SASTA OLIPHANT SCIENCE AWARDS

SOUTH AUSTRALIAN SCIENCE TEACHERS ASSOCIATION

Highly Commended

Programming, Apps & Robotics

Year 9-10

Sukhman Singh
Huylong Vu

Mount Carmel College

ROWE SCIENTIFIC PTY. LTD. www.rowe.com.au

Australian Government
Department of Defence

Government of South Australia
Department for Education

SASTA South Australian Science Teachers Association

Quality Workwear
Dickies
Dickies FORT WORTH
Dickies SHIRTS & PANTS
Trust Label
100 years of HARD WORK
QUALITY WORK
Dickies

<u>VOICE CONTROLLED PROSTHETIC ARM-GUANTLET:</u>

The aim of this entry is to create an affordable prosthetic hand configurable from an external website controlled using voice commands as an alternative to other prosthetics on the market. The scientific purpose of this, is to make a functioning, affordable prosthetic, cheaper than anything non-cosmetic on the market to raise awareness about the power of creating things yourself and to promote scientific education while making a prosthetic for those who cannot afford it.

This, of course, comes with its own system and methods of usage, this project runs python code a raspberry pi 5 (16gb ram) to run pretrained voice recognition models from VOSK that detect gesture commands like “[keyword], grab”. This project also includes access to a website made with React JS that lets you modify certain configurations of the arm, including the keyword/activation word, the position the hand is in (for loud or crowded situations), and the specific AI model the hand is running to let the user pick out and optimize the speed and accuracy of the hand to their liking.

It is important to note that since the hand is at a price point at which, commercially, the majority of prosthetics available are cosmetic, it comes with certain caveats and situations in which the voice command usage is sub-optimal and for that, the web app controls exist for use of the hand. it is not a full replacement of both hands, but if you need a budget-friendly option for one hand, there is are very little better options available for you to use that available as an option if this was on the market.

And to give credit where credit is due, full disclosure, the 3D model of the hand (only the 3D print and elastic cord servo system itself, none of the actual functionality providers like the hand's code, electronics, website, or software design) was downloaded from a tutorial that was partly followed to set up the system of the servo pulling the cord.

Source code:

Arm (main.py):

```
# export GOOGLE_APPLICATION_CREDENTIALS="project-1-455407-
6da299464c24.json"
# curl -X POST -H "Content-Type: application/json" -d '{"passkey" :
"0897465765856865678468764 54HFNGehjjfhgeKJ", "keyword": "jarvis", "RGB
config": [], "model selection" : "small"}' https://magical-hare-
yearly.ngrok-free.app
# curl -X POST -H "Content-Type: application/json" -d '{"passkey" :
"0897465765856865678468764 54HFNGehjjfhgeKJ", "keyword": "jarvis", "RGB
config": [], "model selection" : "small"}' http://127.0.0.1:8080
# ngrok http --url=magical-hare-yearly.ngrok-free.app 8080
# ngrok config add-authtoken
2v6azwRnpMG9V4MUr0JcESEQ6xN_qEX1v4YdTspz68WwMZw1

from flask import Flask, request
from vosk import Model, KaldiRecognizer
import pyaudio
import threading
import json
import time

app = Flask(__name__)

new_config_created = None
new_config = {}

Models = {
    "small" : "vosk-model-small-en-us-0.15",
    "small-ultra" : "vosk-model-en-us-0.22-lgraph",
    "medium" : "vosk-model-en-us-0.22",
    "large" : "vosk-model-en-us-0.42-gigaspeech"
}

app_config = {
    "keyword" : "jarvis",
    "position" : "",
    "model selection" : "medium"
}

commands = {
    "grab" : [0, 0, 0, 0, 0],
    "release" : [0, 0, 0, 0, 0],
    "straighten" : [0, 0, 0, 0, 0],
    "fist" : [0, 0, 0, 0, 0],
    "point" : [0, 0, 0, 0, 0]
}

def on_config_change():
    global new_config_created, app_config
    new_config_created = True
    print(f"\nNEW CONFIG RECIEVED: {new_config} \n")
    time.sleep(0.5)
    new_config_created = False
    app_config = new_config
    return


def transcribe_streaming():

    model = Model(f"Models/{Models[app_config['model selection']]}")
    rec = KaldiRecognizer(model, 16000)

    p = pyaudio.PyAudio()
    stream = p.open(format=pyaudio.paInt16,
                    channels=1,
                    rate=16000,
                    input=True,
                    frames_per_buffer=8000)
    stream.start_stream()

    print("Listening...")

    try:
        while True:
            if not new_config_created:
                data = stream.read(4000, exception_on_overflow=False)
                if rec.AcceptWaveform(data):
                    result = rec.Result()
                    data = json.loads(result)
                    text = data.get("text", "").split(" ")
                    print("You said:", text)

                    for cmd in commands.keys():
                        if cmd in text and text[text.index(cmd) - 1] ==
app_config["keyword"]:
                            print(f"Recognized command: {cmd.upper()}")
                            for move, i in zip(commands[cmd.lower()],
range(len(commands[cmd.lower()]))):
                                print(f"Moving servo {i} by {move}.")
                            break
            if new_config["model selection"] != app_config["model
selection"]:
                model = Model(f"Models/{Models[app_config['model
selection']]}")
                rec = KaldiRecognizer(model, 16000)

            if new_config["position"] != app_config["position"]:

                cmd = new_config["position"]

                print(f"position: {cmd}")

                for move, i in zip(commands[cmd.lower()],
range(len(commands[cmd.lower()]))):
                    print(f"Moving servo {i} by {move}.")

                app_config["position"] = new_config["position"]

            else:
                print(f"{app_config['position']} and
{new_config['position']}")

    except KeyboardInterrupt:
        print("\nStopping...")
        stream.stop_stream()
        stream.close()
        p.terminate()


@app.route("/", methods=["GET", "POST"])
def flask_app():

    global new_config
    try:

        new_data = request.get_data(as_text=True)
        new_data = json.loads(new_data)

    except Exception as e:
        print(e)
        return str(e)

    if new_data["passkey"] == "089746576585686567846876454HFNGehjjfhgeKJ":
        new_data["keyword"] = new_data["keyword"].lower()

        new_config = new_data
        on_config_change()

        return f"Config modified: {new_config}"

    else:
        return "INVALID PASSKEY."

def main():
    try:
        app.run("0.0.0.0", 8080)
    except Exception as e:
        print(f"Error: {e}")

main_thread = threading.Thread(target=main)
transcription_thread = threading.Thread(target=transcribe_streaming)
```

```
main_thread.start()
transcription_thread.start()
```

