UNIVERSIDAD INTERNACIONAL DEL ECUADOR

FACULTAD DE CIENCIAS TÉCNICAS

ESCUELA DE INGENIERÍA MECATRÓNICA

DISEÑO Y CONSTRUCCIÓN DE UN ENTRENADOR INTELIGENTE DE LA PRIMERA FORMA DE TAEKWON-DO (TAEGUK IL JANG)

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

ANEXOS

CHELSEA ALIET ACHIG ARMIJOS

DIRECTOR: ING. GABRIELA ANDALUZ, Msc.

D. M. Quito,

2022

ÍNDICE DE ANEXOS

CÓDIGO FUENTE

Función Principal

```
from flask import Flask
from flask import Response
from flask import request
from flask import jsonify
from flask_cors import CORS, cross_origin
from pymongo import MongoClient
import certifi
import pyrebase
import os
import cv2

from blazepose import evaluate, iniciar, videostart, terminar
from dbcontroller import getposes

app = Flask(__name__)
CORS(app)
app.config['CORS_HEADERS']='Content-Type'
app.config['TESTING'] = True

ca=certifi.where()

Flask.debug=False

global cap

#region MONGO
mongo_uri= 'mongodb+srv://cheliali:livelovelaugh@cluster0.raatz.mongodb.net/test'
db = MongoClient(host=mongo_uri, tlsCAFile=ca).get_database()
#endregion Mongo

#region Firebase
config = {
   "apiKey": "AIzaSyBAM7ePYf0GIulRbdZsLio1SdS6t6UGI4A",
   "authDomain": "first-skein-346416.firebaseapp.com",
   "databaseURL":"https://first-skein-346416.firebaseio.com",
   "projectId": "first-skein-346416",
   "storageBucket": "first-skein-346416.appspot.com",
   "messagingSenderId": "414119301715",
   "appId": "1:414119301715:web:370f9496c2237e0f226ca6",
   "measurementId": "G-K9QX1DERP4"
}
firebase = pyrebase.initialize_app(config)
```

```
storage = firebase.storage()
#endregion


@app.route("/login", methods=['POST'])
@cross_origin()
def login():
    users = db.users
    request_data = request.get_json()
    username = request_data['username']
    password = request_data['password']

    user=users.find_one({'username': username})

    if(user["password"]==password):

        return jsonify("ok")
    else:
        return jsonify("error")

@app.route("/register", methods=['POST'])
@cross_origin()
def register():
    users = db.users
    request_data = request.get_json()

    username = request_data['username']
    password = request_data['password']

    users.insert_one({'username': username, 'password': password})
    return jsonify("ok")

@app.route("/getpoomsaeposes", methods=['GET'])
@cross_origin()
def poomsae1():
    poses=getposes()
    return poses

@app.route("/iniciar", methods=['GET'])
@cross_origin()
def iniciarf():
    global userid
    global posename
    arguments=request.args
    userid=arguments.get('userid')
    posename=arguments.get('pose')
    iniciar(userid,posename)
    return jsonify("ok")

@app.route("/video_feed")
def video_feed():
    global cap
    if os.environ.get('WERKZEUG_RUN_MAIN') or Flask.debug is False:
        cap = cv2.VideoCapture(0)
    return Response(videostart(cap),
    mimetype = "multipart/x-mixed-replace; boundary=frame")

@app.route("/terminar", methods=['GET'])
@cross_origin()
def terminarf():
    terminar()
    return jsonify("ok")

@app.route("/evaluate", methods=['GET'])
@cross_origin()
def evaluation():
    finalGrade = evaluate()
    return jsonify(finalGrade)

if __name__ == "__main__":
    app.run(debug=False)
```

Función Iniciar

```
import cv2
import os
from flask import Flask
from evaluatePose import evaluateVideo

name="out.mp4"

def iniciar(id,pose):
    global userid, posename
    userid = id
    posename = pose
    return
```

