nature portfolio

https://doi.org/10.1038/s41596-022-00731-5

Supplementary information

Small-animal blood exchange is an emerging approach for systemic aging research

In the format provided by the authors and unedited

Supplementary File 1: Arduino code for single pump small animal blood exchange system.

```
#include <Stepper.h>
const int stepsPerRevolution = 200;
// change this to fit the number of steps per revolution
// for your motor
Stepper myStepper(stepsPerRevolution, 9, 10, 11, 12);
// initialize the stepper library on pins 9 through 12:
void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(20);
const int half_cycle = 9933;
  //adjust number of steps here (divisible by 66.6)
  //increase number to increase the volume exchanged
  //decrease number to decrease the volume exchanged
const int exchange_number = 10;
  // input number of exchanges
const int slack = 200;
  //error for slack
const int delay_ = 30;
  // delay between steps, higher delay -> slower speed,
divisible by 5
  // if delay is adjusted, the number of steps must be changed
as well
myStepper.step (half_cycle - slack);
delay(delay_);
for (int x = 0; x < (exchange_number - 1); x++) {
  myStepper.step(-(half_cycle));
  delay(delay_);
  myStepper.step(half_cycle);
  delay(delay_);
}
  myStepper.step(-(half_cycle));
  delay(delay_);
```

Supplementary File 2: Arduino code for double pump small animal blood exchange system using a single motor.

```
#include <Adafruit_MotorShield.h>
#include "utility/Adafruit_MS_PWMServoDriver.h"
#include <Stepper.h>
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
Adafruit_StepperMotor *myMotor1 = AFMS.getStepper(200, 1);
//Specify myMotor1 or myMotor2 depending on which pump is to be used
//myMotor2 is activated by "Adafruit_StepperMotor *myMotor2 = AFMS.getStepper(200, 2);"
void setup() {
  Serial.begin(9600);
const int half_cycle = 9933;
  //adjust number of steps here (divisible by 66.6)
  //increase number to increase the volume exchanged
  //decrease number to decrease the volume exchanged
const int exchange_number = 10;
  // input number of exchanges
const int slack = 200;
  //error for slack
const int delay_ = 30;
  // delay between steps, higher delay -> slower speed, divisible by 5
  // if delay is adjusted, the number of steps must be changed as well
  myMotor1->setSpeed(40);
    //Specify myMotor1 or myMotor2 depending on which pump is to be used
  AFMS.begin();
  myMotor1->step(half_cycle - slack, FORWARD, SINGLE);
    //Specify myMotor1 or myMotor2 depending on which pump is to be used
  delay(delay_);
for (int x = 0; x < (exchange_number - 1); x++) {
  myMotor1->step(half_cycle, BACKWARD, SINGLE);
    //Specify myMotor1 or myMotor 2 depending on which pump is to be used
  delay(delay_);
  myMotor1->step(half_cycle, FORWARD, SINGLE);
    //Specify myMotor1 or myMotor2 depending on which pump is to be used
  delay(delay_);
}
  myMotor1->step(half_cycle, BACKWARD, SINGLE);
    //Specify myMotor1 or myMotor2 depending on which pump is to be used
  delay(delay_);
}
void loop() {}
```

Supplementary File 3: Arduino code for double pump small animal blood exchange system using both motors.

```
#include <Adafruit_MotorShield.h>
#include "utility/Adafruit_MS_PWMServoDriver.h"
#include <Stepper.h>
Adafruit_MotorShield AFMS = Adafruit_MotorShield();
Adafruit_StepperMotor *myMotor1 = AFMS.getStepper(200, 1);
Adafruit_StepperMotor *myMotor2 = AFMS.getStepper(200, 2);
void setup() {
  Serial.begin(9600);
const int half_cycle = 9933;
  //adjust number of steps here (divisible by 66.6)
  //increase number to increase the volume exchanged
  //decrease number to decrease the volume exchanged
const int exchange_number = 10;
  // input number of exchanges
const int slack = 200;
  //error for slack
const int delay_ = 30;
  // delay between steps, higher delay -> slower speed, divisible by 5
  // if delay is adjusted, the number of steps must be changed as well
  myMotor1->setSpeed(80);
  myMotor2->setSpeed(80);
  AFMS.begin();
  for (int a = 0; a < (half_cycle - slack - 1); a++) {
  myMotor1->step(1, FORWARD, DOUBLE);
  delay(5);
  myMotor2->step(1, FORWARD, DOUBLE);
  delay(5 + delay_);
}
for (int x = 0; x < (exchange_number - 1); x++) {
for (int b = 0; b < (half_cycle); b++) {
  myMotor1->step(1, BACKWARD, DOUBLE);
  delay(5);
  myMotor2->step(1, BACKWARD, DOUBLE);
  delay(5 + delay_);
}
for (int c = 0; c < (half_cycle); c++) {
  myMotor1->step(1, FORWARD, DOUBLE);
  delay(5);
  myMotor2->step(1, FORWARD, DOUBLE);
  delay(5 + delay_);
}
}
for (int d = 0; d < (half_cycle); d++) {
  myMotor1->step(1, BACKWARD, DOUBLE);
  delay(5);
  myMotor2->step(1, BACKWARD, DOUBLE);
  delay(5 + delay_);
}
}
void loop() {}
```