Website (App.js):

```
import logo from './logo.png';
import './App.css';
import Button from './Components/Button';
import Input from "./Components/Input"
import { useState } from 'react';
import ModelSelect from './Components/ModelSelect'
import PositionSelect from './Components/PositionSelect';

function App() {

  const [keyword, changeKeyword] = useState("jarvis");
  const [modelSelection, changeModel] = useState("small");
  const [positionSelection, changePosition] = useState("grab");

  function onUserInput(input) {
    changeKeyword(input);
  }

  function onModelSelect(selection) {
    changeModel(selection);
  }

  function onPositionSelect(selection) {
    changePosition(selection);
  }

  function onSubmitClick(keywordParam) {

    if (keywordParam.split(" ").length === 1 && keywordParam !== '') {
        fetch("https://magical-hare-yearly.ngrok-free.app",
            {method: "POST", headers: {"Content-Type": "application/json"},
               body: JSON.stringify({"passkey" :
               "089746576585686567846876454HFNGehjjfhgeKJ",
               "keyword" : keywordParam,
               "position" : positionSelection,
               "model selection" : modelSelection.toString()}),
               "mode" : "no-cors"});

        console.log(`Config successfully modified`);
    }

    else {
        alert("keyword has to be one word.");
    }
  }

  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <br></br>
        <br></br>
        <ModelSelect onModelSelect={onModelSelect} labelText="AI model:
"></ModelSelect>
        <br></br>
        <PositionSelect onPositionSelect={onPositionSelect} labelText="Hand
position: "></PositionSelect>
        <br></br>
        <Input onInput={onUserInput} labelText="Keyword: "></Input>
        <br></br>
        <Button text="submit" onClick={() =>
onSubmitClick(keyword)}></Button>
      </header>
    </div>
  );
}

export default App;
```

Bibliography:

Learnt React JS: https://react.dev

3D model: https://www.viralsciencecreativity.com/post/arduino-flex-sensor-controlled-robot-hand

Tutorial (partly followed, only for the elastic cord-servo system): https://www.youtube.com/watch?v=Fvg-v8FPcjg&t=197s

trained AI models were VOSK to run offline speech recognition.

https://alphacephei.com/vosk/