Función Transmisión de Video

```
def videostart(cap):
    global show
    show=True
    cap.set(cv2.CAP_PROP_FPS,30)
    outmp4 = cv2.VideoWriter(name,cv2.VideoWriter_fourcc(*'mp4v'), 30, (640,480))
    while (show):
        ret, frame = cap.read()
        if ret:
            outmp4.write(frame)
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            (flag, encodedImage) = cv2.imencode(".jpg", frame)
            if not flag:
                continue
            yield(b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' +
                bytearray(encodedImage) + b'\r\n')
        else:
            break
    cap.release()
    outmp4.release()
    cv2.destroyAllWindows()
```

Función Terminar

```
def terminar():
    global show
    show=False
    return
```

Función Evaluar

```
from time import time
import cv2
import mediapipe as mp
import numpy as np
import json
import math

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose

with open('body_poses_model.json', 'rb') as f:
    model = json.load(f)
```

```
def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-b[0])
    angle = np.abs(radians*180.0/np.pi)

    if angle >180.0:
        angle = 360-angle

    return angle

def calculate_distance(p1, p2):
    dist=math.sqrt((p2[0]-p1[0])**2+(p2[1]-p1[1])**2)
    return dist

def evaluateVideo(posename):

    cap=cv2.VideoCapture("out.mp4")
    length = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    print(length)
    cont = 0
    calificacionRef = 0
    videoEnd=False
    calificacionfin = []
    while(cap.isOpened() and not videoEnd):
        ret, frame = cap.read()
        if ret:
            if(cont % 15 == 0):
                calificacionfin=evaluatePose(frame, posename, calificacionRef)
                print(calificacionfin)
        else:
            videoEnd=True
        cont = cont + 1
    return calificacionfin

def evaluatePose(frame, posename, calificacionref):
    try:
        resp = getCurrentAngles(frame)
        currentAngles = resp[0]
        currentDistances = resp[1]
        calificacionesIndividuales=[]
        sumatoria=0
        for angleName in model["poses"][posename].keys():
            if(str(angleName).find("distance") != -1):
                finalval=currentDistances[angleName]*1000
            else:
                powerval=int(model["poses"][posename][angleName] - currentAngles[angleName])**2
                finalval=math.sqrt(powerval)

            for calificacion in model["calificacion"].keys():
                if (int(finalval) in
range(model["calificacion"][calificacion][0],model["calificacion"][calificacion][1])):
                    calificacionesIndividuales.append({"name":angleName, "calificacion":calificacion})
                    sumatoria=sumatoria+int(calificacion)

        promedio=sumatoria/len(calificacionesIndividuales)

        if (int(promedio)>calificacionref):
            cv2.imwrite("frame.jpg", frame)
            calificacionref=int(promedio)
        return calificacionesIndividuales

    except:
        return []

def getCurrentAngles(frame):
    with mp_pose.Pose(
        static_image_mode=True) as pose:

        results=pose.process(frame)
        try:
            landmarks=results.pose_landmarks.landmark

            landmarksCoords={}
            for landMark in model["landmarks"]:
                landMarkValue = mp_pose.PoseLandmark[landMark].value

landmarksCoords[landMark.lower()]=[landmarks[landMarkValue].x,landmarks[landMarkValue].y]

            #Obtener angulos y distancias
            angulos = model["angulos"]
            currentAngles = {}
            currentDistances={}
            for anguloKey in angulos:
                if(str(anguloKey).find("distance") != -1):
```

```
currentDistances[anguloKey]=calculate_distance(landmarksCoords[angulos[anguloKey][0]],land
marksCoords[angulos[anguloKey][1]])
                else:
                    currentAngles[anguloKey] =
calculate_angle(landmarksCoords[angulos[anguloKey][0]],landmarksCoords[angulos[anguloKey
][1]],landmarksCoords[angulos[anguloKey][2]])
            return [currentAngles, currentDistances]

        except:
            pass
```

