.fingers()
    print(f"Пальцев: {fingers}")
    led.number(fingers)

    if fingers >= 4:
        dc_motor.go()
        master.green()
    elif fingers == 1:
```

Практика

Пальцы → Мотор (2/2)

hand_motor

```
19        dc_motor.back()
20        master.red()
21    elif fingers == 0:
22        dc_motor.stop()
23        master.off()
24    sleep(0.3)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Камень -Ножницы-Бумага (1/3)

hand_gesture

```
1  from cubroid import sound, master, led, sleep
2  from cubroid.ai import camera, hand
3  import random
4
5  camera.start()
6  hand.start()
7  sleep(2)
8
9  print("Сыграем в камень, ножницы, бумагу!")
10 print("   Покажите кисть камере целиком!")
11 score = 0
12
13 while True:
14     g = hand.gesture()
15     if g in ["scissors", "rock", "paper"]:
16         choices = ["scissors", "rock", "paper"]
17         robot = random.choice(choices)
18
```

Практика

Камень -Ножницы-Бумага (2/3)

hand_gesture

```
19      print(f"Мой жест: {g}; выбор робота: {robot}")
20
21      if g == robot:
22          master.yellow()
23          led.face("surprise")
24          sound.play("mi")
25          print("Draw!")
26      elif (g == "scissors" and robot == "paper") or (g == "rock" and robot == "scissors") or (g == "paper" and robot == "rock"):
27          score = score + 1
28          master.green()
29          led.face("happy")
30          sound.play("high_do")
31          print(f"Победа! Счёт: {score}")
32      else:
33          master.red()
34          led.face("sad")
35          sound.play("do")
36          print("Defeated!")
```

Практика

Камень -ножницы-бумага (3/3)

hand_gesture

```
37          sleep(2)
38      sleep(0.3)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Введение в язык жестов и распознавание жестов (5 минут)
2	Поиск	Жест руки → действие мотора; проверка реакции (10 минут).
3	Практика	Создайте игру «Камень, ножницы, бумага» и запишите долю побед (20 минут).
4	Подведение итогов	Мозговой штурм: приложения для распознавания жестов с помощью искусственного интеллекта (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Запишите долю побед в десяти раундах

и

Добавьте функцию светодиодного табло!

➡ Следующее занятие

Далее: Распознавание и выражение лиц 😊

Фрагменты, использованные в этом занятии

Пальцы и мотор (hand_motor) | Камень, ножницы, бумага (hand_gesture)

16

Занятие

😊 Камера обнаруживает лицо

Обнаружение лица и анализ его выражения

Сложность: ★★☆

Цели обучения

1. Различать обнаружение лица, установление личности и классификацию выражения.
2. Составлять программу реакции на число обнаруженных лиц и видимое выражение.
3. Обсудить этику ИИ (конфиденциальность распознавания лиц)

Введение

Подумай об этом

😊 По выражению лица мы предполагаем, что чувствует человек.

Всегда ли такое предположение верно?

Модель классифицирует видимое выражение, но не определяет мысли и чувства человека.

Но... всегда ли эта технология хороша?

→ Давайте вместе подумаем о технологиях и этике!

Концепции

Ключевые понятия

1

Обнаружение лица и установление личности — разные задачи

2

Классификация выражений

3

Этика ИИ (конфиденциальность)

Практика

😊 Подсчёт обнаруженных лиц (1/2)

face_count

```
from cubroid import master, sound, led, sleep
from cubroid.ai import camera, face

camera.start()
face.start()
sleep(2)

print("Обнаружение лиц!")

while True:
    n = face.count()
    print(f"Лиц обнаружено: {n}")

    if n == 0:
        master.off()
        led.face("sleepy")
    elif n == 1:
        master.green()
```

Практика

😊 Подсчёт обнаруженных лиц (2/2)

face_count

