Mechanism Design 2 Project

Arduino-Controlled Two-Degree-of-Freedom (DoF) Five-Bar Robot Manipulator

Alp Eren Alpagut 190412028

Haktan Mumcu 180412055

Mehmet Berke Parlat 160412014

Abstract

In this research, we construct a arduino-controlled two-degree-of-freedom (DoF) five-bar robot manipulator. To control the manipulator, we use arduino program after completing the inverse kinematic analysis of the robot manipulator.We determine link lengths using inverse kinematic results.Using the results of the inverse kinematics, the program accurately calculates the joint variables necessary for positioning the end-effector at specified coordinates.

First of all, the link lengths were determined. Then the inverse kinematics of the 5 bar were made. Then the end effector moves linearly in y direction. We achieved this linear rotation by giving equal volts to the motors in opposition to each other with the help of Arduino. Then we designed and produced the links and joints.

Introduction

In the field of robotics, manipulators play a crucial role in performing a wide range of tasks with precision and flexibility. The design and control of robotic manipulators require an in-depth understanding of kinematics to ensure accurate and efficient motion.In this project, we designed and implemented a five-bar manipulator controlled by an Arduino microcontroller. Our manipulator operates with two degrees of freedom (DoF), utilizing servo motors at the joints for accurate positioning and movement.

A unique aspect of our design is the inclusion of two magnets, one placed at the end of the five-bar linkage and the other positioned opposite it. These magnets are oriented to repel each other, creating an additional force component that interacts with the forces generated by the servo motors. This interaction between the repulsive magnetic force and the mechanical force from the motors provides an opportunity to study and calculate the resulting torque on the system.

By analyzing the opposing forces from the magnets and the servo motors, we can gain insights into the torque dynamics of the system. This project not only demonstrates the practical application of inverse kinematics and Arduino-based control but also explores the interplay of magnetic forces and mechanical motion in a robotic context. The findings could have significant implications for future robotic designs and applications.