Controlador de Base de Datos

```
from pymongo import MongoClient
from bson.json_util import dumps
import certifi

ca=certifi.where()
mongo_uri= 'mongodb+srv://cheliali:livelovelaugh@cluster0.raatz.mongodb.net/test'
db = MongoClient(host=mongo_uri, tlsCAFile=ca).get_database()

def getposes():

    cursor=db.poomsaes.find()
    poseslist=list(cursor)

    for poomsae in poseslist:
        poomsae['_id'] = str(poomsae['_id'])

    json_poses=dumps(poseslist)

    return json_poses
```

Modelo de Calificación

```
{
    "landmarks": [
      "LEFT_SHOULDER",
      "LEFT_ELBOW",
      "LEFT_WRIST",
      "LEFT_HIP",
      "LEFT_KNEE",
      "LEFT_ANKLE",
      "RIGHT_SHOULDER",
      "RIGHT_ELBOW",
      "RIGHT_WRIST",
      "RIGHT_HIP",
      "RIGHT_KNEE",
      "RIGHT_ANKLE"
    ],
    "angulos": {
      "leftcodo": ["left_shoulder", "left_elbow", "left_wrist"],
      "leftax": ["left_hip", "left_shoulder", "left_elbow"],
      "lefthip":["left_shoulder","left_hip", "left_knee"],
      "leftknee":["left_hip", "left_knee","left_ankle"],
      "rightcodo": ["right_shoulder", "right_elbow", "right_wrist"],
      "rightax": ["right_hip", "right_shoulder", "right_elbow"],
      "righthip":["right_shoulder","right_hip", "right_knee"],
      "rightknee":["right_hip", "right_knee","right_ankle"],
      "distanceknees":["right_knee","left_knee"],
      "distancefeet":["right_ankle","left_ankle"],
      "distancehipknee":["right_hip","right_knee"]
    },
    "poses": {
      "Arae Izq": {
        "leftcodo": 175,
        "leftax": 20
      },
      "Momtong Izq": {
        "leftcodo": 60,
        "leftax": 10
      },
      "Olgul Izq": {
        "leftcodo": 70,
        "leftax": 150
      },
      "Puno Izq": {
        "leftcodo": 2,
        "leftax": 80
      },
      "Ap Kubi Izq": {
        "distancefeet": 2.3,
        "distanceknees": 1,
        "distancehipknee":1
      },
      "Ap Seogi Izq": {
        "distancefeet": 0.7,
        "distanceknees": 0.5,
        "distancehipknee":1
      },
      "Ap Chagi Izq": {
        "lefthip": 40,
        "leftknee": 170
      },
      "Olgul Der": {
        "rightcodo": 70,
        "rightax": 150
      },
      "Arae Der": {
        "rightcodo": 175,
        "rightax": 20
      },
      "Momtong Der": {
        "rightcodo": 60,
        "rightax": 10
      },
      "Puno Der": {
        "rightcodo": 2,
        "rightax": 80
      },
      "Ap Kubi Der": {
        "distancefeet": 2.3,
        "distanceknees": 1,
        "distancehipknee":1
      },
      "Ap Seogi Der": {
        "distancefeet": 0.7,
        "distanceknees": 0.5,
        "distancehipknee":1
      },
      "Ap Chagi Der": {
        "righthip": 40,
        "rightknee": 170
      }
    },
    "calificacionAngulos":{
      "100":[0,1],
      "95":[1,3],
      "90":[3,5],
      "85":[5,7],
      "80":[7,10],
      "75":[10,13],
      "70":[13,16],
      "60":[16,20],
      "50":[20,30],
      "0":[30,360]
    },
    "calificacionDistancias":{
      "100":[0,0.07],
      "95":[0.07,0.2],
      "90":[0.2,0.35],
      "85":[0.35,0.5],
      "80":[0.5,0.6],
      "75":[0.6,0.7],
      "70":[0.7,0.8],
      "60":[0.8,0.9],
      "0":[0.9,3]
    }
  }
```

