Léptetőmotorok

A léptetőmotorok lényeges tulajdonsága, hogy egy körülforduláshoz hány lépés szükséges. Ezt megadhatják fokban, ekkor az egy lépésre eső szögelfordulást adják meg. Illetve megadhatják az egy körülforduláshoz szükséges lépésszámot. Természetesen fontos paraméter a motor terhelhetősége, teljesítménye és nyomatéka is. A kivezetések száma azért változhat, mert növelhetik a tekercsek számát.

A léptetőmotoroknak alapvetően két fajtája létezik, az unipoláris és a bipoláris. A bipoláris léptetőmotornak általában 4 kivezetése van, míg az unipolárisnak 5 vagy 6.

A léptetőmotorok rövid szakaszos üzemre vannak méretezve. Amikor nem léptet, akkor egy alacsonyabb feszültséget kell rákapcsolni, ellenkező esetben akár le is éghet. Ez a tartófeszültség azért szükséges, hogy a motor ne tudjon kimozdulni az adott pozícióból. Egyébként a motort amikor magára hagyod (kikapcsolod a tápfeszültséget), akkor megpróbál a legközelebbi teljes lépés pozícióba fordulni. Tehát a féllépés pozíció nem stabil. Erre a program tervezésekor figyelni kell..

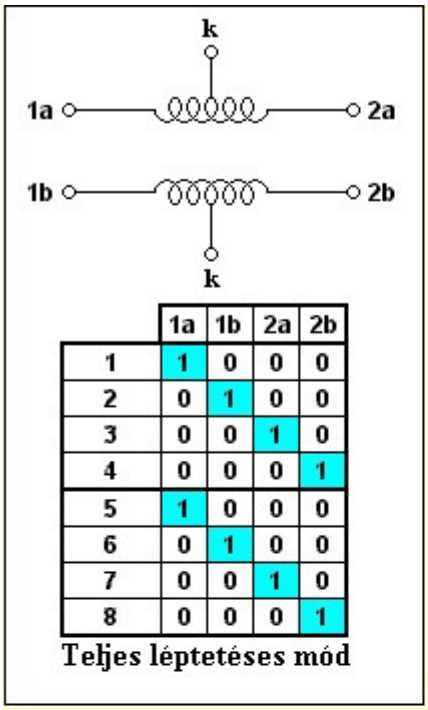
Előnyök:

- Nincs benne szénkefe, így tartósabb lehet (gyakorlatilag a csapágy minősége határozza meg az élettartamot)
- A pontos pozícióba álláshoz nem szükséges bonyolult visszacsatolás (tehát megfelelő vezérlés, és hatátárértéken belüli terhelés esetén biztos, hogy a megfelelő pozícióba fordul)

Hátrányok:

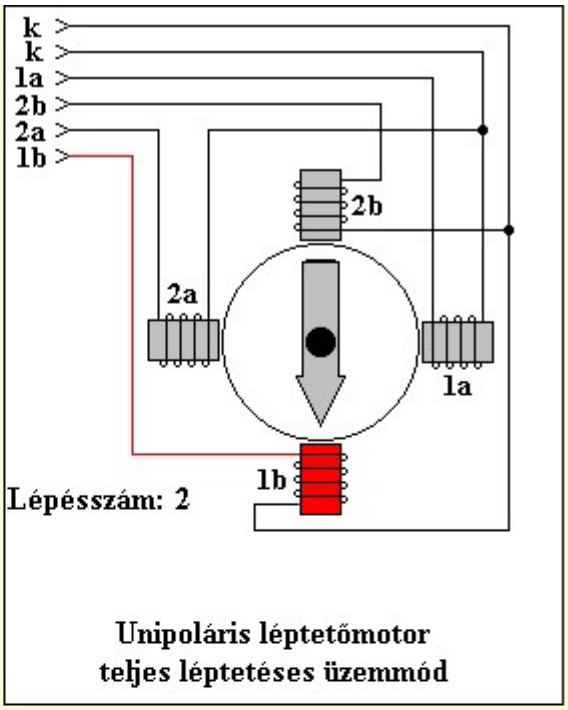
- Alacsony fordulatszám (a maximális fordulatszám tipikusan 500-600

fordulat/perc)
- Azonos teljesítmény mellett nagyobb méret és tömeg
- Nem számítógépes környezetben bonyolultabb vezérlés
- Drágább