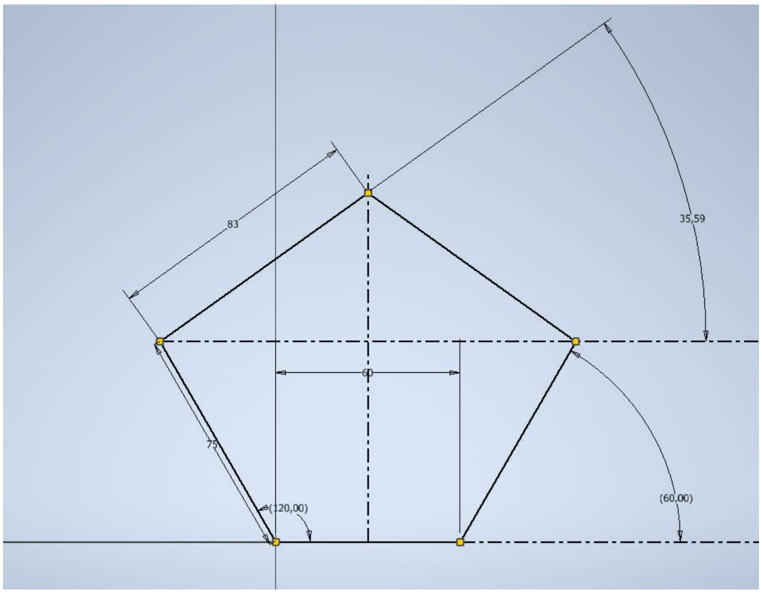


Fig 1-Display of five bar angles and link lengths in Inventor

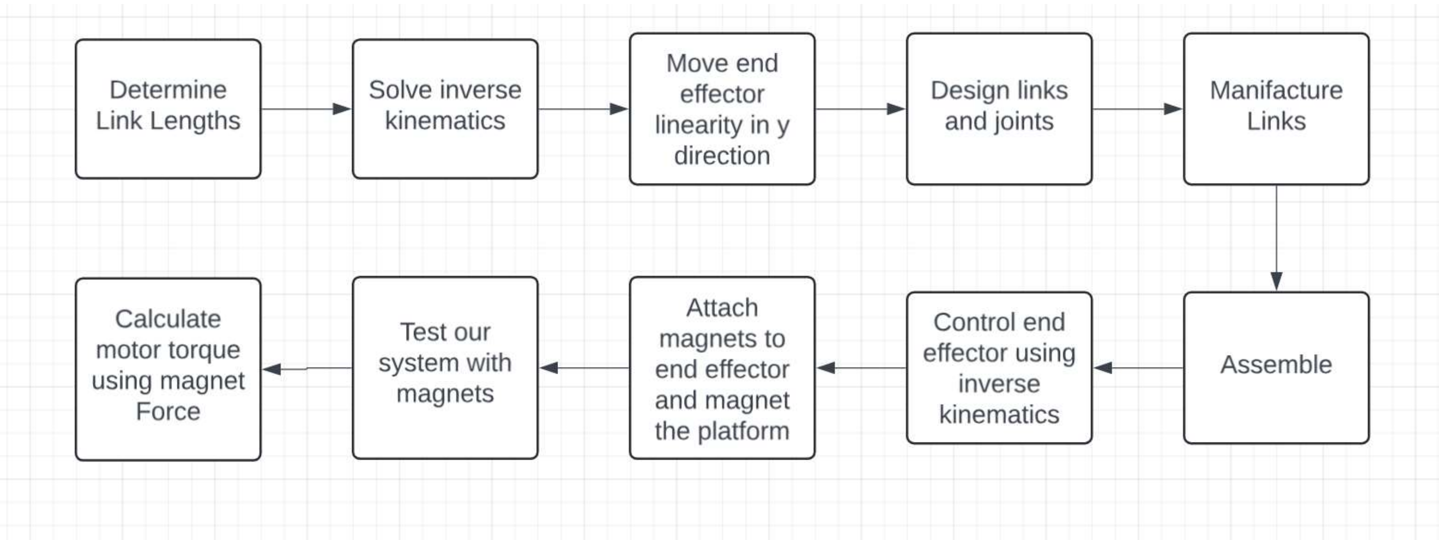
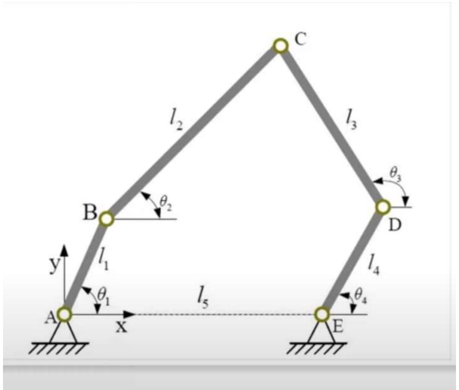


Fig 2-Flow Chart of Project Steps

Calculation of the coordinates



Coordinates of point B

$x_b = l_1 cos(\theta_1)$ = 75*cos(60)=37,5

$y_b = l_1 sin(\theta_1)$=75*sin(60)=64.95

Coordinates of point D

$x_d = l_5 + l_4 cos(\theta_4)$ = 60+75*cos(60)=97,5

$y_d = l_4 sin(\theta_4)$= 75*sin(60)=64.95

Coordinates of point C

$x_c = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) = l_5 + l_4 cos(\theta_4) + l_3 cos(\theta_3)$(1)

$y_c = l_1 sin(\theta_1) + l_2 sin(\theta_2) = l_4 sin(\theta_4) + l_3 sin(\theta_3)(2)$

From eq(1-2) it can be found that θ_1 and θ_4 are independent in the system and θ_2 and θ_3 can be determined by θ_1 and θ_4 as follows

$$\theta_3 = 2\arctan\left(\frac{A \pm \sqrt{A^2 + B^2 - C^2}}{B - C}\right)$$

where;

$A = 2l_3 l_4 sin(\theta_4) - 2l_1 l_3 sin(\theta_1)$

$B = 2l_3 l_5 - 2l_1 l_3 cos(\theta_1) + 2l_3 l_4 cos(\theta_4)$

$$C = {l_1}^2 - {l_2}^2 + {l_3}^2 + {l_4}^2 + {l_5}^2 - 2l_1l_4 sin(\theta_1) sin(\theta_4) - 2l_1l_5 cos(\theta_1) + 2l_4l_5 cos(\theta_4) - 2l_1l_4 cos(\theta_1) cos(\theta_4)$$