Página de Registro e Inicio de Sesión

```
import { Component, OnInit } from '@angular/core';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { DBService } from 'src/app/servicios/database.service';
@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.scss'],
})
export class LoginComponent implements OnInit {
  registroForm!: FormGroup;

  constructor(private fb: FormBuilder, public databaseService: DBService, private router: Router)
{}

  ngOnInit() {
    this.registroForm = this.fb.group({
      username: ['', Validators.required],
      password: ['', Validators.required],
    });
  }

  loginYcrearUsuario(accion: string) {
    if (this.registroForm.invalid) {
      return;
    }
    const { username, password } = this.registroForm.value;
    if (accion == 'register') {
      this.databaseService.createUser({ username, password }).subscribe(() => {
        this.databaseService.username = username;
        localStorage.setItem('username', username);
        this.router.navigateByUrl(`inicio`);
      });
    } else {
      this.databaseService.loginUser({ username, password }).subscribe(() => {
        this.databaseService.username = username;
        localStorage.setItem('username', username);
        this.router.navigateByUrl(`inicio`);
      });
    }
  }
}
```

Página de Inicio

```
import { Component, OnInit } from '@angular/core';
import { trigger, state, style, transition, animate } from '@angular/animations';

import { DBService } from '../../servicios/database.service';
import { HistoryItem, DBHistoryItem, BodyPart } from '../../types/types';
import * as moment from 'moment';

@Component({
  selector: 'app-inicio',
  templateUrl: './inicio.component.html',
  styleUrls: ['./inicio.component.scss'],
  animations: [
    trigger('rowExpansionTrigger', [
      state(
        'void',
        style({
          transform: 'translateX(-10%)',
          opacity: 0,
        })
      ),
      state(
        'active',
        style({
          transform: 'translateX(0)',
          opacity: 1,
        })
      ),
      transition('* <=> *', animate('400ms cubic-bezier(0.86, 0, 0.07, 1)')),
    ]),
  ],
})
export class InicioComponent implements OnInit {
  history!: HistoryItem[];

  constructor(private dbService: DBService) {}

  ngOnInit(): void {
    this.dbService.getHistory().subscribe((res) => {
      this.history = Object.values(res).map((pose) => {
        return {
          ...pose,
          practices: pose.practices.reduce<DBHistoryItem[]>((prev, curr) => {
            curr = {
              ...curr,
              observations: curr.observations
                .filter((o) => o.name !== 'distancehipknee' && o.grade !== '100')
                .map((o) => ({
                  ...o,
                  name: BodyPart[o.name as keyof typeof BodyPart],
                })),
            };

            if (prev.length > 0 && moment(curr.date).isAfter(prev[0].date)) {
              return [curr, ...prev];
            }
            return [...prev, curr];
          }, []),
        };
      });
    });
  }
}
```

Página de Selección de Posición

```
import { Component, OnInit } from '@angular/core';
import { DBService } from '../../servicios/database.service';
import { DBPoomsaes, ModHistory } from '../../types/types';
import { Router } from '@angular/router';

@Component({
  selector: 'app-pose-selection',
  templateUrl: './pose-selection.component.html',
  styleUrls: ['./pose-selection.component.scss'],
})
export class PoseSelectionComponent implements OnInit {
  history!: ModHistory;

  constructor(public dbService: DBService, private router: Router) {}

  ngOnInit(): void {
    if (!this.dbService.poomsaes || this.dbService.poomsaes.length == 0) {
      this.dbService.getPoomsaes().subscribe((resp) => (this.dbService.poomsaes = resp));
    }
    this.dbService.getHistory().subscribe((resp) => (this.history = resp));
  }

  goToPractice(pose: string) {
    this.router.navigateByUrl(`pose/${this.dbService.username}/${pose}`);
  }
}
```