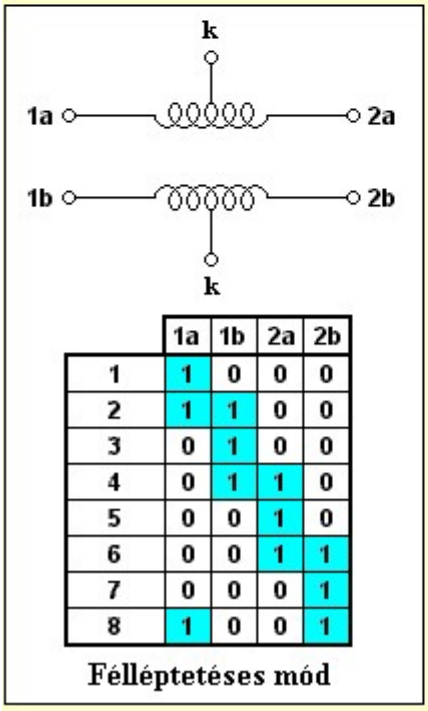


	1a	1b	2a	2b
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

Teljes léptetéses mód

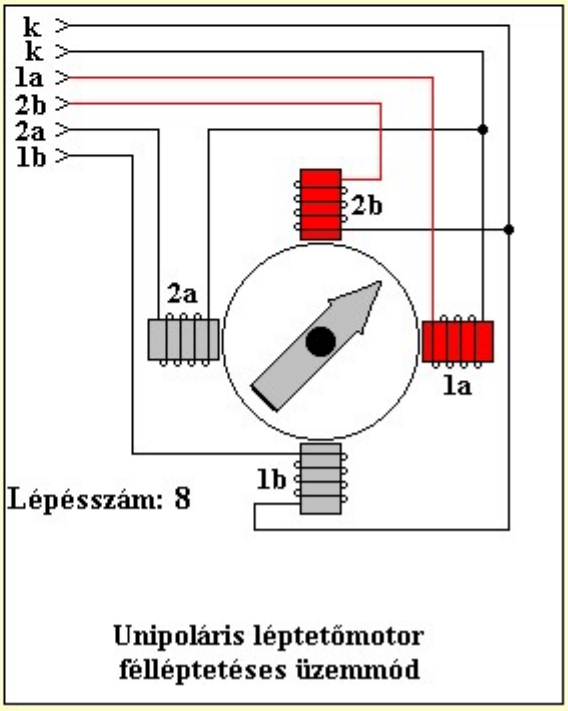


Unipoláris léptetőmotor
teljes léptetéses üzemmód



	1a	1b	2a	2b
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Félléptetéses mód

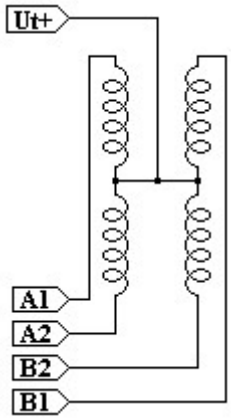


Unipoláris léptetőmotor
félléptetéses üzemmód

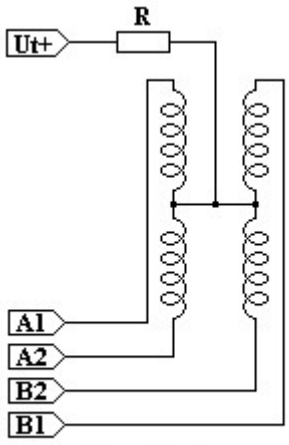
Motorok bekötése

Mint feltűnt itt csak az unipoláris vezérlések vannak lerajzolva. Ennek 2 oka van, egyrészt még nem volt időm a másikra, másrészt egyszerűbb a vezérlése.