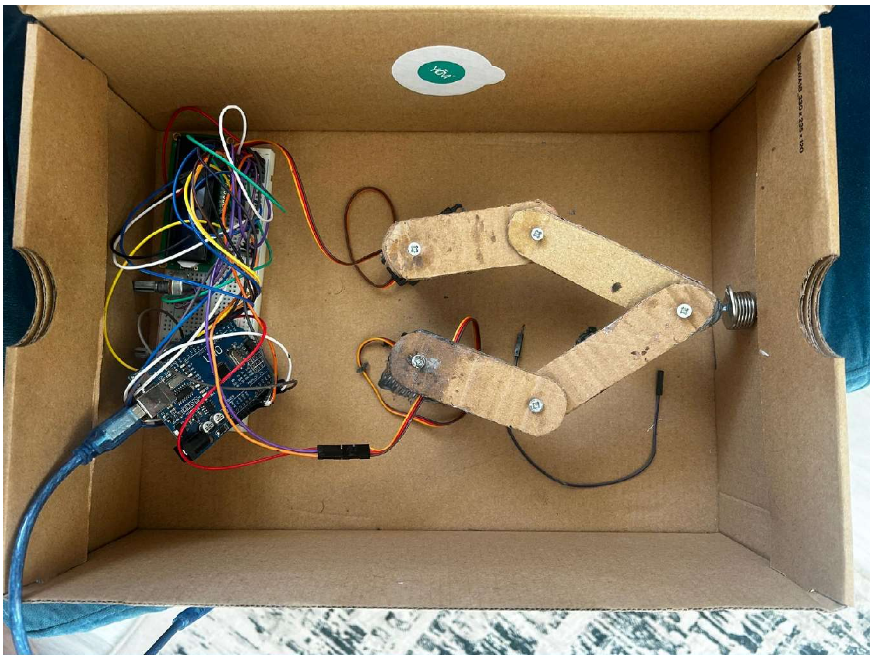


Fig. 4. Electrical connection and whole setup

```matlab
close; clc; clear;
% Parameters
% firstly we define link lengths and the time
l1 =1;l2=3;l3=3;l4=1;l5=2;
% Adjusting the time interval
t=0:0.05:10;
T=0:0.05:9.975;
% then we define angular velocities of the links as shown
omega1 = 2;
omega2 = 1;
% So theta will be omega*time
th1 = omega1*t;
th4 = omega2*t;
% Lets assume point A is at the origin So
A = [0;0];
% Lets assume point E is at x-axis. So the coordinates will be 15 distance away ↙
from the point A in x-direction E = 15*[1;0];
B = [l1*cos(th1);l1*sin(th1)];
D= [l5+l4*cos(th4):l4*sin(th4)]:

a = 2*l3*l4*sin(th4)-2*l1*l3*sin(th1);
b = 2*l3*l5 - 2*l1*l3*cos(th1)+2*l3*l4*cos(th4);
c = l1^2-l2^2 +l3^2 + l4^2 + l5^2 - 2*l1.*l4.*sin(th1).*sin(th4)-2.*l1.*l5.*cos ↙
(th1)+2.*l4.*l5.*cos(th4)-2.*l1.*l4.*cos(th1).*cos(th4);
th3 = 2*atand((a+sqrt(a.^2+b.^2-c.^2))/(b-c));
th2 = asind((l3*sin(th3)+l4*sin(th4)-l1*sin(th1))/l2);
% defining point C
C_x = l1*cos(th1)+l2*cos(th2); C_y=l1*sin(th1)+l2*sin(th2);
C= [C_x;C_y];
% distance of point C from origin(A)
C_d = sqrt(C_x.^2 + C_y.^2);

% by differentiating coordinates with respect to time we can get the
% velocity of point C
% x-component
C_vx = diff(C_x)./diff(t);
% y-component
C_vy = diff(C_y)./diff(t);
% magnitude of the velocity of point C
C_v=sqrt(C_vx.^2 + C_vy.^2);
%by differentiating velocity components with respect to time we can get the ↙
acceleration of point C
C_ax = diff(C_vx)./diff(T);
C_ay = diff(C_vy)./diff(T);
% magnitude of the acceleration of point C
C_a = sqrt(C_ax.^2+C_ay.^2);
for i=1:length(T)
ani = subplot(4,1,1);
% firstly create small circles to show joints clearly
p1_circle = viscircles(A',0.05);
p2_circle = viscircles (B(:,i)',0.05);
p3_circle = viscircles(C(:,i)',0.05); p4_circle = viscircles(D(:,i)',0.05); ↙
```