Página de Práctica

```
import { ThisReceiver } from '@angular/compiler';
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { debounceTime } from 'rxjs';
import { DBService } from 'src/app/servicios/database.service';
import { VideoServiceService } from 'src/app/servicios/video-service.service';
import { GradesResponse } from 'src/app/types/types';
import { environment } from 'src/environments/environment';
import { BodyPart } from '../../types/types';

const baseurl = environment.baseurl;
@Component({
  selector: 'app-pose-practice',
  templateUrl: './pose-practice.component.html',
  styleUrls: ['./pose-practice.component.scss'],
})
export class PosePracticeComponent implements OnInit {
  currentPose!: string;
  userName!: string;
  showVideo: boolean = false;
  showCountDown: boolean = false;
  url: string = ``;
  displayModal: boolean = false;
  loading: boolean = false;
  grade: string = '';
  initialCounter!: number;
  modelPictureURL: string = '';
  observations!: GradesResponse[];

  constructor(
    private route: ActivatedRoute,
    public videoService: VideoServiceService,
    private router: Router,
    private dbService: DBService
  ) {}
  ngOnInit(): void {
    this.initialConfig();
  }

  initialConfig() {
    this.currentPose = this.route.snapshot.paramMap.get('pose') || '';
    this.userName = this.route.snapshot.paramMap.get('name') || '';

    if (!this.dbService.poomsaes || this.dbService.poomsaes.length == 0) {
      this.dbService.getPoomsaes().subscribe((resp) => {
        this.dbService.poomsaes = resp;
        this.start();
      });
    } else {
      this.start();
    }
  }

  start() {
    this.modelPictureURL =
      this.dbService.poomsaes[0].poses.find((pose) => pose.name == this.currentPose)?.picture ||
'';
    this.startProcess();
  }

  startProcess() {
    this.displayModal = false;
    this.initialCounter = 5;
    this.url = `${baseurl}/video_feed?test=${Date().toString()}`;
    const interval = setInterval(() => {
      this.showVideo = true;
      this.showCountDown = true;
      if (this.initialCounter == 0) {
        this.showCountDown = false;
        clearInterval(interval);
        this.videoService
          .startStream(`?userid=${this.userName}&pose=${this.currentPose}`)
          .subscribe((resp) => {
            if (resp == 'ok') {
              this.startPractice();
            } else {
              window.alert('no se detecta persona');
              this.videoService
                .stopStream()
                .pipe(debounceTime(1500))
                .subscribe(() => {
                  this.videoService.evaluate().subscribe(() => {
                    this.router.navigateByUrl('/poses');
                  });
                });
            }
        });
      } else {
        this.initialCounter--;
      }
    }, 1000);
  }
  startPractice() {
    setTimeout(() => {
      this.showVideo = false;
      this.videoService
        .stopStream()
        .pipe(debounceTime(1500))
        .subscribe((_) => {
          this.displayModal = true;
          this.loading = true;

          this.videoService.evaluate().subscribe((resp) => {
            console.log(resp);
            this.observations = resp
              .filter((o) => o.name !== 'distancehipknee' && o.grade !== '100')
              .map((p) => ({
                ...p,
                name: BodyPart[p.name as keyof typeof BodyPart],
              }));
            this.grade =
              resp.length == 0
                ? 'Intente de nuevo'
                : (
                    resp.reduce((prev, cur) => {
                      return prev + Number(cur.grade);
                    }, 0) / resp.length
                  )
                    .toFixed(2)
                    .toString()
                    .concat('/100');

            this.loading = false;
          });
        });
    }, 7000);
  }

  returnToSelection() {
    this.displayModal = false;
    this.router.navigateByUrl(`poses`);
  }
}
```