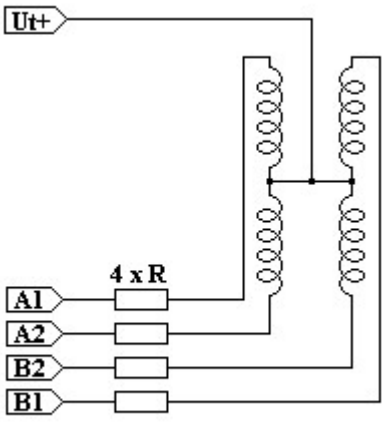
Az is igaz viszont, hogy a bipoláris motorok kicsit erősebbek. Valójában az unipoláris motor nagyon egyszerű bipolárissá alakítani. Ehhez nem kell mást tenni, mint a tekercs



középkivezetését szabadon hagyni, és persze bipoláris vezérléssel ellátni. Elösször a motor bekötése. Az itt közölt kapcsolások mindegyikénél így kell bekötni (elméletben, de rögtön meglátod...) Egyébként a legtöbb unipoláris motort fordítva is bekötheted, tehát a GND kerül a tekercsek közös pontjára... Ezt a kapcsolást csak abban az esetben használhatod, ha a tápfeszültséged stabilizált, és áramkorláttal ellátott.



1.variáció



2. variáció

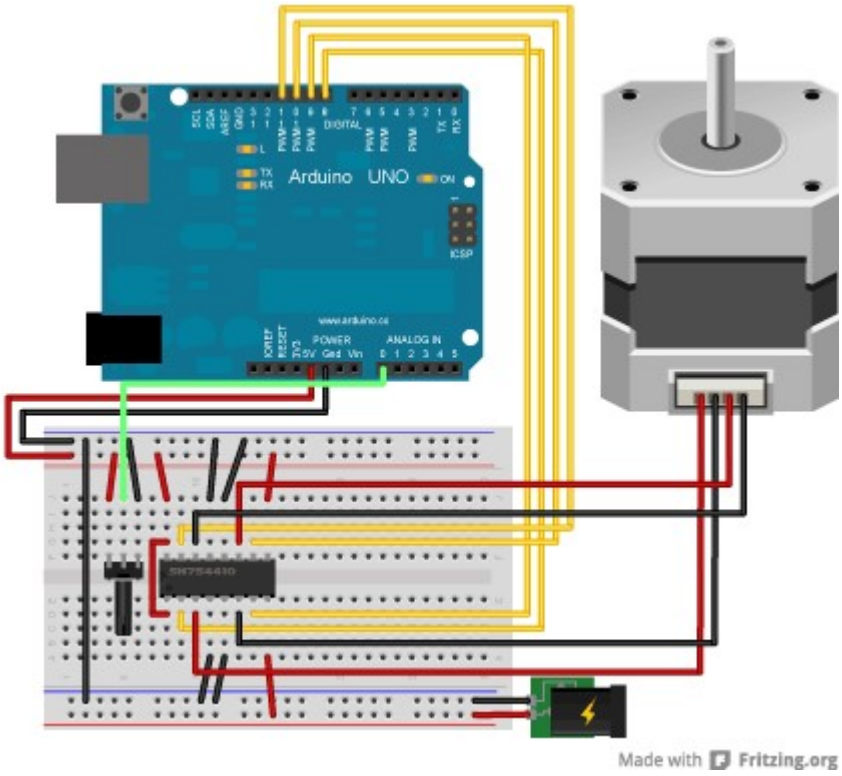
Természetesen ne feledkezzünk meg az áramkorlátról sem (egyszerű stabilizált tápegység esetén). Nem kell megijedni, egy egyszerű ellenállásról van szó, amit a rajznak megfelelően kell bekötni...