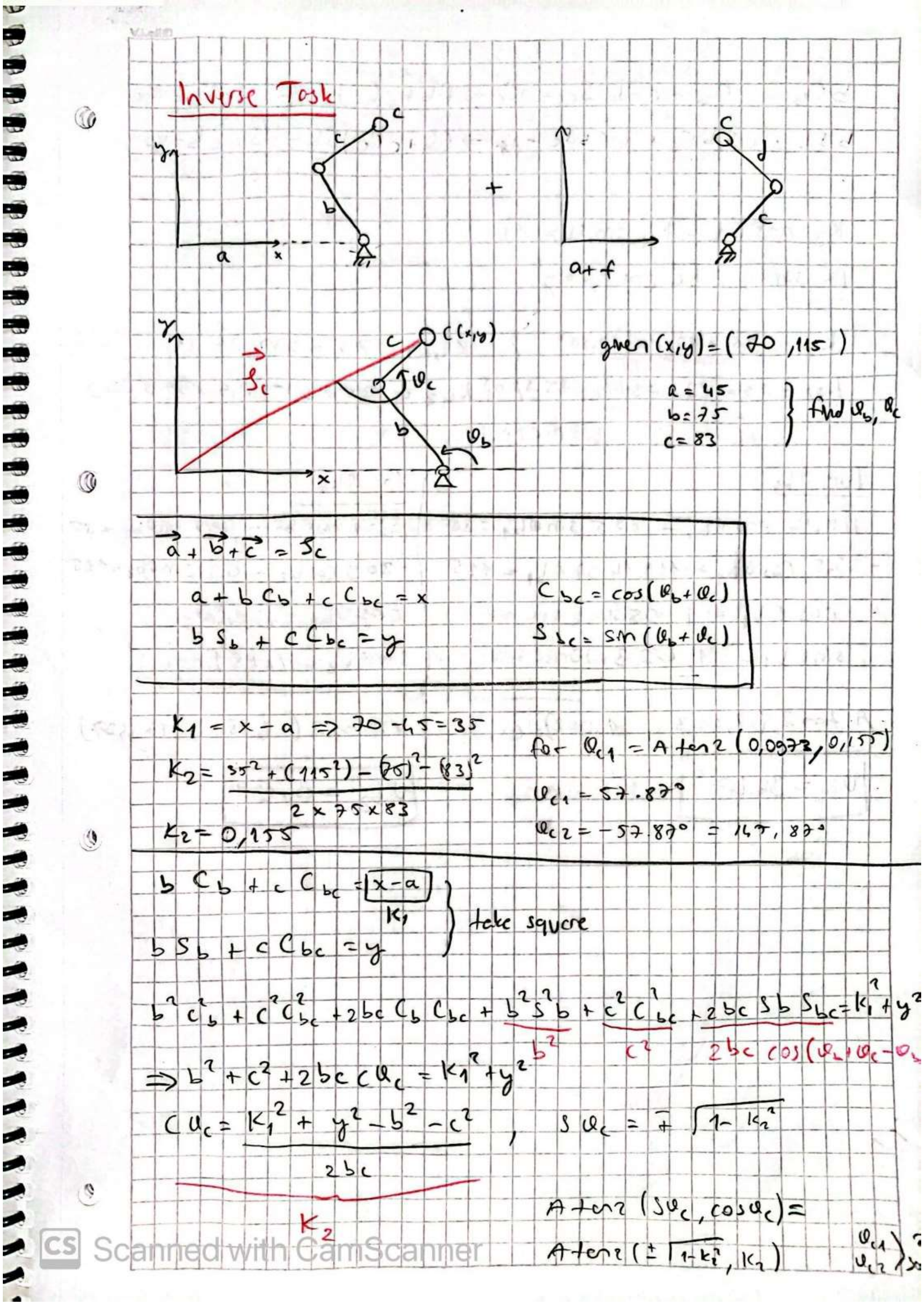
```
p5_circle = viscircles(E',0.05);
% after that show the bars by plotting lines AB_bar = line([A(1) B(1,i)],[A(2) B(2,
i)]); BC_bar = line([B(1,i) C(1,i)],[B(2,i) C(2,i)]); CD_bar = line([C(1,i) D(1,
i)],[C(2,i) D(2,i)]); DE_bar = line([E(1) D(1,i)], [E(2) D(2,i)]);
% adjusting axises axis(ani,'equal');
set(gca,'XLim',[-4 4], 'YLim',[-4 4]);
%plotting position vs time
dis = subplot(4,1,2); plot(dis,t(1:l),C_x(1:i));
set(dis,'XLim',[0,10],'YLim',[-5,5]); xlabel(dis, Time (s)');
ylabel(dis, 'distance wrt A');
% plotting velocity vs time vel = subplot(4,1,3);
plot(vel,t(1:l),C_v(1:i));
set(vel,'XLim', [0 10], 'YLim', [0 500]);
xlabel(vel, 'Time (s)');
ylabel(vel, 'amplitude');
title(vel,'Velocity of C');
grid on;
% plotting amplitude of acceleration vs time acc = subplot(4,1,4);
plot(acc,T(1:l),C_a(1:i));
set(acc,'XLim',[0,10], 'YLim',[0,5000]);
xlabel(acc,'Time (s)');
ylabel(acc,'amplitude');
title(acc,'Acceleration of C');
grid on;
```

