Скрипты были написаны на Python для Linux. Для реализации распознавания голоса была использована библиотека SpeechRecognition. На ее основе переводится запись с микрофона в текст и далее скрипт распознает знакомые слова и в зависимости от содержания запроса пользователя отвечает на команду подготовленным ответом, который воспроизводится с помощью espeak и SpeechSynthesiser.

В роботе Валли установлена веб-камера со встроенным микрофоном, которая и используется для системы распознавания голоса.

**Main.py**

from speech\_recog.speech\_recog import SpeechRecognizer

from speech\_synthesis.speech\_synthesis import SpeechSynthesiser

from commands import commandControl

sr = SpeechRecognizer()

sr.calibrate()

ss = SpeechSynthesiser()

cmdCtrl = commandControl(ss, sr)

# Работает приоритет. Команды выше сработают первыми.

opts = {"cmds": {

"greetings": ("привет", "здорово"),

"menu": ("меню"),

'bill': ("счет", "счёт"),

'order': ("заказ", "заказать"),

'call': ("официант", "валли"),

"none": (""), # На случай если робот распознает пустую строку.

}}

# Проверка наличия вербальных команд в голосовом запросе пользователя.

# Если слово найдено - возвращает имя команды, иначе возвращает имя команды "not\_understand"

def recognizeCommand(text):

print(text)

# cmd - слово слева, verbose\_cmds - список слов справа в opts

for cmd, verbose\_cmds in opts['cmds'].items():

# Это позволяет писать в opts одну вербальную команду

if (isinstance(verbose\_cmds, tuple) == False):

verbose\_cmds = list([verbose\_cmds])

for verbose\_cmd in verbose\_cmds:

if(verbose\_cmd in text):

return cmd

return "none"

def do\_command(cmd):

cmdCtrl.do(cmd)

if \_\_name\_\_ == "\_\_main\_\_":

while True:

do\_command(recognizeCommand(sr.recognize().lower()))

**Commands.py**

from settings import menu

class commandControl:

def \_\_init\_\_(self, ss, sr):

self.ss = ss # Speech Synthiser

self.sr = sr # Speech Recognizer

self.cmds = {

'greetings': self.sayHi,

'menu': self.sayMenu,

'bill': self.sayBill,

'call': self.sayComing,

'order': self.getOrder,

'not\_understand': self.notUnderstand,

'none': self.none

}

def do(self, cmd):

ret = ""

if (cmd in self.cmds):

print(self.cmds[cmd])

ret = self.cmds[cmd]()

if ret:

return ret

def sayHi(self):

self.ss.say("Привет, я робот Валли!")

def sayMenu(self):

text = "Мы подаем "

for i in range(len(menu)):

text += menu[i] + ", "

text += "."

self.ss.say(text)

def sayBill(self):

text="Ваш счет "

text += "310 рублей."

self.ss.say(text)

def getOrder(self):

listen = True

text = ""

while listen:

text = self.sr.recognize().lower()

if ("все" in text):

self.ss.say("На этом все?")

if ("да" in self.sr.recognize().lower):

listen = False

def sayComing(self):

self.ss.say("Я еду!")

def notUnderstand(self):

self.ss.say("Я не понимаю о чем вы говорите.")

def none(self):

self.ss.say("Я не могу понять, что вы сказали.")

**speech\_synthesis.py**

import pyttsx3

from settings import voicepack, rate, volume

class SpeechSynthesiser:

def \_\_init\_\_(self):

self.tts = pyttsx3.init()

self.tts.setProperty('voice', voicepack)

self.tts.setProperty('rate', rate)

self.tts.setProperty('volume', volume)

def say(self, text):

self.tts.say(text)

self.tts.runAndWait()

**speech\_recog.py**

import speech\_recognition as sr

from settings import dev\_index, recognition\_language

class SpeechRecognizer:

def \_\_init\_\_(self):

self.r = sr.Recognizer()

self. mic = sr.Microphone(device\_index = dev\_index)

def calibrate(self):

print("Идет калибровка...")

with self.mic as source:

self.r.adjust\_for\_ambient\_noise(source)

print("Калибровка окончена")

def recognize(self):

res = ""

with self.mic as source:

print("Say something!")

audio = self.r.listen(source)

# recognize speech using Google Speech Recognition

try:

res = self.r.recognize\_google(audio, language = recognition\_language)

except sr.UnknownValueError:

err = "Google Speech Recognition could not understand audio"

print(err)

except sr.RequestError as e:

err = "Could not request results from Google Speech Recognition service; {0}".format(e)

print(err)

return res