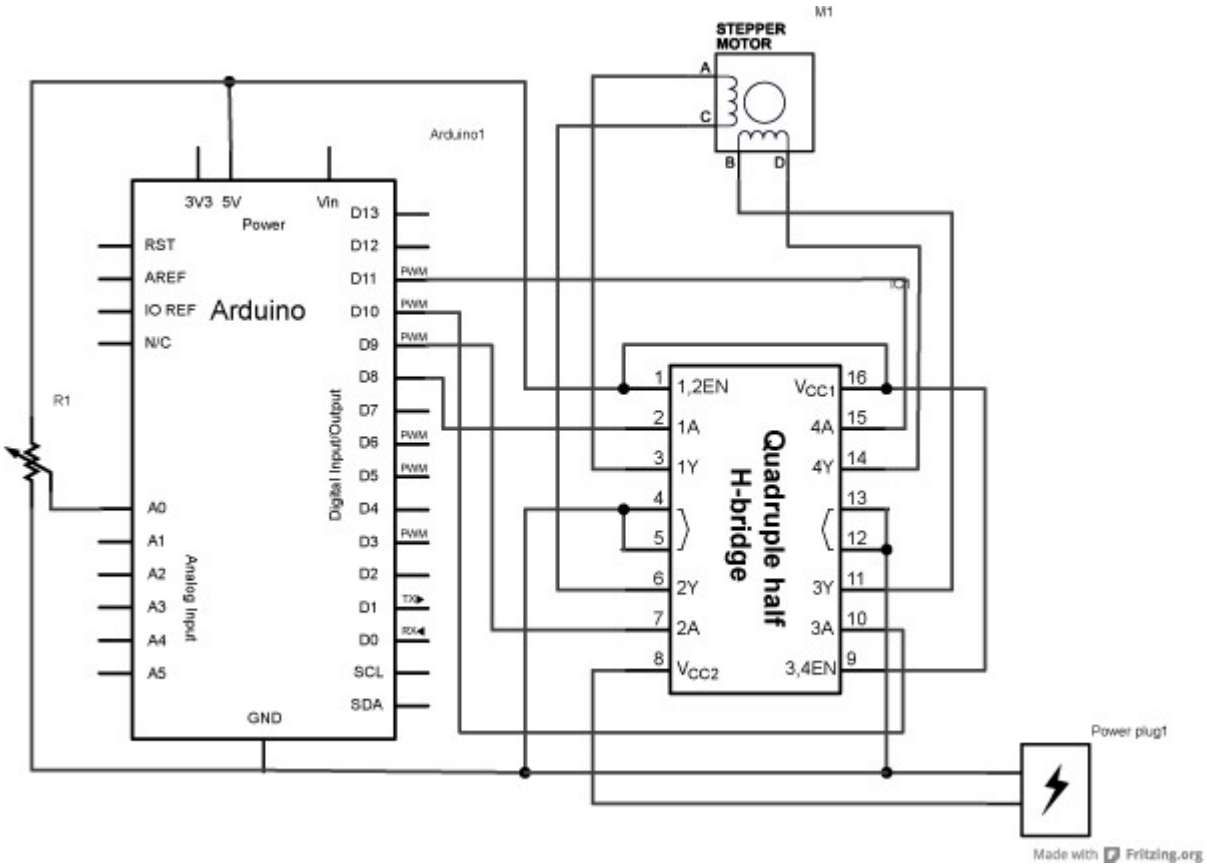
Az 1. variációt csak abban az esetben alkalmazhatod, ha csak teljes léptetéses üzemmódot alkalmazol (tehát egyszerre csak egy tekercsre kerül vezérlés). Amennyiben félléptetéses üzemmódot is szeretnél, akkor mindenképp a 2. variációt kell alkalmazni. Igazság szerint a 2. variációt teljes léptetéses üzemmódnál is alkalmazhatod, azonban egy költséges ellenállásból (Általában több W-os ellenállásról van szó) minek használnál feleslegesen többet a szükségesnél.