Arduino Code

```
#include <Servo.h>
#include <LiquidCrystal.h>

// Create objects for the servo motors
Servo servo1;
Servo servo2;

// Potentiometer pin
const int potPin = A0;

// Servo motor pins
const int servoPin1 = 5;
const int servoPin2 = 6;

// Create object for the LCD (RS, E, D4, D5, D6, D7 pins)
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

// Force
const float force = 26.0; // Newton

// Distances (meters)
const float minDistance = 0.11; // 11 cm
const float maxDistance = 0.15; // 15 cm

// Magnetic force distances (meters)
const float minMagneticDistance = 0.01; // 10 mm
const float maxMagneticDistance = 0.055; // 55 mm

void setup() {
  // Attach servo motor pins
  servo1.attach(servoPin1);
  servo2.attach(servoPin2);

  // Initialize the LCD
  lcd.begin(16, 2);

  // Start serial communication
  Serial.begin(9600);
}


void loop() {
  // Read potentiometer value
  int potValue = analogRead(potPin);

  // Convert potentiometer value to servo angles (0 - 180 degrees)
  int angle = map(potValue, 0, 1023, 0, 180);

  // Set new angles for the servo motors
  servo1.write(angle);
  servo2.write(180 - angle); // Reverse direction by using 180 -
value

  // Calculate distance (in meters)
  float distance = map(potValue, 0, 1023, minDistance * 1000,
maxDistance * 1000) / 1000.0;

  // Torque calculation (in Newton meters)
  float torque = distance * force;

  // Calculate magnetic distance (in meters)
  float magneticDistance = map(potValue, 0, 1023,
minMagneticDistance * 1000, maxMagneticDistance * 1000) / 1000.0;

  // Calculate magnetic torque
  float magneticTorque = force * magneticDistance;

  // Calculate net torque
  float netTorque = torque - magneticTorque;

  // Print values to the serial monitor (optional)
  Serial.print("Servo Angle: ");
  Serial.print(angle);
  Serial.print(" - Net Torque: ");
  Serial.print(netTorque);
  Serial.println(" N*m");

  // Print values to the LCD
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Angle: ");
```

```
  lcd.print(angle);

  lcd.setCursor(0, 1);
  lcd.print("Torque: ");
  lcd.print(netTorque, 3); // Print net torque with 3 decimal
places
  lcd.print(" N*m");

  // Wait briefly
  delay(500);
}
```