```
        led.face("happy")
        sound.play("do")
    elif n >= 2:
        master.rgb(255, 100, 200)
        led.face("love")
        sound.play("sol")
    sleep(0.5)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Выражение → Ответ робота (1/2)

fm_expression

```
from cubroid import led, master, sound, sleep
from cubroid.ai import camera, face_mesh

camera.start()
face_mesh.start()
sleep(2)

print("Измените выражение лица!")
# Выражения: smile, big_smile, surprise, eyes_closed, neutral

while True:
    expr = face_mesh.expression()
    if expr:
        print(f"Выражение: {expr}")

        if expr == "smile" or expr == "big_smile":
            led.face("happy")
            master.green()
```

Практика

Выражение → Ответ робота (2/2)

fm_expression

```
        sound.play("sol")
    elif expr == "surprise":
        led.face("surprise")
        master.yellow()
        sound.play("high_do")
    elif expr == "eyes_closed":
        led.face("sleepy")
        master.rgb(100, 0, 255)
        sound.play("mi")
    else:
        led.face("happy")
        master.off()
sleep(0.5)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Обсудить плюсы и минусы распознавания лиц (5 мин)
2	Поиск	Обнаружение количества лиц → Светодиодный/звуковой отклик (10 минут)
3	Практика	Распознавание выражений лица → программа зеркального отображения эмоций робота (20 минут)
4	Подведение итогов	Этика ИИ: дебаты о распознавании лиц и конфиденциальности (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Задайте не менее трёх реакций на выражение лица и
Сформулируйте правила ответственного использования ИИ.

➡ Следующее занятие

Далее: Распознавание поз

Фрагменты, использованные в этом занятии

😊 Число лиц (face_count) | Выражение и действие (fm_expression)

17

Занятие

Распознавание движений

Распознавание позы

Сложность: ★★★

Цели обучения

1. Понять принципы оценки позы тела.
2. Реализуйте обнаружение действий посредством сравнения координат.
3. Создайте интерактивного танцевального робота.

Введение

Подумай об этом

Знаете ли вы о технологии захвата движения?

Компьютерные персонажи фильмов также повторяют движения человека!

Когда мы поднимаем руки, робот идёт вперёд,

составим программу остановки при их опускании.

Концепции

Ключевые понятия

1

Оценка позы (координаты)

2

Сравнение координат

3

Обнаружение изменения состояния

Практика

Движение рук → Мотор (1/2)

pose_arm

```
from cubroid import dc_motor, master, led, sleep
from cubroid.ai import camera, pose

camera.start()
pose.start()
sleep(2)

print("Поднимите руки для управления роботом!")
print("Руки вверх — вперёд; руки вниз — стоп")
print("Левая рука — налево; правая — направо")

while True:
    left = pose.left_wrist()
    right = pose.right_wrist()
    nose = pose.nose()

    if left and right and nose:
        l_up = left["y"] < nose["y"]
```

Практика

Движение рук → Двигатель (2/2)

pose_arm

```
19      r_up = right["y"] < nose["y"]
20
21      if l_up and r_up:
22          dc_motor.go()
23          master.green()
24          led.face("happy")
25      elif l_up:
26          dc_motor.left()
27          master.rgb(0, 0, 255)
28      elif r_up:
29          dc_motor.right()
30          master.rgb(255, 0, 255)
31      else:
32          dc_motor.stop()
33          master.off()
34          led.face("sleepy")
35  sleep(0.2)
```

Практика

Танцующий робот (1/3)

pose_dance

```
from cubroid import dc_motor, led, sound, master, sleep
from cubroid.ai import camera, pose

camera.start()
pose.start()
sleep(2)

print("Изменяйте позу и управляйте танцем робота!")

prev_action = ""
while True:
    lw = pose.left_wrist()
    rw = pose.right_wrist()
    nose = pose.nose()

    if lw and rw and nose:
        l_up = lw["y"] < nose["y"]
        r_up = rw["y"] < nose["y"]
```

Практика

Танцующий робот (2/3)

pose_dance