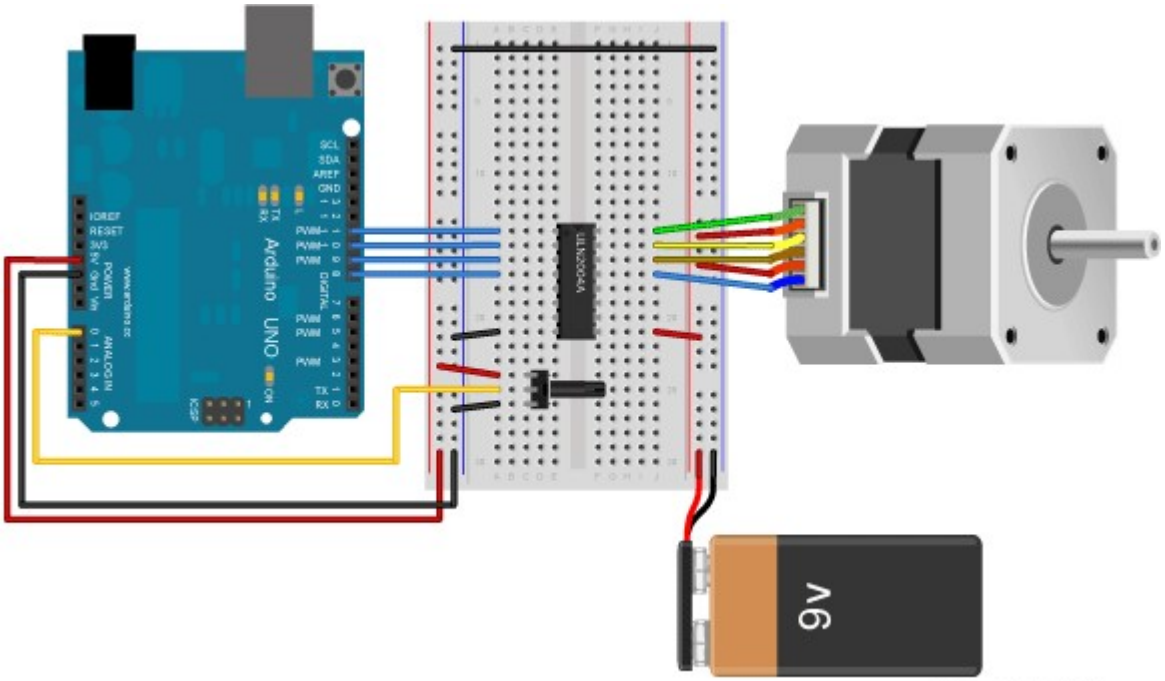
A számítottnál nagyobb értékű ellenállásnak van egy olyan előnye is, hogy nő a léptetési sebesség (mivel csökken az így kialakult R-L tag időállandója). Mint azt valószínűleg sejtetted is, ennek a kisebb nyomaték az ára. Ezenkívül természetesen a rendszered hatásfoka is romlik, hiszen az ellenállás a felesleges energiát "elfűti"...

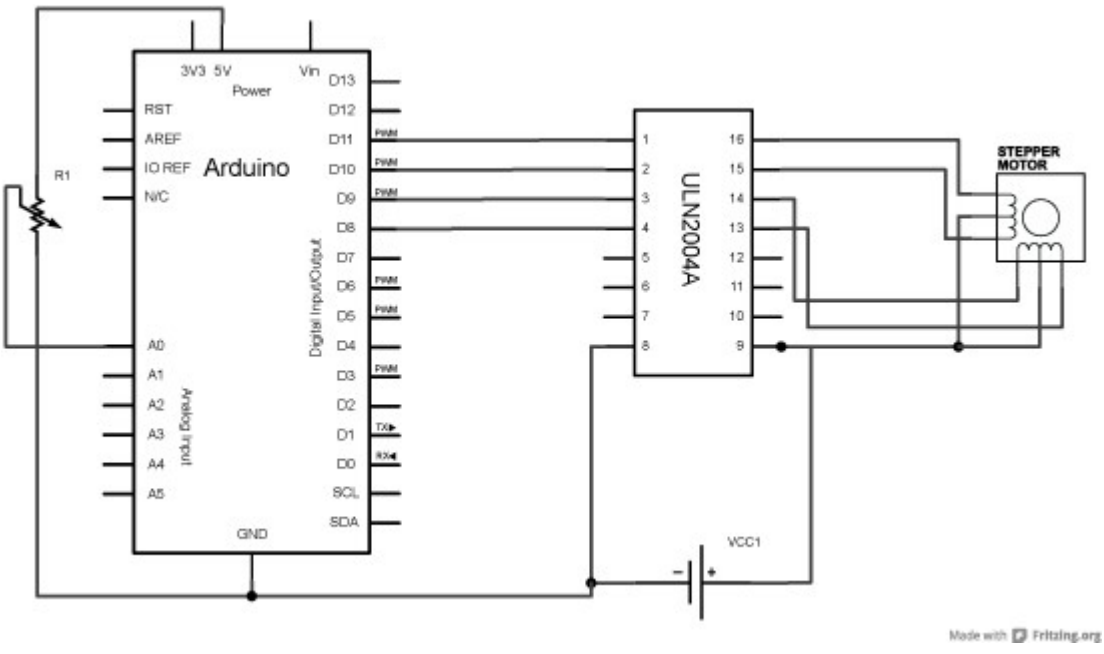
Valójában a leggyakrabban (iparilag) alkalmazott megoldás az úgynevezett chopper kapcsolás. Ebben az esetben a tápfeszültséget a névleges feszültség fölé engedjük, majd amikor az áramerősség elérte a megengedett szintet (ezt egy ellenállással figyeljük), akkor a tekercset rövid időre magára hagyjuk. Ezután természetesen a folyamat indul elölről. FET-et használva kapcsolóelemként szinte veszteségmentes lesz a meghajtás.