Inverse Kinematics Calculation



Inverse Task

y, a, x, b, c, C, +, a+f, C, d, c

y, x, $\vec{S}_c$, c, C(x,y), θ_c, b, θ_b

given $(x,y) = (70, 115)$

$a = 45$, $b = 75$, $c = 83$ } find θ_b, θ_c

$\vec{a} + \vec{b} + \vec{c} = \vec{S}_c$

$a + b C_b + c C_{bc} = x$

$b S_b + c C_{bc} = y$

$C_{bc} = \cos(\theta_b + \theta_c)$

$S_{bc} = \sin(\theta_b + \theta_c)$

$K_1 = x - a \Rightarrow 70 - 45 = 35$

$K_2 = \dfrac{35^2 + (115^2) - (75)^2 - (83)^2}{2 \times 75 \times 83}$

$K_2 = 0,155$

for $\theta_{c1} = \text{Atan2}(0,0973, 0,155)$

$\theta_{c1} = 57.87°$

$\theta_{c2} = -57.87° = 147,87°$

$b C_b + c C_{bc} = \underbrace{\boxed{x-a}}_{K_1}$

$b S_b + c C_{bc} = y$ } take square

$b^2 C_b^2 + c^2 C_{bc}^2 + 2bc C_b C_{bc} + \underbrace{b^2 S^2 b}_{b^2} + \underbrace{c^2 C^2_{bc}}_{c^2} + \underbrace{2bc\, S_b S_{bc}}_{2bc \cos(\theta_b + \theta_c - \theta_b)} = K_1^2 + y^2$

$\Rightarrow b^2 + c^2 + 2bc\, C\theta_c = K_1^2 + y^2$

$C\theta_c = \underbrace{\dfrac{K_1^2 + y^2 - b^2 - c^2}{2bc}}_{K_2}$, $S\theta_c = \mp\sqrt{1 - K_2^2}$

$\text{Atan2}(S\theta_c, \cos\theta_c) =$

$\text{Atan2}(\pm\sqrt{1-K_2^2}, K_2)$ θ_{c1}, θ_{c2}

$$b C_b + c C_b - c S_b S_c = K_1 \Rightarrow \overbrace{(b + c C_c)}^{K_3} C_b - c S_c S_b = K_1$$

$$b S_b + c S_b C_c + c C_b S_c = y \Rightarrow (b + c C_c) S_b + \underbrace{c S_c}_{K_4} C_b = y$$

$$K_3 \cos \theta_b - K_4 \sin \theta_b = K_1$$

$$K_3 \sin \theta_b - K_4 \cos \theta_b = y$$

$$K_{31} = 75 + \underbrace{83 \cos 57.87°}_{119.14} \qquad K_{41} = 75 \times \sin 57.87° > 63.5$$

$$K_{32} = 75 + \underbrace{83 \cos 147.87°}_{4.71} \qquad K_{42} = 75 \times \sin 147.87° > 39.9$$

<u>For θ_{b_1}</u>

$$119.14 \cos \theta_{b_1} - 63.5 \sin \theta_{b_1} = 35$$

$$-63.5 \cos \theta_{b_1} + 119.14 \sin \theta_{b_1} = 115$$

$$\cos \theta_{b_1} = 1,052$$

$$\sin \theta_{b_1} = 1,423$$

$$A\tan 2\,(1,423 \;,\; 1,052)$$

$$\boxed{\theta_{b_1} = 36.47°}$$

<u>for θ_{b_2}</u>

$$4.71 \cos b_2 - 39.9 \sin \theta b_2 = 35$$

$$-39.9 \cos b_2 + 4.71 \sin \theta b_2 = 115$$

$$\cos \theta b_2 = -17,437$$

$$\sin \theta b_2 = -1,181$$

$$A\tan 2\,(-1,18 \;,\; -17,437)$$

$$\boxed{\theta b_2 = -86,12°}$$

Magnetic Force Calculation

$$K = 10^{-7}\,\mathrm{H/m}$$
$$m_1 = 0.7\,\mathrm{T}$$
$$m_2 = 0.7\,\mathrm{T}$$
$$\mu = 4\pi \times 10^{-7}\,\mathrm{H/m}$$
$$r = 10\,\mathrm{mm} = 0.01\,\mathrm{m}$$
$$F = K\frac{m_1 m_2}{\mu r^2}$$
$$r^2 = (0.01\,\mathrm{m})^2 = 0.0001\,\mathrm{m}^2$$
$$F = 10^{-7} \times \frac{(0.7)\times(0.7)}{4\pi\times 10^{-7}\times 0.0001}$$
$$F = 10^{-7} \times \frac{0.49}{4\pi\times 10^{-7}\times 10^{-4}}$$
$$F = 10^{-7} \times \frac{0.49}{1.25664\times 10^{-10}}$$
$$F = 10^{-7} \times 3.898 \times 10^{9}$$
$$F = 3.898 \times 10^{2}\,\mathrm{N}$$