```
19
20      if l_up and r_up and prev_action != "both":
21          dc_motor.speed(200, 50)
22          led.face("love")
23          master.rainbow(0.2)
24          sound.play("high_do")
25          prev_action = "both"
26      elif l_up and not r_up and prev_action != "left":
27          dc_motor.speed(50, 200)
28          led.face("happy")
29          master.rgb(0, 0, 255)
30          sound.play("sol")
31          prev_action = "left"
32      elif r_up and not l_up and prev_action != "right":
33          dc_motor.speed(200, 50)
34          led.face("happy")
35          master.rgb(255, 0, 255)
36          sound.play("mi")
```

Практика

Танцующий робот (3/3)

pose_dance

```
        prev_action = "right"
    elif not l_up and not r_up and prev_action != "down":
        dc_motor.stop()
        led.face("sleepy")
        master.off()
        prev_action = "down"
sleep(0.2)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Введение в технологию захвата движения и варианты ее использования (5 минут)
2	Поиск	Положение рук → управление мотором; наблюдение координат (10 минут).
3	Практика	Танцующий робот: Робот реагирует на мой танец! (20 мин)
4	Показ	Групповая интерактивная демонстрация танца (10 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Карта 4+ действий и
Заполните таблицу соответствия действий и ответов!

➡ Следующее занятие

Далее: Обнаружение объектов и классификация звука

Фрагменты, использованные в этом занятии

Руки и мотор (pose_arm) | Танцующий робот (pose_dance)

18

Занятие

Обнаружение ИИ

Обнаружение объектов и классификация звука

Сложность: ★★★

Цели обучения

1. Понимать обнаружение объектов на основе набора данных COCO.
2. Писать программы, используя результаты обнаружения объектов.
3. Обучайте и подключайте модели классификации звука с помощью TM Audio.

Введение

Подумай об этом

Беспилотные автомобили обнаруживают людей, автомобили, светофоры

Учебная модель обнаруживает объекты классов, на которых была обучена. Она может ошибаться.

Он даже может классифицировать по звуку!

→ Давайте испытаем как визуальный, так и слуховой ИИ!

Концепции

Ключевые понятия

1

Обнаружение объектов (COCO)

2

Оценка уверенности

3

Классификация аудио

Практика

Обнаружение объектов → Мотор (1/2)

detect_basic

```
from cubroid import dc_motor, master, sleep
from cubroid.ai import camera, detect

camera.start()
detect.start()
sleep(2)

print("Обнаружение предметов!")

while True:
    obj = detect.best()
    if obj:
        name = obj["class"]
        score = obj["score"]
        print(f"Обнаружено: {name} ({score}%)")

        if name == "person":
            dc_motor.go()
```

Практика

Обнаружение объектов → Мотор (2/2)

detect_basic

```
        master.green()
    else:
        dc_motor.stop()
        master.yellow()
else:
    dc_motor.stop()
    master.off()
sleep(0.5)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика

Подсчет предметов (1/2)

detect_count

```
from cubroid import sound, master, led, sleep
from cubroid.ai import camera, detect

camera.start()
detect.start()
sleep(2)

print("Посчитаем видимые предметы!")

while True:
    items = detect.all()
    n = len(items)
    print(f"Предметов обнаружено: {n}")

    if n > 0 and n <= 9:
        led.number(n)
        master.green()
        if n >= 3:
```

Практика

Подсчет предметов (2/2)

detect_count

```
        sound.play("high_do")
else:
    led.face("sleepy")
    master.off()

for item in items:
    name = item["class"]
    score = item["score"]
    print(f"  - {name} ({score}%)")
sleep(1)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Практика →

Модель TM Audio (1/2)

audio_basic

```python
from cubroid import dc_motor, master, led, sleep
from cubroid.ai import audio

# Загрузить звуковую модель Teachable Machine
#   Введите URL-адрес на правой панели и загрузите для автоматического подключения.
audio.load("PASTE_TM_Audio_URL_HERE")
sleep(2)

print("Подайте звуковой сигнал!")

while True:
    result = audio.predict()
    name = result["label"]
    score = result["score"]
    print(f"Звук: {name} ({score}%)")

    if name == "Class 1":
        dc_motor.go()
```

Практика → **Модель TM Audio (2/2)**

audio_basic

```
    master.green()
    led.face("happy")
elif name == "Class 2":
    dc_motor.back()
    master.red()
    led.face("surprise")
else:
    dc_motor.stop()
    master.off()
    led.face("sleepy")
sleep(0.5)
```

Нажмите на этот код на панели фрагментов, чтобы автоматически вставить его в редактор!