Érdemes még tudni a léptetőmotorról, hogy a léptetés után (amennyiben tartósan áll a motor), a motorra jutó teljesítményt korlátozni kell, mivel felesleges energiát fogyaszt, és általában erősen melegszik is a motor... (szerencsétlen esetben le is éghet) Erre (az úgynevezett tartófeszültségre)azért van szükség, hogy a motor akkor is megőrizze a pozícióját, ha nem kap vezérlést (ha elmozdul, akkor a vezérlés is "eltéved"). Ezt végezheti egy külső független áramkör (például egy 555 alapú), vagy printerportról való vezérlés esetén egy adatbittel kapcsolhatod át, hogy melyik áramforrásról kapjon tápot a motor, vagy egy másik ellenállást kapcsol az áramkörbe... Persze megteheted azt is, hogy eleve kisebb teljesítményt engedsz a motorra, azonban ilyenkor az elérhető nyomatékod is kisebb lesz (gyengébb lesz a motor).


Arduino UNO
Made with Fritzing.org


M1
STEPPER MOTOR
A
C
B
D
Arduino1
3V3 5V
Power
Vin
RST
AREF
IO REF
N/C
Arduino
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
SCL
SDA
PWM
TX
RX
Digital Input/Output
A0
A1
A2
A3
A4
A5
Analog Input
GND
R1
1,2EN
1A
1Y
2Y
2A
VCC2
VCC1
4A
4Y
3Y
3A
3,4EN
Quadruple half H-bridge
Power plug1
Made with Fritzing.org


Arduino UNO
ULN2004A
9v
Made with Fritzing.org


Arduino
Power
3V3
5V
Vin
RST
AREF
IO REF
N/C
Digital Input/Output
Analog Input
A0
A1
A2
A3
A4
A5
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
SCL
SDA
PWM
TX
RX
R1
ULN2004A
STEPPER MOTOR
VCC1
Made with Fritzing.org

Kódok

```
/*
 * MotorKnob
 *
 * A stepper motor follows the turns of a potentiometer
 * (or other sensor) on analog input 0.
 *
 * http://www.arduino.cc/en/Reference/Stepper
 * This example code is in the public domain.
 */

#include <Stepper.h>
// change this to the number of steps on your motor
#define STEPS 100

// create an instance of the stepper class, specifying
// the number of steps of the motor and the pins it's
// attached to
Stepper stepper(STEPS, 8, 9, 10, 11);

// the previous reading from the analog input
int previous = 0;

void setup() {
  // set the speed of the motor to 30 RPMs
  stepper.setSpeed(30);
}

void loop() {
  // get the sensor value
  int val = analogRead(0);
  // move a number of steps equal to the change in the
  // sensor reading
  stepper.step(val - previous);
  // remember the previous value of the sensor
  previous = val;
}
```

```
/*
 Stepper Motor Control - speed control
 This program drives a unipolar or bipolar stepper motor.
 The motor is attached to digital pins 8 - 11 of the Arduino.
 A potentiometer is connected to analog input 0.
 The motor will rotate in a clockwise direction. The higher the
potentiometer value,
 the faster the motor speed. Because setSpeed() sets the delay between
steps,
 you may notice the motor is less responsive to changes in the sensor value
at
 low speeds.
 Created 30 Nov. 2009
 Modified 28 Oct 2010
 by Tom Igoe
 */

#include <Stepper.h>

const int stepsPerRevolution = 200;  // change this
to fit the number of steps per revolution
// for your motor


// initialize the stepper library on pins 8 through
11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

int stepCount = 0;  // number of steps the motor has
taken

void setup() {
  // nothing to do inside the setup
}

void loop() {
  // read the sensor value:
  int sensorReading = analogRead(A0);
  // map it to a range from 0 to 100:
  int motorSpeed = map(sensorReading, 0, 1023, 0,
100);
  // set the motor speed:
  if (motorSpeed > 0) {
    myStepper.setSpeed(motorSpeed);
    // step 1/100 of a revolution:
    myStepper.step(stepsPerRevolution / 100);
  }
}
```