DATASHEETS

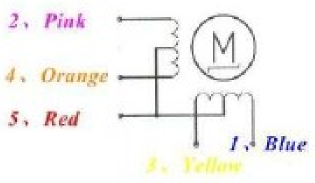
Motor a Pasos 28BYJ-48

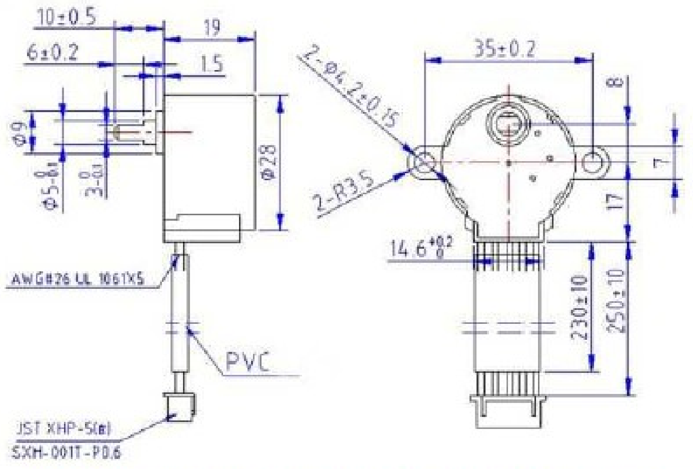
Kiatronics
electronic design and manufacture

28BYJ-48 – 5V Stepper Motor

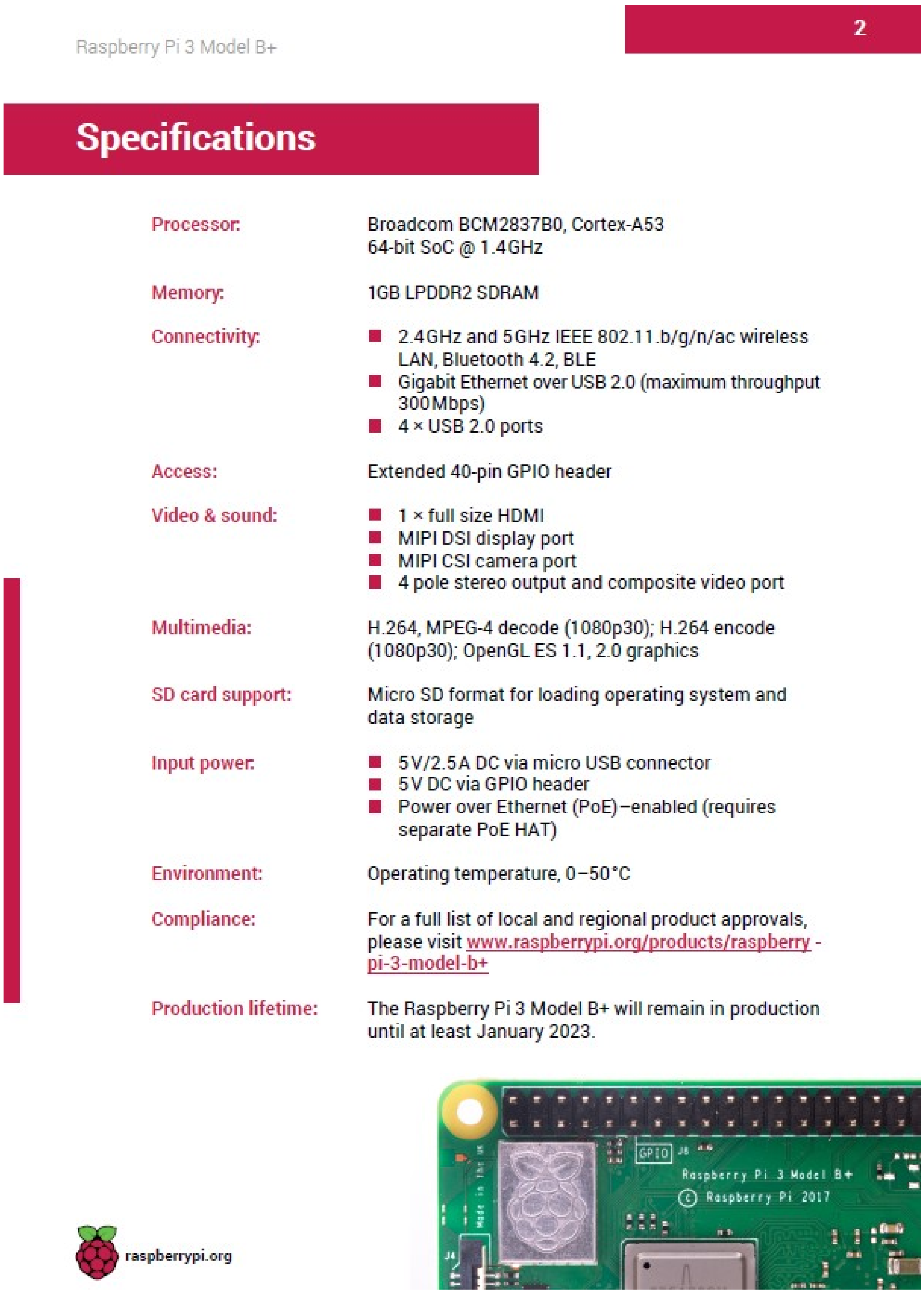
The 28BYJ-48 is a small stepper motor suitable for a large range of applications.