Деятельность

Ход урока

1	Введение	Как беспилотные автомобили распознают объекты? (5 мин)
2	Поиск	Запустить обнаружение объекта: проверить распознавание человека/объекта (10 минут)
3	Практика	Обучение подсчету предметов + обучение классификации звуков TM Audio (20 минут)
4	Подведение итогов	Сравнение различных входных данных AI (визуальных/слуховых) (10 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Программа, реагирующая на определенные объекты +
Обучите модель классификации на собственных примерах.

➡ Следующее занятие

Далее: итоговый проект с ИИ.

Фрагменты, использованные в этом занятии

Предмет и мотор (detect_basic) | Число предметов (detect_count) | Звук и действие (audio_basic)

19

Занятие

Итоговый проект с ИИ

Итоговый проект с ИИ

Сложность: ★★★

Цели обучения

1. Создавайте творческие проекты, свободно комбинируя изученные функции искусственного интеллекта.
2. Подготовьте план проекта.
3. Распределите роли и объедините результаты работы участников.

Введение

Подумай об этом

Объедините все изученные на данный момент функции ИИ, чтобы Давайте создадим своего уникального робота с искусственным интеллектом!

Определение задачи → Проектирование → Реализация → Тестирование.

Как настоящий разработчик!

Концепции

Ключевые понятия

1

Проектное обучение

2

Процесс решения проблем

3

Сотрудничество в команде и роли

Деятельность

Ход урока

1	План	Выбор темы группового проекта с ИИ и подготовка плана (15 минут).
2	Реализация	Кодирование + сборка оборудования + тестирование (25 мин)
3	Середина проверки	Делитесь прогрессом и устраняйте неполадки (5 минут)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Подготовьте план: задача, схема решения и этапы реализации.
напиши это и создай прототип!

➡ Следующее занятие

Далее: Итоговая презентация и размышление

20

Занятие

Итоговый проект с ИИ

Представление проекта и обсуждение результатов

Сложность: ★★★

Цели обучения

1. Эффективно презентуйте результаты проекта
2. Предложите участникам оценить работу друг друга и объяснить свои предложения.
3. Обсудите ход работы, освоенные приёмы и вопросы, которые ещё требуют изучения.

Введение

Подумай об этом

Финальная презентация уже здесь!

Ваш проект робота с искусственным интеллектом

Пришло время представить миру.

Представить проект → обсудить работы → сделать выводы

Концепции

Ключевые понятия

1

Навыки презентации

2

Экспертная оценка

3

Обзор грамотности в области искусственного интеллекта

Деятельность

Ход урока

1	Заключение	Итоговая проверка проекта и подготовка презентации (10 мин)
2	Показ	Презентации групповых проектов ИИ (20 мин)
3	Оценка	Взаимная оценка + журнал саморефлексии (10 мин)
4	Подведение итогов	Обсуждение: Компетенции, необходимые в эпоху ИИ (5 мин)

Подведение итогов

Сегодняшнее задание и следующее занятие

✏ Сегодняшнее задание.

Представление проектов, взаимооценка и обсуждение результатов
напиши и отправь свое портфолио!

➡ Следующее занятие

Учебная программа CuroAI завершена! Поздравляем!

Учебная программа CuroAI завершена!

Отличная работа за 20 занятий О

Робот 10 занятий + ИИ 10 занятий | 38 примеров кода | Задания для самостоятельной работы