Rated voltage :	5VDC
Number of Phase	4
Speed Variation Ratio	1/64
Stride Angle	5.625°/64
Frequency	100Hz
DC resistance	50Ω±7%(25℃)
Idle In-traction Frequency	> 600Hz
Idle Out-traction Frequency	> 1000Hz
In-traction Torque	>34.3mN.m(120Hz)
Self-positioning Torque	>34.3mN.m
Friction torque	600-1200 gf.cm
Pull in torque	300 gf.cm
Insulated resistance	>10MΩ(500V)
Insulated electricity power	600VAC/1mA/1s
Insulation grade	A
Rise in Temperature	<40K(120Hz)
Noise	<35dB(120Hz,No load,10cm)
Model	28BYJ-48 – 5V





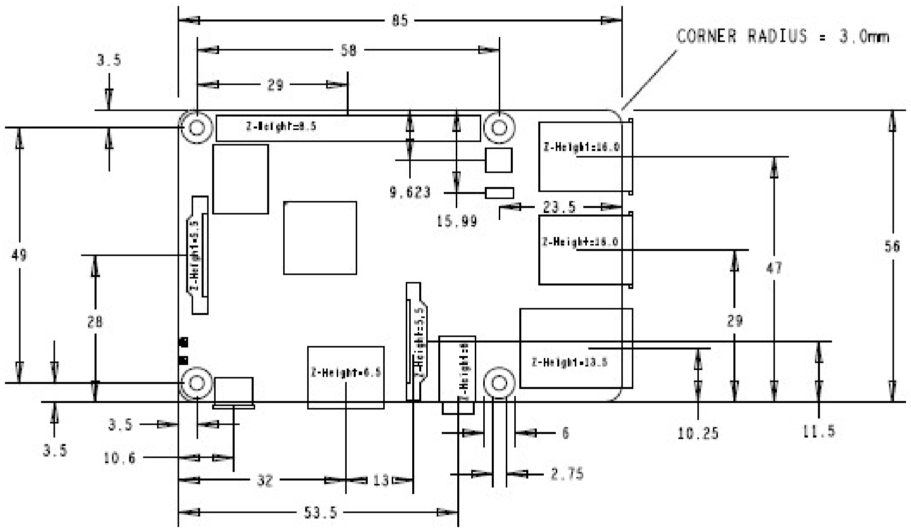
Raspberry Pi 3 modelo B



Specifications

Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	■ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps) ■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	■ 5V/2.5A DC via micro USB connector ■ 5V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50°C
Compliance:	For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry -pi-3-model-b+
Production lifetime:	The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.

Physical specifications



Warnings

- This product should only be connected to an external power supply rated at 5V/2.5A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or only handle it by the edges to minimise the risk of electrostatic discharge damage.

PLANOS MECÁNICOS

PLANOS ELECTRÓNICOS

PLANOS INFORMÁTICOS